Teamwork Makes TEE Work: Open and Resilient Remote Attestation on Decentralized Trust

Xiaolin Zhang, Kailun Qin, Shipei Qu, Tengfei Wang, Chi Zhang, Dawu Gu

***Abstract*—Remote Attestation (RA) enables the integrity and authenticity of applications in Trusted Execution Environment (TEE) to be verified. Existing TEE RA designs employ a centralized trust model where they rely on a single provisioned secret key and a centralized verifier to establish trust for remote parties. This model is however brittle and can be untrusted under advanced attacks nowadays. Besides, most designs only provide fixed functionalities once deployed, making them hard to adapt to different needs on availability, Quality of Service (QoS), etc.**

Therefore, we propose JANUS, an open and resilient TEE RA scheme. To decentralize trust, we, on one hand, introduce Physically Unclonable Function (PUF) as an intrinsic root of trust (RoT) in TEE to provide additional measurements and cryptographic enhancements. On the other hand, we use blockchain and smart contract to realize decentralized verification and result audit. Furthermore, we design an automated turnout mechanism that allows JANUS to remain resilient and offer flexible RA services under various situations. We provide a UC-based security proof and demonstrate the scalability and generality of JANUS by implementing an open-sourced prototype.

***Index Terms*—Remote attestation, Trusted execution environment, Physically unclonable function, Blockchain, Smart contract.**

I. INTRODUCTION

TRUSTED Execution Environment (TEE) has emerged as a cornerstone of protecting sensitive applications by offering isolated areas (*e.g.*, *enclaves*). Remote Attestation (RA) is an essential component to ensure the integrity and authenticity of TEEs, particularly in scenarios where trust must be established between remote parties. Figure 1 depicts a simplified RA workflow standardized in RFC9334 [1]. In this workflow, a verifier $\mathcal{V}rf$ checks whether the application (the attester $\mathcal{A}tt$) in a remote TEE device is compromised. $\mathcal{V}rf$ verifies $\mathcal{A}tt$'s measurement signed by a private key provisioned by the TEE manufacturer, then returns the result to a relying party ($\mathcal{RP}$). $\mathcal{RP}$ can decide whether or not to trust $\mathcal{A}tt$ according to the result. Through RA, various stakeholders, such as cloud service providers, data owners, and developers, can use TEEs assuredly in hostile environments.

Since this workflow assumes a rather simple and intuitive trust model, it has been extended to the passport model and background-check model in [1]. Almost every commercial TEE RA designs such as Intel [2], Azure [3] and AWS [4] adopt these models, as they are easy to use and deploy. However, we notice that the benefits of this model yields its two major limitations:

Xiaolin Zhang, Kailun Qin, Shipei Qu, Tengfei Wang, Chi Zhang and Dawu Gu are with School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University (SJTU), Shanghai 200240, China (email: xiaolinzhang@sjtu.edu.cn; kailun.qin@sjtu.edu.cn; shipeiqu@sjtu.edu.cn; tengfei2019@sjtu.edu.cn; zcsjtu@sjtu.edu.cn; dwgu@sjtu.edu.cn).

Corresponding Author: Chi Zhang, Dawu Gu

Manuscript received XXX, XXX; revised XXX, XXX.

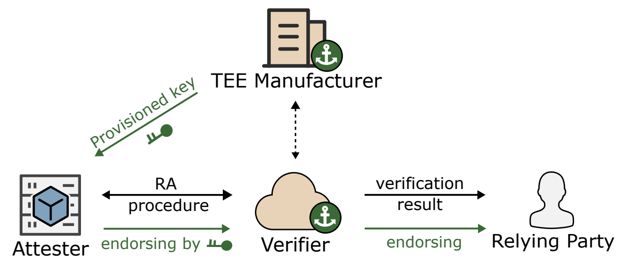


Fig. 1: A typical TEE RA workflow where ⚓ indicates the centralized trusted party, and the green lines indicate the establishment of chain of trust.

Over-centralization leads to untrusted attestation. Current TEE RA schemes are designed upon *centralized trust*, which has been a high-profiled topic that many industrial efforts [5]–[8] are devoted to. This limitation can be boiled down into two following aspects.

- Current TEEs only have a **single and centralized** root of trust (RoT) for RA, *i.e.*, the hardware key provisioned by the manufacturer in Figure 1. It is currently vulnerable to different attacks, which turns out to be loss of trust. Side-channel attacks [9]–[13], fault injection attacks [14]–[17] and other real-world exploits [18], [19] on TEEs have proven that the single centralized RoT is hard to offer trust against more advanced attackers. The RA schemes built upon it would encounter the single point of **untrust** problem and sabotage users' trust in TEEs.
- The verification procedure in RA is only conducted by **a single verifier**. So the trust here is also centralized since the verification service is not executable for others in practice [2]–[4], [20]. A verifier could claim a compromised application to be trusted, or vice versa. Relying parties cannot verify or audit the genuineness of the verification results. It is even worse that the verification services are sometimes deployed by the TEE manufacturer [21]. This trust monopoly would blind the users and nullify the results [22].

Therefore, this over-centralization problem prevents RA from achieving its very purpose, *i.e.*, establishing chain of trust between remote entities. One part being compromised in this chain lead to the failure of the entire process.

Lack of resilience hinders real-world RA deployment. Resilience entails the ability to remain operational against adversity [23], [24], [25], which has been widely pursued in various industrial systems such as cloud services [26]–[29], supply chain [30], PNT (Positioning, Navigation and Timing) and [31]. Most TEE RA schemes are designed to follow a fixed procedure [1] for standard applications like cloud computing [3], [4]. However, unexpected incidents such as network outages, device malfunctions and computing exhaustion can cause failures of cloud infrastructures. Due to the lack of **native resilience**, *i.e.*, the ability to provide flexible functionalities through the design itself, these system failures eventually lead to the failure of deployed RA services.

Moreover, with TEE having being used in more emerging scenarios, RA schemes have stronger incentives to achieve this native resilience. Otherwise, how they offer the services will depend entirely on the underlying (cloud) infrastructures, precluding their use in many other applications.

For the first limitation, MATEE [32] introduces TPM as a backup RoT for TEE, but secret keys isolated by TPM can also be leaked by advanced attacks [33], [34]. OPERA [22] designs customized certificates to offload partial verification functionality, allowing others to deploy their own services. However, it only migrates the trust from one trusted centralized party to yet another. Though SCRAPS [35] and others [36], [37] use blockchain to realize decentralization only in execution while the results cannot be re-verified or audited, let alone the scalability issues. For the second limitation, it appears that current designs [38], [39] do not focus on providing built-in flexibility of RA services under common system failures such as network outages, computing exhaustion, etc., which should be the basic requirements for resilience. Therefore, we ask the question: *How to design a TEE RA scheme with more trusted RoT and verification process, while also providing native resilience for flexible attestation?*

Sketch of our approach. In this paper, we propose JANUS, an open and resilient TEE RA design, to answer this question. JANUS introduces the physically unclonable function (PUF) [40] as the second RoT to decentralize the trust on the TEE manufacturer, offering intrinsic RoT constructions in RA. JANUS realizes efficient decentralized verification by customized smart contracts, offering a more trusted RA verification process. Besides, we design an automated "turnout" mechanism in JANUS to allow participants to freely choose the way to conduct RA by their situations. The attestation turnout diverts the workflow of JANUS just like the railroad turnout guides the train onto a different track.

Contributions. Specifically, this work contributes by:

- To our best knowledge, we are the first to introduce PUF into TEE RA. We design PUF-based mutual attestation protocols that support lightweight end-to-end (E2E) RA functions built around two RoTs. We also give a UC-based [41] security proof for these protocols.
- We also design smart contract-based RA functions to offer public involved verification procedure, which contains an attestation smart contract for transparent decentralized RA verification, and the (first) audit contract to realize the auditability of the results.
- We propose a turnout mechanism to combine the PUF-based protocols and smart contract attestation. It can realize automated attestation switch between these two functionalities, thereby offering native resilience against unexpected system failures in practice.
- We give a Proof-of-Concept (PoC) implementation of JANUS[1]. The comparison results between JANUS and other RA schemes shows that JANUS only has small storage overhead and achieves great scalability and reasonable computing performance.

Scope. We focus on the systematic design of TEE RA schemes in this study. The issues of over-centralization and lack of resilience that exist in most TEE RA designs have grown as urgent industrial concerns nowadays, while not widely explored in academic research. We intend to fill this gap by providing JANUS with a PoC implementation. However, hardware integration of PUF into existing TEE devices is out of scope of this paper, because it requires to redesign the whole circuit layouts and manufacturing procedures of both PUF and TEE with relevant manufacturers' support.

Paper outline. Section II provides background on RA, PUF and smart contract. Section III deepens the motivation of this work. Section IV and V present the threat model, workflow and detailed designs. Section VI offers formal and informal security analysis. Section VII gives a complete evaluation on performance, resilience and security. Section VIII discusses the limitations. Section IX presents relevant studies and Section X concludes our work.

II. BACKGROUND

Remote Attestation (RA) aims to establish trust relations between remote entities. It relies on trusted measurement and cryptographic primitives to extend the chain of trust. RA can be implemented as a challenge-response protocol with shared key agreement. Mutual RA [42], [43] is often needed in practice for distributed applications on separate devices. Figure 1 has given an overview of RA workflow and here we further explain the standard terminology [1] used in this paper:

- *TEE Manufacturer* ($\mathcal{M}fr$) controls the RoT provision of TEE, *e.g.*, root key injection. Intel, Apple [44], Samsung [45] and so on can all be seen as $\mathcal{M}fr$ here.
- *Relying Party* ($\mathcal{RP}$) uses RA to make appraisals for $\mathcal{A}tt$, *i.e.*, determines the trustworthiness of the applications for subsequent actions, such as authorization.
- *Attester* ($\mathcal{A}tt$) refers to a deployed application in TEE. It can prove its status by invoking TEE measuring functions. Multiple $\mathcal{A}tt$ can exist as different system processes on the same TEE device.
- *Verifier* ($\mathcal{V}rf$) acts as a proxy entity for $\mathcal{RP}$ to conduct actual RA operations with $\mathcal{A}tt$. The verification services provided by Google [20], Azure, Amazon, etc can be regarded as the $\mathcal{V}rf$ here. In mutual RA, $\mathcal{V}rf$ can also be the applications to be attested.
- *RA session* means a short period of time starting when $\mathcal{V}rf$ sends an RA request to $\mathcal{A}tt$, and ending when $\mathcal{V}rf$ outputs the results after the verification.

[1]It will be open-sourced at https://sites.google.com/view/janus-ra

Physically Unclonable Functions (PUF), developed from the physical one-way function [40], is a hardware primitive that randomly maps a given challenge to a response. They utilizes minor environmental variations in manufacturing and incorporate physical randomness into its structures. Therefore, each PUF instance has a unique structure and a distinct set of challenge-response pairs (CRPs), rendering duplication nearly impossible. IC PUFs are usually integrated inside the device chips in actual products [46], [47].

Adversaries cannot directly invoke the embedded PUF and obtain CRPs because PUF's I/O bus is coalesced with the chip circuitry and thus not accessible for users. If it uses invasive physical attacks like milling and delayering the chip to recover the stored secrets inside, this will unavoidably alter the original IC status and turn PUF unavailable. Therefore, PUF offers a new way of harnessing the intrinsic physical features to build robust security defenses. We can use PUF as an independent trusted module in TEEs to diversify sources of trust for RA.

Blockchain and Smart Contracts can use transactions (TXs) to record information permanently. Smart contracts can publicly execute predefined tasks over blockchain nodes. This transparency and decentralization are suitable for verifying the measurements in RA based on consensus trust for all parties.

III. Motivation

A. Problems with Real-world TEE RA

The model of single provisioned RoT and single-point-of-verification restrict $\mathcal{RP}$ to the blind trust of $\mathcal{M}fr$ and one verifier. This closed and centralized chain of trust is fragile since only one part of the chain being compromised will directly cause the failure of entire attestations. Unfortunately, nearly all existing TEE RA schemes adopt this design.

Real-world examples: We first analyze several real-world TEE RA design examples.

SGX IAS-based RA [2]: An attesting SGX application gets its measurement using `MRENCLAVE`. It wraps the measurement, a signature signed by an Intel-provisioned key and other data to form an encrypted report. This report can only be decrypted and verified by Intel Attestation Service (IAS).

SGX DCAP-based RA [8]: To avoid this centralized procedure, Intel proposed Data Center Attestation Primitives (DCAP) to support third-party attestation. It publishes the Provisioning Certification Key (PCK) certificates so that others can verify the signatures. This approach offloads partial RA functions. However, users still rely on Intel's single RoT, and only one (non-Intel) verifier can execute the verification.

Microsoft Azure RA [3]: In Azure RA, the measurements of TEE platforms can only be verified by Azure Attestation Service. A token will be issued to $\mathcal{RP}$ if the verification pass. $\mathcal{RP}$ has to trust Azure ($\mathcal{V}rf$) to return the correct results. Other mainstream cloud vendors (*e.g.*, Google, AWS) also use the similar design for their RA services. Compared to Intel SGX RA, the main difference is the replacement of IAS with a vendor-specific service, which only migrates the trust from one trusted centralized party to yet another.

In these examples, participants can only rely on $\mathcal{M}fr$ to establish the RoT (the provisioned keys). Specifically, as long as $\mathcal{V}rf$ accepts the endorsements of $\mathcal{M}fr$ to $\mathcal{A}tt$, the chain of trust in RA is complete. However, this requires another necessary assumption: $\mathcal{V}rf$ should be fully trusted to return the right results. $\mathcal{RP}$ has no other options but to believe that $\mathcal{V}rf$ has faithfully executed the verification. This centralized model attributes trust to the security of provisioned keys and the integrity of $\mathcal{V}rf$, both of which could be undermined by powerful attackers [10]–[13].

Moreover, these examples show that the how $\mathcal{RP}$ can use these RA services is entirely up to how the cloud services are deployed. These RA designs serves more like add-ons to existing cloud services for TEE devices, lack of native resilience to provide flexible functionalities themselves.

Problems Identified: From the above analysis of these real-world examples, we conclude the following problems:

- **P1** Current TEE RA schemes all rely on $\mathcal{M}fr$ to establish a single centralized and provisioned RoT, which can be compromised by advanced attacks.
- **P2** Only one $\mathcal{V}rf$ can execute the verification and $\mathcal{RP}$ have to believe whatever $\mathcal{V}rf$ outputs. Other participants cannot re-verify the results.
- **P3** The functionalities of current schemes are entirely up to the deployed cloud services. If users want to participate in RA services, they must have necessary setups (*e.g.*, stable network) for clouds.

Compared to the previous work [22], [35], [42], [43], we are the first to focus on these problems in TEE RA.

B. Research Challenge

To address these problems, we summarize the following challenges that must be fulfilled. They are the reasons why we choose PUF and blockchain in this work.

How to decentralize TEE RoT *without* using another root key. Achieving decentralization requires a transition from single-party ($\mathcal{M}fr$) RoT to multi-party RoTs. One straightforward approach is to involve other parties in signing the measurements. However, this approach exposes the secret keys of these additional parties to the same risks of compromise from various real-world attacks [10], [12], [17]. In other words, such decentralization does not offer stronger guarantees against advanced attackers than a centralized RoT. Therefore, we need a new way to create genuine decentralized trust.

Our Insight: As explained in §II, the security of PUF sources from its unique physical structure instead of a provisioned key. Due to its ability to provide integrity and authenticity for input challenges, PUF can serve as an additional trusted party to realize measuring and "signing" simultaneously for RA. More importantly, introducing PUF into TEE essentially sets a true intrinsic RoT for attesting platforms. It not only increases the number of RoTs, but also enhances the overall security strengths for RA by offering physics-based independent and unbiased trust, which is never realized in existing research [22], [32], [43].

Also, since PUF manufacturers cannot control or alter the PUF structures, we can even use multiple PUFs from different manufacturers in one device, rendering a more decentralized manner of establishing truly trusted RoTs. Therefore, the intrinsic trust supplemented by PUF is exactly what current TEE RA designs lacks and is difficult to achieve using alternative techniques. More discussions can be seen in §VIII-A.

How to make the verification and results publicly executable and verifiable in TEE RA? Current RA verification is based on private key signatures, which should be publicly verifiable. However in practice, for the ease of management, RA vendors only allow one verifier to execute the verification and claim the results. This centralized trust raises privacy concerns for certain applications and imposes significant computational overhead for distributed attestations [22].

Our Insight: To avoid this centralized design, we need a public "bulletin board" that anyone can use it to engage with the RA verification process, leading to the consensual trust for all participants. Blockchain and smart contract are especially suitable for this. Then $\mathcal{RP}$s do not need to solely trust one verifier to perform the single-point-of-verification. Although blockchain-based solution seems somewhat straightforward, we still need to address subsequent challenges regarding to the specific designs. For example, how to use blockchain to enable efficient results re-verification of historical RA sessions in JANUS? We will further discuss these challenges in §IV-D.

How to realize *native* resilience for flexible TEE RA? A panacea does not exist for this question since different designs would have different ways to realize resilient RA functions.

Our Insight: We argue that an RA scheme is hard to obtain such resilience if only having one attestation functionality. It must have multiple available functionalities with a flexible policy to avoid a fixed structure. As depicted in Fig 2, by an automated turnout mechanism, we allow participants in JANUS to freely choose the suitable way of performing RA in case of common system failures or customized needs, thereby ensuring the resilience.

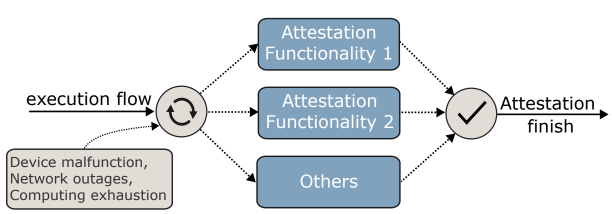


Fig. 2: Illustration of attestation turnout mechanism

C. Desired Properties

Motivated by them, we aim to design an open and resilient TEE RA scheme that achieves these goals:

- *Decentralized RoT*: A new intrinsic and independent RoT will be introduced to provide decentralized trust without using a provisioned key.
- *Open trusted verification and results*: The verification service can be executable for multiple $\mathcal{V}rf$s and others can re-verify the output results.
- *Resilient attestation procedure*: The TEE RA services will be flexible to allow participants to freely perform RA operations under various situations, offering native resilience in the real-world deployment.

IV. JANUS OVERVIEW

A. Design Principles

For the over-centralized trust problem (**P1**, **P2**), we use PUF and smart contract to realize decentralization from the attester side and verifier side, respectively. Thus, we design two distinct RA functions in JANUS, hence the name (an ancient roman guardian god with two faces).

The first (§V-B) is PUF-based mutual attestation protocols built around the double RoTs (*i.e.*, PUF and TEE root key) to enable decentralized trust, thereby solving **P1** from the attester side. The second part (§V-C) is smart contract-based RA functionalities that realize decentralized verification and result audits. This guarantees their transparency and solves **P2** from the verifier side. The two parts together form the off-chain and on-chain RA function of JANUS.

To realize resilience for JANUS (**P3**), we let the two parts to collaborate with each other by a smart contract-based turnout mechanism. This collaboration yields greater openness and offers automation to freely switch between off-chain and on-chain attestation (§V-D). This collaboration and the above decentralization present how to design an open and resilient TEE RA scheme by "teamwork" of different techniques.

B. System Architecture

Roles: Except for $\mathcal{M}fr, \mathcal{A}tt, \mathcal{V}rf$, we define the following roles for participants in JANUS:

- *Attestation Manager*: A trusted third-party (TTP) service for bootstrapping the system. It does not appear in the online attestation after the initialization.
- *Aggregator*: Resource-rich $\mathcal{A}tt$ or $\mathcal{V}rf$ who will aggregate blockchain transactions for others.
- *Auditor*: Attested legitimate $\mathcal{A}tt$ or $\mathcal{V}rf$ who will audit the attestation results of requested RA sessions.

The Attestation Manager is only for the global calculation in initialization. Its behaviors can be verified publicly (§V-A) and thus can be realized by any global service providers. It will not hurt the decentralization in practice.

Workflow: We present the workflow of JANUS in Figure 3. It has three phases: Provision, Initialization, and Attestation.

- *Provision* (offline): The TEE devices are provisioned with TEE secret materials like device-specific private keys. Also, PUF instances will be embedded into the devices as an additional RoT to provide intrinsic trust.
- *Initialization* (offline): $\mathcal{A}tt$ and $\mathcal{V}rf$ and other participants first set up a consortium blockchain. Then Attestation Manager groups $\mathcal{A}tt$ and $\mathcal{V}rf$ and helps generate keys for them. The Manager also uploads registration manifests to the blockchain and completes the initialization. Details can be seen in §V-A
- *Attestation* (online): For a complete RA session in JANUS, $\mathcal{V}rf$ and $\mathcal{A}tt$ first determine the way of attestation by turnout. Then $\mathcal{V}rf$ sends a challenge message to start the off-chain attestation (§V-B) or on-chain attestation §V-B. Last, the attestation result will be determined and may be audited in the future.

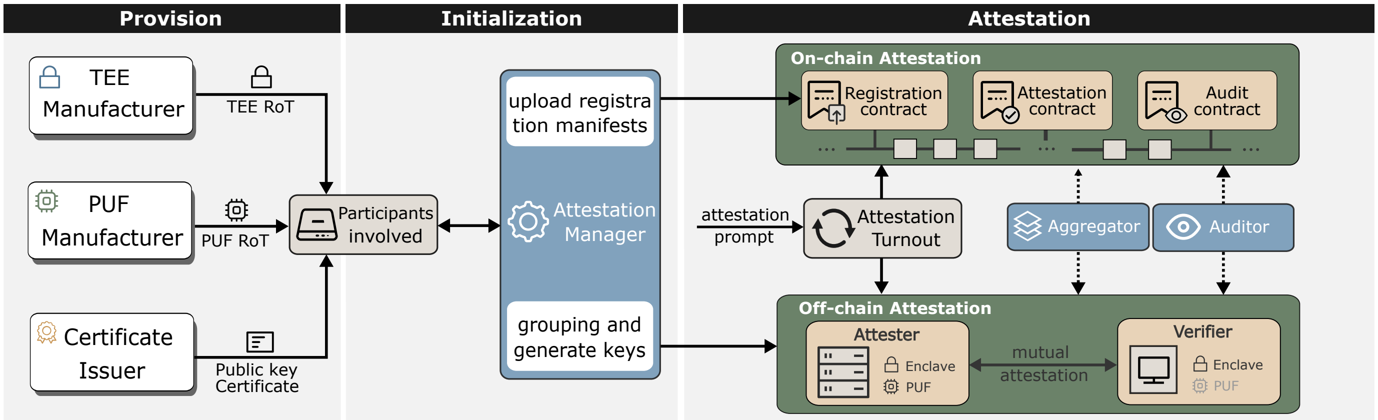


Fig. 3: Architecture and workflow of JANUS

C. Threat Model

We consider an adversary $\mathcal{A}dv$ who can launch attacks against $\mathcal{A}tt, \mathcal{V}rf$ and attempts to fool the attestation or tamper with the results in JANUS. Its goal is to access and alter applications in TEE without being caught. Specifically, in our threat model, $\mathcal{A}dv$ can:

- **A1** (**Channel attacks**) Launch man-in-the-middle (MiTM) attacks to eavesdrop, intercept, modify and replay messages between $\mathcal{A}tt$ and $\mathcal{V}rf$ in the open channel.
- **A2** (**DoS attacks**) Sending numerous erroneous attestation requests to $\mathcal{A}tt$ to exhaust its resources, and make it unavailable to legitimate RA requests in the end.
- **A3** (**Device Hijack**) Crack and re-run the devices of $\mathcal{A}tt$ or $\mathcal{V}rf$ to reveal the device keys in semi-invasive manners like side-channel analysis or external storage read-out.
- **A4** (**Integrated Circuit Delayering**) Use invasive measures like Focused Ion beam (FIB) and Laser Voltage Probing (LVP) to mill and delayer the chip, and then physically extract the fused keys in TEE.
- **A5** (**PUF Modeling attacks**) Collect or recover raw PUF CRPs from the communication packets during the online attestation. $\mathcal{A}dv$ can use them to train a PUF model by machine learning to obtain other valid CRPs.

These attacks may help $\mathcal{A}dv$ generate signed measurements or fool the verification procedure. Besides, $\mathcal{A}dv$ is allowed to corrupt the verifiers to directly modify the results. Therefore, we also consider the following attacks here:

- **A6** (**Rogue verifier**) $\mathcal{A}dv$ corrupts a legitimate $\mathcal{V}rf$ and makes it output the verification results as $\mathcal{A}dv$'s wishes.
- **A7** (**Partial verifier collusion**) $\mathcal{A}dv$ corrupts a group of legitimate verifiers, and lets them collude to make a wrong attestation claim on attesters.

Our threat model provides a more comprehensive description of $\mathcal{A}dv$' capabilities in practice. As shown in Table I, JANUS allows adversaries to conduct all these attacks while other schemes can only partially resist them.

TABLE I: Threat models of different (TEE) RA schemes

Attacks	A1	A2	A3	A4	A6	A7
SGX [2]	●	○	◐	○	○	○
OPERA [22]	●	◐	◐	○	○	○
MATEE [42]	●	○	◐	○	○	○
LIRA-V [43]	●	○	◐	○	○	○
SCRAPS [35]	●	●	◐	○	●	●
JANUS	●	●	●	●	●	●

●: Resisted; ◐: Partially resisted; ○: not resisted;

D. Design Challenges

C1: Maintaining scalability in an end-to-end (E2E) protocol. Common approaches of establishing secure E2E links between two parties use heavyweight primitives (TLS) or redundant interactions (two-party computation), limited in efficiency and scalability. This challenge is particularly tricky in our work since PUF is a hardware-unique primitive that each instance has a unique CRP domain. Pre-sharing each ones' CRPs would bring tremendous overhead instead.

C2: Avoiding attestation data (privacy) leakage on the blockchain. The consortium blockchain used here may allow rogue verifiers to probe data on-chain and recover others' output or deduce others' secrets. For example, raw measurements cannot be stored since rogue verifiers may directly obtain them to perform MiTM attacks.

C3: Snapshotting the off-chain attestation protocols. To address **P2**, taking all data of RA sessions as "snapshots" is infeasible, because this causes excessive storage overhead and latency on-chain. Therefore, we need a lightweight and secure structure to summarize the sessions, *i.e.*, verifying the snapshots is equivalent to verifying each step of the protocol.

C4: Realizing automation for attestation turnout. The attestation functionalities provided by JANUS should be closely merged with each other to achieve resilience for various applications. The turnout mechanism must be self-instructing and interconnect with each part, without requiring global operators to manage the RA procedures.

V. JANUS DESIGNS

We present the detailed design of JANUS in this section. As described in §IV-A, we realize two attestation functionalities in JANUS, *i.e.*, off-chain and on-chain, and an automated turnout mechanism to combine the two. The notations used through this paper are given as follows.

Notations: Let $\{0,1\}^*$ be the set of all bit strings and $\emptyset$ be an empty set. For any $n \in \mathbb{N}$, $\{0,1\}^n$ is the set of all n-bit strings. $X \leftarrow\!\$\, \{0,1\}^n$ means the random selection of a string X from $\{0,1\}^n$. For some $X \in \{0,1\}^n$, $|X|$ denotes the length of X. For some $X, Y \in \{0,1\}^n$, $X||Y$ denotes their concatenation, and $X \oplus Y$ denotes their bitwise XOR result. The lengths of PUF challenges and responses are set to be the same in this study, denoted by n unless otherwise specified. $H_K(\cdot)$ denotes a keyed hash function.

A. System Initialization

The offline initialization is conducted by the Attestation Manager, who helps generate IDs and keys. The Manager is *not* involved in the online attestation. W.L.O.G., we assume that verifiers do not necessarily have PUF to demonstrate the compatibility of our designs.

① **Setup and Grouping.** The participants first create their consortium blockchain accounts, *i.e.* a public-private key pair $(\mathsf{pk}, \mathsf{sk})$ with personal address $\mathsf{addr} = H(\mathsf{pk})$. Then the Manager does the following:

- Grouping: Grouping is used to organize the participants in JANUS by their locations or functions. For example, the attested applications on the same device can be naturally assigned in the same group (group id gid). The use of grouping will be further shown in ②.
- Attesters: Let $\mathsf{aid} = \mathsf{addr}||\mathsf{DSN}||\mathsf{gid}$. DSN (Device Serial Number) identifies on which device the attester $\mathcal{A}tt_{\mathsf{aid}}$ is deployed.
- Verifiers: Let $\mathsf{vid} = \mathsf{addr}||\mathsf{VON}||\mathsf{gid}$ where SON (Verifier Owner Number) specifies which owner $\mathcal{V}rf_{\mathsf{vid}}$ belongs to. vid is similar to aid, which facilitates the identity switch in the mutual attestation.

② **Protocol keys generation.** Then Manager helps generate the protocol key, including a group key (unique for each group) and a communication key (unique for each one).

- Attesters: Each $\mathcal{A}tt_{\mathsf{aid}}$ generates an initial PUF challenge C_{init}, which can be chosen arbitrarily and made public. Then $\mathcal{A}tt_{\mathsf{aid}}$ uses PUF to get $R_{\mathsf{aid}} = \textit{puf}\ (C_{\mathsf{init}})$ and $MR_{\mathsf{aid}} = \textit{puf}\ (R_{\mathsf{aid}})$, and submits R_{aid} to the Manager. The Manager calculates and returns the group key $K_{\mathsf{gid}} = \oplus_{i=1}^{N} R_i$ for a group G_{gid} with N attesters. $\mathcal{A}tt_{\mathsf{aid}}$ now calculates its communication key $MK_{\mathsf{aid}} = K_{\mathsf{aid}} \oplus R_{\mathsf{aid}} \oplus MR_{\mathsf{aid}}$.
- Verifiers: Each $\mathcal{V}rf_{\mathsf{vid}}$ randomly selects a communication key $s\mathsf{vid} \leftarrow\!\$\, \{0,1\}^n$, then submits s_{vid} to the Manager and receives their group key $S_{\mathsf{gid}} = \oplus_{j=1}^{M} s_j$.
- Manager: For each communication key, it encrypts the key using all group keys to compute the handshake materials $\mathcal{H}$. For example, for MK_i of $\mathcal{A}tt_i$, $\mathcal{H}_i = \{\ldots, \mathsf{Enc}_{K_j}(MK_i), \mathsf{Enc}_{K_i}(MK_i), \ldots \mathsf{Enc}_{S_k}(MK_i), \ldots\}$. All $\mathcal{H}$ will be uploaded to the blockchain in ③. This pre-computation can address **C1** and thus significantly improve the E2E scalability of JANUS.

Finally, $\mathcal{A}tt_{\mathsf{aid}}$ stores $(C_{\mathsf{init}}, MK_{\mathsf{aid}})$ and $\mathcal{V}rf_{\mathsf{vid}}$ stores $(s_{\mathsf{vid}}, S_{\mathsf{gid}})$. $\mathcal{A}tt_{\mathsf{aid}}$ does not need to store the group key since they can recover $K_{\mathsf{gid}} = MK_{\mathsf{aid}} \oplus MR_{\mathsf{aid}} \oplus R_{\mathsf{aid}}$ by PUF. Note that the communication keys are designed to be known by the other participant in the protocol (§V-B2). Our PUF-based designs ensures that an adversary obtaining several communication keys cannot derive any group key, eliminating the usage of secure storage (seen in §V-B3).

③ **Registration to the blockchain.** The Manager uploads registration manifests to the blockchain, including $\mathcal{H}$, device configurations, and measurements. The configurations would contain CPU SVN versions, TCB sizes, and others. The measurements fall into two types here:

- Traditional Measurement: For verifiers, their measurements are traditional enclave measurements M, *e.g.*, hash digests of the built enclaves' meta-data [2].
- PUF-based Nested Measurement: For attesters, their measurements RM are obtained by $RM = \textit{puf}\ (M)$, *i.e.*, using both PUF and TEE.

The Manager would upload $(\mathsf{pid}, H(RM_{\mathsf{aid}}||\mathsf{aid}||\mathsf{pid}))$ or $H(M_{\mathsf{vid}}||\mathsf{vid})$ instead of raw measurements to prevent the sloppy attesters in **challenge C2**. The registration contract handling these manifests is given in §V-C.

B. PUF-based Attestation Protocols

To address **P1**, we use PUF as the second but intrinsic RoT to generate authenticated measurements. The designed protocols form the off-chain RA function of JANUS.

1) Local attestation: JANUS supports the local attestation (LA) [2] that allows enclaves on the same device to attest each other. These enclaves can belong to different $\mathcal{A}tt$. As illustrated in Figure 4, the core idea of our LA protocol is based on PUF and message authentication code (MAC).

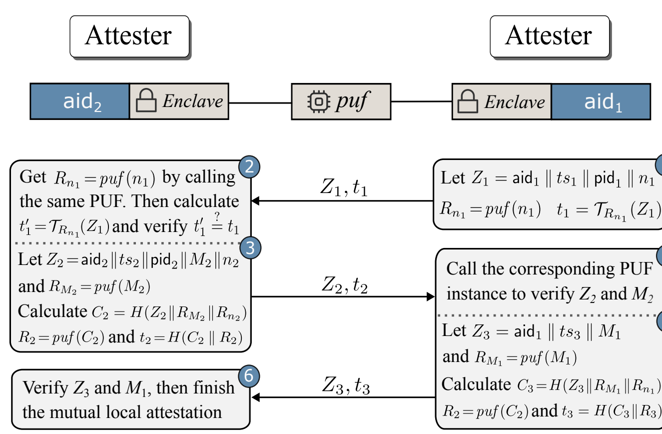


Fig. 4: The off-chain local attestation protocol of JANUS

① $\mathcal{A}tt_{\mathsf{aid}_1}$ uses aid_1, timestamp ts_1, local PUF id pid_1 and a nonce $n_1 \in \{0,1\}^n$ to form Z_1. Then it invokes PUF to get R_{n_1}. A MAC algorithm $\mathcal{T}$ like HMAC [48] is used with R_{n_1} being the key.
② $\mathcal{A}tt_{\mathsf{aid}_2}$ reproduces R_{n_1} by using n_1 as the PUF challenge. (Z_1, t_1) is considered valid only if $\mathcal{A}tt_{\mathsf{aid}_2}$ gets the same t_1.
③ After verifying (Z_1, t_1), $\mathcal{A}tt_{\mathsf{aid}_2}$ uses a different PUF instance pid_2 to get R_{M_2}. It obtains $C_2, R_2 = \textit{puf}(C_2)$ and gets t_2.
④ $\mathcal{A}tt_{\mathsf{aid}_1}$ uses the specified PUF instance to get R_{n_2} and re-calculated C_2, R_2, thereby verifying (Z_2, t_2).
⑤ $\mathcal{A}tt_{\mathsf{aid}_1}$ follows the same steps in ③ to generate (Z_3, t_3). Its measurement M_1 is also contained in Z_3 to attest $\mathcal{A}tt_{\mathsf{aid}_2}$.
⑥ If the verification passes for both enclaves, it proves that they indeed exist on the same TEE platform.

In Figure 4, two attesters $\mathcal{A}tt_{\mathsf{aid}_1}, \mathcal{A}tt_{\mathsf{aid}_2}$ control different enclaves but can both invoke the device's PUF instances. We let $\mathcal{A}tt_{\mathsf{aid}_1}$ initiate the session and generates the challenge message (Z_1, t_1) by step ①. When receiving (Z_1, t_1) from an untrusted local channel, $\mathcal{A}tt_{\mathsf{aid}_2}$ needs to invoke the same PUF to verify the tag. The *double hash-double PUF* operation in step ③ and ⑤ provides PUF-based integrity and authenticity to the tag generation. Our protocol avoids leaking PUF responses to the channel, or adversaries can collect raw CRPs to perform PUF modeling attacks.

Moreover, $\mathcal{A}tt_{\mathsf{aid}_1}$ and $\mathcal{A}tt_{\mathsf{aid}_2}$ can conveniently use a private PUF response as their shared secret key, eliminating the need of public key operations.

2) Remote attestation: This protocol is the core of off-chain RA function of JANUS. Note that "off-chain" here means that the computation takes place off-chain, although it may be required to fetch data from chain. The main procedure of this protocol is shown in Figure 5.

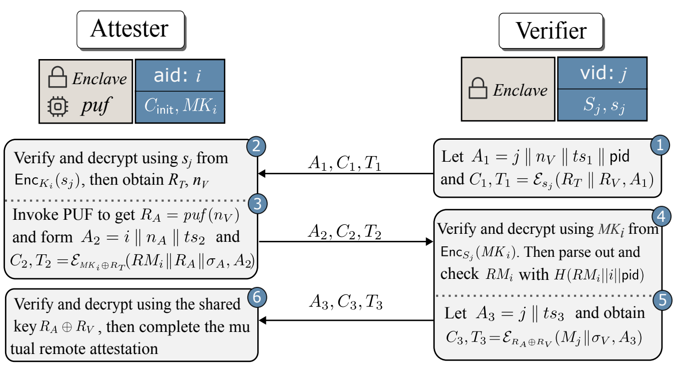


Fig. 5: The off-chain remote attestation protocol of JANUS

Before starting the RA session, the verifier $\mathcal{V}rf_j$ retrieves $\mathsf{Enc}_{S_j}(MK_i)$ and $(\mathsf{pid}, H(RM_i||i||\mathsf{pid}))$ from the blockchain. Then it initiates the RA session as follows.

Sending remote attestation challenge. $\mathcal{V}rf_j$ first constructs the challenge message (A_1, C_1, T_1):

① $\underline{(A_1, C_1, T_1)}$: $\mathcal{V}rf_j$ forms A_1 and samples $R_T, R_V \leftarrow\!\!\$ \{0,1\}^n$ where R_T is a temporary random string and R_V is a part of the shared key. Now $\mathcal{V}rf_j$ uses an authenticated encryption (AE) algorithm $\mathcal{E}$ to encrypt $R_T||R_V$ with A_1 as the associated data (AD) and s_j as the key. Thus, $\mathcal{V}rf_j$ gets the ciphertext C_1 and the tag T_1, then sends (A_1, C_1, T_1) to the attester.

Responding with signed measurement. Upon receiving (A_1, C_1, T_1), $\mathcal{A}tt_i$ learns the vid$j$ and retrieves $\mathsf{Enc}_{K_i}(s_j)$, $H(M_j||j))$ from the blockchain.

② $\underline{\text{verify } (A_1, C_1, T_1)}$: $\mathcal{A}tt_i$ gets R_i, MR_i by PUF and recovers $K_i = MK_i \oplus MR_i \oplus R_i$. It uses the decryption algorithm $\mathcal{D}$ to verify (A_1, C_1) and obtain R_T, R_V.

③ $\underline{(A_2, C_2, T_2)}$: After $\mathcal{A}tt_i$ verifies the challenge, it generates the nested measurement $RM_i = puf\ (M_i)$. Then it uses the attestation key to sign $RM_i||R_A$ where $R_A = puf_{\mathsf{pid}}\ (n_V)$ is another part of the shared session key. The signature σ_A is then appended after $RM_i||R_A$ and they are together encrypted by $\mathcal{E}$. Finally, $\mathcal{A}tt_i$ sends (A_2, C_2, T_2) as the response back to $\mathcal{V}rf_j$.

Verifying with response. $\mathcal{V}rf_j$ decrypts $\mathsf{Enc}_{S_j}(MK_i)$ to get MK_i so that it can calculate $MK_i \oplus R_T$.

④ $\underline{\text{verify the measurement}}$: $\mathcal{V}rf_j$ uses the certificate Cert_i to verify σ_A. If the verification passes, $\mathcal{V}rf_j$ parses out RM_i to re-calculate $H(RM_i||i||\mathsf{pid})$ and compares it with the one on-chain. If the measurement is valid, then $\mathcal{V}rf_j$ completes the attestation with $\mathcal{A}tt_i$.

⑤ $\underline{(A_3, C_3, T_3)}$: $\mathcal{V}rf_j$ now can obtain the shared session key $SK = R_A \oplus R_V$. Then it forms A_3 and generates its enclave measurement M_j. It calls $\mathcal{E}$ to encrypt M_j along σ_V using the shared key SK.

In step ⑥, $\mathcal{A}tt_i$ can verify (A_3, C_3, T_3) and the measurement M_j using the same procedure in ④. Then it finishes the attestation with $\mathcal{V}rf_j$ and obtains $SK = R_A \oplus R_T$.

3) Protocol Summary: We briefly summarize the above attestation protocols and give the following remarks.

- **Double RoTs**: With PUF being an extra RoT, $\mathcal{A}dv$ would fail the attestation even if it can extract the TEE root keys. This decentralized trust achieved by double RoTs can offer stronger security guarantees for TEE RA and succinctly addresses **P1**.
- **Tradeoff**: By the pre-computation of handshake materials, the PUF-based RA protocol achieves lightweight and scalable E2E communication at the cost of one query to blockchain. Participants can periodically cache the potential entries in idle states, minimizing the latency of the just-in-time query.
- **Key Protection**: The designed key structures of $\mathcal{A}tt$ use PUF response MR_{aid} to mask the secret R_{aid}. Then even if MK_{aid} is leaked, $\mathcal{A}dv$ will learn nothing about R_{aid} and K_{gid}. This masking technique can be applied to SK if stronger protection for **A3** is required.

C. Smart Contracts for Attestation

We design three smart contracts in JANUS to realize participant registration, on-chain attestation, and attestation audit. We first introduce the registration contract, which is required by both off-chain and on-chain attestation.

1) Registration Contract: As mentioned in §V-A, thi contract is used to store registration manifests. It manipulates the blockchain as an immutable distributed database $\mathcal{L}$ through key-value pairs and offers two functionalities:

① UPLOAD MANIFESTS. The Attestation Manager can upload devices' measurements, encrypted key materials and other data through this contract. The contract stores them with unique blockchain addresses as indexes in $\mathcal{L}$.

② RETRIEVE ON-CHAIN DATA. The blockchain nodes (initialized devices) are allowed to retrieve any data stored on-chain. They can use a specified address to query the block entry in $\mathcal{L}$ without direct interactions with the contract.

The registration contract avoids the risk of a single point of failure caused by a central server, thereby offering communication scalability for RA in large-scale network.

2) Attestation Contract: This contract realizes on-chain attestation for JANUS. It involves multiple verifiers in a single verification execution. As a result, both the execution and results of RA verification would be publicly aware and admitted, thereby addressing **problem P2**. The overall procedure of this contract is illustrated in Figure 6.

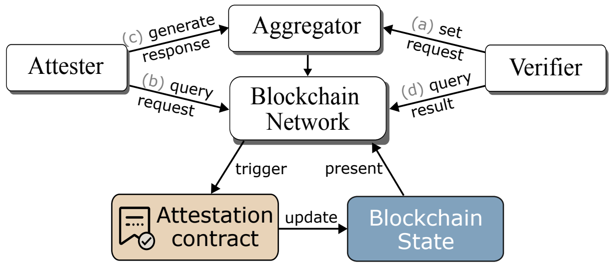


Fig. 6: Workflow of attestation smart contract

Contract description: The verifier first generates a request transaction to trigger the contract and the attester, *e.g.*, changing the state nonce n under a given request blockchain address which $\mathcal{A}tt_{\mathsf{aid}}$ periodically queries. When $\mathcal{A}tt_{\mathsf{aid}}$ learns of a coming request (the nonce being changed), it generates $RM_{\mathsf{aid}}, \sigma_A$, and gets $C = \mathsf{Enc}_{MK_{\mathsf{aid}}}(n||RM_{\mathsf{aid}}||\sigma_A)$. It wraps C in a valid transaction and submits it to a nearby Aggregator. Aggregators will wrap and submit received transactions in a batch to the blockchain.

Functionalities: Main functionalities of our attestation contract, formalized in Algorithm 1, are given as follows.

Algorithm 1: Smart contract for on-chain attestation

Input : transaction list l
Output: trust state st

<u>BATCH ATTESTATION</u>

1 **for** a transaction $tx \in l$ **do**
2   **if** tx.signature is valid **then**
3     $st \leftarrow$ON-CHAIN ATTESTATION(tx)
4   **Output:** st
5   **end**
6 **end**

<u>ON-CHAIN ATTESTATION</u>

7 $c \leftarrow \mathsf{Enc}_{S_g}(MK_{\mathsf{aid}}),\ \ t \leftarrow H(RM_{\mathsf{aid}}||\mathsf{aid}||\mathsf{pid}),\ \ \mathsf{pid}$ `/* retrieve them from blockchain */`
8 $MK_{\mathsf{aid}} \leftarrow \mathsf{Dec}_{S_g}(c), C \leftarrow tx.\mathsf{payload}$ `// parse tx`
9 $P \leftarrow \mathsf{Dec}_{MK_{\mathsf{aid}}}(C),\quad (RM^*_{\mathsf{aid}}, \sigma_A) \leftarrow \mathsf{Parse}(P)$
10 $t^* \leftarrow H(RM^*_{\mathsf{aid}}||\mathsf{aid}||\mathsf{pid})$
11 **if** σ_A is invalid or $t \neq t^*$ **then**
12   **Output:** untrusted
13 **end**
14 **Output:** trusted

① <u>ON-CHAIN ATTESTATION</u>: As shown in line 7, verifiers use MK_{aid} to obtain the measurement RM_{aid} and the signature σ_A. They can follow the same verification steps in §V-B2 and determine the final attestation result on RM_{aid} through the consensus algorithm.

② <u>BATCH ATTESTATION</u>: The contract allows verifiers to verify multiple measurements in a transaction batch at once. This improves the network's computing resource utilization and expedite the attestation process.

Specialized Applications: Our contract benefits distributed services like Function as a Service (FaaS) [49]. FaaS allows developers to deploy their applications in small functions. Due to the highly scattered deployment of the code, traditional RA schemes may suffer from efficiency issues as they have to attest each function individually. In JANUS, distributed functions can be attested efficiently through this contract.

3) Audit Contract: Though the transparency of verification results is ensured in smart contract-based attestation, it should be also guaranteed for end-to-end attestation protocols in §V-B2. To achieve this, we design an audit contract as shown in Figure 7. This contract allows other participants in JANUS to re-verify the result of a historical end-to-end RA session. It helps evaluate the device status and identify compromised verifiers or attesters in time.

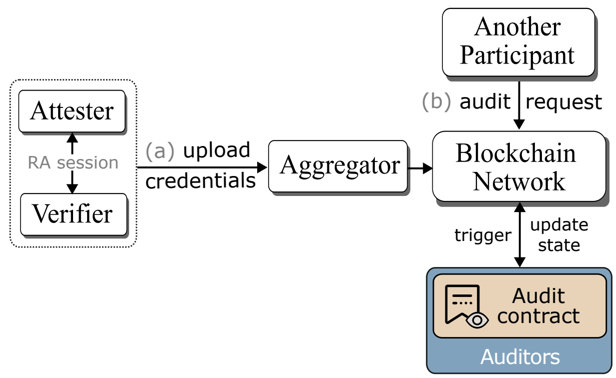


Fig. 7: Workflow of on-chain audit smart contract

Contract description: In the real world, devices are often dynamically configured, exhibiting significant heterogeneity. The audit contract enables the assessment of the devices' trust level based on their configurations like CPU SVN and vendor policy. An example metric is shown in the appendix. The trust level of a device indicates whether its protocol data, *e.g.*, measurements and output results, should be audited or not.

However, uploading all protocol messages for all devices is not feasible (**C3**), so we propose a new data structure, *credential*, to snapshot an end-to-end RA session. Here, $\mathcal{A}tt_{\mathsf{aid}}$ calculates $cr_1 = H_{K_{\mathsf{gid}}}(H_{s_{\mathsf{vid}}}(m_1||m_2||\mathsf{aid}||\mathsf{vid}))$ and $\mathcal{V}rf_{\mathsf{vid}}$ calculates $cr_2 = H_{S_{\mathsf{gid}}}(H_{MK_{\mathsf{aid}}}(m_1||m_2||\mathsf{aid}||\mathsf{vid}))$. A pair of credentials (cr_1, cr_2) ensures that both parties have executed the protocol and acquired the expected output, *i.e.*, obtained each other's key (K_{gid} or s_{vid}) and measurements (m_1 or m_2). (cr_1, cr_2) will be submitted to the blockchain through an Aggregator for auditing. To this end, We define a new trusted entity, Auditor, who is allowed to obtain others' keys from the Manager. Attested participants can become an Auditor and genuinely executes this contract in their TEE.

Functionalities: The audit contract provides three functionalities, as formalized in Algorithm 2.

① <u>TRUST RATING</u>: Devices can upload their configurations to the blockchain during initialization (§V-C1). Auditors can be requested to calculate a trust rate of a given device id by this contract. Trust rates serve as a reference for the likelihood of a device becoming compromised.

② <u>ATTESTATION REVIEW</u>: Auditors can obtain keys and measurements from the Manager to verify the credentials. As

Algorithm 2: Smart contract for attestation audit

Input : transaction list l
Output: device trust rate r, audit state st

TRUST RATING

1 **if** tx.signature is valid **then**
2 conf $\leftarrow$ Parse(tx.payload)// device config
3 $r \leftarrow$ Evaluate(conf)
4 **Output:** r
5 **end**

SPOT CHECK:

6 $n \leftarrow\$ [0, l.\text{size} - 1]$, $tx \leftarrow l[n]$
/* randomly pick one */
7 **if** tx.signature is valid **then**
8 $st \leftarrow$ATTESTATION REVIEW(tx)
9 **Output:** st
10 **end**

ATTESTATION REVIEW

11 $K_{\text{gid}}, S_{\text{gid}}, MK_{\text{aid}}, s_{\text{vid}}, m_1 \leftarrow RM_{\text{aid}}, m_2 \leftarrow M_{\text{vid}}$
/* retrieve them from the Manager */
12 $(cr_1, cr_2) \leftarrow$ Parse(tx.payload) // credentials
13 $cr_1^* \leftarrow H_{K_{\text{gid}}}(H_{s_{\text{vid}}}(m_1 || m_2 || \text{aid} || \text{vid}))$
14 $cr_2^* \leftarrow H_{S_{\text{gid}}}(H_{MK_{\text{aid}}}(m_1 || m_2 || \text{aid} || \text{vid}))$
15 **if** $cr_1 = cr_1^*$ and $cr_2 = cr_2^*$ **then**
16 **Output:** trusted
17 **end**
18 **Output:** untrusted

shown in line 13, 14, they can re-calculate (cr_1^*, cr_2^*) and compare them with (cr_1, cr_2) in a submitted transaction.

③ SPOT CHECK: Auditing every credential would bring tremendous computation overhead. Therefore, we allow Auditors to spot check some credentials, *i.e.*, treat the audit results of some randomly picked credentials as the results of the whole input list. Combined with trust rating, spot check greatly improves audit efficiency and minimizes the risk of missing compromised participants

Specialized Applications: Google zero trust architecture (BeyondCorp) [50] and CISA Trusted Internet Connections [51] both require a trust-level design and auditing capability. To the best of our knowledge, JANUS is the first RA design that is applicable to those applications. It can fulfill their requirements for real-world device attestations.

The attestation and audit contract thoroughly address P2 by enabling decentralized verification and providing consensual and auditable results. Every RA session in JANUS are publicly transparent, preventing malicous verifiers from exclusively claiming wrong attestation results.

D. Attestation Turnout

Now JANUShas several open trusted functionalities based on PUF and smart contract, but it still lacks resilience (**P3**) under unexpected system failures. To this end, turnout mechanism is designed to divert the RA workflow and make different functionalities collaborate with each other, just like the railroad turnout in the real world.

As shown in Figure 8, our turnout mechanism is instantiated by a turnout smart contract and switch rules. When encountering network outages and other situations, $\mathcal{V}rf$ and $\mathcal{A}tt$ can use them to asynchronously determine how they will perform RA. No other entities or global operators are required here, so the $\mathcal{V}rf$ and $\mathcal{A}tt$ can automatically control their attestation flow, which addresses **challenge C4**.

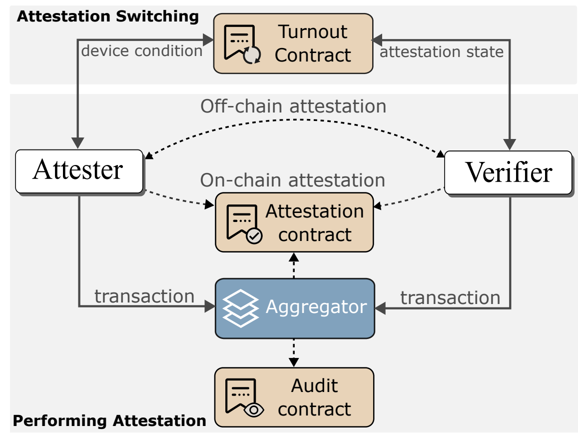


Fig. 8: Holistic view of attestation turnout in JANUS

Turnout smart contract: This contract maintains two fields for every $\mathcal{A}tt$ to change the attestation flow: device condition (`dc`) and attestation state (`as`). Both fields take one of the values in {`off-chain`, `on-chain`, `both`}. Note that `dc` is only set by $\mathcal{A}tt$ and `as` is set by anyone who wants to conduct RA with $\mathcal{A}tt$, *i.e.*, some verifier.

`dc` indicates the way in which $\mathcal{A}tt$ prefers to conduct the attestation, *i.e.*, off-chain protocol or on-chain smart contract. In practice, the device where $\mathcal{A}tt$ resides may experience network outages, low battery power, stressed computing resources, etc. `dc` gives $\mathcal{A}tt$ the leeway to adapt these predicaments. `as`, on the other hand, indicates how $\mathcal{V}rf$ will initiate an RA session with $\mathcal{A}tt$. $\mathcal{V}rf$ first checks $\mathcal{A}tt$'s `dc`, and if `dc=both`, then $\mathcal{V}rf$ can set `as` to `on-chain` or `off-chain`. Otherwise, `as` should only be set the same as `dc`.

For example, if $\mathcal{A}tt$ cannot establish E2E links currently, it can trigger the turnout contract to update `dc` to `on-chain` in the blockchain (always online). Then $\mathcal{V}rf$ will not choose to conduct off-chain RA with $\mathcal{A}tt$ after quering `dc`. $\mathcal{A}tt$ and $\mathcal{V}rf$ thus change their attestation flow independently without interference of any other entities.

Switch rules in Turnout: We now give some switch rules to instruct the turnout procedure.

SWITCH TO ON-CHAIN: After ensuring `dc` $\neq$ `off-chain`, $\mathcal{V}rf$ should choose on-chain smart contract-based attestation with $\mathcal{A}tt$ when encountering the listed cases:

- $\mathcal{V}rf$ cannot establish E2E communication with $\mathcal{A}tt$.
- The attestation results need to be acknowledged by at least a quorum of verification services.
- $\mathcal{A}tt$'s application is deployed in a distributed manner.

To this end, $\mathcal{V}rf$ needs to update `as` to `on-chain` through the turnout contract. Then it generates a request transaction to perform on-chain attestation (§V-C2).

SWITCH TO OFF-CHAIN: If `dc`$\neq$`on-chain`, $\mathcal{V}rf$ may choose to perform off-chain PUF-based attestation protocols under the following cases:

- $\mathcal{V}rf$ and $\mathcal{A}tt$ only conduct a routine attestation (periodically executed for regular check).
- Higher verification efficiency is required for real-time systems like IoV and IoT.
- $\mathcal{V}rf$ needs off-chain attestation for other reasons, *e.g.*, privacy concerns, personal needs.

In the first two cases, $\mathcal{V}rf$ are not required to update `as`, thereby saving unnecessary interactions with the contract. It directly sends the challenge of step ① in §V-B2 to start the attestation. For the third case, $\mathcal{V}rf$ should first update `as` to `off-chain` before the attestation. $\mathcal{V}rf$ and $\mathcal{A}tt$ are indeed required to submit their credentials afterward in all three cases.

These switch rules enable $\mathcal{V}rf$ and $\mathcal{A}tt$ to switch between on-chain or off-chain attestation seamlessly. They can serve as basic guidelines when administrators are configuring customized policies for a TEE RA system. This paper does not delve into the specifics of turnout configurations since they will differ across various RA systems.

VI. JANUS SECURITY

A. Security Proof of PUF-based Protocols

We first formalize the PUF definition and give the idealized PUF properties as follows.

Definition 1: (Ideal PUF) Let P be a PUF family where $R = \mathit{puf}(C)$ for $\forall \mathit{puf} \in P$ and $C, R \in \{0,1\}^*$. P is an ideal PUF family if $|C| = |R| = n$ and for any probabilistic polynomial-time (PPT) distinguisher $\mathcal{A}$, we have,

- (Pseudorandomness) $\forall \mathit{puf} \in P, \forall C \in \{0,1\}^n$,
$$\left|\Pr\left[\mathcal{A}^{\mathit{puf}(C)} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\$} \Rightarrow 1\right]\right| \leq \frac{1}{2^n} \cdot \epsilon, \quad (1)$$
where ϵ is a negligible value related to n and \$ denotes a random bits oracle.
- (Uniqueness) $\forall \mathit{puf}_1, \mathit{puf}_2 \in P, \forall C \in \{0,1\}^n$, let $p(C) = \mathit{puf}_0(C) \oplus \mathit{puf}_1(C)$,
$$\left|\Pr\left[\mathcal{A}^{p(C)} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\$} \Rightarrow 1\right]\right| \leq \frac{1}{2^n} \cdot \epsilon'. \quad (2)$$
- (Unclonability) Each PUF instance has a unique physical structure. Any attempt to evaluate a pre-installed PUF instance in simulation environments such as hardware debugging would change its response generation process and turn it into a different instance.

In this work, we utilize Universally Composable (UC) framework [41] to depict the security of our off-chain protocols in §V-B1 and §V-B2. UC aims to rigorously demonstrate the (in)distinguishability of the protocol execution in the real world and ideal world. In the ideal world, an ideal functionality $\mathcal{F}$ describes the abstracted interfaces that a protocol wishes to achieve. Therefore, we propose the first mutual attestation ideal functionality $\mathcal{F}_{att}$ in the appendix.

Then a Simulator $\mathcal{S}$ in the ideal world emulates the protocol execution and behaviors of the real-world adversary $\mathcal{A}$. The protocol realizes the ideal functionality if an environment machine $\mathcal{Z}$ cannot distinguish the observed output is from the real world or the ideal world. For JANUS, we have the following theorem to prove,

Theorem 1: If the AEAD algorithm $(\mathcal{E}, \mathcal{D})$ provides ciphertext indistinguishability under chosen ciphertext attack (IND-CCA), the hash function H is collision resistant and *puf* is an ideal PUF, then the PUF-based attestation protocols of JANUS with respect to a global database $\mathcal{L}$ UC-realizes $\mathcal{F}_{att}$ against a static adversary $\mathcal{A}$.

Proof The complete proof is given in the appendix. □

B. Security Analysis

We now give the security analysis against different attacks and informally describe how JANUS is capable of defending against them.

- **Channel attacks**: JANUS utilizes cryptographic techniques to prevent $\mathcal{A}dv$ to replay, modify, forge any messages in an open channel. The AEAD algorithm $(\mathcal{E}, \mathcal{D})$ ensures data confidentiality, integrity, and authenticity, while the PUF-based MAC structure in LA guarantees the authenticity of (Z, t).
- **DoS attacks**: The AEAD algorithm is a symmetric cipher and implementation-friendly to constrained devices. Thus, no excessive computation burdens are imposed on attesters. Moreover, the blockchain can be considered a robust distributed system that can withstand numerous data-retrieval requests.
- **Device Hijack**: The $\mathcal{A}dv$ may repeatedly invoke the attestation procedure to generate desired side-channel leakage after physically hijacking the device. However, due to PUF, K_{gid} remains secure even if MK_{aid} is deducted out by power side-channel analysis.
- **Integrated Circuit Delayering**: Traditional schemes usually rely on the security of provisioned TEE root keys. As an additional intrinsic RoT, PUF itself plays the same role as the keys. This secrecy comes from the unpredictable physical structures rather than an externally key. Unlike TPM or other mechanisms, this property ensures $\mathcal{A}dv$ cannot pass the attestation if it retrieves the TEE root keys by chip delayering.
- **PUF modeling attacks**: Our protocols and smart contracts do not expose PUF CRPs during transmission, preventing $\mathcal{A}dv$ from gathering raw CRPs to train a PUF model. Also, JANUS is PUF-agnostic and can be adapted to various modeling-resilient PUF designs.
- **Rogue verifier and collusion**: The attestation contract in §V-C2 prevents one or some rogue verifiers from intentionally declaring the wrong attestation results. The audit contract can also minimize the impact that a rogue verifier caused.

VII. IMPLEMENTATION AND EVALUATION

A. JANUS *Implementation*

We provide a PoC implementation of JANUS in this paper. For off-chain attestation protocols, we use a LPC55S69 evaluation board as the verifier and an Dell EMC R750 Server as the attester. LPC55S69 enables ARM TrustZone and has dual

TABLE II: Performance of off-chain remote attestation of JANUS

Items	Local Attestation			Remote Attestation			Binary size (KB)	PUF ((8,8)-IPUF)	Communication Latency
	Overall	Challenge	Response	Overall	Challenge	Response			
SGX Server	1.18 ms	0.12ms	0.81 ms	4.98 ms	1.64 ms (③)	0.04 ms (②) 2.16 ms (⑥)	548	2267 LUT + 295 FF, 14ms†	3 ms ∼ 7 ms
LPC55S69		-		510.46 ms	3.47 ms (①) 164.12 ms (④)	341.98 ms (⑤)	348		

†: including the latency of serial port communication;

core Arm Cortex-M33 up to 150 MHz. It is a widely used general-purpose IoT device [52]. The EMC Server is with 512 GB RAM and with Intel Xeon Gold 6330 3.1 GHz processor enabling SGX. We set an (8,8)-IPUF [53] instance on a Digilent Nexys board with Xilinx Artix-7 embedded. IPUF is connected to the EMC Server to be the second RoT. The LPC board communicates with the EMC server using socket API. They simulate the scenario where distributed application modules require mutual RA.

For on-chain smart contracts, we use Hyperledger Sawtooth [54] to implement our smart contracts in Docker containers, as in [35], [37], The contracts are deployed on the EMC server and thus will be running with the attester code. We allow remote devices to access the Sawtooth network via REST API and the exposed container ports. Then the LPC board can interact with the Sawtooth blockchain.

B. Prototype Evaluation

1) Off-Chain Protocol Evaluation: Table III compares the theoretical performance of the PUF-based RA protocol in §V-B2 with SGX and LIRA-V [43].

Theoretical comparison: We can see that JANUS has a compact protocol structure using three-round interactions to complete mutual attestation. Compared to other protocols, JANUS only needs asymmetric primitives in measurement signing, resulting in lower computation overhead on resource-constrained platforms. And JANUS is the only one to achieve UC-based provably security. Also, JANUS does not necessitate the pre-storage of cryptographic materials for peer-to-peer communication, reducing storage overhead to constant complexity. Therefore, the off-chain protocol of JANUS is lightweight and scalable for numerous devices.

Microbenchmark: The performances of the protocols are given in Table II. We here choose ASCON [55] (the new NIST lightweight AEAD standard), SHA256 and ECDSA with secp256r1. The devices genuinely execute the PUF-based RA protocol (∼1000 LoC) meanwhile we use `gettimeofday()` and hardware cycle counter `KIN1_DWT_CYCCNT` to record the latency.

TABLE III: Theoretical comparison of different RA protocols

Properties	SGX	LIRA-V	JANUS
Mutual RA	✗	✓	✓
Anti-DoS	✗	✗	✓
Security proof	✗	✗	✓
No. of RoT	1	1	2
Signature calls	2	2	2
Key agreement	ECDH	ECDH	PUF XOR
Protocol round	2	3	3

The local attestation protocol (∼1500 LoC) involving no asymmetric cryptographic operations can be completed in less than 1.2 ms on SGX enclaves in the server. The majority of computation latency for the RA protocol comes from the signing and verifying operation that is about 157.7 ms and 337.30 ms on the LPC board. The AEAD-based design can save considerable computation resources. Also, the raw executable files are 548 KB and 348 KB on each platform, making it resource-friendly and highly portable.

2) On-Chain Smart Contract Evaluation: We implement the smart contracts in Python (∼600 LoC) and the corresponding client in C (∼800 LoC) that allows devices to form and send the transactions to Sawtooth Network.

Microbenchmark: Table IV summarizes the overall performance of on-chain attestation of JANUS. We measure the performance from two perspectives: (1) client preparation and (2) contract execution. For (1), except the attestation data generation, the required operations by Sawtooth including data serialization and transaction wrapping bring much overhead actually. Hence, as shown in Table IV, the computation latency on the board is about 300 ms∼400 ms and the server is about 3 ms∼4 ms. For (2), off-chain data submissions are executed faster than transaction payload verification. The submission latency of three contracts are all less than 5 ms and is acceptable for real-time systems. Our tests also find out that the LPC board requires about 12 ms to retrieve the stored data on-chain through unix sockets in local network. These results show that our off/on-chain interactions do not adversely affect the overall performance of the entire RA procedure in JANUS.

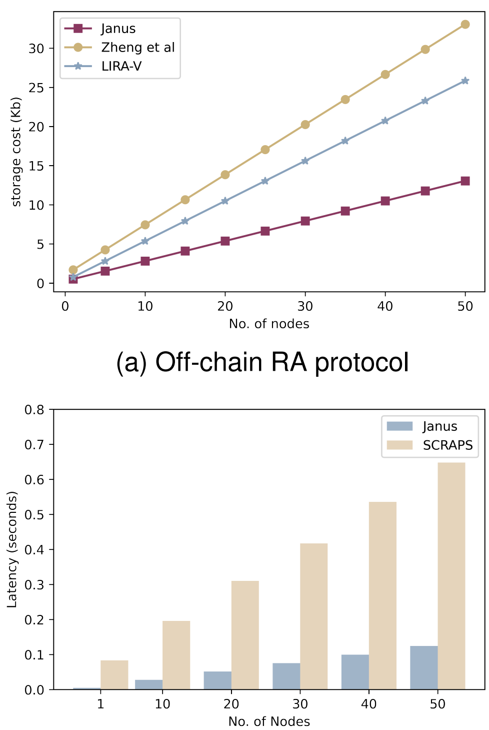
3) Scalability Evaluation: We present the scalability evaluation of JANUS's off-chain and on-chain attestation

Baseline selection: We choose [56], [43] as the off-chain comparison baseline and [35] for on-chain. [56] presents an E2E PUF-based authentication protocol and [35] involves the design of RA protocol for two remote devices. On the other hand, [35] also uses smart contract for attestation and thus is partially similar with JANUS's on-chain part.

Comparison: Figure 9a illustrates the storage cost incurred by a $\mathcal{V}rf$ when communicating with other $\mathcal{A}tts$. Compared with [43], [56], JANUS requires the minimal cryptographic materials in E2E communication. Figure 9b provides a comparison of on-chain attestation latency between JANUS and SCRAPS [35]. JANUS exhibits a slower upward trend as the number of $\mathcal{A}tt$ increases and save many interaction latency than SCRAPS. JANUS can offer better scalable off-chain and on-chain attestation services for large-scale networks.

TABLE IV: Performance of onchain attestation of JANUS

Side	Item	Attestation		Audit		Turnout	
Contract	Script size	8.0 KB		8.0 KB		5.7 KB	
	Off-chain Data submission	2.5 ms		3.7 ms		4.4 ms	
	TX Payload Verification	29.6 ms		8.3 ms		-	
	Total calls	3		2		2	
Client	**Platform**	Challenge	Attestation report	Credentials	Request	dc update	as submission
	EMC SGX Server	3.15 ms	3.75 ms	3.11 ms	3.08 ms	3.06 ms	3.17 ms
	LPC55S69	341.26 ms	486.25 ms	333.81 ms	324.75 ms	325.85 ms	338.60 ms



(a) Off-chain RA protocol

(b) On-chain Attestation

Fig. 9: Scalability evaluation of JANUS when increasing the involved attested participants

VIII. DISCUSSION

In this section, we discuss the detailed reasons of why PUF is a preferred primitive in TEE RA, and additional efforts of integrating PUF into real-world TEEs

A. Choosing PUF over Other Hardware-based Techniques

PUF internalizes environmental randomness on its physical structure and provides a native combination of hardware protection and cryptographic one-wayness, which is never realized by other primitives. The reasons why we choose PUF instead of TPM, HSM or other techniques are as follows:

New security guarantees: PUF guarantees trusted measuring by the unique intrinsic structures instead of a given secret key. However, TPM or HSM establish RoT by securing the fused private keys and cryptographic operations, which is similar with TEE. Thus, they cannot provide new security guarantees. On the other hand, PUF eliminates the target (*i.e.*, keys) of many real-world attacks by a new security mechanism. It can offer stronger guarantees to enforce decentralized trust and enables JANUS to have extra advantages against advanced attackers (**A4**).

Independent trust: Once the fabrication is completed and the chip is mounted onto the motherboard, PUF reserves an untouched area of circuits that cannot be modified by others (even its manufacturer). Therefore, plus the "keyless" security mechanism it offers, PUF creates a new source of trust that is truly independent of the TEE manufacturer. However, in TPM 2.0, the keys for attestation can be injected by specified parties [57], which may be vulnerable to supply chain attacks and not a not fully trusted process. Therefore, TPM is hard to provide independent trust compared to PUF.

Succinct usage: Compared to other techniques, PUF only exposes challenge and response port. When PUF splits the original centralized RoT, its succinct usage benefits the security designs around it, as shown by our protocols, which are compatible with existing TEE functions in a new workflow.

B. Integrating PUF into TEEs

Intel [46], NXP [47] and Xilinx [58] have released products with integrated PUF. However, what they use is SRAM PUF that has only one response and thus is insufficient for our designs. With PUF being standardized [59], [60] and more commercialized, a hardware ecosystem with built-in stronger PUF can be expected in the near future. Here we envision a possible roadmap to demonstrate the feasibility.

The off-the-shelf hardware integration would require xPU (*e.g.*, CPU, GPU, IPU and NPU) extensions and new xPU-PUF interface design. Potential efforts include: (1) On the PUF side, it should be defined as an attestable fixed function module. It should be setup with a clear secure model (*e.g.*, side-channel resistant). (2) On the xPU side, new hardware extensions or drivers should be added to cooperate with the installed PUF. Then RA related instructions (e.g., measurement or signing) could optionally be enforced by PUF. (3) The (off-die) physical link between xPU and PUF should be encrypted. It may also need access control and atomic data import/export. The xPU-PUF interface should be securely designed with minimal necessary operations.

C. Limitations

We identify two limitations of JANUS. First, PUF here must work *after* TEE to generate a nested measurement. It decentralizes the attestation endorsements in a sequential way.

Therefore, we cannot use PUF alone to obtain measurements in JANUS in case of TEE failures. We intend to address this limitation in the future. Second, JANUS does not provide a formalized turnout policy. The smart contract-based mechanism is particular for the on/off-chain design. It is also difficult to be evaluated due to the lack of baselines. This limitation may not be well addressed since turnout does not have a universal form for different RA schemes.

IX. Related Work

RA has been a necessary component of many TEEs except for SGX, such as Sanctum [61], Keystone [62], TDX [63], and SEV [64]. Many studies have focused on topics like TOCTOU [65], verifiable structure [66], [67], real-time RA [68]–[70]. Here we will investigate various RA schemes from the view of the architectural features.

Decentralized RA Design. As mentioned in §III-A, Intel DCAP-based RA [8] delegates its authority and allows non-Intel parties to author their SGX attestation. OPERA [22] offloads Intel-centric RA functionality to OPERA servers by customized certificates. It enables enclaves to deploy their own service though they are still chained on Intel. To avoid the involvement of TTPs in mutual RA, MAGE [42] separately generates enclaves' identities and measurements, thereby resolving the cycling dependency issue. MATEE [32] provides an independent key structure and measurement procedure in TPM, so it can cooperate with the original TEE RoT and achieves the multimodal attestation.

The decentralization in these approaches is not from the establishment of underlying RoT. However, JANUS, by the integration of PUF, introduces a different trust mechanism from TEE (and TPM) to build a truly intrinsic RoT.

PUF-based Protocols. PUF was first introduced for measurement in [71], [72]. However, the measurements in [71]–[73] are verified using emulated PUF models, which are unreliable and vulnerable to model extraction attacks. Therefore, PUF CRPs can be pre-stored in a central server [74] or blockchain [75], but this would leak raw CRPs to adversaries for modeling attack. PAtt [76] adopts the idea of PUF on programmable controllers for attestation. Dijk *et al*. [77] use PUF to construct an extended interface for attestation key insulation. Compared with these work, JANUS is the first solution to apply the trust measurement capability of PUF to TEE with scalable attestation protocols.

PUF has been used as a security primitive in many scenarios [78]–[80] other than RA. Chatterjee *et al*. designed anonymous PUF-based authentication protocols [81] [82] for IoT. PUF-RAKE [83] is a key exchange protocol with CRP obfuscation. Zheng *et al*. [56] designed a provably secure mutual authentication protocol for peer PUF devices.

Collective Attestation. SEDA [84] and LISA [85] utilizes a multiway tree data structure for organizing the one-to-many swarm attestation. SANA [86] allows attesters to publish self-attested evidence to the verifier. DIAT [38] decomposes the attesting program into modules for collective attestation in autonomous systems. FeSA [87] realizes automatic attestation through distributed federated learning. PASTA [88] uses Schnorr multi-signature to facilitate the validations. TM-Coin [36], BARRET [89], and others [35], [37], utilize blockchain to perform the verification collaboratively. These schemes only remains scalable on specific scenarios while JANUS can adapt to different conditions by turnout.

X. Conclusion and Future Work

To decentralize the trust establishment in traditional TEE RA designs, we propose using PUF as an additional intrinsic RoT to break the closed RA architecture dominated by TEE manufacturers. Our PUF-based mutual attestation protocols are lightweight in terms of both computation and storage. And to eliminate the centralization in single-party verification, we design several smart contracts to enable measurements to be publicly verifiable and permanently reviewable for all participants. Our turnout mechanism can naturally combine the different attestation functionalities of JANUS to form a resilient RA service. JANUS thoroughly addresses the issues of over-centralization and lack of resilience that exist in nearly all TEE RA schemes. It can be a comprehensive solution of increasing trust in TEE and shed some light on designing a more secure and trusted TEE RA paradigm.

In the future, we plan to overcome the aforementioned drawbacks of JANUS. We will design a dedicated PUF-based measuring mechanism for kinds of platforms including TEE. It is expected to achieve lightweight runtime measuring by PUF. Building on this and JANUS framework, we will explore how to use PUF to provide runtime measurement and design PUF-based heterogeneous TEE network.

Appendix A
Security Proof

As mentioned in §VI-A, the following definition formalizes the basic idea of UC-based security proof.

Definition 2: (UC-security) A protocol π UC-realizes the ideal functionality $\mathcal{F}$, if for any PPT adversary $\mathcal{A}$ there exists a simulator S such that no PPT environment machine $\mathcal{Z}$ exists to distinguish whether it contacts with π and $\mathcal{A}$ in the real world or $\mathcal{S}$ and $\mathcal{F}$ in the ideal world. We have,

$$REAL_{\pi,\mathcal{A},\mathcal{Z}} \approx IDEAL_{\mathcal{F},\mathcal{S},\mathcal{Z}} \tag{3}$$

where $\approx$ denotes the computational indistinguishability.

The formalized ideal functionality $\mathcal{F}_{\mathcal{A}tt}$ for mutual attestation protocols is shown in Figure 10. We omit the repeated entry check of tape records on $\mathsf{aid}, \mathsf{vid}, \mathsf{ssid}$ in $\mathcal{F}_{\mathcal{A}tt}$. Assuming the system is parameteraized with 1^λ, we take the remote attestation protocol π_{RA} as an example to prove Theorem 1 as follows.

Proof We construct a simulator $\mathcal{S}$ that precisely emulates $\mathcal{A}$'s behaviors. We assume that a PPT adversary $\mathcal{A}$ can statically corrupt any party in π_{RA}, *i.e.*, $\mathcal{V}rf$ or $\mathcal{A}tt$, to obtain its internal state and tape input and output. $\mathcal{S}$ runs an internal copy of $\mathcal{A}$ and then proceeds in the following cases.

Case 1. *$\mathcal{A}tt$ and $\mathcal{V}rf$ are both honest.*

$\mathcal{A}$ in the real world only obtains the channel messages if $\mathcal{A}tt$ and $\mathcal{V}rf$ are not corrupted. So $\mathcal{S}$ in the ideal world need to simulate the messages exchanged between $\mathcal{A}tt$ and $\mathcal{V}rf$ in the real

Functionality: $\mathcal{F}_{att}$

Initialization:

- Upon receiving $(\textsc{Measurement}, \mathsf{id}, \mathsf{prog})$ from a party P, generate platform information $\mathcal{I}$ and obtain measurement $m := \mathsf{memms}(\mathsf{prog}, \mathcal{I})$, then let $\mathcal{L}[\mathsf{id}, \mathsf{prog}] := (\mathsf{measured}, \mathcal{I}, m)$ and return $(\textsc{Measured}, \mathsf{id})$ to P.
- Upon receiving $(\textsc{Provision}, \mathsf{id})$ from P,
 (a) generate the platform key set $\mathcal{K}$ and helper data $\mathcal{W}$, update $\mathcal{L}[\mathsf{id}, :].\mathsf{append}(\mathcal{K}, \mathcal{W})$;
 (b) let $\mathcal{L}[\mathsf{id}, :].\mathsf{state} = \mathsf{provisioned}$ and return $(\textsc{Provisioned}, \mathcal{W}, \mathsf{id})$ to P.

Mutual Report:

- Upon receiving $(\textsc{Challenge}, i, j, \mathsf{prog}_i, \mathsf{ssid})$ from P_i,
 (a) send the message to $\mathcal{S}$ and update $\mathcal{L}[i, \mathsf{prog}_i].\mathsf{append}(\mathsf{ssid})$;
 (b) if receive $(\textsc{Challenged}, A, C, T, \mathsf{ssid})$ from $\mathcal{S}$, send it to P_j and set $\mathcal{L}[j, \mathsf{prog}_j].\mathsf{state} = \mathsf{challenged}$.
- Upon receiving $(\textsc{MutReport}, i, j, \mathsf{prog}_j, \mathsf{ssid})$ from P_j,
 (a) send the message along with $(\mathcal{I}_i, \mathsf{prog}_i, \mathcal{I}_j, \mathsf{prog}_j)$ to $\mathcal{S}$ and update $\mathcal{L}[j, \mathsf{prog}_j].\mathsf{append}(\mathsf{ssid})$;
 (b) if receive $(\textsc{MutAttReport}, \langle A_i, C_i, T_i\rangle, \langle A_j, C_j, T_j\rangle, \sigma_i, \sigma_j, \mathsf{ssid})$ from $\mathcal{S}$, set $\mathcal{L}[i, \mathsf{prog}_i].\mathsf{state} = \mathsf{challenged}$. Then send $(\textsc{Attestation}, A_i, C_i, T_i, \mathsf{ssid})$ and $(\textsc{Attestation}, A_j, C_j, T_j, \mathsf{ssid})$ to P_i, P_j, respectively.

Mutual Verification:

- Upon receiving $(\textsc{Verification}, i, j, A_i, C_i, T_i, A_j, C_j, T_j, \mathsf{ssid})$ from P_i or P_j,
 (a) obtain the signature $\sigma_i := \mathsf{Recov}_{\mathcal{K}\ _i, \mathcal{W}_i}(A_i, C_i, T_i)$ and $\sigma_j := \mathsf{Recov}_{\mathcal{K}_j, \mathcal{W}_j}(A_j, C_j, T_j)$;
 (b) if the party is corrupted, or the submitted record and the signature have not appeared or appear more than once, set $f[i, j] := 1$ or $f[j, i] := 1$ accordingly; otherwise let $f[i, j]$ or $f[j, i]$ be the result of $\mathsf{PrfVrf}_{\mathcal{K}, \mathcal{W}}(\sigma, \mathcal{I}, m)$;
 (c) set $\mathcal{L}[i, \mathsf{prog}_i].\mathsf{state} = \mathcal{L}[j, \mathsf{prog}_j].\mathsf{state} = \mathsf{attested}$ and send $(\textsc{Attested}, f[i, j], \mathsf{ssid})$, $(\textsc{Attested}, f[j, i], \mathsf{ssid})$ to P_i, P_j, respectively.

Fig. 10: Ideal Functionality of Mutual Attestation

world. Upon receiving PROVISION from $\mathcal{F}_{\mathcal{A}tt}$, $\mathcal{S}$ randomly selects $\mathcal{K}_{\mathcal{A}tt} := (MK_{\mathsf{id}}, K)$ and $\mathcal{K}_{\mathcal{V}rf} := (s_{\mathsf{id}}, S)$. Upon receiving CHALLENGE, $\mathcal{S}$ obtains $R_V, R_A, R_T \leftarrow\!\!\$\ \{0,1\}^n$ and simulates MUTATTREPORT by randomly sampling PUF-based measurement RM and attestation proof σ. Then it uses AEAD encryption $\mathcal{E}$ to get (A, C, T) for $\mathcal{A}tt$ and $\mathcal{V}rf$. It sends these messages to the internal adversary $\mathcal{A}$ who outputs them to the environment machine $\mathcal{Z}$. We denote $\mathsf{Adv}^{\mathrm{IND}}_{\pi_{RA}, \mathcal{F}_{att}}(Z)$ as the probability upper bound of Z distinguishing them with the real-world messages, and we have $\mathsf{Adv}^{\mathrm{IND}}_{\pi_{RA}, \mathcal{F}_{att}}(Z) \leq 3\mathsf{Adv}^{\mathrm{IND}}_{\mathrm{AEAD}}(Z) + \mathsf{Adv}^{\mathrm{IND}}_{\mathrm{PUF}}(Z) + 2\mathsf{Adv}^{\mathrm{EUF}}_{\mathrm{Sig}}(Z) + 2\mathsf{Adv}^{\mathrm{Coll}}_{\mathrm{H}}(Z) = \epsilon(\lambda)$ where $\epsilon(\lambda)$ is a negligible value $(negl)$ related to λ.

Case 2. *$\mathcal{A}tt$ is corrupted and $\mathcal{V}rf$ is honest.*

If $\mathcal{A}$ corrupts $\mathcal{A}tt$ in the real world, $\mathcal{S}$ can also obtain the historical states of $\mathcal{A}tt$. Therefore, $\mathcal{S}$ can get $\mathcal{K}_{\mathcal{A}tt}$ to decrypt the old (C, T) to obtain the measurement RM and recover the attestation proof σ during the challenge phase. Then, it can use these extracted materials to calculate $\mathcal{A}tt$'s output as if they are generated by $\mathcal{A}$ in the real world execution. However, since $\mathcal{S}$ cannot obtain the states of $\mathcal{V}rf$, it has to simulate $\mathcal{V}rf$'s output to $\mathcal{F}_{att}$, as in the **Case 1**. Consequently, the advantage of $\mathcal{Z}$ to distinguish $\mathcal{V}rf$'s output should not exceed $2\mathsf{Adv}^{\mathrm{IND}}_{\mathrm{AEAD}}(Z) + \mathsf{Adv}^{\mathrm{EUF}}_{\mathrm{Sig}}(Z) + \mathsf{Adv}^{\mathrm{wColl}}_{\mathrm{H}}(Z) = \epsilon(\lambda)$.

Case 3. *$\mathcal{A}tt$ is honest and $\mathcal{V}rf$ is corrupted.*

Similar to **Case 2**, if $\mathcal{A}$ corrupts $\mathcal{V}rf$ in the real world, $\mathcal{S}$ can obtain (s, S) to recover its measurement M and the corresponding proof. Then $\mathcal{S}$ simulates $\mathcal{A}tt$'s output and forms (A, C, T) as the input to $\mathcal{F}_{att}$. Since $\mathcal{A}tt$ is assumed to be equipped with PUF, $\mathcal{S}$ has to randomly sample a PUF response as the measurement. $\mathcal{Z}$ needs to break the indistinguishability game of ideal PUF, thereby having the advantage $\mathsf{Adv}^{\mathrm{IND}}_{\pi_{RA}, \mathcal{F}_{att}}(Z) \leq \mathsf{Adv}^{\mathrm{IND}}_{\mathrm{AEAD}}(Z) + \mathsf{Adv}^{\mathrm{EUF}}_{\mathrm{Sig}}(Z) + \mathsf{Adv}^{\mathrm{wColl}}_{\mathrm{H}}(Z) + \mathsf{Adv}^{\mathrm{IND}}_{\mathrm{PUF}}(Z) = \epsilon(\lambda)$.

Case 4. *$\mathcal{A}tt$ and $\mathcal{V}rf$ are both corrupted.*

This case is straightforward since $\mathcal{S}$ can extract all states in $\mathcal{A}tt$ and $\mathcal{V}rf$.

□

Note that the local attestation protocol can also be proven under the same ideal functionality. Therefore, the full off-chain attestation procedure of JANUS can achieve UC-security by directly applying the UC-composable theorem in [41].

APPENDIX B
COMPATIBILITY REAL-WORLD TEE REMOTE ATTESTATION

Compatibility: JANUS is designed to have a regular attestation procedure as we expected. It can be easily integrated with real-world RA schemes by changing the payload field. Table V summarizes the attestation report formats of mainstream TEE RA schemes. Moreover, since the off-chain remote attestation protocol in §V-B2 preserves a symmetric structure for both parties, our protocol can be scalable and lightweight regardless of whether attesters and verifiers have PUF installed. For on-chain attestation, the smart contracts are loosely coupled with device hardware configurations. JANUS focuses more on the architectural design of attestation and is open to underlying TEE specifications, making it compatible with different systems with heterogeneous devices.

Table V summarizes the format of several real-world TEE RA designs, including Intel SGX, Azure and ACK-TEE

TABLE V: Real-world Remote Attestation Report Fields

Field	Intel SGX	ACK-TEE	Azure for SGX	Azure for SEV-SNP	Azure for TPM
SVN	isvsvn, cpusvn	svn	svn	bootloader_svn, tee_svn guest_svn, microcode_svn	tpmversion
Measurement	mrenclave	mrenclave	mrenclave	launchmeasurement	-
Signer	mrsigner	mrsigner	mrsigner	authorkeydigest	aikPubHash
Signature	signature	signature	signature	signature	signature
Others	base_name, report_data, isvpoid	product_id	product_id, report_data	imageid, report_id report_data	vbsReportPresent, secureBootEnabled

(**A**libaba Cloud **C**ontainer Service for **K**ubernetes). This table can serve as the evaluation metric of the trust rating in §V-C3. Auditors can label each device based on the device configuration details in the table to reflect its trust confidence level. Additionally, this table can also serve as a reference for integrating JANUS's attestation report into real-world TEE products.

ACKNOWLEDGMENTS

The authors would like to thank the editors and anonymous reviewers for their work. This work is supported by National Key R&D Program of China (No. 2022ZD0120304) and Shanghai Science and Technology Innovation Action Program (No. 22511101300).

REFERENCES

[1] H. Birkholz, D. Thaler, M. Richardson, N. Smith, and W. Pan, "Remote ATtestation procedureS (RATS) Architecture," RFC 9334, Jan. 2023. [Online]. Available: https://www.rfc-editor.org/info/rfc9334

[2] V. Costan and S. Devadas, "Intel SGX explained," *Cryptology ePrint Archive*, 2016, https://eprint.iacr.org/2016/086.pdf.

[3] Microsoft, "Microsoft Azure Attestation," https://azure.microsoft.com/en-us/products/azure-attestation, 2021.

[4] AWS Nitro Enclaves, "Cryptographic attestation," https://docs.aws.amazon.com/enclaves/latest/user/set-up-attestation.html, 2023.

[5] Open Compute Project, "Caliptra, a new specification for an open source silicon root of trust (RoT)," https://www.opencompute.org/documents/caliptra-silicon-rot-services-09012022-pdf, 2021.

[6] OpenTitan, "OpenTitan Homepage," https://opentitan.org/, 2023.

[7] Intel, "Project Amber: Increasing Trust in Confidential Computing," https://www.intel.com/content/www/us/en/security/project-amber.html, 2023.

[8] V. Scarlata, S. Johnson, J. Beaney, and P. Zmijewski, "Supporting Third Party Attestation for Intel® SGX with Intel® Data Center Attestation Primitives," *Product Brief*, 2018.

[9] M. Lipp, D. Gruss, R. Spreitzer, C. Maurice, and S. Mangard, "ARMageddon: Cache attacks on mobile devices," in *25th USENIX Security Symposium*, 2016, pp. 549–564.

[10] P. Kocher, J. Horn, A. Fogh, D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, M. Schwarz, and Y. Yarom, "Spectre attacks: Exploiting speculative execution," in *2019 IEEE Symposium on Security and Privacy*, 2019, pp. 1–19.

[11] M. Lipp, M. Schwarz, D. Gruss, T. Prescher, W. Haas, A. Fogh, J. Horn, S. Mangard, P. Kocher, D. Genkin, Y. Yarom, and M. Hamburg, "Meltdown: Reading kernel memory from user space," in *27th USENIX Security Symposium*, 2018, pp. 973–990.

[12] S. van Schaik, M. Minkin, A. Kwong, D. Genkin, and Y. Yarom, "Cacheout: Leaking data on intel cpus via cache evictions," in *2021 IEEE Symposium on Security and Privacy*, 2021, pp. 339–354.

[13] M. Lipp, A. Kogler, D. F. Oswald, M. Schwarz, C. Easdon, C. Canella, and D. Gruss, "PLATYPUS: software-based power side-channel attacks on x86," in *42nd IEEE Symposium on Security and Privacy*, 2021, pp. 355–371.

[14] A. Tang, S. Sethumadhavan, and S. J. Stolfo, "CLKSCREW: exposing the perils of security-oblivious energy management," in *26th USENIX Security Symposium*, 2017, pp. 1057–1074.

[15] A. Cui and R. Housley, "BADFET: defeating modern secure boot using second-order pulsed electromagnetic fault injection," in *11th USENIX Workshop on Offensive Technologies*, 2017.

[16] P. Qiu, D. Wang, Y. Lyu, and G. Qu, "VoltJockey: Breaching trustzone by software-controlled voltage manipulation over multi-core frequencies," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 195–209.

[17] R. Buhren, H. N. Jacob, T. Krachenfels, and J. Seifert, "One Glitch to Rule Them All: Fault Injection Attacks Against AMD's Secure Encrypted Virtualization," in *2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021, pp. 2875–2889.

[18] S. van Schaik, A. Kwong, D. Genkin, and Y. Yarom, "SGAxe: How SGX Fails in Practice," https://sgaxe.com/files/SGAxe.pdf, 2021.

[19] S. van Schaik, A. Seto, T. Yurek, A. Batori, B. AlBassam, C. Garman, D. Genkin, A. Miller, E. Ronen, and Y. Yarom, "SoK: SGX.Fail: How stuff get eXposed," https://sgx.fail, 2022.

[20] Google Cloud, "Remote attestation of disaggregated machines," https://cloud.google.com/docs/security/remote-attestation, 2022.

[21] Intel, "Remote Attestation," https://www.intel.com/content/www/us/en/developer/tools/software-guard-extensions/attestation-services.html, 2023.

[22] G. Chen, Y. Zhang, and T.-H. Lai, "OPERA: Open Remote Attestation for Intel's Secure Enclaves," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. ACM, Nov. 2019, pp. 2317–2331.

[23] C. Dukes, "Committee on national security systems (cnss) glossary," *CNSSI, Fort 1322 Meade, MD, USA, Tech. Rep*, vol. 1323, pp. 1324–1325, 2015.

[24] J. T. F. T. Initiative *et al.*, *SP 800-39. managing information security risk: Organization, mission, and information system view*. National Institute of Standards & Technology, 2011.

[25] K. L. Dempsey, N. S. Chawla, L. A. Johnson, R. Johnston, A. C. Jones, A. D. Orebaugh, M. A. Scholl, and K. M. Stine, "SP 800-137. information security continuous monitoring (ISCM) for federal information systems and organizations," 2011.

[26] Microsoft, "Cloud-native resiliency," https://learn.microsoft.com/en-us/dotnet/architecture/cloud-native/resiliency, 2022.

[27] ——, "Azure platform resiliency," https://learn.microsoft.com/en-us/dotnet/architecture/cloud-native/infrastructure-resiliency-azure, 2022.

[28] Google cloud, "Infrastructure design for availability and resilience," https://services.google.com/fh/files/misc/infrastructure_design_for_availability_and_resilience_wp.pdf, 2020.

[29] Mezzalira, Luca and Hyatt, Laura and Denti, Vittorio and Jaupaj, Zamira, "Let's Architect! Creating resilient architecture," https://aws.amazon.com/cn/blogs/architecture/lets-architect-creating-resilient-architecture/, 2022.

[30] H. Min, "Blockchain technology for enhancing supply chain resilience," *Business Horizons*, vol. 62, no. 1, pp. 35–45, 2019.

[31] The Department of Homeland Security (DHS) Science and Technology Directorate (S&T) , "Resilient Positioning, Navigation, and Timing (PNT) Reference Architecture," https://www.dhs.gov/science-and-technology/publication/resilient-pnt-reference-architecture, 2022.

[32] A. Galanou, F. Gregor, R. Kapitza, and C. Fetzer, "MATEE: Multimodal attestation for trusted execution environments," in *Proceedings of the 23rd ACM/IFIP International Middleware Conference*, Nov. 2022, pp. 121–134.

[33] D. Moghimi, B. Sunar, T. Eisenbarth, and N. Heninger, "TPM-FAIL: TPM meets timing and lattice attacks," in *29th USENIX Security Symposium (USENIX Security 20)*, 2020, pp. 2057–2073.

[34] H. N. Jacob, C. Werling, R. Buhren, and J.-P. Seifert, "faultpm: Exposing amd ftpms' deepest secrets," 2023.

[35] L. Petzi, A. E. B. Yahya, A. Dmitrienko, G. Tsudik, T. Prantl, and S. Kounev, "SCRAPS: Scalable Collective Remote Attestation for Pub-Sub IoT Networks with Untrusted Proxy Verifier," in *31st USENIX Security Symposium*, 2022, pp. 3485–3501.

[36] J. Park and Kwangjo Kim, "TM-Coin: Trustworthy management of TCB measurements in IoT," in *IEEE International Conference on Pervasive Computing and Communications Workshops*, 2017, pp. 654–659.

[37] S. F. J. J. Ankergård, E. Dushku, and N. Dragoni, "PERMANENT: Publicly Verifiable Remote Attestation for Internet of Things Through Blockchain," *Foundations and Practice of Security*, vol. 13291, pp. 218–234, 2022.

[38] T. Abera, R. Bahmani, F. Brasser, A. Ibrahim, A.-R. Sadeghi, and M. Schunter, "DIAT: Data Integrity Attestation for Resilient Collaboration of Autonomous Systems," in *Proceedings 2019 Network and Distributed System Security Symposium*, 2019.

[39] A. Ibrahim, A. Sadeghi, G. Tsudik, and S. Zeitouni, "DARPA: device attestation resilient to physical attacks," in *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks (WISEC)*, 2016, pp. 171–182.

[40] R. Pappu, "Physical One-Way Functions," Ph.D. dissertation, MIT, Sep. 2002.

[41] R. Canetti, "Universally composable security: A new paradigm for cryptographic protocols," in *Proceedings 42nd IEEE Symposium on Foundations of Computer Science*. IEEE, 2001, pp. 136–145.

[42] G. Chen and Y. Zhang, "MAGE: Mutual Attestation for a Group of Enclaves without Trusted Third Parties," in *31st USENIX Security Symposium*, 2022, pp. 4095–4110.

[43] C. Shepherd, K. Markantonakis, and G. Jaloyan, "LIRA-V: lightweight remote attestation for constrained RISC-V devices," in *IEEE Security and Privacy Workshops*, 2021, pp. 221–227.

[44] Apple Platform Security, "Secure Enclave," https://support.apple.com/en-gb/guide/security/sec59b0b31ff/web, 2021.

[45] Samsung Developers, "SAMSUNG TEEGRIS," https://developer.samsung.com/teegris/overview.html, 2019.

[46] Intel Agilex Device Security User Guide, "Intrinsic ID Physically Unclonable Function (PUF)," https://www.intel.com/content/www/us/en/docs/programmable/683823/21-3/intrinsic-id-physically-unclonable-function.html, 2021.

[47] NXP, "LPC5500 Series," https://www.nxp.com/products/processors-and-microcontrollers/arm-microcontrollers/general-purpose-mcus/lpc5500-arm-cortex-m33:LPC5500_SERIES, 2021.

[48] M. Bellare, R. Canetti, and H. Krawczyk, "Keying hash functions for message authentication," in *16th Annual International Cryptology (CRYPTO)*, ser. Lecture Notes in Computer Science, vol. 1109, 1996, pp. 1–15.

[49] Google cloud, "Cloud Functions," https://cloud.google.com/functions, 2023.

[50] ——, "BeyondCorp: A new approach to enterprise security." https://cloud.google.com/beyondcorp, 2023.

[51] Cybersecurity and I. S. Agency, "Trusted Internet Connections 3.0," https://www.cisa.gov/sites/default/files/publications/CISA%20TIC%203.0%20Traditional%20TIC%20Use%20Case.pdf, 2021.

[52] NXP, "High Efficiency Arm Cortex-M33-Based Microcontroller Family," https://www.nxp.com/products/processors-and-microcontrollers/arm-microcontrollers/general-purpose-mcus/lpc5500-arm-cortex-m33/high-efficiency-arm-cortex-m33-based-microcontroller-family:LPC55S6x, 2023.

[53] P. H. Nguyen, D. P. Sahoo, C. Jin, K. Mahmood, U. Rührmair, and M. van Dijk, "The Interpose PUF: Secure PUF design against state-of-the-art machine learning attacks," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 243–290, 2019.

[54] K. Olson, M. Bowman, and J. Mitchell, "Sawtooth: An Introduction," https://www.hyperledger.org/wp-content/uploads/2018/01/Hyperledger_Sawtooth_WhitePaper.pdf, 2018.

[55] C. Dobraunig, M. Eichlseder, F. Mendel, and M. Schläffer, "ASCON v1. 2: Lightweight authenticated encryption and hashing," *Journal of Cryptology*, vol. 34, no. 3, p. 33, 2021.

[56] Y. Zheng, W. Liu, C. Gu, and C.-H. Chang, "PUF-based Mutual Authentication and Key Exchange Protocol for Peer-to-Peer IoT Applications," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–18, 2022.

[57] Trusted Computing Group, "TPM 2.0 Keys for Device Identity and Attestation," https://trustedcomputinggroup.org/wp-content/uploads/TPM-2p0-Keys-for-Device-Identity-and-Attestation_v1_r12_pub10082021.pdf, 2021.

[58] AMD Xilinx, "Physically Unclonable Function," https://docs.xilinx.com/r/en-US/am011-versal-acap-trm/Physically-Unclonable-Function?tocId=8Lcvsmt7Vkdglr~m8UuUeg, 2022.

[59] "Information security, cybersecurity and privacy protection — Physically unclonable functions — Part 1: Security requirements," International Organization for Standardization, Standard, Dec. 2020.

[60] "Information security, cybersecurity and privacy protection — Physically unclonable functions — Part 2: Test and evaluation methods," International Organization for Standardization, Standard, May 2022.

[61] V. Costan, I. A. Lebedev, and S. Devadas, "Sanctum: Minimal hardware extensions for strong software isolation," in *25th USENIX Security Symposium, USENIX Security 16*, 2016, pp. 857–874.

[62] D. Lee, D. Kohlbrenner, S. Shinde, K. Asanović, and D. Song, "Keystone: An Open Framework for Architecting Trusted Execution Environments," in *Fifteenth EuroSys Conference*, 2020, pp. 38:1–38:16.

[63] J. Yao, "Intel® Trust Domain Extensions," https://www.intel.com/content/www/us/en/developer/articles/technical/intel-trust-domain-extensions.html, 2023.

[64] R. Buhren, C. Werling, and J.-P. Seifert, "Insecure Until Proven Updated: Analyzing AMD SEV's Remote Attestation," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 1087–1099.

[65] I. De Oliveira Nunes, S. Jakkamsetti, N. Rattanavipanon, and G. Tsudik, "On the TOCTOU Problem in Remote Attestation," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021, pp. 2921–2936.

[66] I. D. O. Nunes, K. Eldefrawy, N. Rattanavipanon, M. Steiner, and G. Tsudik, "VRASED: A Verified Hardware/Software Co-Design for Remote Attestation," in *28th USENIX Security Symposium*, 2019, pp. 1429–1446.

[67] I. D. O. Nunes, K. Eldefrawy, N. Rattanavipanon, and G. Tsudik, "APEX: A verified architecture for proofs of execution on remote devices under full software compromise," in *29th USENIX Security Symposium*, Aug. 2020, pp. 771–788.

[68] S. Surminski, C. Niesler, F. Brasser, L. Davi, and A.-R. Sadeghi, "RealSWATT: Remote Software-based Attestation for Embedded Devices under Realtime Constraints," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021, pp. 2890–2905.

[69] Z. Sun, B. Feng, L. Lu, and S. Jha, "OAT: Attesting Operation Integrity of Embedded Devices," in *2020 IEEE Symposium on Security and Privacy*, 2020, pp. 1433–1449.

[70] J. Wang, Y. Wang, A. Li, Y. Xiao, R. Zhang, W. Lou, Y. T. Hou, and N. Zhang, "ARI: Attestation of Real-time Mission Execution Integrity," in *31st USENIX Security Symposium*, 2023.

[71] S. Schulz, A.-R. Sadeghi, and C. Wachsmann, "Short paper: Lightweight remote attestation using physical functions," in *Proceedings of the Fourth ACM Conference on Wireless Network Security*, 2011, pp. 109–114.

[72] Ü. Kocabas, A. R. Sadeghi, C. Wachsmann, and S. Schulz, "Poster: Practical embedded remote attestation using physically unclonable functions," in *Proceedings of the 18th ACM Conference on Computer and Communications Security*, 2011, pp. 797–800.

[73] J. Kong, F. Koushanfar, P. K. Pendyala, A.-R. Sadeghi, and C. Wachsmann, "PUFatt: Embedded Platform Attestation Based on Novel Processor-Based PUFs," in *Proceedings of the 51st Annual Design Automation Conference*, 2014, pp. 1–6.

[74] M. N. Aman, M. H. Basheer, S. Dash, J. W. Wong, J. Xu, H. W. Lim, and B. Sikdar, "Hatt: Hybrid remote attestation for the internet of things with high availability," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7220–7233, 2020.

[75] U. Javaid, M. N. Aman, and B. Sikdar, "Defining trust in IoT environments via distributed remote attestation using blockchain," in *Mobihoc 20: ACM International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing*, 2020, pp. 321–326.

[76] H. R. Ghaeini, M. Chan, R. Bahmani, F. Brasser, L. Garcia, J. Zhou, A.-R. Sadeghi, N. O. Tippenhauer, and S. Zonouz, "PAtt: Physics-based Attestation of Control Systems," in *22nd International Symposium on Research in Attacks, Intrusions and Defenses, RAID 2019*, 2019, pp. 165–180.

[77] M. van Dijk, D. Gurevin, C. Jin, O. Khan, and P. H. Nguyen, "Autonomous secure remote attestation even when all used and to be used

digital keys leak," *Cryptology ePrint Archive*, 2021, https://eprint.iacr.org/2021/602.pdf.
[78] M. Umar, S. H. Islam, K. Mahmood, S. Ahmed, Z. Ghaffar, and M. A. Saleem, "Provable Secure Identity-Based Anonymous and Privacy-Preserving Inter-Vehicular Authentication Protocol for VANETS Using PUF," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 11, pp. 12 158–12 167, 2021.
[79] T. Alladi, Naren, G. Bansal, V. Chamola, and M. Guizani, "Secauthuav: A novel authentication scheme for UAV-ground station and UAV-UAV communication," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 12, pp. 15 068–15 077, 2020.
[80] X. Zhang, D. Gu, T. Wang, and Y. Huang, "Old School, New Primitive: Towards Scalable PUF-based Authenticated Encryption Scheme in IoT," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1–1, 2023.
[81] U. Chatterjee, V. Govindan, R. Sadhukhan, D. Mukhopadhyay, R. S. Chakraborty, D. Mahata, and M. M. Prabhu, "Building PUF Based Authentication and Key Exchange Protocol for IoT without Explicit CRPs in Verifier Database," *IEEE Transactions on Dependable and Secure Computing*, vol. 16, no. 3, pp. 424–437, May 2019.
[82] U. Chatterjee, D. Mukhopadhyay, and R. S. Chakraborty, "3PAA: A Private PUF Protocol for Anonymous Authentication," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 756–769, 2021.
[83] M. A. Qureshi and A. Munir, "PUF-RAKE: A PUF-Based Robust and Lightweight Authentication and Key Establishment Protocol," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 4, pp. 2457–2475, Jul. 2022.
[84] N. Asokan, F. Brasser, A. Ibrahim, A.-R. Sadeghi, M. Schunter, G. Tsudik, and C. Wachsmann, "SEDA: Scalable embedded device attestation," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, 2015, pp. 964–975.
[85] X. Carpent, K. ElDefrawy, N. Rattanavipanon, and G. Tsudik, "Lightweight Swarm Attestation: A Tale of Two LISA-s," in *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*. ACM, 2017, pp. 86–100.
[86] M. Ambrosin, M. Conti, A. Ibrahim, G. Neven, A.-R. Sadeghi, and M. Schunter, "SANA: secure and scalable aggregate network attestation," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016, pp. 731–742.
[87] B. Kuang, A. Fu, Y. Gao, Y. Zhang, J. Zhou, and R. H. Deng, "FeSA: Automatic Federated Swarm Attestation on Dynamic Large-Scale IoT Devices," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–16, 2022.
[88] F. Kohnhäuser, N. Büscher, and S. Katzenbeisser, "A practical attestation protocol for autonomous embedded systems," in *2019 IEEE European Symposium on Security and Privacy*, 2019, pp. 263–278.
[89] M. Bampatsikos, C. Ntantogian, C. Xenakis, and S. C. A. Thomopoulos, "BARRETT: BlockchAin Regulated REmote aTTestation," in *IEEE/WIC/ACM International Conference on Web Intelligence*, 2019, pp. 256–262.

Xiaolin Zhang received from Xidian University of China, Xi'an, his B.S. degree in cyber security in 2020. He is currently a Ph.D. candidate under successive postgraduate and doctoral program at School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai. His research interests include design of PUF-based security schemes and cryptography. He has published scientific papers on IEEE TCAD.

Kailun Qin is currently a DEng candidate at School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University (SJTU), Shanghai. He obtained his master's degrees respectively from IMT Atlantique (France) and Nanjing University of Science and Technology (China) in 2017. His research interests include confidential computing, trusted execution environment and system security. He has been an active speaker at multiple cloud and security industry conferences.

Shipei Qu received from Xidian University of China, Xi'an, his B.S. degree in Electronic Engineering in 2020. He is currently a Ph.D. candidate at School of Electronic Information and Electrical Engineering,Shanghai Jiao Tong University, Shanghai. His research interests include hardware security and cryptography.

Tengfei Wang received the B.E. degree and the M.E. degree from Tianjin University, Tianjin, China, in 2016 and in 2019, respectively. He is currently pursuing the D.E degree from the School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai. His current research interests include circuit and system designs for cryptosystems. He has published scientific papers on IEEE TVLSI and other academic journals.

Chi Zhang is currently a research associate at School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai. He received the B.S. degree in computer science and technology from Southeast University in 2014, and the Ph.D. degree in computer science and technology from Shanghai Jiao Tong University in 2022. His research interests include cryptographic engineering, hardware security such as side-channel analysis and fault inject analysis, and cryptography. He has published scientific papers on TCHES, IEEE TVLSI, TIFS and DAC, AsianHost and other academic journals and conferences.

Dawu Gu is a distinguished professor at School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University (SJTU), Shanghai. He received from Xidian University of China, Xi'an, his B.S. degree in applied mathematics in 1992, and his M.S. degree in 1995 and Ph.D. degree in 1998 both in cryptography. His research interests include crypto algorithms, crypto engineering, and system security. He was the winner of Chang Jiang Scholars Distinguished Professors Program in 2014 by Ministry of Education of China. He won the National Award of Science and Technology Progress in 2017. He is the vice president of the Chinese Association for Cryptologic Research (CACR). He has got over 200 scientific papers in top academic journals and conferences, such as CRYPTO, IEEE S&P, ACM CCS, NDSS, TCHES. He also served as PC members of international conferences such as ASIACRYPT, ASIACCS, ACNS, ISC, ISPEC, ICICS, Globecom, IEEE TrustCom, and NSS for more than 30 times.