Formal Verification of xXxXx, BSc proj (Autumn 2024)

Mathias Emil Lystlund Rasmussen

October 22, 2024

Contents

1 Introduction **5**

2 The Entity Attestation TOKEN(EAT) draft-ietf-rats-eat-31 **5**

2.1 Motivation 5

2.2 Remote attestation 5

2.3 Remote ATtestation procedureS (RATS) 5

2.3.1 Evidence 6

2.3.2 Root of trust and TPM 6

2.3.3 PCR/inside Attester (NOT DONE) 7

2.3.4 Verifier 8

2.4 Layerd Attestation and Composite devices 10

2.5 Entity Attestation Token (EAT) 10

2.6 Jason web token (JWT) 10

2.7 Properties 11

3 Tamarin-prover **11**

4 My model in Tamarin-prover **11**

5 Conclusion **11**

Abstract

Acknowledgement

1 Introduction

help library, or head to our plans page to choose your plan.

2 The Entity Attestation TOKEN(EAT) draft-ietf-rats-eat-31

Entity Attestation Token or EAT is a protocol ... . EAT provides

2.1 Motivation

2.2 Remote attestation

Remote Attestaion (RA) is a security concept/method used to verify the integrity and trustworthiness of a system. In RA, a peer (The Attester) produces "believeble" information about itself (evidence) to enable a remote peer (The relying party) to decide whether to consider the attester a trustworthy peer or not, and what level of access that the attester should have in your system.
The concept of Remote attestation is not new, and is used all over and in many systems. The problem with remote attestation is, that it has traditionally been implemented in a proprietary manner, instead of something that is standardized.[1][2]

2.3 Remote ATtestation procedureS (RATS)

As mentioned, the previous problem with attestation was that our systems were proprietary systems for special use cases as supposed to something that was broad and standardized. The Internet Engineering Task Force (IETF) took these prior examples of deployed systems of attestation and tried to standardized them.
So RATS is a standardized architecture (not a protocol), which purpose is to create standardized formats within existing/new protocols/systems, in order to achieve mutual attestation. [2][1]

In the RATS architecture the attester produces evidence about it self and sends it to a verifier. The verifier then evaluates the evidence to what is called "Attestation Result". The verifier then sends the attestation result to the relying party, that then decides what access to give the attester.[1][2]

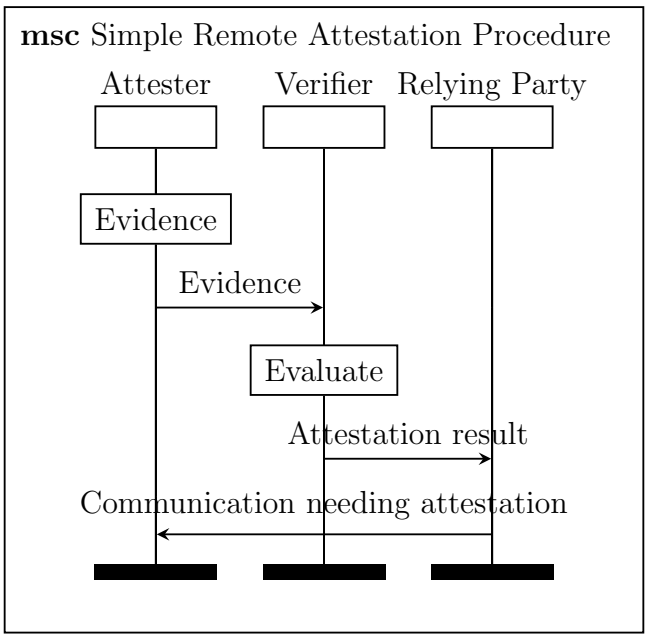

Figure 1: A simplified representation of the Remote Attestation Procedure.

The attester can be any IoT device, computer, server, smart phone etc, trying to gain access to a network, system, or resources and needs to be trusted by the system that it is interacting with.

2.3.1 Evidence

In order for an attester to be able to be trusted by a relying party, the attester has to share information about it self. This information is vital for the relying party, since it doesn't want to share resources and secret information to just anyone. In order for an attester to be trusted, it has to live up to certain requirements set by the relying party. These can include running a "new enough" software version, not having modified software running, a maintenance log etc. The purpose of the information send to the relying party is to convince it, that you are in a "god enough state" to share information with. A reason why a relying party might want to know which software version a device is running could be that there is a known vulnerability in older versions, where adversaries might be able to extract the data send to the attester device.[1][2]

2.3.2 Root of trust and TPM

If an attester is compromised/in a bad state, then the relying party and verifier can't trust the evidence/information produced by the attester. A typical solution to this problem is to use a "root of trust". A root of trust is a trusted component or mechanism that provides trustworthy appraisal for other components. It does so, by collecting, protecting and signing information in the system/IoT device. This information can describe attributes of the system, such as hardware, software, firmware etc. [1][3]
An example of a "root of trust" is the Trusted Platform Module (TPM). A TPM is a small chip, with minimal resources and computational power. TPM's have build in cryptographic functionality and support different cryptographic algorithms, such as

SHA256.
The fundemental concept of a TPM is to guarantee integrity and confidentiality for the evidence data, even when the operating system has been compromised. Note that a TPM dosen't guarantee security against an adversary that has physical access to the hardware of the device [3].
A TPM will have a unique endorsement key, which was generated under the manufactoring process and is not extractable from the TPM. In order to verify the TPM, the TPM uses an endorsement key (EK) certificate, which is a x.509 certificate generated by the TPM manufacturer and is therefore verifiable by the manufacturer. In order to verify that a TPM and its EK certificate is associated with a device, they use a platform certificate provided by the system manufacturer, which associates platform metadata with the TPM identity. [4][3]
The TPM generates an attestation key (AK) with the EK. The AK can be verified to be linked to the EK. The private portion of the Ak never leaves the TPM. The AK is used by the TPM to sign the evidence related to the state of the device.

2.3.3 PCR/inside Attester (NOT DONE)

TPM Providing a space that can be considered trustworthy even if the system has been compromised. Endorsement key (EK) - generated at the factory - NOT extractable - Private part of EK is fixed for the entire lifetime of the TPM

EK certificate (generated by the TPM manifactorer) ties the EK key Bach to the manufactorer Platform certificate (generated by the system manifacturer) ties platform (various bits of platform meta data (e.g. model number, seriel number, xxx) with the TPM identity. It can be used to verify that the EK key corresponds to the machine

EK key is NOT used for signing - only used for encryption !!

For signing a second set og keys - attestation keys AK - are used A number of different AK's can exist. AK's are provided/loaded on the TPM (challenge(using a response protocol - encrypted by the public part of EK). AK's are used to sign PCR values - to ensure that the provides values are from the TPM on the correct system/platform!

PCR - Platform Configuration Register Secure storage contained in the TPM used to hold a register of PCR values can NOT be modified directly by OS or firmware Every time a change to a register I requested, the new hash value is generate as a hash og the old value and the new value (and perhaps also the next register value). In this way each value hold a complete log of the history in one value (block chain !!)

Examples of types of data stored in registers: Hash of firmware - Hash of bootloader Hash of kernal Etc.
Remote attestation: Generate a quote from the TPM. It contains the signed values of the PCR registers together with the measurement log, containing all changes to all the measured values. The verifier verifies the signature of the quote, and then verifies the validity of the measurement log by hashing them using the same algorithm as the TPM/PCR Then the verifier known that the measurement log is valid - all values are valid - and can then make some kind of security policy decision based on the measurement values

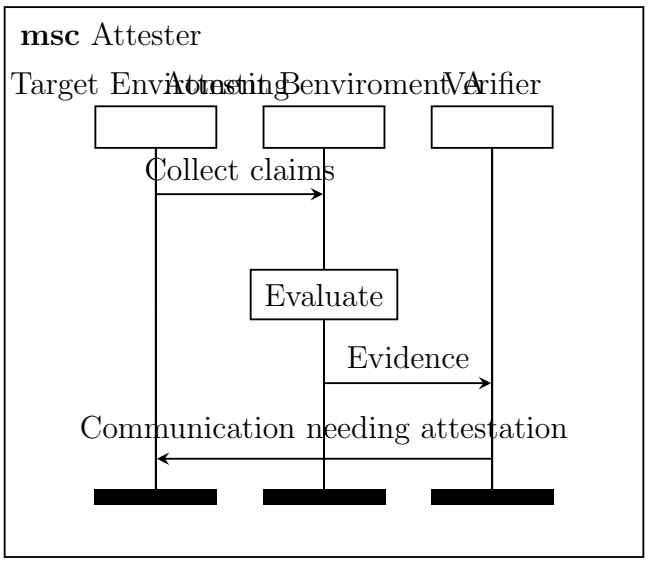

Figure 2: Attester produces evidence: CURRENTLY NOT DONE

2.3.4 Verifier

The verifiers role in the attestation process, is to evaluate the evidence provided by the attester and determine the trustworthiness of the attesters state. The verifier does this by consulting a Verifier Owner. The Verifier Owner sends what is called a "Appraisal policy for Evidence". This appraisal policy for evidence defines what evidence is acceptable and how the evidence should be processed, and what endorsements are required.[1][2]
The verifier can then contact and receive endorsements from the different endorsers related to the attester. The Endorser can be a manufacturer of different hardware associated with the attester. This endorser can as an example verify signatures used by different hardware, like a TPM.[1][2]
The verifier can then consult a Reference Value Provider supplies the verifier with reference values. The reference values are known-good values, which are used to evaluate the evidence from the attester. These "good" reference values can be software updates with no major security faults, hardware configurations, etc.
Following these the verifier will produce an Attestation result. The attestation result is an assertion that the attester is in a good state, bad state or maybe its a lot more refined then that.
So the attestation result can be as simple as a boolean, yes the system is good or no the system is not good, or it can be a long list of different evaluations of different aspects of the attester.[1][2]

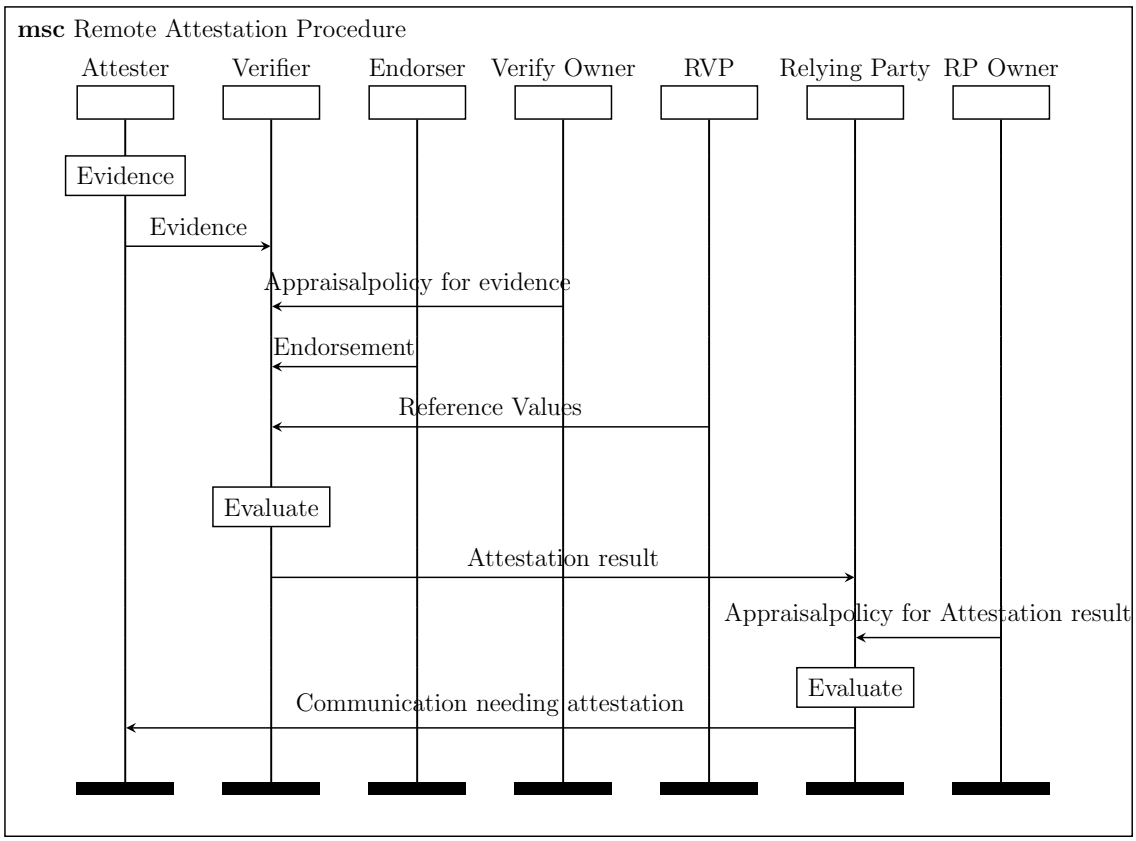

Figure 3: A more in depth depiction of the different steps in a remote attestation procedure. In this figure Reference Value Provider has been reduced to RVP and the Relying Party Owner has been reduced to RP Owner. The boxes with text e.g. the box with Evidence, indicates an internal process within that given entity.[1][2]

In figure 3 we see an attester, in an internal process, produce evidence and send that evidence to the verifier.
The Verifier then consults an Appraisalpolicy for evidece, Endorser and a Reference Value provider inorder to evaluate the evidence in the for of an attestation result.
The Verifier then sends the attestation result to the relying party that the atterser wishes to communicate with/gain resources from. The relying party then consults a relying party owner, which provides an appraisal policy for attestation result. This policy serves as a guideline for the relying party to evaluate whether the attester meets the required security and trust standards before granting access or resources.

Note that the different entities shown in figure 3: Attester, Verifier, Endorser, Verify Owner, etc, are displayed as different entities. This doesn't have to be the case, and in many cases, these entities overlap and are the same device.

2.4 Layerd Attestation and Composite devices

2.5 Entity Attestation Token (EAT)

The specific specification of EAT format is The Entity Attestation Token (EAT) draft-ietf[1]-rats-eat-31 [6]. The EAT protocol follows the operational model described in the RATS Architecture [1][6].

EAT is a standardized format used for conveying evidence, in the form of claims about the attester. These claims can include information about the hardware, software, or other attributes essential for the verifier to verify the state of the attester.
The IEFT specification descries a series of possible claims that can be used in an EAT. Which claims are used is very much context dependent - some claims are only relevant for a hardware entity and some are only relevant for a software entity.

The IEFT specification devides claims into 3 groups:

- Claims to ensure freshness e.g. eat_nonce
- Claims describing then entity
- Claims describing the token

An EAT is encoded in either JavaScript Object Notation (JSON) or Concise Binary Object Representation (CBOR), depending on the use case. An EAT is build on JSON Web Token (JWT) and CBOR Web Token (CWT). [6] The following example is a JSON format for the payload for the JWT and is also found in [6]:

```
{
    "eat_nonce": "MIDBNH28iioisjPy",
    "ueid":      "AgAEizrK3Q",
    "oemid":     76543,
    "swname":    "Acme IoT OS",
    "swversion": "3.1.4"
}
```

This payload includes of a nonce, which ensures freshness. The Universal Entity ID (ueid) is a "serial number" uniquely identifying the attester device. The Original Equipment Manufacturer ID (oemid) can be used to identify the hardware manufacturer of components in the attester device. The software name (swname) is used to identify the name and version of software running, and the software version (swversion) is the current software version running on the attester.[6]

2.6 Jason web token (JWT)

A Jason Web Token or JWT, is a... Token based authentication...

[1]The Internet Engineering Task Force (IETF), is a standards development organization for the Internet[5]

2.7 Properties

The Entity Attestation Token protocol should comply with the following properties to its security:

- **Authenticity**
  The protocol must ensure that all attestation tokens and claims are generated by a legitimate entity. ...
- **Integrity**
  The protocol must guarantee that the attestation data (EAT) has not been tampered with during transmission. An attacker should not be able to modify any claims within an EAT or craft a new token which would be accepted by a verifier.
- **Freshness and non-replyability**
  The protocol must ensure that the evidence produced by the attester is fresh, meaning that the data about the attester reflects the current state of the entity. The nonce is used for this, and ensures that an attacker cannot resend a previously valid attestation token.
- **evt Confidentiality with JOSE**
- **Limitations on Hardware Security**

In order to verify that the EAT protocol adheres to the listed security properties, the protocol will be modelled in Tamarin-prover. This tool will be explained in a later section, in order to understand why a proof created via the tool is useful when assessing the correctness of a protocol.

3 Tamarin-prover

4 My model in Tamarin-prover

5 Conclusion

References

[1] Birkholz Henk, Thaler Dave, Richardson Michael, Smith Ned, and Pan Wei. Remote attestation procedures (rats) architecture, 2023. URL: https://datatracker.ietf.org/doc/rfc9334/.

[2] Confidential Computing Consortium. Remote attestation procedures (rats) architecture, 2021. URL: https://www.youtube.com/watch?v=j30PhEWKgG4.

[3] Confidential Computing Consortium. Tpm based attestation - how can we use it for good?" - matthew garrett (lca 2020), 2020. URL: https://www.youtube.com/watch?v=FobfM9S9xSI&ab_channel=linux.conf.au.

[4] admin@trustedcomputinggroup.org. Trusted platform module library part 1: Architecture, 20019. URL: https://trustedcomputinggroup.org/wp-content/uploads/TCG_TPM2_r1p59_Part1_Architecture_pub.pdf.

[5] IETF. Introduction to the ietf, 2024. URL: https://www.ietf.org/about/introduction/.

[6] Lundblade Laurence, Mandyam Giridhar, O'Donoghue Jeremy, and Wallace Carl. The entity attestation token (eat) draft-ietf-rats-eat-31, 2024. URL: https://datatracker.ietf.org/doc/draft-ietf-rats-eat/.