1 SANA: Secure and Scalable Aggregate Network Attestation

1.1 Introduction

1.1.1 Overview

Embedded devices are often security and privacy critical, because they possibly have consequences in the physical world. One common attack is to modify or replace a device's firmware, as part of a larger attack scenario. Safe and secure operation of a device needs to be ensured to prevent this type of attack. It is important to guarantee its software integrity, e.g., via remote software attestation. Remote software attestation is an interactive protocol that allows a prover to prove its software integrity to a remote verifier. This is usually achieved by signing integrity-protected memory regions. Attestation of individual smart devices is a well established research area. However, to date there is a lack of viable approaches to securely scale device attestation to a very large number of devices:

1.1.2 Goals

1. scalability i.e., it efficiently verifies the integrity of a large collection of devices by means of a novel signature scheme, which allows aggregation of attestation proofs.
2. public verifiability, i.e., the produced aggregate signature can be verified by any one knowing the (aggregate) public key.
3. enabled untrusted aggregation, i.e., compromise (including physical tampering) of aggregating nodes does not affect the integrity of the attestation process.

1.1.3 Approach

- A novel signature scheme, called Optimistic Aggregate Signature (OAS) allows the aggregation of signatures on different attestation responses, while having a verification overhead that is constant in the size of the network.
- SANA is the first collective attestation scheme for networks of embedded devices that supports high dynamicity and adheres to common assumptions made in single-prover attestation. SANA leverages OAS over aggregation trees to provide highly scalable attestation of large device populations, in a single round-trip.
- The performance of SANA is analyzed on three state-of-the-art security architectures for low-end embedded devices (e.g., SMART, TrustLite, and TyTAN), for networks of up to 1, 000, 000 devices, in order to demonstrate its scalability.

1.2 Background & Related Work

1.2.1 Collective Attestation

SEDA, made a first step towards a collective attestation, i.e., the scalable attestation of large groups of interconnected devices. The main focus of SEDA is efficiency and applicability to low-end devices, rather than security in the presence of a strong adversary. Our proposed collective attestation protocol SANA overcomes the limitations of SEDA by

- Requiring minimal trust anchor in hardware only for the attested devices.
- Allowing aggregation to be performed by largely untrusted nodes, which are only required for availability.
- Limiting the effect of successful attacks on the hardware of an attested device to the device itself, i.e., it will not affect the attestation of other devices.

Similarly, Denial-of-Service attacks on one device in SANA will not affect other devices. Finally, SANA informs the verifier with ids as well as software configurations of the devices that failed attestation.

1.2.2 Aggregate Multi-Signatures

A new signature scheme Optimistic Aggregate Signature (OAS) is proposed, that

- Allows signatures on distinct messages to be aggregated.
- Provides a signature verification algorithm that is constant in the number of signers.

The communication overhead of the scheme is linear in the number of different messages, while the computational overhead is linear in the number signers who signed a different message than the default one. However, this number is assumed limited. Finally, we present a pairings-based construction of OAS, and combine it with aggregation trees, providing unlimited scalability.

1.3 Design

1.3.1 System Model

SANA is a protocol on network (G) between the following logical entities: prover (P), aggregator (A), owner (O), and verifier (V). A prover composes a proof of integrity of its software configuration, which is sent to a remote verifier. Provers can have different software and hardware configuration. However, we expect the majority of them to have a good software configuration (Good provers). An aggregator has the purpose of relaying messages between entities, and collecting and aggregating attestation responses

from provers, or other aggregators. The entity O represents the network owner or operator, responsible for the deployment, as well as the maintenance, of every prover Prover in G. A physical device in G can embed the functionalities of every logical component described above, or a combination of them.

1.3.2 Protocol

Each prover is initialized with the cryptographic material needed to execute SANA collective. The initialization is performed in a secure environment, and preferably, but not necessarily, by O. At a given time, a verifier V, which possesses an appropriate attestation token generated by O, may attest G. Note that, if V and O are two distinct entities, the token is securely exchanged offline. In order to attest the network, V chooses an aggregator and sends it an attestation request containing an attestationtoken. The request is flooded in the network forming a logical aggregation tree, that has provers as leaf nodes, and aggregators as intermediate nodes. Leaf nodes of the aggregation tree, i.e., provers create their attestation response and send it to their parent nodes. Aggregators, i.e., non-leaf nodes, in turn, aggregate the attestation responses received from their child nodes, and forward the result to their parents. Finally, the aggregated report is received and verified by V.

1.3.3 Requirements

- Unforgeability and Freshness: If the attestation hardware of a prover is unchanged and a correct verifier was able to validate the aggregate attestation result including a given prover, then the claimed integrity measurement reflects an actual software configuration of this prover at a time during this protocol run.
- Completeness: If the attestation hardware of provers is unchanged and a correct verifier was able to validate the aggregate attestation result for a given set of provers, then all provers actually reported their software configuration in the given protocol run.
- Scalablity: The protocol allows a verifier to attest a large network of devices. The communication and computational complexity for prover and verifier must be at most logarithmic in the number of devices in the collection.
- Public Verifiability: In a public key domain, the collective attestation evidence collected by a verifier can be verified by any party. In this case, the Unforgeability requirement only proves the state of the prover within the time window between generation of the challenge by the owner, and the receipt of the evidence from the verifier.
- Privacy Preservation: Verification does not require detailed knowledge of the configuration of G (e.g., its topology).

- Heterogeneity: The protocol is applicable to networks with heterogeneous devices. The scheme can use any integrity measurement mechanism used by devices in G.
- Availability: If all participants are honest and the network is available then the protocol produces collective attestation evidence.
- Limiting DoS: It should not be possible to run a global DoS attack over the whole network through one device.

1.3.4 Assumptions

It is assumed that all provers in G correctly implement the minimal hardware features required for secure remote attestation. A potential implementation of P could have: a Read Only Memory (ROM) that stores the protocol code and the related cryptographic key(s), and a simple Memory Protection Unit (MPU), that restricts access to cryptographic key(s) to protocol code only, and ensures secrecy of the key(s) through non-interruptible, and clean execution of the protocol code. The assumption is made that the owner O to be trusted. Finally, all cryptographic schemes used in our protocol are assumed to be secure.

1.3.5 Implementation

SANA was implemented on TyTAN as isolated tasks, which are protected via secure boot. Further, the MPU was configured such that only SANA's tasks can access the protocols secret data. For example, according to rule #2 in the MPU table, the OAS secret key is only read accessible to createResponse(). TyTAN is a security architectures for embedded systems, that is based on TrustLite. TyTAN provides hardware-assisted isolation of system components with real-time execution. Isolation is fundamental to protect critical components against unintended access by other potentially malicious components. In TyTAN, a Memory Protection Unit restricts access to data, to the task that owns this data. Moreover, both authenticity and confidentiality of the tasks' code and data are based on secure boot.

1.4 Results & Evaluation

1.4.1 Communication Overhead

A token T with z good configurations consists of 20z + 58 bytes, and a challenge Ch of 20z + 78 bytes. A response α has size 32 + 32w + 20m bytes, where m is the number of distinct bad software configurations and w is the number of distinct OAS public keys of bad provers. The communication overhead of the each aggregator is, sending at most 32 + (20z + 78)g + 32w + 20m bytes and receiving at most 20z + 78 + 32g + 32w + 20m bytes where g is the number of neighbours. Finally, every prover P sends 84 bytes and receives 20z + 78 bytes.

1.4.2 Memory Cost

Each Prover in G stores the ids and values of s counters, its OAS secret key, its identity certificate, and the public key O. The storage overhead for every P is estimated as 10s + 228 bytes, where s is the number of counters used by O.

1.4.3 Performance

The aggregation tree approach allows provers, and aggregators on the same depth of the tree, to perform their computations in parallel. However, the OAS signature aggregation at depth d depends on the signature creation computations at depth d + 1. Consequently, the overall run-time of the SANA depends on the depth ($d = f(n + a) \in O(log(n + a))$) of the aggregation tree generated for the graph of the network, the number of neighbors of each aggregator, and the number of bad provers. The run-time t of SANA is estimated as

$$t =\leq \left[110d + \sum_{i=0}^{d} (32w + 20m)\right] \times t_{tx} + \sum_{i=0}^{d} p \times t_{agg} + d \times (t_{ver} + t_{hash}) + t_{sign} \quad (1)$$