# Attestation: A Taxonomy and Evaluation

WILL THOMAS, The University of Kansas, USA

Remote attestation is a method through which principals may exchange information in order to establish trust. Many varying methods for achieving this goal have been proposed and explored, and the applications of attestation are extremely far reaching. For this reason, diverging nomenclature has appeared, along with a general lack of an overall attestation view across the many sub-classifications of attestation. In this review, we will attempt to unify the nomenclature used for attestation, as well as establish a general taxonomy for attestation methods. Further, we will summarize the prominent current methods for attestation and review the strengths and weaknesses of each, while exploring typical attestation use cases.

CCS Concepts: • **Security and privacy** → *Trust frameworks.*

Additional Key Words and Phrases: Attestation, Trust, Verification

**ACM Reference Format:**
Will Thomas. 2023. Attestation: A Taxonomy and Evaluation. 1, 1 (December 2023), 14 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

## 1 INTRODUCTION

In this paper, we will attempt to set-up a general framework for describing and categorizing attestation, as well as a more in-depth exploration of the benefits and limitations of the specific attestations paradigms. In order to achieve this goal, we must start with some common definitions necessary to describe nearly all attestation scenarios.

*Definition 1.1 (**Attestation**).* **Attestation** is the process of producing a verifiable report of evidence from a system as presented by Coker et al. [10], Haldar et al. [14], Sadeghi and Stüble [36].
The primary motivation behind attestation is not as a malware preventive measure[1], but rather a detection method to allow for the revocation of trust from a corrupted principal.

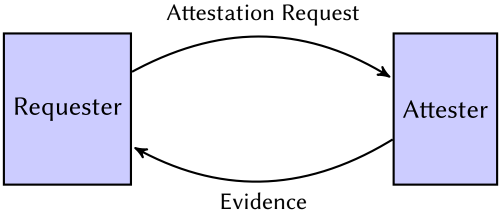

Fig. 1. Remote attestation.

[1] as Loscocco et al. [22] established perfect prevention is impossible

Author's address: Will Thomas, The University of Kansas, Institute for Information Sciences, 2335 Irving Hill Rd, Lawrence, KS, 66045, USA, 30wthomas@ku.edu.

XXXX-XXXX/2023/12-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Attestation has additionally been referred to as “semantic remote attestation”, which represents that the process of attestation will involve two main (typically separate/remote) principals that exchange evidence regarding the semantics/behaviors of the system, ultimately to establish trust in the faithful execution of some semantics by a certain principal (the **Attester**):

*Definition 1.2 (**Attestation Request**).* The Attestation Request (Fig 1) is a request made from the Requester to the Attester asking for attestation to take place. The Attester then returns evidence to the Requester.

*Definition 1.3 (**Requester**).* The **Requester** is the principal that sends Attestation Request’s to some Attester.
The Requester will also have some method of validating/verifying that the returned Evidence from the Attestation Request is valid. For this reason, the Requester is also on occassion referred to as the **Verifier** or **Appraiser**.

*Definition 1.4 (**Attester**).* The **Attester** is the principal that will receive Attestation Request’s from a Requester, process them, and return evidence to the Requester.
Other word frequently used to refer to this principal are: **Prover** and occassionally **Target**. The usage of **Target** is only appropriate when the Attester is itself the target of an Attestation Request, and not just a servicing principal that performs **Measurement** on some other entity.

Although in many cases, the Requester will also be the principal that requires evidence about a Target system, there may be an additional principal involved if the Attestation Request to the Attester is made through a Third Party. In this case, we need a **Relying Party**.

*Definition 1.5 (**Relying Party**).* The **Relying Party** is a principal who wishes to access the evidence report regarding an Attester for the purpose of making a **Trust Decision**.

*Definition 1.6 (**Trust Decision**).* Martin et al. [24] established a group of properties that will help establish “trust” in an principal.
These properties are:

- strong identity
- composition of “good” parts
- observation of “good” behavior

The definition of “good” must remain ambigious when we analyze principals from an abstract level, and the determination of “good” is left as a problem for measurement and the Attestation System Administrator.
Many times in Attestation however, the nuances of these requirements and the various ways to establish trust are quite complicated. These properties and “trust” in general are explored further in Section 4: Attestation Requirements

### 1.1 Attestation Types

Depending on the relationships between the Relying Party, Requester, and Attester; the type of Attestation can change. The 5 main possible relationships are depicted in Figure 2, where $X, Y, Z$ represent different principals in an attestation.

| Relying Party | Requester | Attester | Attestation Type |
|---|---|---|---|
| $X$ | $X$ | $X$ | Self-Attestation/Auditing |
| $X$ | $X$ | $Y$ | Direct Attestation |
| $X$ | $Y$ | $Y$ | Strong Report Attestation |
| $X$ | $Y$ | $X$ | Third-Party Auditing |
| $X$ | $Y$ | $Z$ | Third-Party Attestation |

Fig. 2. Table of Attestation Types

Note that an obvious extension to these types would involve when the Attester and Target are not one and the same. This case usually would be considered a form of **Delegated Attestation**, but more nuance may apply. Extending this classification is left to future work.

*1.1.1 Self-Attestation/Auditing.* In **Self-Attestation/Auditing**, the same principal is the Relying Party, Requester, and Attester. This represents a special case of Attestation that is not remote at all. In Self-Attestation, evidence is produced by the principal as a means to verify to itself or a possible end-user that the system is not compromised.

This form of attestation in practice is typically not recognized as attestation and would best be compared to a form of malware detection or anti-virus software that runs on a computer.

*1.1.2 Direct Attestation.* In **Direct Attestation**, the Relying Party and the Requester are the same principal, and the Attester is separate. This is a typical attestation use-case where the Relying Party has the ability to Appraise/Verify the Attestation Evidence returned by the Attester.

*1.1.3 Strong Report Attestation.* **Strong Report Attestation** is a type of Attestation where a Trusted Component of the single principal comprising the Requester and Attester is able to gather evidence as to that principals state, reliably verify that evidence, and serve reports to any Relying Party.

This type of Attestation is in practice very difficult to realize as having a high-level Trusted Component of a principal that can both gather and verify evidence securely is difficult to construct.

*1.1.4 Third-Party Auditing.* **Third-Party Auditing** is very similar to Self-Attestation, except the Verification of the evidence the principal gathers is left to a Third-Party entity rather than itself.

This is another conceivable method where the actual attestation process is very similar to a malware detection process that utilizes a Third-Party database to identify adversary actions.

*1.1.5 Third-Party Attestation.* In **Third-Party Attestation**, all principals are distinct. However, a certain level of implicit trust must exist between the Relying Party and the Requester to believe that the report the Requester generates for consumption by the Relying Party is a faithful representation of the state of the evidence provided by the Attester. This implicit trust can be resolved to some extent by layering attestation protocols (Section 3.2.1: Layered Attestation).

Third-Party Attestation is very similar to Direct Attestation, except for the Relying Party either lacks the ability to Appraise evidence provided by the Attester, or the Attester does not trust the Relying Party with its evidence, but will trust the Requester to Appraise it.

Ultimately, this is one of the most flexible types of attestation, and any general purpose attestation protocol must support Third-Party Attestation.

## 2 CLASSIFICATION

There are two main classes of Attestations: **Software** and **Hardware**, along with an additional **Hybrid** sub-class. The primary distinquishing difference between these two classes is the existence/utilization of a **Trusted Hardware Component**.

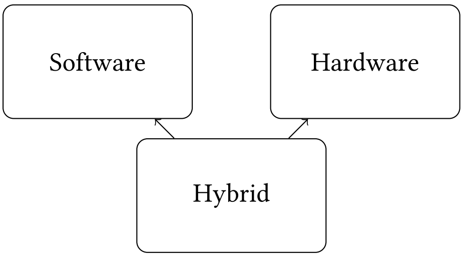

Fig. 3. Attestation Class hierarchy

*Definition 2.1 (**Trusted Hardware Component**).* A hardware component (that is implicitly trusted to have been constructed correctly) that provides the following functions:

(1) A secure hardware private key
(2) A secure random number generator
(3) Tamper resistant memory storage
(4) Cryptographic functions

The Trusted Computing Group [39] in their specification for the Trusted Platform Module (TPM), include all these features.

### 2.1 Hardware Attestation

This class of attestation is is characterized by its requirement for a **Trusted Hardware Component** to establish a Hardware Root of Trust (RoT). This Root of Trust will utilize the tamper resistant memory storage to store attestation evidence, and then sign it with the secure hardware private key, and the provided cryptography functions. This ensures that this tamper-proof hardware component will capture evidence of any malicious actions and not be erasable without corrupting a presumed uncorruptible component of the system.

Hardware Attestation also proposes that the Root of Trust be used as a base from which additional trust decisions can be made moving up to a final Binary/Property/Behavior/etc. of a principal. This Chain of Trust concept interplays well with Layered Attestation presented later.

Hardware Attestation has proven particularly relevant to **Binary Based Attestation** as the tamper resistant memory storage of a Trusted Hardware Component provides a secure location to store evidence regarding a binary that cannot be erased by transient malware. Additionally, tamper resistant memory storage such as the Program Control Registers (PCRs) of the TPM can be used to verify secure boot occurs. A common extension to the definition of Trusted Hardware Component requires that boot is either securely completed, or boot evidence is recorded within the Trusted Hardware Component to be verified later.

Kühn et al. [20] demonstrated how a TPM can be used as a hardware root of trust. Using just a Trusted Computing Group (TCG) trusted hardware component (such as the TPM) a viable hardware based attestation system can be built, without other OS modifications.

## 2.2 Software Attestation

Software based attestation is a class of attestation that is useful for incorporating into legacy systems, or utilizing in cases where the ability for hardware support is limited (such as low-power embedded devices, IoT, etc.). Software based attestation has three main components:

(1) Does not require specialized hardware to complete attestation requests.
If specialized hardware is required for an attestation mechanism, it should be classified as hardware based attestation
(2) Relies on the timeliness of attestation request responses to establish trust in the attestation evidence.
This is implemented through a requirement that a specific, constant, amount of computational and memory resources (typically 100%) will be allocated to serving the attestation request as soon as the attester receives it. Thus the time for the attestation request to be served should be relatively constant as well. The verifer/requester should have an understanding of the expected time it will take for an attestation request to be served, and invalidate any evidence/responses that are returned after a time longer than the attestation should've taken. This helps thwart spoofing attacks where an adversary could utilize a cloned, uncorrupted version of the system they have taken over to generate good evidence in response to attestation requests. The issue of spoofing attacks is solved in hardware based attestation through the hardware root of trust signing attestation evidence; essentially the uncloneable secret key is used.
In order for spoofing attacks to take place in software based attestation, the adversary must be able to dispatch any attestation request to its uncorrupted "exemplar" system and respond with the evidence from that system just as quickly as it would take for it to do the attestation evidence gathering and computation on itself in real-time. This is generally considered to be highly improbably and thus trust in the attestation evidence is merited should it be returned in a timely fashion.
(3) Requires a direct communication method between the attester/prover and requester/verifer.
This requirement (although some research has proposed possible alternatives) is primarily enforced by requirement (2). The unreliability of a multiple hop/indirect communication method would make trusting software based attestation results too difficult, so it is generally considered required that direct communication be established.

Armknecht et al. [3] presents a strong background on software based attestation systems, demonstrating the inherent strengths in that it can be easily applied in many areas. The great weakness of software attestation schemes lies in the (typical) requirement of the software attestion service to hijack all computing and memory resources of the prover/attester to ensure the timely return of evidence to a relying party/verifier. This is because software attestation implicitly relies on a timing argument for it to be considered secure. The returned evidence package must be received within a certain time constraint, that is generally considered infeasible for an adversary to have falsified.

One great limitation of software based attestation is that it suffers from the Time-of-Check-Time-of-Use (TOCTOU) problem. Transient malware with the ability to erase itself poses a great problem for Software Attestation as without a hardware root of trust to track and preserve any record of malware interference, software attestation may return "good" evidence due to an inability to detect the malware that has erased its presence. De Oliveira Nunes et al. [12], Nunes et al. [26] all explore the primary issues of Software Attestation and propose solutions to the TOCTOU problem and dealing with a transient adversary.

A particular vulnerability of Software based attestation is Computation Denial of Service (CDoS). Bampatsikos et al. [4] propose using a Public Ethereum Network (PEN) to solve the CDoS vulnerability of software based attestation systems. As previous mentioned, software based attestation approaches require an extremely timely computation and return of the attestation evidence in order for the evidence to be verified and approved. This is typically achieved through shunting all computational and memory resources of the prover/attester towards the attestation request. This opens up software based attestation to an obvious DoS vulnerability if attestation requests are repeatedly made. The solution proposed by Bampatsikos et al. [4] with their system BARRETT is a PEN for which cryptocurrency must be mined for each attestation request, otherwise they will be dropped without any attestation evidence gathering or computation taking place. This is stronger than a nonce or timestamp approach that may thwart some, but not all CDoS attacks.

### 2.3 Hybrid Attestation

Hybrid attestation, as one might expect, combines features of both hardware based and software based attestation. Rather than the strict requirement that hardware not be utilized in the attestation process, some specific (likely trusted) hardware component will be used in hybrid attestation. However, this hardware component will not reach the level of complexity, trust, or power that the **Trusted Hardware Component** provides, or a pure hardware based attestation solution would be better. Some examples of hybrid based attestation systems have been assembled here:

- Javaid et al. [17] particularly utilized Physical Unclonable Functions (PUF) as a mechanism for hybrid attestation measurement; requring hardware support for PUFs, but software mechanism for verifying the integrity of the results. They additionally make use of a Blockchain as an attestation entity registration/identification system. The use of a blockchain as an identification server for attestation systems is a common theme that falls into the realm of **Explicit Identity Management** (similarly to a central **Certificate Authority** or other **Trusted Third Party**).
- Further hybrid attestation work is explored by Mondal et al. [25] in there system PReFeR, where instead of utilizing PUF's, Physically Related Functions (PRF's) are used. PRF's do not require hardware support for secure storage, and also will not require computationally intensive crypto operations. Since this is a hybrid attestation scheme, the timeliness of a response is critical for validating its integrity. These means that the computational intensivity of the attestation measurements matters a lot for performance reasons. Reducing the amount of computation that must occur dynamically through Hardware Performance Counters (HPCs) is a key feature of the PReFeR work that keeps it a hybrid solution requiring some hardware support, but mitigating many limitations of software attestation such as TOCTOU. Ultimately, PRF's also allow for easy non-Binary attestation methodologies, as PUF's will need a large database of possible acceptable Binary attestation results for verification, where as PRF's will not.

## 3 ATTESTATION METHODS

In this section we will explore some of the general methodologies used by attestation systems. These methodologies are a characterization of the measurement target that the attestation is being performed on. Differing methodologies offer different advantages: some are much easier to implement, but harder to verify; others have much more complex measurement processes, but offer greater flexibility when it comes to making trust decisions.

*Definition 3.1 (**Binary Based Attestation**).* **Binary Based Attestation** was the first, and most obvious, attestation method. Using a simple comparison between the hash of a running program binary, and comparing it to a "golden value" that the binary should have, you can tell if the running binary is the expected binary. Many of the early attestation definitions and systems (Brickell et al. [6], Gu et al. [13], IETF RATS Working Group [16]) implemented binary attestation. Despite the robustness of Binary Attestation, it is severely limited when viewed in light of modern constantly evolving and changing systems. Binary Attestation has one acceptable "golden value" that will be considered "good" during the appraisal/verification process. This limitation is addressed by the many alternative attestation methods that have been proposed following Binary Attestation.

*Definition 3.2 (**Direct Anonymous Attestation (DAA)**).* **Direct Anonymous Attestation (DAA)** is a method of **Binary, Hardware Attestation** first introduced by Brickell et al. [6] that specifically upholds the principle of anonymity. A **Trusted Execution Environment (TEE)** is established with the use of a **Trusted Platform Module (TPM)** (or extensibly any other **Trusted Hardware Component**). The TPM has a specific **Attestation Identity Key (AIK)** that can be verified through the use of a **Trusted Third Party (TTP)** that acts as a **Certificate Authority (CA)**. This form of attestation has extremely niche use cases, and manages to present Attestation Reports to Relying Partys allowing make trust decisions without disclosing the identity of the Attester.

*Definition 3.3 (**Privilege-Based Attestation**).* Rauter et al. [33] propose PRIBA (PRivilege-Based remote Attestation) as an extension of binary attestation. One of the great limitations of binary attestation is that knowledge of all possible "good" configurations of an attestation target/attester is extremely difficult. Instead, the good configurations of the attester are established through binary analysis of the end executable to be attested to. This executable is analyzed to determine "modules" that it depends on, in the form of system call APIs. These modules are then themselves attested to for their possible good configurations. If a binary utilizes modules that are themselves attested to and shown to be in a "good" state, then the overall binary is considered to be "good". In this way, PRIBA is an inductive approach to attestation, as the binary needs to be made of good parts to be verified. However, it does not quite meet the standards for Section 3.2.1: Layered Attestation that will be explored later.

*Definition 3.4 (**Property Based Attestation**).* **Property based attestation** was introduced by Sadeghi and Stüble [36] as a more general model of attestation, that would help solve some of the limitations of Binary Attestation. Rather than verifying a specific binary is running with specific configurations, property based attestation attempts to establish that a certain "property" of a system exists. For this reason, property based attestation has a wider range of acceptable evidence that will lend towards convincing a verifer/requester that the property they are looking to see in the attester actually exists.
This was further elaborated upon by Chen et al. [9] where they actually designed a general protocol for which property based attestation could occur.
Some examples of properties we would want to verify about a system are:

- Up to date software
- Running virus checker
- Kernel Integrity Checks (Loscocco et al. [23])

*Definition 3.5 (**Behavioral/Policy Attestation**).* Behavioral Attestation was a general high-level method for attestation introduced by Li et al. [21] for verifying that a system policy or behavior is being abided by.

*Definition 3.6 (**Model-Based Behavioral Attestation**).* This method for attestation is an extension of **Behavioral/Policy Attestation** introduced by Alam et al. [1]. They utilized Usage Control (UCON) as a model for attester policy, and then gather evidence that the UCON model is not violated. Specifically, Model-Based Behavioral Attestation identifies system behaviors and associations between executing processes. They are codified in a low-level policy language and as long as no events violate the provided policy, this is evidence of a "good" system.
Specifically, lack of evidence of events violating the model implies a "good" attester.

*Definition 3.7 (**Delegated Attestation**).* Delegated Attestation is a hybrid attestation method introduced by Ammar et al. [2] which involves augmenting an attestation system with a dedicated piece of attestation hardware. While each attester itself will not require specialized hardware support, their software will be modified to transmit all events to this dedicated "Attestation Proxy" that will manage attestation evidence securely.
This approach particularly thwarts TOCTOU issues that arise in software based attestation systems by adding the specialized hardware based attestation system. Transient malware will be recorded by the trusted hardware components of the Attestation Proxy and cannot be erased, even if the malware erases traces of itself from the original attester.

## 3.1 Attestation Timings

There are two main types of Attestation Timings:

(1) On-Demand:
The Attestation Request is serviced once it arrives and fresh evidence is gather at the time of attestation servicing

(2) Periodic:
This is typically completed with some form of self-measurement and attestation results caching. Particularly, attestation evidence is gather occassionally and stored in the memory of the attester. This evidence is then returned to the requester when an attestation request arrives.

## 3.2 Attestation Patterns

### *3.2.1 Layered Attestation.*

**Layered Attestation** is a general pattern for attestation that involves Attestation Service Providers (ASPs) that can execute either arbitrary measurements, or further attestation protocols themselves. The expressive power of ASPs is the key feature of layered attestation, as an the same attestation protocol may act differently depending on the target of its attestation.

It is worth noting that ASPs are not non-deterministic[2], but that they may take variable arguments based upon the target of attestation that will describe a targets structure; allowing for an inductive approach to be taken by all ASPs.

[2] Although nothing but the Appraisal/Verification process most likely necessitating determinicity would forbid it

Helble et al. [15] introduce a set of general patterns that attestation requests will likely take and establish the security benefits and limitations of each approach. These mechanisms all utilize layered attestation, and are themselves meant to be layered upon one another to create more complex attestation protocols.

In the same realm of property based attestation, Domain Specific Languages for attestation have been developed to help create attestation protocols that can achieve fine-grained selection of properties you wish to verify. Specifically, the Copland Consortium [11] developed Copland, a DSL meant for attestation, along with an accompanying verified Attestation Manager (Petz et al. [30]). Attestation DSL's allow for specific Attestation Service Providers (ASPs) to represent a measurement of a specific property.

Petz and Alexander [28] design and verify a Attestation Virtual Machine, as a running executable which can consume Attestation Requests in the form of Copland protocols.

Further, they demonstrate how the assumptions necessary for securely gathered attestation evidence via an Attestation Manager or Virtual Machine can be satisfied with a strict process isolation mechanism, such as the seL4 microkernel (Klein et al. [18]).

*3.2.2 Swarm Attestation.* Swarm attestation is an attestation pattern that will utilize many attesters in a parallel fashion to efficiently make trust decisions. Carpent et al. [7], Kuang et al. [19], Wedaj et al. [41] all explore using swarm attestation to efficiently attest to a large number of attesters with a single protocol.

The key characteristics of swarm attestation are:

- Multiple Attesters
- A propagating attestation request across all Attesters in the swarm
- A protocol that is more efficient to operate in parallel rather than individually.

Additionally, a swarm attestation will typically have a non-standard network topology where the connection between the Requester and Attesters does not necessarily need to be direct. If all Attesters are directly connected to the Requester, it may be better characterized as **Distributed Attestation**.

## 4 ATTESTATION REQUIREMENTS

Now that we have established a taxonomy of attestation, we will turn our attention towards the evaluation of attestation systems. In particular, we will review the evolving requirements of attestation systems that have been devised to secure them and the results they provide.

### 4.1 5 Principles of Attestation

One of the most pivotal works on the requirements for Remote Attestation was developed my Coker et al. [10]. In this work, Coker et al. outlined 5 principles that attestation frameworks must incorporate:

1. (Freshness) Evidence should reflect the system state at the time of evidence gathering
2. (Comprehensive Information) Attestation should have sufficient access to gather relevant evidence and evidence should provide a comprehensive view of the system.
3. (Constrained Disclosure) The attester/target should be able to control what information about their system is disclosed and to whom it is disclosed.
4. (Semantic Explicitness) Evidence should have a well-defined formal semantics to inform the trust judgement that the appraiser/verifier will undertake.

(5) (Trustworthy Mechanism) A fundamental part of trusting attestation evidence results is trusting the attestation infrastructure. Therefore, evidence of the “good” construction of the attestation infrastructure should be provided in evidence to the verifier.

The keen reader will notice that we have already presented examples of attestation frameworks that may abide by these principles but can still be easily exploited. One such example would be the case of software binary attestation, where the evidence reflects the state of a running executable at exactly measurement time. However, transient malware may be able to erase evidence of its presence then restore itself before a security critical operation occurs (the TOCTOU problem). The attestation system abided by all 5 principles, but still has a glaring vulnerability[3].

### 4.2 14 Goals of Attestation

Given that the 5 Principles are insufficient to prevent all attacks on attestation protocols, extensions to the principles were done by Banks et al. [5], Usman et al. [40] to expand upon the requirements necessary for securing attestation systems.

Usman et al. established 14 goals of attestation schemes in [40]:

(1) Acceptably secure remote attestation
(2) Protection of Verifier, Target, and Attestation Agent.
(3) Secure communication between Target and Verifier
(4) Attestation Agent cannot modify Target state.
(5) Key protection
(6) Integrity of the Attestation Agent state.
(7) Attestation Agent has (read-only) access to the full Target state
(8) Controlled invocation – first-to-last integrity measurement
(9) Atomicity – interruption-free execution of integrity measurements.
(10) Functional correctness of the integrity measurement.
(11) Freshness of Attestation Response.
(12) Confidentiality of the channel between Target and Verifier
(13) Integrity of the channel between Target and Verifier.
(14) Availability of the channel between Target and Verifier.

It is evident in this compilation of goals that not all attestation schemes would likely need to satisfy them, but any scheme that could satisfy all goals would likely be robust to most attacks.

### 4.3 Quality of Attestation

Carpent et al. [8] introduced the concept of a Quality of Attestation Metric[4], particularly to apply to Section 3.1: Attestation Timings and Software based attestation.

Their original formulation had 3 main variables and is shown in Figure 4. This notion of attestation quality based upon the freshness of information takes inspiration from the Principles and Goals previously outlined, while also incorporating the costs to achieve such implementations.

### 4.4 Protocol Orderings

As is evidenced by the complexities of the attestation system requirements previously presented, there is a lot of nuance to making a robust attestation system.

One particular method for ensuring attestation yields quality results (especially applicable to Section 3.2.1: Layered Attestation) is utilizing a protocol ordering. Ramsdell [31], Ramsdell et al.

[3] One major factor for this is that the work establishing the 5 Principles operated primarily in the context of hardware based attestation; where the TOCTOU problem is essentially solved by a **Trusted Hardware Component**

[4] Along with Quality of Swarm Attestation around the same time (Carpent et al. [7])

| Var | Definition | Benefits | Limitations |
|---|---|---|---|
| $T_M$ | time between consecutive measurements | High detection rate | High burden on the Attester |
| $T_C$ | time between consecutive attestation requests | Fast detection | High burden on Attester and Requester |
| $f$ | freshness of latest measurement | Fresh information | High burden on Attester |

Fig. 4. Quality of Attestion table from Carpent et al. [8]

[32] first introduced the concept of validating layered attestation protocols based upon an analaysis with a model checker. This model checker would simulate adversary actions, and give possible attacks based upon the protocol. Further work was done by Petz et al. [29] utilizing these tools in conjunction with the Flexibles Mechanisms for Remote Attestation [15] to show the specific desired properties of an attestation protocol itself could be verified with model checking.

Protocol ordering was extended upon by Rowe [34, 35] where the key insight that attack tree on protocols could be ordered based upon homomorphisms between attack graphs[5]. Utilizing protocol orderings lends to upholding the original Principle of "Trustworthy Mechanism", but also adds another layer of analysis to attestation. Ultimately, trust in an attestation does not depend solely on the attestation mechanism, or the attestation targets, but the ordering that attestations of those targets take place.

## 5 ATTESTATION USE CASES

In this section, we will explore some of the common use cases for attestation that have not been explored in this paper already. In one of its most fundamental forms, remote attestation has been used as a method for attesting that a program is executing correctly. Correct execution was extended by Gu et al. [13] beyond the idea that the original program binary is the expected binary to be running, but also that the inputs to the program are the same (or fall within some "good" range of expected inputs). Programs are considered to be modelable as a Finite State Machine (FSM), and thus the attestation of a program binary with its given inputs is a traversal of this FSM that yields evidence regarding the integrity of the running program.

### 5.1 Cloud Infrastructure

Verifying that a cloud infrastructure can both be trusted as not corrupted, and not disclosing any trade secrets, is a popular target for attestation efforts. Ott et al. [27] introduced a hardware agnostic representation of attestation entities/measurement targets that helps make deciding whether to trust a complicated infrastructure more understandable.

Additionally, a newly proposed target (Sultana et al. [37]) for attestation is in the realm of programmable dataplanes. Programmable dataplanes offer many benefits, but also are exposed to possible attacks if they are misconfigured. It is proposed that attestation can act not only as evidence of correct execution, but as an auditing and configuration-verification tool. Ultimately this was an expression of property based remote attestation without a specific class of attestation in mind.

### 5.2 Internet of Things

Of all the targets for attestation, the most frequently explored are low-power embedded devices, particularly those that make up the Internet of Things (IoT). The primary reason to target these

[5] Generally a homomorphism from attack tree A to B meant that attack A was more easily completed than attack B

devices for attestation is the resource-constrained nature of the devices, making it infeasible to maintain an anti-malware solution with high overhead.

IoT attestation typically takes the form of software attestation, as the resource constraints are the reason attestation is being utilized in the first place and hardware attestation typically requires more resources. One modern example of software based IoT attestation was ERASMUS (Efficient Remote Attestation via Self-Measurement for Unattended Settings) a periodic, self-measuring attestation system presented by Carpent et al. [8]. Numerous other examples of software based embedded device attestation exist as well [12, 17, 26], etc. Despite possible resource limitations, IoT has also been demonstrated to be compatible with hardware based attestation approaches. Tan et al. [38] established a method by which hardware based binary attestation of Wireless Sensor Networks could be achieved through the use of a TPM.

## 6 FUTURE WORK

While we attempted to be as complete as possible in the development of this taxonomy, there are far too many novel methods of attestation to be explored in this paper. Further work would attempt to seek out more proposed attestation methods and either classify them with the existing taxonomy established in this paper, or extend our classification framework to fit them.

One obvious area mentioned in Section 1.1: Attestation Types that could be extended was the Attestation Type Table (Fig 2). We analyzed the possible scenarios when we have 3 distinct principals involved, one for the Relying Party, Requester, and Attester, but no treatment was given to the case when the Attester and Target were different. This extension would be valuable as pre-existing cases where the Attester and Target differ exist and fall into a unexplored part of this taxonomy.

An additional extension to this work would be exploring further how the roles played by the Relying Party and Requester may be changed, and how the type of attestation they perform may change with it. For example, the role of the requester as both the initiator of an attestation request (without necessarily being reliant on the outcome of that attestation request) and Verifier of evidence, may have to change when considering that possibly just a Third-Party verifier is required, but does not initiate requests. Such cases are explored by Brickell et al. [6], Helble et al. [15] and can be forced into our existing taxonomy by ignoring some of the implicit ordering that exists during an attestation request, but giving them more first class treatment with an extension of the taxonomy would be preferred.

Expanding upon the recent work by Usman et al. [40] and the development of the 14 Goals of attestation schemes would be a valuable future work. Particularly, discovering a minimal set of goals that must be satisfied by certain attestation classes, types, methods, and patterns, in order for them to be robust to attacks would be a beneficial extension of the taxonomy. In particular it would ensure that when new attestation systems are developed that they satisfy all goals needed to make their style of attestation robust.

## 7 CONCLUSION

Overall, we have explored and established a standard taxonomy for Attestation. Many different types, classes, methods, and patterns for Attestation exist, each with their own benefits and limitations. The specific type of attestation that should be used is a function of the environment in which the attestation must take place, and a solution tailored for that context can be built using the standard conventions outlined in this review.

## REFERENCES

[1] Alam, M., Zhang, X., Nauman, M., Ali, T., Seifert, J.P.: Model-based behavioral attestation. In: Proceedings of the 13th ACM Symposium on Access Control Models and Technologies. pp. 175–184. SACMAT '08, Association for Computing Machinery, New York, NY, USA (2008), https://doi.org/10.1145/1377836.1377864

[2] Ammar, M., Crispo, B., De Oliveira Nunes, I., Tsudik, G.: Delegated attestation: Scalable remote attestation of commodity cps by blending proofs of execution with software attestation. In: Proceedings of the 14th ACM Conference on Security and Privacy in Wireless and Mobile Networks. pp. 37–47. WiSec '21, Association for Computing Machinery, New York, NY, USA (2021), https://doi.org/10.1145/3448300.3467818

[3] Armknecht, F., Sadeghi, A.R., Schulz, S., Wachsmann, C.: A security framework for the analysis and design of software attestation. In: Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security. p. 1–12. CCS '13, Association for Computing Machinery, New York, NY, USA (2013), https://doi.org/10.1145/2508859.2516650

[4] Bampatsikos, M., Ntantogian, C., Xenakis, C., Thomopoulos, S.C.A.: Barrett blockchain regulated remote attestation. In: IEEE/WIC/ACM International Conference on Web Intelligence - Companion Volume. pp. 256–262. WI '19 Companion, Association for Computing Machinery, New York, NY, USA (2019), https://doi.org/10.1145/3358695.3361752

[5] Banks, A.S., Kisiel, M., Korsholm, P.: Remote attestation: A literature review (2021)

[6] Brickell, E., Camenisch, J., Chen, L.: Direct anonymous attestation. In: Proceedings of the 11th ACM conference on Computer and communications security. pp. 132–145. ACM (2004)

[7] Carpent, X., ElDefrawy, K., Rattanavipanon, N., Tsudik, G.: Lightweight swarm attestation: A tale of two lisa-s. In: Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security. pp. 86–100 (2017)

[8] Carpent, X., Tsudik, G., Rattanavipanon, N.: Erasmus: Efficient remote attestation via self-measurement for unattended settings. In: 2018 Design, Automation & Test in Europe Conference & Exhibition (DATE). pp. 1191–1194. IEEE (2018)

[9] Chen, L., Landfermann, R., Löhr, H., Rohe, M., Sadeghi, A.R., Stüble, C.: A protocol for property-based attestation. In: Proceedings of the First ACM Workshop on Scalable Trusted Computing. pp. 7–16. STC '06, Association for Computing Machinery, New York, NY, USA (2006), https://doi.org/10.1145/1179474.1179479

[10] Coker, G., Guttman, J., Loscocco, P., Herzog, A., Millen, J., O'Hanlon, B., Ramsdell, J., Segall, A., Sheehy, J., Sniffen, B.: Principles of remote attestation. International Journal of Information Security 10(2), 63–81 (June 2011)

[11] Copland Consortium: The copland framework website (2019), https://copland-lang.org

[12] De Oliveira Nunes, I., Jakkamsetti, S., Rattanavipanon, N., Tsudik, G.: On the toctou problem in remote attestation. In: Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security. p. 2921–2936. CCS '21, Association for Computing Machinery, New York, NY, USA (2021), https://doi.org/10.1145/3460120.3484532

[13] Gu, L., Ding, X., Deng, R.H., Xie, B., Mei, H.: Remote attestation on program execution. In: Proceedings of the 3rd ACM Workshop on Scalable Trusted Computing. pp. 11–20. STC '08, Association for Computing Machinery, New York, NY, USA (2008), https://doi.org/10.1145/1456455.1456458

[14] Haldar, V., Chandra, D., Franz, M.: Semantic remote attestation – a virtual machine directed approach to trusted computing. In: Proceedings of the Third Virtual Machine Research and Technology Symposium. San Jose, CA (May 2004)

[15] Helble, S.C., Kretz, I.D., Loscocco, P.A., Ramsdell, J.D., Rowe, P.D., Alexander, P.: Flexible mechanisms for remote attestation. ACM Transactions on Privacy and Security (TOPS) 24(4), 1–23 (2021)

[16] IETF RATS Working Group: Remote ATtestation ProcedureS (RATS). https://datatracker.ietf.org/wg/rats/about/ (2023)

[17] Javaid, U., Aman, M.N., Sikdar, B.: Defining trust in iot environments via distributed remote attestation using blockchain. In: Proceedings of the Twenty-First International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing. p. 321–326. Mobihoc '20, Association for Computing Machinery, New York, NY, USA (2020), https://doi.org/10.1145/3397166.3412801

[18] Klein, G., Elphinstone, K., Heiser, G., Andronick, J., Cock, D., Derrin, P., Elkaduwe, D., Engelhardt, K., Kolanski, R., Norrish, M., Sewell, T., Tuch, H., Winwood, S.: sel4: formal verification of an os kernel. In: SOSP '09: Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles. pp. 207–220. ACM, New York, NY, USA (2009)

[19] Kuang, B., Fu, A., Yu, S., Yang, G., Su, M., Zhang, Y.: Esdra: An efficient and secure distributed remote attestation scheme for iot swarms. IEEE Internet of Things Journal 6(5), 8372–8383 (2019)

[20] Kühn, U., Selhorst, M., Stüble, C.: Realizing property-based attestation and sealing with commonly available hard-and software. In: Proceedings of the 2007 ACM workshop on Scalable trusted computing. pp. 50–57. ACM (2007)

[21] Li, X.y., Shen, C.x., Zuo, X.d.: An efficient attestation for trustworthiness of computing platform. In: 2006 International Conference on Intelligent Information Hiding and Multimedia. pp. 625–630 (2006)

[22] Loscocco, P.A., Smalley, S.D., Muckelbauer, P.A., Taylor, R.C., Turner, S.J., Farrell, J.F.: The inevitability of failure: The flawed assumption of security in modern computing environments. In: In Proceedings of the 21st National Information Systems Security Conference. pp. 303–314 (1998)

[23] Loscocco, P.A., Wilson, P.W., Pendergrass, J.A., McDonell, C.D.: Linux kernel integrity measurement using contextual inspection. In: Proceedings of the 2007 ACM workshop on Scalable trusted computing. pp. 21–29. STC '07, ACM, New

York, NY, USA (2007), http://doi.acm.org/10.1145/1314354.1314362

[24] Martin, A., et al.: The ten page introduction to trusted computing. Tech. Rep. CS-RR-08-11, Oxford University Computing Labratory, Oxford, UK (2008)

[25] Mondal, A., Gangopadhyay, S., Chatterjee, D., Boyapally, H., Mukhopadhyay, D.: Prefer : Physically related function based remote attestation protocol. ACM Trans. Embed. Comput. Syst. 22(5s) (sep 2023), https://doi.org/10.1145/3609104

[26] Nunes, I.D.O., Eldefrawy, K., Rattanavipanon, N., Steiner, M., Tsudik, G.: Vrased: A verified hardware/software co-design for remote attestation. In: 28th USENIX Security Symposium (USENIX Security 19). pp. 1429–1446 (2019)

[27] Ott, S., Kamhuber, M., Pecholt, J., Wessel, S.: Universal remote attestation for cloud and edge platforms. In: Proceedings of the 18th International Conference on Availability, Reliability and Security. ARES '23, Association for Computing Machinery, New York, NY, USA (2023), https://doi.org/10.1145/3600160.3600171

[28] Petz, A., Alexander, P.: An infrastructure for faithful execution of remote attestation protocols. Innovations in Systems and Software Engineering (2022)

[29] Petz, A., Jurgensen, G., Alexander, P.: Design and formal verification of a copland-based attestation protocol. In: Proceedings of the 19th ACM-IEEE International Conference on Formal Methods and Models for System Design. pp. 111–117. MEMOCODE '21, Association for Computing Machinery, New York, NY, USA (2021), https://doi.org/10.1145/3487212.3487340

[30] Petz, A., et al.: copland-avm. https://github.com/ku-sldg/copland-avm (2020)

[31] Ramsdell, J.: Chase: A model finder for finitary geometric logic. https://github.com/ramsdell/chase (2020)

[32] Ramsdell, J.D., Guttman, J.D., Millen, J.K., O'Hanlon, B.: An analysis of the caves attestation protocol using cpsa. Technical Remport MTR090213, MITRE, Center for Integrated Intelligence Systems, Bedford, MA (December 2009)

[33] Rauter, T., Höller, A., Kajtazovic, N., Kreiner, C.: Privilege-based remote attestation: Towards integrity assurance for lightweight clients. In: Proceedings of the 1st ACM Workshop on IoT Privacy, Trust, and Security. p. 3–9. IoTPTS '15, Association for Computing Machinery, New York, NY, USA (2015), https://doi.org/10.1145/2732209.2732211

[34] Rowe, P.D.: Confining adversary actions via measurement. Third International Workshop on Graphical Models for Security pp. 150–166 (2016)

[35] Rowe, P.D.: On orderings in security models. In: Protocols, Strands, and Logic: Essays Dedicated to Joshua Guttman on the Occasion of his 66.66 th Birthday, pp. 370–393. Springer (2021)

[36] Sadeghi, A., Stüble, C.: Property-based attestation for computing platforms: caring about properties, not mechanisms. In: Proceedings of the 2004 workshop on New security paradigms. pp. 67–77. ACM (2004)

[37] Sultana, N., Shands, D., Yegneswaran, V.: A case for remote attestation in programmable dataplanes. In: Proceedings of the 21st ACM Workshop on Hot Topics in Networks, HotNets 2022, Austin, Texas, November 14-15, 2022. pp. 122–129. ACM (2022), https://doi.org/10.1145/3563766.3564100

[38] Tan, H., Hu, W., Jha, S.: A tpm-enabled remote attestation protocol (trap) in wireless sensor networks. In: Proceedings of the 6th ACM Workshop on Performance Monitoring and Measurement of Heterogeneous Wireless and Wired Networks. pp. 9–16. PM2HW2N '11, Association for Computing Machinery, New York, NY, USA (2011), https://doi.org/10.1145/2069087.2069090

[39] Trusted Computing Group: TCG TPM Specification. Trusted Computing Group, 3885 SW 153rd Drive, Beaverton, OR 97006, version 1.2 revision 103 edn. (July 2007), https://www.trustedcomputinggroup.org/resources/tpm_main_specification/

[40] Usman, A.B., Cole, N., Asplund, M., Boeira, F., Vestlund, C.: Remote attestation assurance arguments for trusted execution environments. In: Proceedings of the 2023 ACM Workshop on Secure and Trustworthy Cyber-Physical Systems. pp. 33–42. SaT-CPS '23, Association for Computing Machinery, New York, NY, USA (2023), https://doi.org/10.1145/3579988.3585056

[41] Wedaj, S., Paul, K., Ribeiro, V.J.: Dads: Decentralized attestation for device swarms. ACM Trans. Priv. Secur. 22(3), 19:1–19:29 (Jul 2019), http://doi.acm.org/10.1145/3325822