This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2025.3618876

TruChord: A Secure Communication Framework for Hybrid SDIoT Architecture Based on Chord Overlay

Yu Zhang, Bei Gong, *Member, IEEE*, Zahid Halim *Senior Member, IEEE*, Hisham Alasmary *Member, IEEE*, Muhammad Waqas, *Senior Member, IEEE*, and Iftekhar Ahmad *Senior Member, IEEE*

***Abstract*—As a highly integrated and comprehensive application of the new generation of information technology, IoT inevitably comes with security issues. The communication security of IoT devices is the foundation for their large-scale development. To address the problems of missing cross-domain identity authentication, low routing efficiency, and frequent malicious attacks in hybrid SDIoT, we propose a secure and trusted communication framework for IoT based on overlay networks, namely TruChord. This framework introduces DICE technology to generate device firmware identifiers, constructs a trust chain, and realizes hardware binding of device identities and firmware integrity measurement. Meanwhile, it designs a domain-specific identifier mapping mechanism, which integrates SDN domain DPID, IP domain prefix, and firmware identifiers to make physically adjacent nodes logically adjacent on the ring. Finally, it proposes a hierarchical trust aggregation protocol, which ensures secure communication between nodes through intra-domain local verification and cross-domain global aggregation with the help of a trust mechanism. Through formal proof, TruChord can resist Sybil attacks and collusion attacks. Simulation experiments show that TruChord outperforms the comparison frameworks in core indicators such as end-to-end delay, network traffic, and convergence time, providing an efficient solution for cross-domain secure communication of IoT devices.**

***Index Terms*—Hybrid SDIoT, Chord overlay,the trust mechanism, Cross-domain communication security.**

I. Introduction

FRom smart home devices to industrial sensors, the Internet of Things (IoT) has expanded the interaction between humans and technology from discrete device control to full-domain life scenarios. The traditional Internet of Things (IoT) architecture faces the dual challenges of insufficient scalability and sharply increasing management complexity in large-scale scenarios, while the centralized control mechanism of Software-Defined Networking (SDN) provides a systematic solution to these problems.The traditional IP network cannot be immediately replaced by the SDN architecture, thus forming a hybrid SDN architecture where the two network architectures coexist. CPqD, the R&D center of Brazilian Telecom, developed the RouterFlow mechanism [1], which was the first to realize the interconnection between SDN and IP networks. Google achieved the interconnection between SDN and IP by adding a Border Gateway Protocol (BGP) component to the controller. This hybrid Software-Defined IoT (SDIoT) architecture is projected to become increasingly prevalent [2].According to Gartner's prediction, 50% of enterprises will deploy SDN by 2025 to address the complexity of traditional network operation and maintenance [3], [4].

In hybrid SDIoT, fundamental differences exist between SDN autonomous domains and IP autonomous domains [5]. The IP domain adopts a decentralized autonomous model, where node identities are primarily identified by 48-bit MAC addresses or 32-bit/128-bit IP addresses. In contrast, the SDN domain employs a centralized management and control model, where the identities of nodes (such as switches and edge terminals) are dynamically assigned by the controller (e.g., DeviceID defined in flow tables). Identifiers based on "controller ID + device ID" are mostly in 64-bit/128-bit custom formats. These differences make it difficult for nodes to recognize each other's identities during cross-domain communication [6], [7]. For instance, an IP node cannot verify the controller-assigned identifier of an SDN node. Meanwhile, identity authentication of nodes in the IP domain relies on locally stored certificates (e.g., X.509) or pre-shared keys, with verification performed independently by each node. In the SDN domain, authentication depends on pre-shared keys between the controller and nodes or centrally issued short-term certificates, and verification is managed by the controller. Therefore, during cross-domain communication, nodes in the SDN domain and IP domain cannot parse the encoding rules of each other's identifiers. Additionally, when an IP node attempts to verify an SDN node, it cannot confirm the legitimacy of the node's identifier because it cannot access the root certificate of the SDN controller. Conversely, when the SDN controller verifies an IP node, it is difficult to determine the node's real identity due to the weak correlation between IP identifiers and hardware (e.g., MAC addresses can be forged). Malicious nodes can exploit the vulnerabilities in the aforementioned mechanisms to launch multi-dimensional attacks, including:

Y. Zhang is with the School of Information Sciences and Technology, Zhengzhou Normal University, Henan, 450044, China (e-mails: zhangyu@zznu.edu.cn)

B. Gong is with the Faculty of Information Technology, Beijing University of Technology, Beijing 100124 China (e-mails: gongbei@bjut.edu.cn)

Z. Halim is with the Department of Information Management, National Yunlin University of Science and Technology, Douliou, Yunlin 64002, Taiwan (e-mails: zahidh@yuntech.edu.tw)

H. Alasmary is with the Department of Computer Science, King Khalid University, Abha, Saudi Arabia. (e-mails: alasmary@kku.edu.sa)

M. Waqas is the School of Computing and Mathematical Sciences, Faculty of Engineering and Science, University of Greenwich, UK and also with the School of Engineering, Edith Cowan University, Perth, 6007 WA, Australia (e-mails: engr.waqas2079@gmail.com)

I. Ahmad is with the School of Engineering, Edith Cowan University, 6027 Perth, Australia (e-mails: i.ahmad@ecu.edu.au)

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2025.3618876

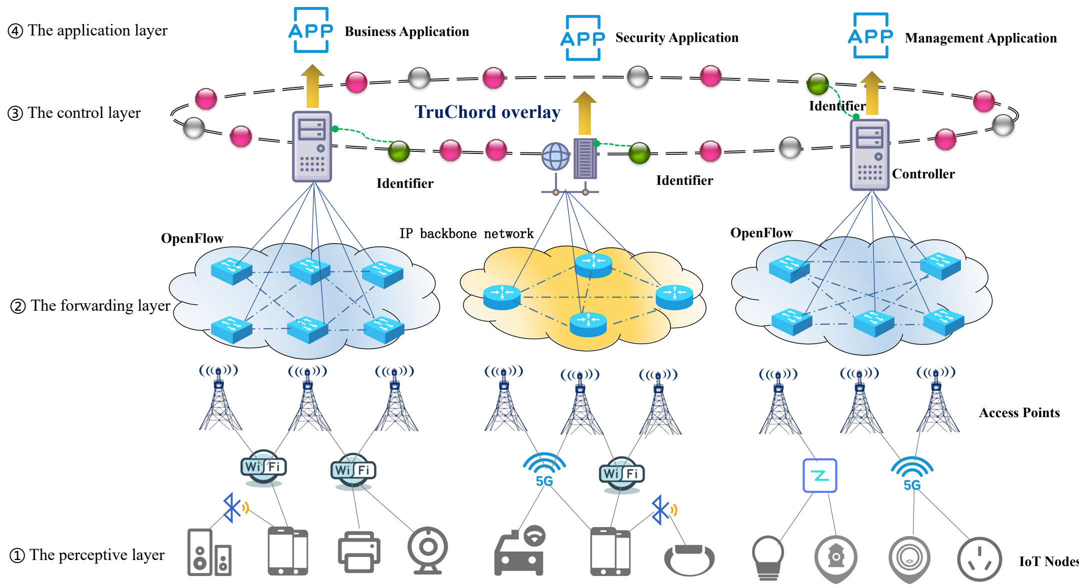


Fig. 1. System model of hybrid SDIoT

device impersonation and unauthorized access, data theft and instruction tampering, and trust chain contamination [8]–[10]. Thus, there is an urgent need to address the issues of inconsistent cross-domain identity identifiers and authentication challenges in hybrid SDIoT.

Overlay Network, as a logical network built on top of a physical network, is not restricted by the underlying physical network topology and can mask the differences between heterogeneous networks [11], [12]. Moreover, all data in the overlay network is processed locally [13]. Even if the underlying layer is monitored, attackers still cannot obtain information on the overlay nodes, and the data security at the logical layer can still be guaranteed, thereby improving the system security [14].Chord is a ring-shaped overlay network proposed by the Massachusetts Institute of Technology and the University of California, Berkeley. It uses a 160-bit identifier space with 2^{160} identifiers to map network device objects onto the ring [15]. In this paper, SDIoT is adopted as the underlying network infrastructure, and cross-domain interconnection is realized through the Chord overlay network. Meanwhile, end-to-end security mechanisms are deployed at the logical layer to construct a trusted and secure IoT framework, namely TruChord. The overall network structure is shown in Fig. 1. By leveraging the characteristics of the overlay network, TruChord realizes unified identity identification through node ID hash mapping. Specifically, the IP addresses of nodes in the IP domain and the controller identifiers of nodes in the SDN domain are mapped into 160-bit Chord node IDs via one-way functions (e.g., SHA-256). All nodes exist with unique IDs in the overlay network, which masks the heterogeneous differences between SDN nodes and IP nodes and enables cross-domain interconnection.

In traditional Chord networks, node identities are generated merely through hashing, lacking hardware-level binding. Malicious nodes can forge multiple identities to join the network, making it vulnerable to Sybil attacks [16], [17]. To ensure node identity authentication, in TruChord, the logical ID generation mechanism for IoT nodes no longer relies solely on random hashing. Instead, it is combined with the Device Identity Composition Engine (DICE) to achieve hardware binding, confirming the legitimacy of the physical identity of IoT nodes and preventing forged nodes from joining. DICE is a lightweight root-of-trust security architecture standard launched by the Trusted Computing Group (TCG) [18]. With the help of the lightweight root-of-trust DICE, it ensures that IoT devices establish a trusted chain from the boot phase, preventing IoT devices from being implanted with malicious firmware or having their identities forged, and fundamentally curbing the risk of node identity forgery [19], [20].

In this paper, we first construct a trusted device identification system at IoT terminals to realize device integrity verification, thereby ensuring that nodes cannot be tampered with and providing trusted identity information for IoT nodes. Then, through the TruChord overlay network, we map physically adjacent nodes in the underlying physical network to adjacent positions on the ring, thereby reducing lookup delay. Finally, by integrating the IoT trust mechanism, we achieve intra-domain local verification and cross-domain global aggregation to ensure secure communication between nodes. The main contributions of this paper are as follows:

- A hybrid SDIoT secure communication framework based on Chord overlay is designed, and a TruChord overlay construction scheme with sub-domain node identifiers is proposed. The SDN and IP domain address spaces are partitioned into sub-domains, and a mapping scheme for proximity node identifiers is established using the OpenFlow/IP prefix address space and identity information to enhance node lookup and location efficiency.
- A DICE technology is integrated into the TruChord security architecture as a trusted root to complete the trust chain transfer, achieve device measurement and integrity verification, and generate IoT device firmware identifiers to provide identity authentication for device network access.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2025.3618876

- A trusted communication protocol based on TruChord is proposed to determine the trustworthiness of nodes, based on the trust values of $Requester$, $Networker$ and $Target$ through the trust mechanism, and resists switch attacks, bad-mouth attacks, and vote stuffing attacks.

Simulation experiments show that the TruChord framework achieves more balanced load distribution within regions. By reducing cross-domain hops through domain-specific mapping, it effectively lowers query latency and can provide accurate trust decisions with a high probability. It is significantly superior to the comparison frameworks in terms of both efficiency and security.

II. Related Work

Extensive research has been conducted on Chord overlay architectures. In Chord [15], nodes directly map keys and values based on name and location using a hash function. The existing literature improves upon the standard Chord ring. In [21], a reverse routing table and a fast table with frequently accessed information are designed to enable bidirectional routing by completing node location and lookup in both counterclockwise and clockwise directions. In [22], the focus is on encrypting routing messages to resist eavesdropping, but it retains the original identity management mechanism of Chord and thus cannot solve the problem of cross-domain identity inconsistency in hybrid networks. In [23], a trust evaluation model is introduced, but it relies on trust propagation between nodes and lacks a hardware-level root of trust, making it vulnerable to identity forgery attacks.

To guarantee the trusted identity authentication of IoT nodes, trusted integrity verification is generally implemented by hardware, such as Trusted Platform Module (TPM) and TrustZone [24]. The DICE technology development for resource-constrained IoT devices is a lightweight, trusted solution representing the state-of-theart development of trusted computing technology. The research [25] focuses on an IoT device, which is a radio frequency identification (RFID) reader, employing DICE technology. Through an analysis of the system's startup process, the study devises a method for establishing and extending trust chains. Moreover, it introduces a DICE-based scheme for measuring and updating devices in an IoT environment. DICE uses a secret device identifier to measure the integrity of software modules sequentially during device startup, and the process in this way generates a series of secret values.

Regarding the SDIoT security framework, the existing works [26]–[28] focused on centralized security policy management in the SDN domain, enhancing intra-domain security through unified controller management and control. However, it neglects compatibility design with the IP domain, resulting in the formation of "security islands" in cross-domain scenarios and hindering the collaboration of heterogeneous networks. The research [29] concentrates on the integrity protection of SDN flow tables, without involving a unified identity mechanism between heterogeneous nodes or cross-domain transmission of trust chains, thus failing to meet the requirements for trusted interaction among nodes in hybrid networks.

III. Proposed Model and Framework

A. TruChord Node Trusted Identity Verification

DICE leverages a Unique Device Secret (UDS) and firmware hash to generate a Critical Device Identity (CDI), ensuring a hard binding between device identity and firmware. As the fundamental hardware-level identifier, the UDS is permanently embedded in the device's BOOT ROM by IoT manufacturers and is accessible exclusively to the DICE engine. The chain of trust originates from the UDS, with hierarchical measurements extending from L1 to L3, as illustrated in Fig. 2. The device measurement and verification process based on DICE is described below:

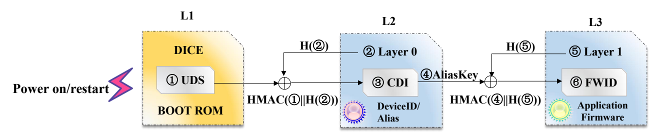


Fig. 2. DICE trust chain transfer process

1) L1: (Hardware Initialization: Generating CDI from UDS): Layer0 constitutes the device's most fundamental boot code. The hash function measures the code executed in Layer 0, producing $BootHash = SHA-256(Layer0)$. Subsequently, the UDS s combined with BootHash using a Hash-based Message Authentication Code (HMAC) function to generate $CDI = HMAC(UDS, BootHash)$. Following this, all data in memory is erased to prevent snooping or inference of any UDS-related information, thereby enhancing IoT device security. Finally, the DICE engine transfers control to L2, as depicted in Fig. 3.

2) L2: (Identity Generation: Interaction between Device Identity and Alias Keys): The CDI is used to generate the device's unique permanent identity (called $DeviceID$) and an alias key (called $AliasKey$). The $DeviceID$ serves as the device's lifetime identity requiring strict protection, for which a self-signed $DeviceID$ certificate is generated. The $AliasKey$, employed for daily communications, mitigates private key exposure risks and generates its associated certificate. Specifically, the alias certificate is created by signing the public key of the AliasKey with the private key of the $DeviceID$: $AliasCert = Sign(DeviceID_{priv}, AliasKey_{pub})$. The hash

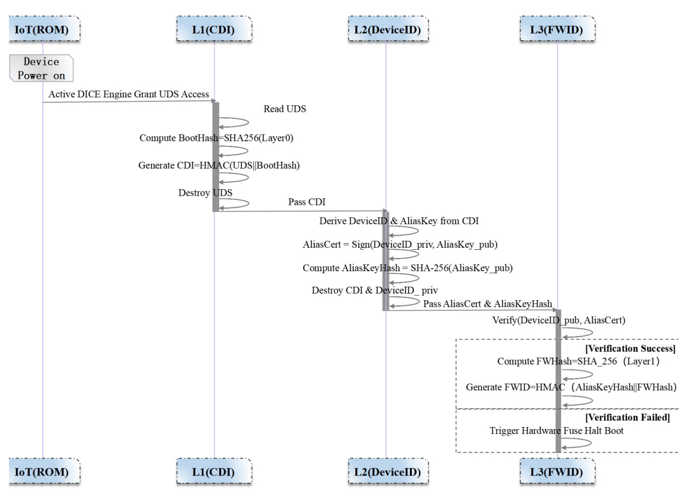


Fig. 3. L1 to L3 transfer process

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2025.3618876

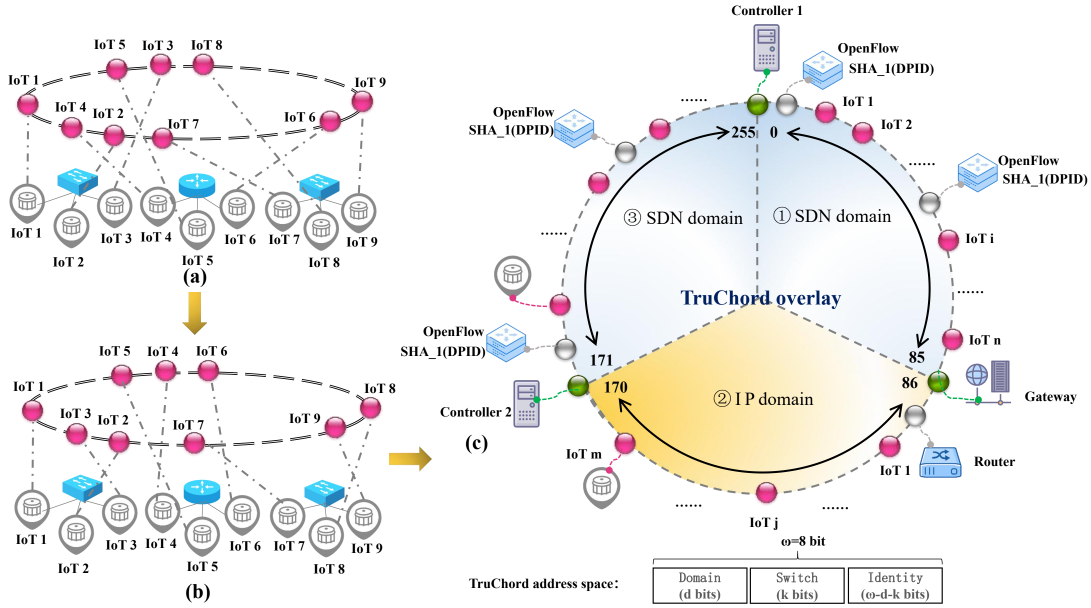


Fig. 4. TruChord identifier generation and domain-based mapping (a) direct mapping (b) domain-based mapping (c) address space

of $AliasKey_{pub}$ is computed as $AliasKeyHash = SHA-256(AliasKey_{pub})$. All CDI values and $DeviceID$ private keys are then purged from memory and registers. Only $AliasCert$ and $AliasKeyHash$ are passed to L3, with control transferred to L3, as illustrated in Fig. 3.

3) L3: (Firmware measurement, trust chain extended to the application layer): Verifies the authenticity of identities transmitted from L2, measures application firmware integrity, and completes the trust chain closure. Uses the DeviceIDpub pre-stored in the device's secure enclave to validate the signature of $AliasCert$. Upon successful verification, confirms that $AliasKey_{pub}$ has not been tampered with. Reads the application firmware from Layer1's sensors and computes its hash as $FWHash = SHA\_256(Layer1)$. Combines AliasKeyHash from L2 with FWHash to generate the Firmware Identifier: $FWID = HMAC(AliasKeyHash||FWHash)$. Completes a cryptographically secured trust chain as shown in Fig. 3. Any physical modification altering firmware configuration will result in $FWID$ validation failure, ensuring runtime integrity.

Based on the DICE trust chain transmission mechanism, the measurement reference values are stored in the $DeviceID$ certificate and $AliasCert$, and a certificate chain is formed simultaneously to verify the device identity and provide trusted proof. IoT devices report the $FWID$ to the gateway, which verifies the $AliasKeyHash$ through the pre-stored $DeviceID_{pub}$. After confirming that the firmware has not been tampered with, the sensor is allowed to access the network. When a device is attacked, it will trigger $DICE's$ integrity re-measurement and terminate program execution, thereby ensuring that IoT devices are not tampered with and preventing the spread of attacks.

B. TruChord node lookup location algorithm

TruChord is based on the Chord mechanism, which employs a consistent hash function (SHA_256) to hash the attributes of the IoT nodes. The hash values are then mapped to the overlay network, generating x-bit identifiers. These node identifiers are organized in a clockwise ring formation, spanning from the smallest to the largest values in the range of $[0, 2^{x-1}]$. In Chord, node IDs are randomly assigned through hashing, which may result in adjacent nodes on the ring being geographically far apart in the physical network. Queries thus need to cross multiple autonomous domains, increasing actual latency [30]. In TruChord, we combine Datapath Identifier (DPID) with switch hardware addresses and assign identifiers according to the network topology, making the antecedent/successor relationship of their identifiers consistent with those in the underlying physical network. This ensures that IoT nodes connected to the same switch belong to the same domain in TruChord. In Fig. 4(a), for instance, consider the mapping of IoT 1 and IoT 4 on the TruChord ring, in which they appear to be neighbors. During network communication or node positioning, directly searching for network nodes may increase node query latency. Therefore, we have modified the mapping method of identifiers on the TruChord overlay network. As shown in Fig. 4(b), the nodes connected to the same switch (such as IoT1, IoT2, and IoT3) also maintain a precursor/successor relationship on the overlay. All nodes are distributed on the TruChord according to their domains.

TruChord uses a ring approach to organize its overlay address space, which partitions this $\omega - bits$ address space into $d - bits$ for the domain number, $k - bits$ for the switch number, and $\omega - d - k$ bits for the IoT identity number. Under the hybrid SDIoT network architecture, the identifier space is evenly distributed across the TruChord ring based on the number of sub-domains, as can be seen in Figure 3(c). Within this configuration, the TruChord space is equally divided into three parts, including two SDN domains and one IP domain. Suppose that the TruChord ring can be represented by an 8-bit address space ($\omega = 8$), and the overall TruChord overlay can be subdivided into M sub-domains. Then, the domains across the ring can be evenly allocated $d = 2^{\omega}/M$

bits each. Consequently, the address space of SDN domain and IP domain in TruChord overlay has the following value: $D_{SDN1} = \{0, 85\}, D_{IP} = \{86, 170\}, D_{SDN2} = \{171, 255\}$.

The set of nodes in the SDN domain, denoted as V_{SDN}, can be represented as $V_{SDN_i} = \{v_{c_i}, v_{o_i1}, ..., v_{o_im}, v_{n_i1}, v_{n_i2}, ..., v_{n_in}\}$. This set includes a controller node v_{c_i}, m OpenFlow switches and middleware nodes $v_{o_ij}, 1 \leq j \leq m, m \in \mathbb{Z}^+$, and n IoT nodes $v_{n_ij}, 1 \leq j \leq n, n \in \mathbb{Z}^+$. Based upon the number of OpenFlow switches, the address space of the SDN domain is further divided into 2^k evenly-spaced sub-domains within the TruChord framework. The respective controller generates the initial k bits. In the OpenFlow subdomain, the controller generates these k bits as a node identifier corresponding to the switch number within the SDN domain, while the rest of $\omega - d - kbits$ bits are uniquely determined by hashing the node identity information.The DPID uniquely identifies the OpenFlow switch [31]. A new OpenFlow switch must send its DPID and port statistics to the controller when it connects to the controller. DPID is a 64-bit binary, with the higher 16 bits determined by the manufacturer (In Mininet [32], it is assigned by its controller), which uniquely identifies an OpenFlow switch. The lower 48 bits of DPID are derived from the MAC address of the switch. IoT nodes connect to the OpenFlow switch to retrieve their DPID respective values, which are then used to calculate the initial bits through a hash function. The initial bit sequence is then mapped onto the TruChord ring associated with the switch. This procedure ensures that the identifiers assigned to the IoT nodes, connected to a common switch, have the same beginning value and are in the same domain within the TruChord overlay network, as illustrated in Fig. 4(c). The hash value of the node's integrity-verified identity information, denoted as FWID, is combined with the initial k bits to generate a unique IOT node identifier, referred to as $CID_{SDN_{IoT_i}}$. The calculation of $CID_{SDN_{IoT_i}}$ is formulated in Eq. (1) and the algorithm flow is shown in Fig. 5..

$$CID_{\mathrm{SDN}_{IoT_i}} = \{SHA\_256(DPID)|SHA\_256(FWID_{IoT_i})\} \quad (1)$$

Let us use V_{IP} to denote the set of nodes in IP domain, defined as $V_{IP} = \{v_{s1}, ..., v_{sp}, v_{n_i1}, ..., v_{n_iq}\}$. Here v_{si} represents a network device such as a router or switch in the $i-th$ sub-domain, where $1 \leq i \leq p$, with $p \in \mathbb{Z}^+$. Furthermore, q denotes v_{n_ij} IoT nodes, which are connected to the same ith router/switch, where $1 \leq j \leq q$, with $q \in \mathbb{Z}^+$.

When the node identifier is obtained by applying hashing over the IP address, it leads to yet another challenge, which is that multiple nodes with adjacent node identifiers on the TruChord ring are far apart in the physical network, resulting in a heightened lookup latency, as depicted in Fig. 4(a). The entire Internet is still in a state of global transition from IPv4 to IPv6. The 32-bit address in IPv4 comprises a net service identity (NSID) and a host identity (HID), where NSID provides the geographic distribution information in the physical network. The connection between the network number (NSID) and IoT node identity information (FWID) can be used for the mapping of IPv4 nodes in the IP domain to the TruChord ring, as formulated in Eq. (2)

$$CID_{IPv4_{IoT_i}} = \{SHA\_256(NSID)|SHA\_256(FWID_{IoT_i})\} \quad (2)$$

The IPv6 network protocol employs a 128-bit hierarchical address space, including a higher-order 64-bit network prefix and a lower-order 64-bit host identifier. The high 64-bit network prefix is further divided into a 13-bit Top-Level Aggregation Identifier (TLA), a 24-bit Next-Level Aggregation Identifier (NLA) and a 16-bit Site-Level Aggregation Identifier (SLA), respectively. The TLA, which is usually associated with a large ISP or address manager, manages a larger block of addresses and assigns addresses to the NLA (small and medium-sized ISPs). NLAs, in turn, assign addresses to the SLA (subnets) as well as network hosts. The process of mapping Pv6 nodes to the TruChord ring in the IP domain is formulated in Eq. (3).

$$\begin{aligned} CID_{IPv6_{IoT_i}} = & SHA\_256(TLA \;\&\; NLA \;\&\; SLA) \\ & |SHA\_256(FWID_{IoT_i}). \end{aligned} \quad (3)$$

The domain-specific address mapping of TruChord ensures minimal node localization query latency when the logical nodes on the ring are consistent with the physical topology. In Fig. 4(c), the ring has N nodes, divided into M subdomains ($M = 3$), with each domain containing $D = N/M$ nodes. For inter-domain routing, routing is performed among the M domain header nodes, resulting in a hop count of $O(logM)$. For intra-domain routing, routing is conducted among the D nodes, with a hop count of $O(logD) = O(log(N/M))$. Therefore, the total hop count is $O(logM) + O(log(N/M)) = O(logM + log(N/M))$. When $N = 10^6$, the hop count of traditional Chord is $O(log10^6) = 20$. In TruChord, when $M = 10^3$, the hop count is $O(log10^3 + log10^3) = O(20)$, which is equivalent to that of traditional Chord. However, in practice, physically adjacent nodes can further reduce intra-domain hops (e.g., down to 5 hops), making the total hop count $O(10 + 5) = O(15)$, and the average query latency can be improved by 25%.

Physically adjacent nodes may become targets of the same attacker, resulting in simultaneous failure of multiple nodes. In TruChord, the $DeviceID$ certificates of physically adjacent nodes are usually issued by the same root CA. Nodes can pre-exchange and cache the DICE certificate chains of adjacent nodes to form a local certificate chain pool. In cross-domain scenarios, there is no need to request the certificate chain from a remote CA or cross-domain nodes when verifying physically adjacent nodes. For example, when node A (physically adjacent to node B) verifies the identity of B, it can directly retrieve the public key of B's $DeviceID$ certificate from the local cache to verify the signature of AliasCert. The differences between Chord and TruChord are shown in Table 1.

IV. TruChord Security Communication Protocol

Nodes achieve trusted access through the identity authentication of IoT terminals described in Section III-A. IoT nodes are mapped to the TruChord overlay network in accordance

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2025.3618876

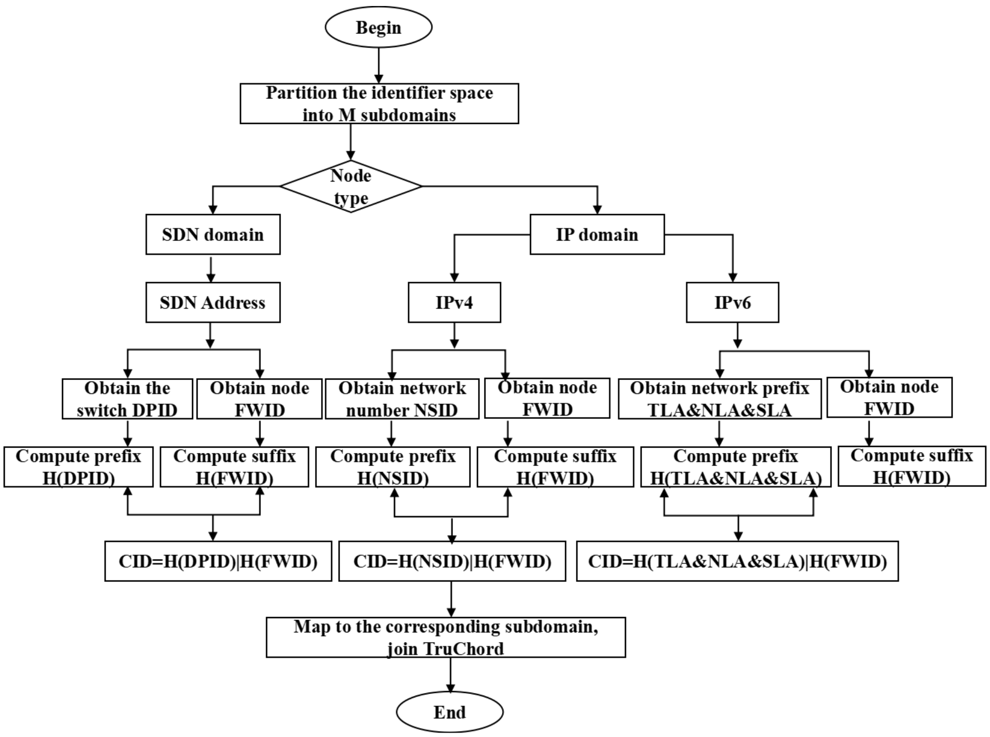


Fig. 5. The algorithm flow

TABLE I
THE DIFFERENCES BETWEEN CHORD AND TRUCHORD

ine	Chord	TruChord
ine Average Query Latency	O(logN)	O(logM+logN/M)
ine Load balancing degree	Obvious local hotspots	More balanced load within the region
ine Trust chain verification	Relies on cross-domain certificate chain queries (ms)	Local certificate pool verification (μs)
ine		

with Section III-B. In the TruChord secure communication framework, a requester needs to confirm whether the target node is trusted and thus needs to query the node for its trust value. In the hybrid SDIoT architecture, the key to secure communication between IoT nodes lies in solving the problem of determining node trust values during cross-domain access, thereby ensuring the trustworthiness of data interaction in heterogeneous network environments.

A. Node Communication Model

Fig. 6(a) provides an illustration of cross-domain access for a hybrid SDIoT Network node that adheres to TruChord. Suppose that an IoT $Requester$ node ($NodeA(Vn_{12n})$ in Figure 4), in the SDN domain, is searching for the $Target$ node ($NodeB(Vn_{2n})$ in Figure 4), residing in the IP domain. The $Requester$ ($NodeA(Vn_{12n})$ first sends a request to its OpenFlow switch, $V_{o_{12}}$, and uses the $lookup$ function to find out the communication addresses of OpenFlow switch, $V_{o_{12}}$, and controller, V_{c_1}, in its domain. The controller in the SDN domain manages all OpenFlow switches and determines the trustworthiness of network devices. Consequently, V_{c_1} communicates with the gateway, denoted as V_{c_2}, in the IP domain to locate the $Target$ ($NodeB(Vn_{2n})$) after finding the corresponding router/switch, as determined by the Border Gateway Protocol (BGP). The $Requester$ gradually approaches the $Target$ by routing messages. By utilizing the unique node identifier, CID, the network queries the node trust value from the node trust list, $TrustList$. This query determines the node's trustworthiness. After the $Requester$ communicates with the $Target$ ($NodeB$), it provides a feedback report to the Network node along the path.

B. Security Communication Protocol

In the TruChord secure communication framework, the requesting node is required to verify the trustworthiness of the $Target$ node and query the Network node for its trust value. The computational procedure for determining node trust values is not outlined here; instead, the citation [33] can be referred to select an appropriate method for trust value calculation.

1) Query: In the first step, determine whether the target $NodeB$ belongs to the SDN domain based on its CID value. The trust value (tv_{dst}) of the target, $NodeB$, is established based on the $TrustList$ maintained by the SDN control node. If $NodeB$ is part of the IP domain, it works as a decentralized network node and manages the storage of trust values in accordance with the P2P networking principles.

Let's assume that the $Target$ carries out a search operation using $lookup(CID_{nk_i})$ and ends up locating n network nodes,

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2025.3618876

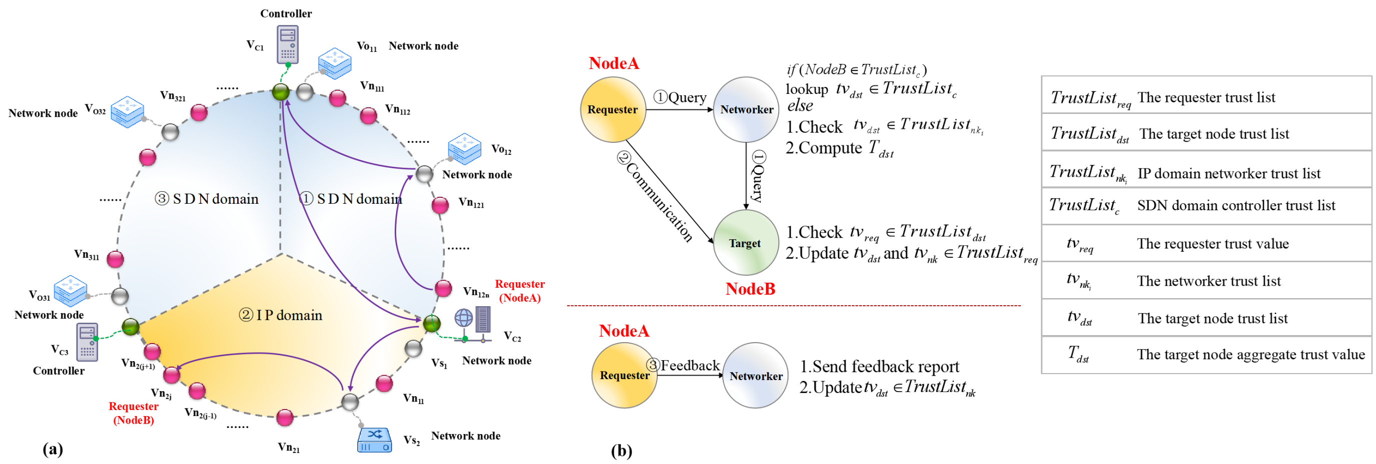


Fig. 6. Cross-domain node communication (a) Node request (b) Security communication protocol

aggregating the trust values of these n network nodes to calculate the aggregated trust value, T_{dst}, of the $Target$, as formulated in Eq. (4)

$$T_{dst} = \frac{\sum_{0}^{n} (tv_{nk_i\_dst} \cdot tv_{nk_i})}{n}. \tag{4}$$

where, $tv_{nk_i\_dst}$ represents the under-evaluation trust value of the $Target$ provided by the network nodes. tv_{nk_i} is the local trust value of the Network node, which is stored in the trust list ($TrustList_{req}$) of the $Requester$. The variable n denotes the number of network nodes that store the trust value of the $Target$.

The T_{dst} aggregated trust value of the $Target$ is then compared with the trust threshold, θ. If $T_{dst} > \theta$, the $Target$ is considered trustworthy, after which the next steps of this secure communication protocol are executed. If a Network node comes under attack, its trustworthiness can be maliciously compromised, potentially providing fake data on behalf of an otherwise trusted node.

2) Communication: Once the $Requester$ is assured that the $Target$ is trusted, it sends a communication request along its own node identifier, CID_{req}, to the $Target$. The $Target$, in turn, examines the trust value, tv_{req}, of the $Requester$, as stored in the $Target$ node's trust list, referred to as $TrustList_{dst}$. Comparing it with the trust threshold θ, if $tv_{req} > \theta$, the requester node is considered trustworthy, ensuring secure communication. After the communication is over, the $Requester$ carries out an evaluation of the communication process and calculates a trust value, tv_{dst}, for the target node providing communication service. At the same time, based on the course of the communication, the $Requester$ evaluates the Network nodes and determines a trust value, $tv_{req_{dst}}$. The computational procedure to determine the trust value of the ith Network node, tv_{nk_i}, is formulated in Eq. (5)

$$tv_{nk_i} = 1 - (|tv_{dst_i} - tv_{nk_i\_dst}|). \tag{5}$$

where tv_{dst_i} represents the trust value for the $Target$ providing communication services. Simultaneously, the $Requester$ calculates the trust value for the $Target$. In addition to that, $tv_{nk_I\_dst}$ is the trust value that the Network node assigns to the $Target$ node.

The "on-off attack" strategy exploits the tendency of trust systems to forget, wherein a malicious node raises the trust value for a brief period of time to deceive the device, and then launches an attack. To effectively counter such On-off attacks, it is necessary not only to consider the latest trust value but also to pay more attention to the historical trust values of the node, assigning them a greater weight. A time decay factor $\gamma(0 < \gamma < 1)$ is introduced, and the calculations for the network node tv_{nk_i} and the target node tv_{dst} are shown in Eqs. (6) and (7) as follows:

$$tv_{nk_t} = \frac{\sum_{t=1}^{T} (tv_{nk_{i,t}} \cdot \gamma^{T-t})}{\sum_{t=1}^{T} \gamma^{T-t}} \tag{6}$$

$$tv_{dst_t} = \frac{\sum_{t=1}^{T} (tv_{dst_{i,t}} \cdot \gamma^{T-t})}{\sum_{t=1}^{T} \gamma^{T-t}} \tag{7}$$

Here, T represents the time window (for the most recent 100 interactions, $T = 100$), denoting the time steps within the window. $tv_{nk_{i,t}}$ is the trust value of network node i in the t-th interaction. γ^{T-t} is the weight of the t-th behavior: recent behaviors ($t \approx T$) have higher weights, while the weights of earlier behaviors ($t < T$) decay significantly over time. Given $\gamma = 0.9$, the weight of recent malicious behaviors is 1, whereas the weight of early normal behaviors is only $\gamma^9 9 \approx 0.006$. When the proportion of recent malicious behaviors reaches 30%, the probability that the trust value exceeds the threshold is constrained to $P \leq 0.3 \times (1 - \gamma^{30}) \approx 5\%$. By weighting the node's historical behaviors according to the time window, even if a malicious node intermittently disguises itself as normal, the high weight of its recent malicious behaviors will continuously lower its trust value. It is difficult to "whitewash" itself through short-term normal behaviors, thus being stably identified.

Network nodes intentionally send negative feedback to benign devices or increase the trust value of damaged devices, forming bad-mouth attacks and vote stuffing attacks [34], [35]. A bad-mouth attack refers to a malicious node deliberately submitting a falsely low trust value to a benign node; for example, maliciously rating the trust value of a normal node from 0.8 to 0.2. Suppose there are m feedback values from network nodes to the target node, forming a set $\{\mathrm{tv}_{nk_dst,1}, \mathrm{tv}_{nk_dst,2}, ..., tv_{nk_dst,m}\}$. Its mean μ and standard deviation σ are as shown in Eqs. (8) and (9):

$$\mu = \frac{1}{m}\sum_{i=1}^{m} \mathrm{tv}_{nk_dst,i} \tag{8}$$

$$\sigma = \sqrt{\frac{1}{m-1}\sum_{i=1}^{m} \left(\mathrm{tv}_{nk_dst,i} - \mu\right)^2} \tag{9}$$

Where $\mathrm{tv}_{nk_dst,i}$ represents the feedback trust value of the i-th node on the target node. If $|\mathrm{tv}_{nk_dst,i} - \mu| > 3\sigma$, the feedback is judged as abnormal and excluded from the trust value aggregation calculation. For example, when $\mu = 0.8$ and $\sigma \approx 0.05$, the normal feedback of benign nodes is concentrated in $[0.7, 0.9]$. A malicious low feedback (such as 0.2) satisfies $|0.2 - 0.8| = 0.6 > 3 * 0.05 = 0.15$, and thus will be filtered out as it is far beyond the 3σ range. Conversely, vote stuffing attacks refer to multiple malicious nodes colluding to submit falsely high trust values, such as raising the trust value of a malicious node from 0.3 to 0.9. For groups with highly consistent feedback values, a "collusion suspicion" detection is triggered. The feedback similarity Sim of trust values calculated within the group is shown in Eq. (10).

$$Sim = 1 - \frac{1}{m-1}\sum_{i=1}^{m} |\mathrm{tv}_{nk_dst,i} - \mu| \tag{10}$$

When $Sim < \tau(\tau$ is a preset similarity threshold, such as $\tau = 0.8$), the group is determined to be suspected of collusion. The feedback trust values of all nodes in the group are uniformly attenuated, and the updated node trust value $tv'_{nk_dst,i}$ is calculated as shown in Eq. (11).

$$tv'_{nk_dst,i} = tv_{nk_dst,i} \times \delta \tag{11}$$

Where δ is the attenuation factor, which is dynamically adjusted according to the degree of collusion suspicion (the smaller the Sim, the smaller the value of δ). Suppose there are k colluding nodes among m nodes that submit trust values greater than 0.9, while benign nodes submit the real value of 0.3. After attenuation, the aggregated trust value of the target node is $\mathrm{tv}_{nk_dst} \leq (k \times 0.9 \times \delta + (m-k) \times 0.3)/m$. When $k/m < 30\%$ and $\delta = 0.5$, $\mathrm{tv}_{nk_dst} < 0.5$, which is lower than the trust threshold of 0.6, proving that the probability of a successful attack is constrained to $P \leq k/(2m)$. That is, the probability that malicious nodes collude to raise the target node's trust value above the trust threshold will not exceed half of the proportion of colluding nodes (k/m).

3) Feedback: It is the responsibility of the $Requester$ to generate a feedback report for the $Target$, which contains the trust value, tv_{req_dst}. The $Requester$ transmits its own CID_{req} value, the feedback report and the $Target$ identifier, CID_{dst}. The updated Target trust value is calculated as in Eq. (12).

$$tv_{dst_s} = \frac{tv_{dst_{s-1}} \cdot (s-1) + tv_{req\_dst}}{s}. \tag{12}$$

where s represents the number of updates to the target node's trust value. By calculating the historical trust value of the target node, the influence of the previous trust values is retained, and then it is added to $tv_{req_{dst}}$ in the feedback report to update the network node's trust value.

V. Prototype System Simulation

PeerSim, as cited in [36], is a generic simulation platform designed for the simulation of structured P2P overlay networks. PeerSim automatically generates a network topology. Likewise, in this simulated network, three types of nodes are established, which include the SDN nodes, IPv4 nodes and IPv6 nodes [37]. The detailed configuration of the simulation environment is shown in Table 2.

TABLE II
Key Simulation Parameters

ine Parameter	value
ine Simulator	PeerSim 1.0.5
ine total Nodes N	4096/8192/16384/32768
ine SDN Domains	3(1 controller + 20 OVS per domain)
ine IP Domains	2(10 routers per domain)
ine Intra-domain Latency	10 ms
ine Inter-domain Latency	100 ms
ine Malicious Node Ratio	10%–50%
ine Trust threshold (θ)	0.6
ine	

A. Analysis of the Network Stability

The evolution of the frequency of accesses to the $Target$ is illustrated in Fig. 7(a). As the number of accesses rises up to 500, the proportion of nodes involved in the process gradually increases. This shift in access frequency is represented graphically by a dot on the figure, denoting the average key value. As the network grows in size, the average key witnesses a consistent variation. Moreover, within each sub-domain, the average number of lookups conforms to the time complexity of $O(\log N/M)$.

Figure 7(b) illustrates the end-to-end latency for the four architectures. We gradually increased the number of domains from 1 to 4096 while keeping the maximum hop tolerance value within 15 hops. In the hybrid SDIoT architecture, SDN, IPv4, IPv6 and TruChord modes are independently used for node location and lookup. As per the literature [15], the average hop count increases with the logarithmic number of nodes in the Chord environment, approximately averaging at $\frac{1}{2}\log_2 N$ hops. The average number of hops in the realistic scenarios typically remains between 6 to 7 hops, depending on the number of nodes. Moreover, when each hop involves cross-domain transition, the average lookup latency should remain

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2025.3618876

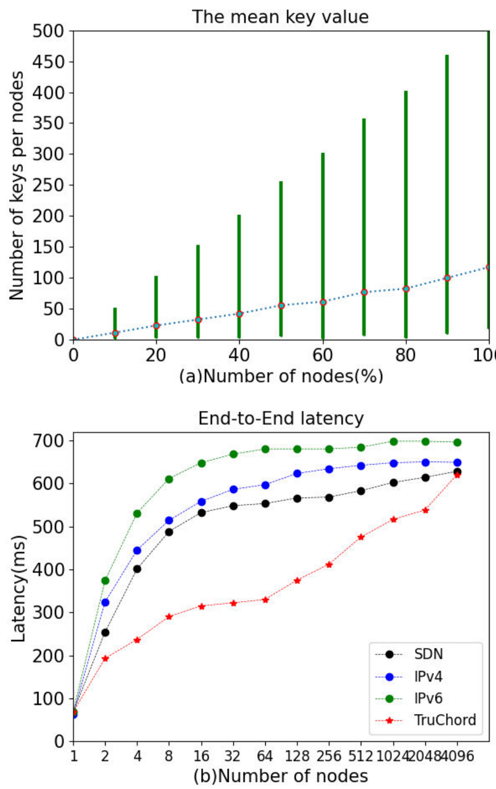


Fig. 7. The node access key value and end-to-end latency in TruChord (a)The mean key value (b) End-to-End latency

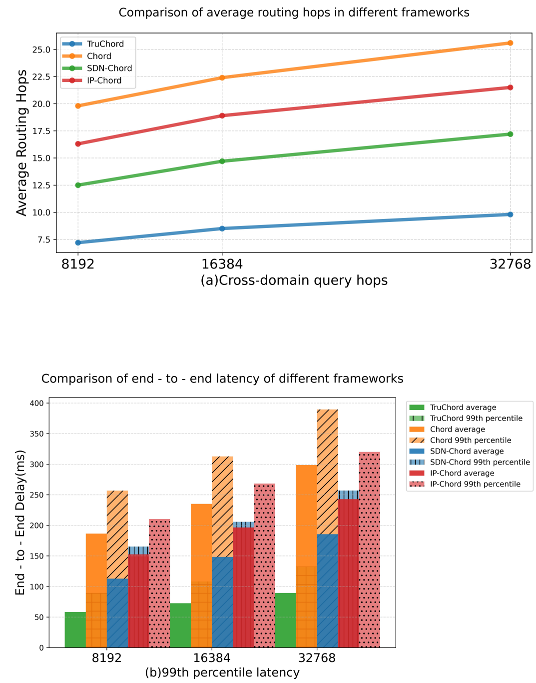


Fig. 8. End-to-end delay and routing hops under different node scales (a) Cross-domain query hops (b) 99th percentile latency

within the range of 600 to 700 milliseconds, as depicted in Figure 6(b). Since the TruChord loop can modify the node identity generation process, the nodes with the same prefix (DPID, IPv4, and IPv6) can be co-located adjacent to each other. This, in turn, results in a high hit ratio during the lookup process, which can effectively reduce the lookup latency.

To verify the scalability and superiority of TruChord in large-scale IoT scenarios, we expanded the node scale to three scenarios: 8192, 16384, and 32768. Figure 8(a) shows the performance of end-to-end delay and routing hops under the four frameworks in the three node scales. Thanks to the domain identifier mapping, TruChord realizes the optimization of routing hops in ultra-large-scale IoT scenarios. The average routing hops of TruChord increase most gently with the growth of node scale, being approximately 7.5 hops when there are 8192 nodes and about 10 hops when there are 32768 nodes. The number of hops in Chord increases linearly with the node scale. The hop growth of SDN-Chord and IP-Chord is between the two, but still much higher than that of TruChord. TruChord keeps physically adjacent nodes adjacent on the logical ring and optimizes routing efficiency through domain mapping, thereby reducing the number of cross-domain jumps. To evaluate the stability of the network and its performance under extreme conditions, we compared the robustness of different architectures through the performance of the 99th percentile latency, as shown in Figure 8(b). The gap between the 99th percentile latency and the average latency of TruChord is the smallest. When there are 32,768 nodes, the 99th percentile latency is approximately 130 ms, which is only 1.44 times the average value, reflecting the stability of domain-based routing. The 99th percentile latency of Chord is more than 1.3 times its average latency, indicating that this architecture has large latency fluctuations in dynamic scenarios. The 99th percentile latencies of SDN-Chord and IP-Chord are also significantly higher than that of TruChord. Since TruChord only updates the information of nodes within the same domain, and the DICE trust chain strongly binds device identities, the latency fluctuations are jointly reduced.

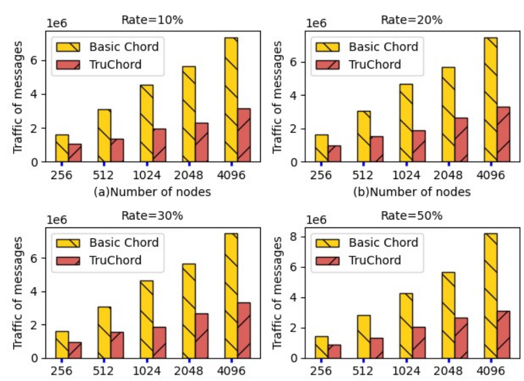


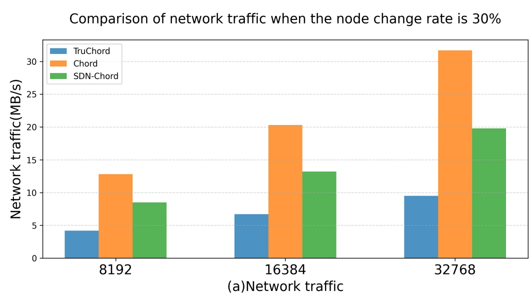
Fig. 9. Analysis of the traffic generated by messages generated at different node variation rates.

The network dynamism caused by nodes' joining, dropping or failure tests TruChord's ability to maintain consistency. This network churn requires the exchange of additional messages that increase network traffic. The nodes are assumed

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2025.3618876

to join/exit randomly, following a Gaussian distribution [38]. The rate of change reflects the proportion of IoT nodes joining or exiting the network with respect to the total number of nodes in a certain time interval. In the simulation setup, the rate of change is used to represent the nodes exiting the network, and this value varies between 10% to 50%. The simulated ‘experiment compares TruChord with the standard Chord protocol to evaluate the traffic generated by each while maintaining the overlay network at different node variation rates.

The advantage of TruChord's identifier assignment scheme is that it ensures that precursor and successor nodes within the same domain on the overlay can be physically located in the same switch/router sub-domain. Fig. 9 analyses and compares the traffic generated by messages generated at different node variation rates, for both Basic Chord and TrueChord network architectures.Moreover, the traffic generated in the TruChord framework is much lower than in the standard Chord at all node variation rates. This reduction can be attributed to the reduced number of hops required in the TruChord framework. Remarkably, the total traffic is reduced by more than 50% when the total number of nodes reaches 4096.



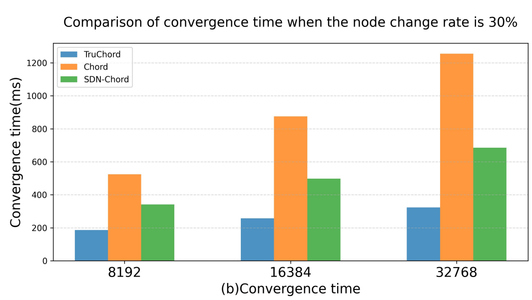


Fig. 10. Comparison of different node scales (a)Network traffic (b)Convergence time

When nodes dynamically join or exit, TruChord only updates the information of nodes within the same domain, thereby reducing global message interactions and demonstrating significant advantages in network traffic and convergence time. In Fig. 10(a), the network traffic of TruChord is only 33% to 36% of that of the standard Chord. When $N = 8192$, the traffic of TruChord is 4.2 MB/s. Since nodes only notify neighbors in the same domain when exiting, it avoids the global broadcasting of Chord, resulting in a traffic lower than the 12.8 MB/s in Chord. When the traffic of TruChord increases to 9.5 MB/s, the growth rate is 126%. The traffic growth shows a linear relationship with the node scale. The traffic of the Chord architecture increases to 31.7 MB/s, with a growth rate of 147%, which is higher than that of TruChord. When nodes change dynamically, TruChord only triggers topology updates within the same domain, resulting in the lowest traffic interaction overhead. Fig. 10(b) reflects the rapid response capability to dynamic node changes through convergence time. When $N = 8192$, due to the fast update of flow tables within the SDN domain and the synchronization of only adjacent nodes within the IP domain without the need for global routing table reconstruction, the convergence time of TruChord is 186 ms, while that of Chord is 524 ms. When $N = 32768$, the convergence time of TruChord is 324 ms, with a growth rate of 74%, while that of Chord is 1256 ms, with a growth rate of 140%. Therefore, the convergence time of TruChord is significantly lower than that of the comparison frameworks. Its domain-based architecture has a rapid response capability to the dynamic joining or exiting of nodes, which adapts to the mobility requirements of the Internet of Things.

C. Analysis of the architectural security

The primary goal of the attacking nodes is to gain control of the network and manipulate benign nodes to either introduce as many malicious services as possible or disrupt the services provided by legitimate nodes. In a simulation scenario setup to analyze the architecture-level security of Truchord, a set of IoT nodes are randomly selected, either as $Requester$ or $Target$, to simulate 3000 lookups.The red line in Fig. 11(a) represents the $Requester$ node communication rate (RCR) for untrusted nodes. As the number of malicious nodes increases, there is a point where, before 40% of the malicious nodes are compromised, the probability of the requesting node engaging in communication with an untrusted node reaches zero. Conversely, the blue line indicates the correct Node recognition rate (NRR) for trusted and malicious nodes. As the number of malicious nodes reaches 40%, the accuracy of recognizing trusted vs malicious nodes gradually diminishes.The red line in Fig. 11(a) represents the $Requester$ node communication rate (RCR) for untrusted nodes. As the number of malicious nodes increases, there is a point where, before 40% of the malicious nodes are compromised, the probability of the requesting node engaging in communication with an untrusted node reaches zero. Conversely, the blue line indicates the correct Node recognition rate (NRR) for trusted and malicious nodes. As the number of malicious nodes reaches 40%, the accuracy of recognizing trusted vs malicious nodes gradually diminishes.

Fig. 11(b) shows the process of each device connecting to a different number of network nodes in a scenario where 20% of the network nodes are malicious.In case, a large number of network nodes are connected to the $Requester$ in the system, the $Requester$ can derive more accurate trust values from the Network node. Fig. 11(b) shows that as soon as the number of nodes connected to a device exceeds three, the node communication rate (RCR) nearly approaches 100%, while the

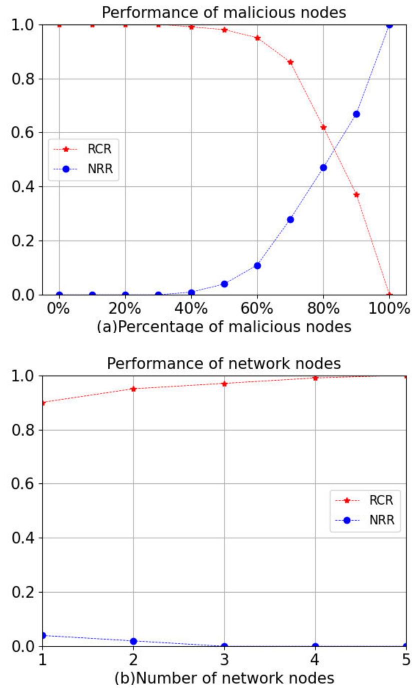


Fig. 11. Analysis of connectivity and identification of malicious nodes in hybrid SDIoT network (a) Performance of malicious nodes with different percentages. (b) Performance of different numbers of network nodes connected by each device.

node recognition (NRR) almost drops to zero. This indicates that the $Requester$ can attain highly reliable trust values by connecting to a few network nodes.

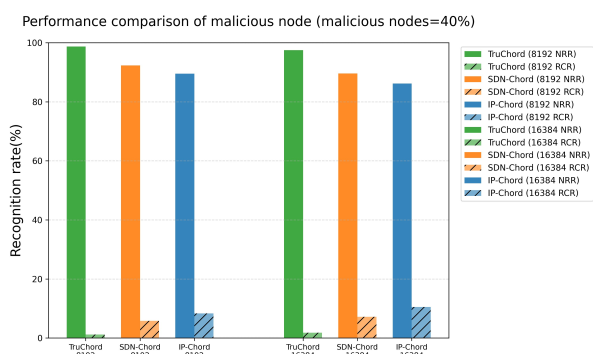


Fig. 12. rio of 40% malicious nodes with different node scales

Under different scales of IoT nodes, the NRR value of TruChord is always greater than 97.5% and the RCR value is always less than 2%, which is significantly better than the comparison frameworks. When the node scale increases from 8192 to 16384, the NRR of TruChord only decreases by 1.2% and the RCR only increases by 0.6%, which means that multi-node redundant aggregation reduces the impact of a single malicious node, as shown in Fig. 12. TruChord suppresses the false trust values of malicious nodes through weighted fusion by means of intra-domain local trust aggregation and cross-domain global verification, thus reducing the interference of trust values from malicious nodes. At the same time, the strong binding of the DICE trust chain to device identities ensures that the identities of nodes participating in the aggregation are credible, further improving the recognition accuracy.

VI. Security Proof

In the hybrid SDIoT environment based on the TruChord framework, attackers launch attacks by exploiting cross-domain identity differences, vulnerabilities in trust mechanisms, and other means. Through formal proof, TruChord can resist Sybil attacks and collusion attacks.

A. Sybil Attack Prevention

An attacker attempts to occupy key positions in the TruChord ring to manipulate routing by forging multiple identities to access the network. The probability of a successful attack is denoted by $Pr(G_1)$. That is, the attacker generates a forged DeviceID certificate $Cert'_D$ or an Alias certificate $Cert'_A$, in an attempt to make the forged node n' pass the DICE verification $Verify(Cert'_D, Cert'_A) = True$.

Suppose an attacker attempts to forge g fake nodes, they need to generate g sets of different $DeviceID$ certificates. Since the generation of $DeviceID$ depends on UDS, the trust chain described in Section III-A of this paper requires the completion of L1 operations to calculate the value of CDI = HMAC(UDS, BootHash). However, attackers cannot obtain UDS. Forging a $DeviceID$ must satisfy $Sig(\text{DeviceID}'_{pub}) = Sig(UDS \leftarrow CDI')$. As can be seen from Section III-A, each IoT node's $DeviceID$ in TruChord is strongly bound to the physical device through the DICE trust chain. The $DeviceID$ is uniquely generated by UDS, which is a hardware-unique key solidified in the BOOT ROM and only accessible by the DICE engine. According to the collision resistance of SHA-256, the hardware solidification feature of UDS results in a probability of 2^-256 for the collision between the forged $CDI\prime$ and the real CDI. Combined with the node mapping mechanism of TruChord in Section III-B, the identifier of the forged node must match the physical topology prefix (such as DPID, IP prefix), which further increases the difficulty of forgery. The success probability of a Sybil attack is $Pr(G_1) \leq 2^{-256}$, and the forgery probability is negligible in practical scenarios.

B. Analysis of the architectural security

Attackers collude to manipulate trust values. Suppose there are k malicious nodes collaborating to provide false trust values, causing the aggregated trust value of the target node to exceed the trust threshold $\theta = 0.6$, thereby resulting in the malicious node being judged as trustworthy. The success probability of the attack is denoted by $Pr(G_2)$.

Suppose the actual trust value of a node is $tv_j \in [0, 0.3]$, and malicious nodes provide a false value of $tv'_i = 1$. The aggregated trust value calculated according to Eq. (4) is $\hat{T}(t) = \frac{\sum_{i=1}^{k} tv'_i + \sum_{j=k+1}^{n} tv_j}{n} = \frac{k+(n-k)0.3}{n}$, where k is the number of malicious nodes and n is the total number of nodes. When the proportion of malicious nodes is $\alpha = k/n \leq 0.4$,

$\hat{T}(t) \leq \frac{0.4n+0.6n0.3}{n} = 0.58 < \theta = 0.6$, and thus the collusion attack fails. When $\alpha > 0.4$, $\hat{T}(t)$ may exceed θ. However, the simulation results in Fig. 11(b) in Section V-C show that at this time, the node recognition rate (NRR) decreases as α increases, and success is only possible when $n \leq 3$; when $n \geq 3$, the NRR is close to 100%, and the collusion attack is suppressed. Therefore, the necessary conditions for a successful collusion attack are $\alpha > 0.4$ and $n \leq 2$. In TruChord, $n \geq 3$ by default, and $Pr(G_1) \leq 5\%$. To sum up, since the aggregated trust value in TruChord adopts a weighted average, and the trust value of network nodes is dynamically adjusted according to historical feedback, the weight of malicious nodes is reduced, thereby preventing collusion attacks.

VII. CONCLUSION

To address issues such as difficult cross-domain identity authentication, low routing efficiency, and frequent malicious attacks in hybrid SDIoT architectures, this paper proposes the TruChord framework. It breaks through the cross-domain security bottlenecks of hybrid SDIoT through "hardware root of trust + domain-based routing + aggregated trust". The DICE trust chain realizes full-link measurement from the hardware UDS to the application FWID, reducing the probability of device identity forgery to the theoretical limit. Domain identifier mapping embeds the physical topology into the Chord logical ring, resulting in an average routing hop count of only 10 for ultra-large-scale nodes, which is 60% less than that of the standard Chord. The hierarchical trust aggregation protocol suppresses malicious collusion through dynamic weights, maintaining a 97.5% recognition accuracy even in scenarios with 40% malicious nodes. Through formal security proof, the TruChord security architecture can resist Sybil attacks and collusion attacks in the hybrid SDIoT environment. Simulation verification shows that TruChord outperforms the comparison frameworks in core indicators such as end-to-end delay, network traffic, and convergence time, providing an efficient solution for cross-domain secure communication of IoT devices. In the future, we can further explore the dynamic update of the trust chain in edge computing scenarios and the in-depth integration of 5G slicing and domain-based routing to promote the large-scale deployment of trusted IoT.

ACKNOWLEDGMENTS

This work is supported by Key Technologies R & D Program of Henan (242102210143). The authors extend their appreciation to the Deanship of Scientific Research at King Khalid University for funding this work through Large Groups Project under grant number RGP.2/637/46.

REFERENCES

[1] M. R. Nascimento, C. E. Rothenberg, M. R. Salvador, C. N. Corrêa, S. C. De Lucena, and M. F. Magalhães, "Virtual routers as a service: the routeflow approach leveraging software-defined networks," in *Proceedings of the 6th International Conference on Future Internet Technologies*, 2011, pp. 34–37.

[2] S. Tu, A. Badshah, H. Alasmary, and M. Waqas, "Eake-wc: Efficient and anonymous authenticated key exchange scheme for wearable computing," *IEEE Transactions on Mobile Computing*, pp. 1–12, 2023.

[3] A. Montazerolghaem and M. H. Yaghmaee, "Load-balanced and qos-aware software-defined internet of things," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 3323–3337, 2020.

[4] M. Dai, L. Luo, J. Ren, H. Yu, and G. Sun, "Psaccf: Prioritized online slice admission control considering fairness in 5g/b5g networks," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 6, pp. 4101–4114, 2022.

[5] G. Xu, S. Xu, X. Fan, Y. Cao, Y. Mao, Y. Xie, and X.-B. Chen, "Rat ring: Event driven publish/subscribe communication protocol for iiot by report and traceable ring signature," *IEEE Transactions on Industrial Informatics*, 2025.

[6] G. Sun, Y. Zhang, H. Yu, X. Du, and M. Guizani, "Intersection fog-based distributed routing for v2v communication in urban vehicular ad hoc networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 6, pp. 2409–2426, 2019.

[7] Q. Cai, J. Chen, D. Luo, G. Sun, H. Yu, and M. Guizani, "Deter-pay: A deterministic routing protocol in concurrent payment channel network," *IEEE Internet of Things Journal*, vol. 11, no. 19, pp. 31 206–31 220, 2024.

[8] Y. Zhang, Y. Hong, M. Guizani, S. Wu, P. Zhang, and R. Liu, "A multi-layer information dissemination model and interference optimization strategy for communication networks in disaster areas," *IEEE Transactions on Vehicular Technology*, vol. 73, no. 1, pp. 1239–1252, 2023.

[9] F. Xu, B. Duo, Y. Xie, G. Pan, Y. Yang, L. Zhang, Y. Ye, T. Bao, T. A. Gulliver, and Y. Wang, "Multi-uav assisted mixed fso/rf communication network for urgent tasks: Fairness oriented design with drl," *IEEE Transactions on Vehicular Technology*, 2024.

[10] X. Wu, B. Zou, C. Lu, L. Wang, Y. Zhang, and H. Wang, "Dynamic security computing framework with zero trust based on privacy domain prevention and control theory," *IEEE Journal on Selected Areas in Communications*, 2025.

[11] M. Haus, M. Waqas, A. Y. Ding, Y. Li, S. Tarkoma, and J. Ott, "Security and privacy in device-to-device (d2d) communication: A review," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 2, pp. 1054–1079, 2017.

[12] M. Waqas, S. Tu, Z. Halim, S. U. Rehman, G. Abbas, and Z. H. Abbas, "The role of artificial intelligence and machine learning in wireless networks security: Principle, practice and challenges," *Artificial Intelligence Review*, vol. 55, no. 7, pp. 5215–5261, 2022.

[13] P. Wang, W. Song, H. Qi, C. Zhou, F. Li, Y. Wang, P. Sun, and Q. Zhang, "Server-initiated federated unlearning to eliminate impacts of low-quality data," *IEEE Transactions on Services Computing*, vol. 17, no. 3, pp. 1196–1211, 2024.

[14] M. Haus, M. Waqas, A. Y. Ding, Y. Li, S. Tarkoma, and J. Ott, "Security and privacy in device-to-device (d2d) communication: A review," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 2, pp. 1054–1079, 2017.

[15] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," *ACM SIGCOMM computer communication review*, vol. 31, no. 4, pp. 149–160, 2001.

[16] X. Zhao, T. Wang, Y. Li, B. Zhang, K. Liu, D. Liu, C. Wang, and H. Snoussi, "Target-driven visual navigation by using causal intervention," *IEEE Transactions on Intelligent Vehicles*, vol. 9, no. 1, pp. 1294–1304, 2023.

[17] J. Xiao, Y. Ren, J. Du, Y. Zhao, S. Kumari, M. J. Alenazi, and H. Yu, "Calra: practical conditional anonymous and leakage-resilient authentication scheme for vehicular crowdsensing communication," *IEEE Transactions on Intelligent Transportation Systems*, 2024.

[18] TCG, "Trusted computing group," [EB/OL], https://www.trustedcomputinggroup.org.

[19] J. Hu, H. Jiang, S. Chen, Q. Zhang, Z. Xiao, D. Liu, J. Liu, and B. Li, "Wishield: privacy against wi-fi human tracking," *IEEE Journal on Selected Areas in Communications*, vol. 42, no. 10, pp. 2970–2984, 2024.

[20] Z. Cao, L. Huang, T. Wang, Y. Wang, J. Shi, A. Zhu, T. Shi, and H. Snoussi, "Understanding the dimensional need of noncontrastive learning," *IEEE Transactions on Cybernetics*, 2025.

[21] J. P. Xiong, Y. W. Zhang, P. L. Hong, and J. S. Li, "Chord6: Ipv6 based topology-aware chord," in *Joint International Conference on Autonomic and Autonomous Systems and International Conference on Networking and Services-(icas-isns' 05)*. IEEE, 2005, pp. 4–4.

[22] P. Karrupusamy, "Advanced metering infrastructure with secure chord lookup protocol for iot systems," *J. Electric. Eng. Autom.*, vol. 2, no. 3, pp. 112–117, 2021.

[23] W. Zhang, B. Sun, and Y. Sun, "Trustchord: chord protocol based on the trust management mechanism," in *2010 International Conference on*

Advanced Intelligence and Awareness Internet (AIAI 2010). IET, 2010, pp. 64–67.

[24] K. Kalkan and K. Rasmussen, "Trusd: Trust framework for service discovery among iot devices," *Computer Networks*, vol. 178, p. 107318, 2020.

[25] K. K. Karmakar, V. Varadharajan, S. Nepal, and U. Tupakula, "Sdn-enabled secure iot architecture," *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6549–6564, 2020.

[26] Y. Jia, F. Zhong, A. Alrawais, B. Gong, and X. Cheng, "Flowguard: An intelligent edge defense mechanism against iot ddos attacks," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9552–9562, 2020.

[27] X. Ai, A. Badshah, S. Tu, M. Waqas, and I. Ahmad, "An improved ultra-lightweight anonymous authenticated key agreement protocol for wearable devices," *IEEE Transactions on Mobile Computing*, vol. 24, no. 5, pp. 4543–4557, 2025.

[28] S. Tu, A. Badshah, H. Alasmary, and M. Waqas, "Eake-wc: Efficient and anonymous authenticated key exchange scheme for wearable computing," *IEEE Transactions on Mobile Computing*, vol. 23, no. 5, pp. 4752–4763, 2024.

[29] A. Panchenko, A. Mitseva, and S. Knabe, "Whisperchord: Scalable and secure node discovery for overlay networks," in *2021 IEEE 46th Conference on Local Computer Networks (LCN)*. IEEE, 2021, pp. 170–177.

[30] B. Gong, C. Guo, C. Guo, C. Guo, Y. Sun, M. Waqas, and S. Chen, "Slim: A secure and lightweight multi-authority attribute-based signcryption scheme for iot," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 1299–1312, 2024.

[31] N. Shukla, A. Bamal, S. Srivastava, and M. Pandey, "Software defined underlay-aware structured peer-to-peer overlay," in *Proceedings of the 19th International Conference on Distributed Computing and Networking*, 2018, pp. 1–10.

[32] R. L. S. De Oliveira, C. M. Schweitzer, A. A. Shinoda, and L. R. Prete, "Using mininet for emulation and prototyping software-defined networks," in *2014 IEEE Colombian conference on communications and computing (COLCOM)*. IEEE, 2014, pp. 1–6.

[33] Y. Zhang, B. Gong, Y. Wu, G. Zheng, and Z. Diao, "A trusted remote data trading scheme in hybrid sdn for intelligent internet of things," *Wireless Communications and Mobile Computing*, vol. 2023, no. 1, p. 7463722, 2023.

[34] H. Jiang, P. Ji, T. Zhang, H. Cao, and D. Liu, "Two-factor authentication for keyless entry system via finger-induced vibrations," *IEEE Transactions on Mobile Computing*, vol. 23, no. 10, pp. 9708–9720, 2024.

[35] C. Fu, G. Liu, K. Yuan, and J. Wu, "Nowhere to h^{2}ide: Fraud detection from multi-relation graphs via disentangled homophily and heterophily identification," *IEEE Transactions on Knowledge and Data Engineering*, 2024.

[36] "Peersim: A peer-to-peer simulator," [EB/OL], http://peersim.sourceforge.net.

[37] Y. Liu, X. Dong, E. Zio, and Y. Cui, "Event-triggered multiple leaders formation tracking for networked swarm system with resilience to noncooperative nodes," *IEEE Transactions on Cybernetics*, 2025.

[38] M. I. Ribeiro, "Gaussian probability density functions: Properties and error characterization," *Institute for Systems and Robotics, Lisboa, Portugal*, 2004.