RATS Working Group M. Richardson
Internet-Draft Sandelman Software Works
Intended status: Informational C. Wallace
Expires: May 6, 2021 Red Hound Software
W. Pan
Huawei Technologies
November 02, 2020

Use cases for Remote Attestation common encodings
draft-richardson-rats-usecases-08

Abstract

This document details mechanisms created for performing Remote Attestation that have been used in a number of industries. The document initially focuses on existing industry verticals, mapping terminology used in those specifications to the more abstract terminology used by the IETF RATS Working Group.

The document aspires to describe possible future use cases that would be enabled by common formats.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on May 6, 2021.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction

The recently chartered IETF RATS WG intends to create a system of attestations that can be shared across a multitude of different users.

This document exists as place to collect use cases for the common RATS technologies in support of the IETF RATS charter point 1. This document is not expected to be published as an RFC, but remain open as a working document. It could become an appendix to provide motivation for a protocol standards document.

End-user use cases that would either directly leverage RATS technology, or would serve to inform technology choices are welcome, however.

2. Terminology

Critical to dealing with and contrasting different technologies is to collect terms which are compatible, to distinguish those terms which are similar but used in different ways.

This section will grow to include forward and external references to terms which have been seen. When terms need to be disambiguated they will be prefixed with their source, such as "TCG(claim)" or "FIDO(relying party)"

Platform attestations generally come in two categories. This document will attempt to indicate for a particular attestation technology falls into this.

2.1. Static attestations

A static attestation says something about the platform on which the code is running.

2.2. Session attestations

A session attestation says something about how a session key used in a connection such as TLS connection was created. It is usually the result of evaluating attestations that are attached to the certificates used to create such a session.

2.3. Statements

The term "statement" is used as the generic term for the semantic content which is being attested to.

2.4. Hardware Root Of Trust

[SP800-155] offers the following definition for root of trust.

"Roots of Trust are components (software, hardware, or hybrid) and computing engines that constitute a set of unconditionally trusted functions. Reliable and trustworthy BIOS integrity measurement and reporting depend upon software agents; each software agent relies upon Roots of Trust, and the level of trustworthiness in each agent depends on its Roots of Trust. BIOS integrity measurement requires the coordination of a Measurement Agent to harvest measurements, a Storage Agent to protect the measurements from modification until they can be reported, and a Reporting Agent to reliably report the measurements. Each of these agents has a corresponding Root of Trust (Root of Trust for Measurement, etc.) These Roots of Trust must act in concert and build on each other to enable reliable and trustworthy measurement, reporting, and verification of BIOS integrity measurements."

SP800-155 uses the terms RoT for Reporting, Storage and Measurement, but not RoT for Verification - it uses "Verification Agent". Though it is assumed the verifier is trustworthy.

However, [tcgglossary] (page 9) includes a RoT for Verification (RTV) as well.

The TCG Glossary also offers a general definition for Root of Trust "A component that performs one or more security-specific functions, such as measurement, storage, reporting, verification, and/or update.

It is trusted always to behave in the expected manner, because its misbehavior cannot be detected (such as by measurement) under normal operation. "

[SP800-147B] defines RoT for Update (RoTU) and RoTU verification (RoTU-v).

The TCG definition seems more concise than the NIST, but gets to the same point.

For the purpose of this documenet, a hardware root of trust refers to security functionality that is trusted to behave in the expected manner, because its misbehavior cannot be detected under normal operation and resists soft exploits by encapsulating the functionality in hardware.

2.5. Template for Use cases

Each use case will consist of a table with a number of constant fields, as illustrated below. The claim names will be loosely synchronized with the EAT draft. The role workflow (formerly "attestation type") will be described in the architecture draft. It will describe two classes of workflow: the passport type (Attestee sends evidence to Attester, receives signed statment, which is sent to relying party), or the background check type (Attestee sends measurements to Relying party, Relying Party checks with Attester).

Use case name: Twelve Monkeys

Who will use it: Army of the Twelve Monkeys SDO

Attester: James Cole

Relying Party: Dr. Kathryn Reilly

Message Flow: Passport

Claims used as evidence: OEM Identity, Age Claim, Location Claim, ptime Claim

Description: James Cole must convince Dr. Reilly he is from the future, and not insane.

3. Requirements Language

This document is not a standards track document and does not make any normative protocol requirements using terminology described in [RFC2119].

4. Overview of Sources of Use Cases

The following specifications have been covered in this document:

- o The Trusted Computing Group "Network Device Attestation Workflow" [I-D.fedorkow-rats-network-device-attestation]
- o Android Keystore
- o Fast Identity Online (FIDO) Alliance attestation,

This document will be expanded to include summaries from:

- o Trusted Computing Group (TCG) Trusted Platform Module (TPM)/Trusted Software Stack (TSS)
- o ARM "Platform Security Architecture" [I-D.tschofenig-rats-psa-token]
- o Intel SGX attestation [intelsgx]
- o Windows Defender System Guard attestation [windowsdefender]
- o Windows Device Health Attestation [windowshealth]
- o Azure Sphere Attestation [azureattestation]: https://azure.microsoft.com/enus/resources/azure-sphere-device-authentication-andattestation-service/en-us/
- o IETF NEA WG [RFC5209]

Additional sources are welcome and requested.

5. Use case summaries

This section lists a series of cases where an attestation is done.

5.1. Device Capabilities/Firmware Attestation

This is a category of claims

Use case name: Device Identity

Who will use it: Network Operators

Attester: varies

Message Flow: varies

Relying Party: varies

Claims used as evidence: TBD

Description: Network operators want a trustworth report of identity and version of information of the hardware and software on the machines attached to their network. The process starts with some kind of Root of Trust that provides device identity and protected storage for measurements. The mechanism performs a series of measurements, and expresses this with an attestation as to the hardware and firmware/software which is running.

This is a general description for which there are many specific use cases, including [I-D.fedorkow-rats-network-device-attestation] section 1.2, "Software Inventory"

5.1.1. Relying on an (third-party) Attestation Server

Use case name: Third Party Attestation Server

Who will use it: Network Operators

Message Flow: background check

Attester: manufacturer of OS or hardware system

Relying Party: network access control systems

Claims used as evidence: TBD

Description: The measurements from a heterogenous network of devices are provided to device-specific attestation servers. The attestation servers know what the "golden" measurements are, and perform the appropriate evaluations, resulting in attestations that the relying parties can depend upon.

5.1.2. Autonomous Relying Party

Use case name: Autonomous

Who will use it: network operators

Message Flow: passport

Attester: manufacturer of OS or hardware system

Relying Party: peer systems

Claims used as evidence: TBD

Description: The signed measurements are sent to a relying party which must validate them directly. They are not sent to a third party. (It may do so with the help of a signed list of golden values, or some other process). The relying party needs to validate the signed statements directly.

This may occur because the network is not connected, or even because it can not be connected until the equipment is validated.

5.1.3. Proxy Root of Trust

Use case name: Proxy Root of Trust

Who will use it: network operators

Message Flow: passport

Attester: manufacturer of OS or hardware system

Relying Party: peer systems

Claims used as evidence: TBD

Description: A variety of devices provide measurements via their Root of Trust. A proxy server collects these measurements, and (having applied a local policy) then creates a device agnostic attestation. The relying party can validate the claims in a standard format.

5.1.4. network scaling - small

Use case name: Network scaled - small

Who will use it: enterprises

Message Flow: background check

Attester: manufacturer of OS or hardware system

Relying Party: network equipment

Claims used as evidence: TBD

Description: An entire network of systems needs to be validated (such as all the desktops in an enterprise's building). The infrastructure is in the control of a single operator and is already trusted. The network can be partitioned so that machines that do not pass attestation can be quarantined. A 1:1 relationship between the device and the relying party can be used to maintain freshness of the attestation.

5.1.5. network scaling - medium

Use case name: Network scaled - medium

Who will use it: larger enterprises, including network operators

Message Flow: passport

Attester: manufacturer of OS or hardware system

Relying Party: network equipment

Claims used as evidence: TBD

Description: An entire network of systems needs to be validated: such as all the desktops in an enterprise's building, or all the routers at an ISP. The infrastructure is not necessarily trusted: it could be subverted, and it must also attest. The devices may be under a variety of operators, and may be mutually suspicious: each device may therefore need to process attestations from every other device. An NxM mesh of attestations may be untenable, but a system of N:1:M relationships can be setup via proxy attestations.

5.1.6. network scaling - large

Use case name: Network scaled - large

Who will use it: telco/LTE operators

Message Flow: passport

Attester: manufacturer of OS or hardware system

Relying Party: malware auditing systems

Claims used as evidence: TBD

Description: An entire network of systems need to be continuously attested. This could be all of the smartphones on an LTE network, or every desktop system in a worldwide enterprise. The network operator wishes to do this in order to maintain identities of connected devices more than to validate correct firmware, but both situations are reasonable.

5.2. Hardware resiliency / watchdogs

Use case name: Hardware watchdog

Who will use it: individual system designers

Message Flow: passport

Attester: manufacturer of OS or hardware system

Relying Party: bootloader or service processor

Claims used as evidence: TBD

Description: One significant problem is malware that holds a device hostage and does not allow it to reboot to prevent updates to be applied. This is a significant problem, because it allows a fleet of devices to be held hostage for ransom. Within CyRes the TCG is defining hardware Attention Triggers that force a periodical reboot in hardware.

This can be implemented by forcing a reboot unless attestation to an Attestation Server succeeds within the period interval, and having a reboot do remediation by bringing a device into compliance, including installation of patches as needed.

This is unlike the previous section on Device Attestation in that the attestation comes from a network operator, as to the device's need to continue operating, and is evaluated by trusted firmware (the relying party), which resets a watchdog timer.

5.3. IETF TEEP WG use case

Use case name: TAM validation

Who will use it: The TAM server

Message Flow: background check

Attester: Trusted Execution Environment (TEE)

Relying Party: end-application

Claims used as evidence: TBD

Description: The "Trusted Application Manager (TAM)" server wants to verify the state of a TEE, or applications in the TEE, of a device. The TEE attests to the TAM, which can then decide whether to install sensitive data in the TEE, or whether the TEE is out of compliance and the TAM needs to install updated code in the TEE to bring it back into compliance with the TAM's policy.

5.4. Confidential Machine Learning (ML) model

Use case name: Machine Learning protection

Who will use it: Machine Learning systems

Message Flow: TBD

Attester: hardware TEE

Relying Party: machine learning model owner

Claims used as evidence: TBD

Description: An example use case is where a device manufacturer wants to protect its intellectual property in terms of the ML model it developed and that runs in the devices that its customers purchased, and it wants to prevent attackers, potentially including the customer themselves, from seeing the details of the model. This works by having some protected environment (e.g., a hardware TEE) in the device attest to some manufacturer's service, which if attestation succeeds, then the manufacturer service releases the model, or a key to decrypt the model, to the requester. If a hardware TEE is involved, then this use case overlaps with the TEEP use case.

5.5. Critical infrastructure

Use case name: Critical Infrastructure

Who will use it: devices

Message Flow: TBD

Attester: plant controller

Relying Party: actuator

Claims used as evidence: TBD

Description: When a protocol operation can affect some critical system, the device attached to the critical equipment wants some assurance that the requester has not been compromised. As such, attestation can be used to only accept commands from requesters that are within policy. Hardware attestation in particular, especially in conjunction with a TEE on the requester side, can provide protection against many types of malware.

5.5.1. Computation characteristics

Use case name: Shared Block Chain Computational claims

Who will use it: Consortia of Computation systems

Message Flow: TBD

Attester: computer system (physical or virtual)

Relying Party: other computer systems

Claims used as evidence: TBD

Description: A group of enterprises organized as a consortium seeks to deploy computing nodes as the basis of their shared blockchain system. Each member of the consortium must forward an equal number of computing nodes to participate in the P2P network of nodes that form the basis of the blockchain system. In order to prevent the various issues (e.g. concentration of hash power, anonymous mining nodes) found in other blockchain systems, each computing node must comply to a predefined allowable manifest of system hardware, software and firmware, as agreed to by the membership of the consortium. Thus, a given computing node must be able to report the (pre-boot) configuration of its system and be able to report at an y time the operational status of the various components that make-up its system.

The consortium seeks to have the following things attested: system configuration, group membership, and virtualization status.

This is a peer-to-peer protocol so each device in the consortium is a relying party. The attestation may be requested online by another entity within the consortium, but not by other parties. The attestation needs to be compact and interoperable and may be included in the blockchain itself at the completion of the consensus algorithm.

The attestation will need to start in a hardware RoT in order to validate if the system is running real hardware rather than running a virtual machine.

5.6. Virtualized multi-tenant hosts

Use case name: Multi-tenant hosts

Who will use it: Virtual machine systems

Message Flow: TBD

Attester: virtual machine hypervisor

Relying Party: network operators

Claims used as evidence: TBD

Description: The host system will do verification as per 5.1.

The tenant virtual machines will do verification as per 5.1

The network operator wants to know if the system _as a whole_ is free of malware, but the network operator is not allowed to know who the tenants are.

This is contrasted to the Chassis + Line Cards case (To Be Defined: TBD).

Multiple Line Cards, but a small attestation system on the main card can combine things together. This is a kind of proxy.

5.7. Cryptographic Key Attestation

Use case name: Key Attestation

Who will use it: network authentication systems

Message Flow: TBD

Attester: device platform

Relying Party: internet peers

Claims used as evidence: TBD

Description: The relying party wants to know how secure a private key that identifies an entity is. Unlike the network attestation, the relying party is not part of the network infrastructure, nor do they necessarily have a business relationship (such as ownership) over the end device.

5.7.1. Device Type Attestation

Use case name: Device Type Attestation

Who will use it: mobile platforms

Message Flow: TBD

Attester: device platform

Relying Party: internet peers

Claims used as evidence: TBD

Description: This use case convinces the relying party of the characteristics of a device. For privacy reasons, it might not identify the actual device itself, but rather the class of device. The relying party can understand from either in-band (claims) or out-of-band (model numbers, which may be expressed as a claim) whether the device has trustworthy features such as a hardware TPM, software TPM via TEE, or software TPM without TEE. Other details such as the availability of finger-print readers or HDMI outputs may also be inferred.

5.7.2. Key storage attestation

Use case name: Key storage Attestation

Who will use it: secure key storage subsystems

Message Flow: TBD

Attester: device platform

Relying Party: internet peers

Claims used as evidence: TBD

Description: This use case convinces the relying party only about the provenance of a private key by providing claims of the storage security of the private key. This can be conceived as a subset of the previous case, but may be apply very specifically to just a keystore. Additional details associated with the private key may be provided as well, including limitations on usage of the key.

Key storage attestations may be consumed by systems provisioning public key certificates for devices or human users. In these cases, attestations may be incorporated into certificate request protocols (e.g., EST {#rfc7030}, CMP {#rfc4210}, ACME {#rfc8555}, SCEP [I-D.gutmann-scep], etc.) and processed by registration authorities or certification authorities prior to determining contents for any issued certificate.

5.7.3. End user authorization

Use case name: End User authorization

Who will use it: authorization systems

Message Flow: TBD

Attester: device platform

Relying Party: internet peers

Claims used as evidence: TBD

Description: This use case convinces the relying party that the digital signatures made by the indicated key pair were done with the approval of the end-user/device-operator. This may also be considered possible subset of the device attestation above, but the attestation may be on a case-by-case basis. The nature of the approval by the end-user would be indicated. Examples include: the user unlocked the device, the user viewed some message and acknowledge it inside an app, the message was displayed to the user via out-of-app control mechanism. The acknowledgements could include selecting options on the screen, pushing physical buttons, scanning fingerprints, proximity to other devices (via bluetooth beacons, chargers, etc)

5.8. Geographic attestation

Use case name: Location attestation

Who will use it: geo-fenced systems

Message Flow: passport (probably)

Attester: secure GPS system(s)

Relying Party: internet peers

Claims used as evidence: TBD

Description: The relying party wants to know the physical location (on the planet earth) of the device. This may be provided directly by a GPS/GLONASS/Galileo module that is incorporated into a TPM. This may also be provided by collecting other proximity messages from other device that the relying party can form a trust relationship with.

5.8.1. I am here

The simplest use case is the claim of some specific coordinates.

5.8.2. I am near

The second use case is the claim that some other devices are nearby. This may be absolute ("I am near device X, which claims to be at location A"), or just relative, ("I am near device X"). This use could use "I am here" or "I am near" claims from a 1:1 basis with device X, or use some other protocol. The nature of how the proximity was established would be part of this claim. In order to defeat a variety of mechanisms that might attempt to proxy ("wormhole") radio communications, highly precise clocks may be required, and there may also have to be attestations as to the precision of those clocks.

An additional example of being near would be for the case where two smartphones can establish that they are together by recording a common random movement, such as both devices being shaken together. Each device may validate the claim from the other (in a disconnected fashion), or a third party may validate the claim as the relying party.

This could be used to establish that a medical professional was in proximity of a patient with implanted devices who needs help.

5.8.3. You are here

A third way to establish location is for a third party to communicate directly with the relying party. The nature of how this trust is established (and whether it is done recursively) is outside of the scope here. What is critical is that the identity of "You" can be communicated through the third party in a way that the relying party can use, but other intermediaries can not view.

5.9. Connectivity attestation

Use case name: Connectivity attestation

Who will use it: entertainment systems

Message Flow: TBD

Attester: hardware-manufacturer/TEE

Relying Party: connected peer

Claims used as evidence: TBD

Description: The relying party wants to know what devices are connected. A typical situation would be a media owner needing to know what TV device is connected via HDMI and if High-bandwidth Digital Content Protection (HDCP) is intact.

5.10. Component connectivity attestation

Use case name: Component connectivity

Who will use it: chassis systems with pluggable components

Message Flow: background check

Attester: line card

Relying Party: management/control plane software

Claims used as evidence: TBD

Description: A management controller or similar hardware component wants to know what peripherals, rack scale device or other dynamically configurable components are currently attached to the platform that is under management controller control. The management controller may serve as attestation verifier over a local bus or backplane but may also aggregate local attestation results and act as a platform attester to a remote verifier.

5.11. Device provenance attestation

Use case name: RIV - Device Provenance

Who will use it: Industrial IoT devices

Message Flow: passport

Attester: network management station

Relying Party: a network entity

Claims used as evidence: TBD

Description: A newly manufactured device needs to be onboarded into a network where many if not all device management duties are performed by the network owner. The device owner wants to verify the device originated from a legitimate vendor. A cryptographic device identity such as an IEEE802.1AR is embedded during manufacturing and a certificate identifying the device is delivered to the owner onboarding agent. The device authenticates using its 802.1AR IDevID to prove it originated from the expected vendor.

The device chain of custody from the original device manufacturer to the new owner may also be verified as part of device provenance attestation. The chain of custody history may be collected by a cloud service or similar capability that the supply chain and owner agree to use.

[I-D.fedorkow-rats-network-device-attestation] section 1.2 refers to this as "Provable Device Identity", and section 2.3 details the parties.

5.12. DNS privacy policy

Use case name: DNS-over-TLS or DNS-over-HTTPS server privacy policy

Who will use it: enterprises and browsers and BYOD operating systems

Message Flow: passport

Attester: review agency

Relying Party: browsers and operating systems

Claims used as evidence: DNS server identity, privinfo (see draft-reddy-dprive-dprive-privacy-policy)

Description: Users want to control how their DNS queries are handled by DNS servers so they can configure their system to use DNS servers that comply with their privacy expectations.

This use case communicates an attestion from a DoH server to a web browser or equivalent in a desktop or mobile operating system. The attester is a third party which has performed some kind of review of the DNS server. This may include significant levels of Device Capability attestation as to what is running and how it is configured (see Section 5.1), in which case this is a form of Proxy Root of Trust (Section 5.1.3).

5.13. Safety Critical Systems

Use case name: Safety Critical Systems

Who will use it: Power plants and other systems that need to assert their current state, but which can not accept any inputs from the outside. The corollory system is a black-box (such as in an aircraft), which needs to log the state of a system, but which can never initiate a handshake.

Message Flow: background check

Attester: web services and other sources of status/sensor information

Relying Party: open

Claims used as evidence: the beginning and ending time as endorsed by a Time Stamp Authority, represented by a time stamp token. The real time clock of the system itself. A Root of Trust for time; the TPM has a relative time from startup.

Description: These requirements motivate the creation of the Time Base Unidirectional Attestation (TUDA) [I-D.birkholz-rats-tuda], the output of TUDA are typically a secure audit log, where freshness is determined by synchronization to an source of external time.

The freshness is preserved in the evidence by the use of a Time Stamp Authority (TSA) which provides Time Stamp Tokens (TST).

5.14. Trusted Path Routing

Use case name: Trusted Path Routing

Who will use it: Service Providers want to offer a trustworthy transport service to Government, Military, Financial, and Medical end-users.

Message Flow: background check model for a centralized controller based alternative, and passport model for a router/switch distributed alternative.

Attester: Routers/switches

Relying Party: Network Controllers and Peer Routers/Switches

Claims used as evidence: TPM Quotes, log entries passed into TPM PCRs, trustworthiness levels appraised by Verifiers, and included in passports.

Description: There are end-users who believe encryption technologies like IPsec alone are insufficient to protect the confidentiality of their highly sensitive traffic flows. These end-users want their sensitive flows to be forwarded across just those network devices currently appraised as trustworthy by the TCG-RIV use case.

[I-D.voit-rats-trusted-path-routing] discusses two alternatives for exchanging traffic with end-user customer identified "sensitive subnets". Traffic going to and from these subnets will transit a path where the IP layer and above are only interpretable by those network devices recently evaluated as trustworthy.

These two alternatives are:

Centralized Trusted Path Routing: For sensitive subnets, trusted end-to-end paths are pre-assigned through a network provider domain. Along these paths, attestation evidence of potentially transited components has been assessed. Each path is guaranteed to only include devices meeting the needs of a formally defined trustworthiness level.

Distributed Trusted Path Routing: Through the exchange of attestation evidence between peering network devices, a trusted topology is established and maintained. Only devices meeting the needs of a formally defined trustworthiness level are included as members of this topology. Traffic exchanged with sensitive subnets is forwarded into this topology.

6. Technology users for RATS.

6.1. Trusted Computing Group Remove Integrity Verification (TCG-RIV)

The TCG RIV Reference Document addresses the problem of knowing if a networking device should be part of a network, if it belongs to the operator, and if it is running appropriate software. The work covers most of the use cases in Section 5.1.

This proposal is available as [I-D.fedorkow-rats-network-device-attestation]. The goal is to be multi-vendor, scalable and extensible. The proposal intentionally limits itself to:

- "non-privacy-preserving applications (i.e., networking, Industrial IoT)",

- the firmware is provided by the device manufacturer

- there is a manufacturer installed hardware root of trust (such as a TPM and boot ROM)

Service providers and enterprises deploy hundreds of routers, many of them in remote locations where they're difficult to access or secure. The point of remote attestation is to:

- identify a remote box in a way that's hard to spoof

- report the inventory of software was launched on the box in a way that cannot be spoofed, that is undetectably altered by a "Lying Endpoint"

The use case described is to be able to monitor the authenticity of software versions and configurations running on each device. This allows owners and auditors to detect deviation from approved software and firmware versions and configurations, potentially identifying infected devices. [RFC5209]

Attestation may be performed by network management systems. Networking Equipment is often highly interconnected, so it's also possible that attestation could be performed by neighboring devices.

Specifically listed to be out of scope for the first generation includes: Linux processes, composite assemblies of hardware/software created by end-customers, and equipment that uses Sleep or Hibernate modes. There is an intention to cover some of these are topics in future versions of the documents.

The TCG-RIV Attestation leverages the TPM to make a series of measurements during the boot process, and to have the TPM sign those measurements. The resulting "PCR" hashes are then available to an external verifier.

A critical component of the RIV is compatibility with existing TPM practice for attestation proceedures, as spelled out in the TCG TAP Informational Model [tapinfomodel] and TPM architecture specifications [tpmarchspec].

The TCG uses the following terminology:

- Device Manufacturer

- Attester ("device under attestation")
- Verifier (Network Management Station)
- "Explicit Attestation" is the TCG term for a static (platform) attestation
- "Implicit Attestation" is the TCG term for a session attestation
- Reference Integrity Measurements (RIM), which are signed my device manufacturer and integrated into firmware.
- Quotes: measured values (having been signed), and RIMs
- Reference Integrity Values (RIV)
- devices have a Initial Attestation Key (IAK), which is provisioned at the same time as the IDevID [ieee802-1AR]
- PCR - Platform Configuration Registry (deals with hash chains)

The TCG document builds upon a number of IETF technologies: SNMP (Attestation MIB), YANG, XML, JSON, CBOR, NETCONF, RESTCONF, CoAP, TLS and SSH. The TCG document leverages the 802.1AR IDevID and LDevID processes.

6.2. Android Keystore system

[keystore] describes a system used in smart phones that run the Android operation system. The system is primarily a software container to contain and control access to cryptographic keys, and therefore provides many of the same functions that a hardware Trusted Platform Module might provide.

The uses described in section Section 5.7 are the primary focus.

On hardware which is supported, the Android Keystore will make use of whatever trusted hardware is available, including use of a Trusted Execution Environment (TEE) or Secure Element (SE). The Keystore therefore abstracts the hardware, and guarantees to applications that the same APIs can be used on both more and less capable devices.

A great deal of focus from the Android Keystore seems to be on providing fine-grained authorization of what keys can be used by which applications.

XXX - clearly there must be additional (intended?) use cases that provide some kind of attestation.

Android 9 on Pixel 2 and 3 can provided protected confirmation messages. This uses hardware access from the TPM/TEE to display a message directly to the user, and receives confirmation directly from the user. A hash of the contents of the message can provided in an attestation that the device provides.

In addition, the Android Keystore provides attestation information about itself for use by FIDO.

QUOTE: Finally, the Verified Boot state is included in key attestation certificates (provided by Keymaster/Strongbox) in the deviceLocked and verifiedBootState fields, which can be verified by apps as well as passed onto backend services to remotely verify boot integrity

6.3. Fast IDentity Online (FIDO) Alliance

The FIDO Alliance [fido] has a number of specifications aimed primarily at eliminating the need for passwords for authentication to online services. The goal is to leverage asymmetric cryptographic operations in common browser and smart-phone platforms so that users can easily authentication.

The use cases of Section 5.7 are primary.

FIDO specifications extend to various hardware second factor authentication devices.

Terminology includes:

- "relying party" validates a claim
- "relying party application" makes FIDO Authn calls
- "browser" provides the Web Authentication JS API
- "platform" is the base system
- "internal authenticator" is some credential built-in to the device
- "external authenticator" may be connected by USB, bluetooth, wifi, and may be an stand-alone device, USB connected key, phone or watch.

FIDO2 had a Key Attestation Format [fidoattestation], and a Signature Format [fidosignature], but these have been combined into the W3C document [fido_w3c] specification.

A FIDO use case involves the relying party receiving a device attestation about the biometric system that performs the identication of the human. It is the state of the biometric system that is being attested to, not the identity of the human!

FIDO does provides a transport in the form of the WebAuthn and FIDO CTAP protocols.

According to [fidotechnote] FIDO uses attestation to make claims about the kind of device which is be used to enroll. Keypairs are generated on a per-device _model_ basis, with a certificate having a trust chain that leads back to a well-known root certificate. It is expected that as many as 100,000 devices in a production run would have the same public and private key pair. One assumes that this is stored in a tamper-proof TPM so it is relatively difficult to get this key out. The use of this key attests to the the device type, and the kind of protections for keys that the relying party may assume, not to the identity of the end user.

7. Examples of Existing Attestation Formats.

This section provides examples of some existing attestation formats.

7.1. Android Keystore

Android Keystore attestations take the form of X.509 certificates. The examples below package the attestation certificate along with intermediate CA certificates required to validate the attestation as a certificates-only SignedData message [RFC5652]. The trust anchor is available here: [keystore_attestation].

The attestations below were generated using the generateKeyPair method from the DevicePolicyManager class using code similar to the following.

```
KeyGenParameterSpec.Builder builder = null;
if(hasStrongBox) {
        builder = new KeyGenParameterSpec.Builder(
                        m_alias,
                        KeyProperties.PURPOSE_SIGN | KeyProperties.PURPOSE_VERIFY
 | KeyProperties.PURPOSE_ENCRYPT | KeyProperties.PURPOSE_DECRYPT)
                        .setKeySize(2048)
                        .setDigests(KeyProperties.DIGEST_NONE, KeyProperties.DIGE
ST_SHA256)
                        .setBlockModes(KeyProperties.BLOCK_MODE_CBC, KeyPropertie
s.BLOCK_MODE_GCM)
                        .setEncryptionPaddings(KeyProperties.ENCRYPTION_PADDING_R
SA_PKCS1, KeyProperties.ENCRYPTION_PADDING_RSA_OAEP)
                        .setSignaturePaddings(KeyProperties.SIGNATURE_PADDING_RSA
_PSS, KeyProperties.SIGNATURE_PADDING_RSA_PKCS1)
                        .setUserAuthenticationRequired(false)
                        .setIsStrongBoxBacked(true)
                        .setUnlockedDeviceRequired(true);
}
else {
        builder = new KeyGenParameterSpec.Builder(
                        m_alias,
                        KeyProperties.PURPOSE_SIGN | KeyProperties.PURPOSE_VERIFY
 | KeyProperties.PURPOSE_ENCRYPT | KeyProperties.PURPOSE_DECRYPT)
                        .setKeySize(2048)
                        .setDigests(KeyProperties.DIGEST_NONE, KeyProperties.DIGE
ST_SHA256, KeyProperties.DIGEST_SHA384, KeyProperties.DIGEST_SHA512)
                        .setBlockModes(KeyProperties.BLOCK_MODE_CBC, KeyPropertie
s.BLOCK_MODE_CTR,KeyProperties.BLOCK_MODE_GCM)
                        .setEncryptionPaddings(KeyProperties.ENCRYPTION_PADDING_R
SA_PKCS1, KeyProperties.ENCRYPTION_PADDING_RSA_OAEP)
                        .setSignaturePaddings(KeyProperties.SIGNATURE_PADDING_RSA
_PSS, KeyProperties.SIGNATURE_PADDING_RSA_PKCS1)
                        .setUserAuthenticationRequired(false)
                        .setIsStrongBoxBacked(false)
                        .setUnlockedDeviceRequired(true);
}
builder.setAttestationChallenge(challenge_bytes);

KeyGenParameterSpec keySpec = builder.build();
AttestedKeyPair akp = dpm.generateKeyPair(componentName, algorithm, keySpec, idAt
testationFlags);
```

7.1.1. TEE

Annotations included below are delimited by ASN.1 comments, i.e., -. Annotations should be consistent with structures described here: [keystore_attestation].

```
   0 1172: SEQUENCE {
   4  764:   SEQUENCE {
   8    3:     [0] {
  10    1:       INTEGER 2
         :       }
  13    1:     INTEGER 1
  16   13:     SEQUENCE {
  18    9:       OBJECT IDENTIFIER
```

```
        :         sha256WithRSAEncryption (1 2 840 113549 1 1 11)
```

```
  29    0:        NULL
         :        }
  31   27:      SEQUENCE {
  33   25:        SET {
  35   23:          SEQUENCE {
  37    3:            OBJECT IDENTIFIER serialNumber (2 5 4 5)
  42   16:            PrintableString 'c6047571d8f0d17c'
         :            }
         :          }
         :        }
  60   32:      SEQUENCE {
  62   13:        UTCTime 01/01/1970 00:00:00 GMT
  77   15:        GeneralizedTime 07/02/2106 06:28:15 GMT
         :        }
  94   31:      SEQUENCE {
  96   29:        SET {
  98   27:          SEQUENCE {
 100    3:            OBJECT IDENTIFIER commonName (2 5 4 3)
 105   20:            UTF8String 'Android Keystore Key'
         :            }
         :          }
         :        }
 127  290:      SEQUENCE {
 131   13:        SEQUENCE {
 133    9:          OBJECT IDENTIFIER rsaEncryption (1 2 840 113549 1 1 1)
 144    0:          NULL
         :          }
 146  271:        BIT STRING, encapsulates {
 151  266:          SEQUENCE {
 155  257:            INTEGER
         :              00 B5 3A 83 61 A2 85 CC D2 D6 25 7F 07 0B B4 A0
         :              F6 FE 05 01 C9 55 CB 0D 18 D2 C6 79 BA 82 12 67
         :              75 8D 5B F3 24 D3 F8 EA 99 82 7D 1F 5E CD 77 D6
         :              99 11 13 FF 18 C9 3D 4D 01 C5 8E E9 04 E7 17 E2
         :              88 12 2B B9 A1 77 2F C2 4F 57 78 98 4E E3 DE 7A
         :              1B 18 BE D3 ED C9 59 A0 24 50 E1 FA AC 81 B6 DA
         :              80 B0 BD 48 AD 26 9C 4A 4E CE 54 17 58 C1 F4 F8
         :              7F 3C 5D 8F C8 2C 2A 7B 18 95 B3 D4 E0 3A C8 9D
         :                      [ Another 129 bytes skipped ]
 416    3:            INTEGER 65537
         :            }
         :          }
         :        }
 421  347:      [3] {
 425  343:        SEQUENCE {
 429   14:          SEQUENCE {
 431    3:            OBJECT IDENTIFIER keyUsage (2 5 29 15)
 436    1:            BOOLEAN TRUE
```

```
 439    4:           OCTET STRING, encapsulates {
 441    2:             BIT STRING 4 unused bits
        :               '1100'B
        :             }
        :           }
 445  323:         SEQUENCE {
 449   10:           OBJECT IDENTIFIER '1 3 6 1 4 1 11129 2 1 17'
 461  307:           OCTET STRING, encapsulates {  -- Attestation Extension
 465  303:             SEQUENCE {            -- KeyDescription
 469    1:               INTEGER 2           -- attestationVersion (KM3)
 472    1:               ENUMERATED 1        -- attestationSecurityLevel (TrustedE
nv.)
 475    1:               INTEGER 3           -- keymasterVersion
 478    1:               ENUMERATED 1        -- keymasterSecurityLevel (TrustedEnv
.)
 481    9:               OCTET STRING 'challenge'    -- attestationChallenge
 492    0:               OCTET STRING        -- reserved
        :                 Error: Object has zero length.
 494   44:               SEQUENCE {          -- softwareEnforced
 496    8:                 [701] {           -- creationDateTime
 500    6:                   INTEGER 01 64 47 2A 4B 64
        :                   }
 508   28:                 [709] {                -- attestationApplicationId
 512   26:                   OCTET STRING, encapsulates {
 514   24:                     SEQUENCE {         -- AttestationApplicationId
 516   20:                       SET {            -- package_infos
 518   18:                         SEQUENCE {    -- AttestationPackageInfo
 520   13:                           OCTET STRING 'AndroidSystem'  -- package_nam
e
 535    1:                           INTEGER 1   -- version
        :                           }
        :                         }
 538    0:                       SET {}     -- signature_digests
        :                       }
        :                     }
        :                   }
        :                 }
 540  229:               SEQUENCE {          -- hardwareEnforced
 543   14:                 [1] {             -- purpose
 545   12:                   SET {
 547    1:                     INTEGER 0    -- KeyPurpose.ENCRYPT
 550    1:                     INTEGER 1    -- KeyPurpose.DECRYPT
 553    1:                     INTEGER 2    -- KeyPurpose.SIGN
 556    1:                     INTEGER 3    -- KeyPurpose.VERIFY
        :                     }
        :                   }
 559    3:                 [2] {             -- algorithm
 561    1:                   INTEGER 1       -- Algorithm.RSA
        :                   }
 564    4:                 [3] {             -- keySize
 566    2:                   INTEGER 2048
```

```
        :                   }
 570   11:                 [5] {             -- digest
 572    9:                   SET {
 574    1:                     INTEGER 4    -- Digest.SHA256
 577    1:                     INTEGER 5    -- Digest.SHA384
 580    1:                     INTEGER 6    -- Digest.SHA512
        :                     }
        :                   }
 583   14:                 [6] {             -- padding
 585   12:                   SET {
 587    1:                     INTEGER 4    -- PaddingMode.RSA_PKCS1_1_5_ENCRYPT
 590    1:                     INTEGER 2    -- PaddingMode.RSA_OAEP
 593    1:                     INTEGER 3    -- PaddingMode.RSA_PKCS1_1_5_SIGN
 596    1:                     INTEGER 5    -- PaddingMode.RSA_PSS
        :                     }
        :                   }
 599    5:                 [200] {           -- rsaPublicExponent
 603    3:                   INTEGER 65537
        :                   }
 608    2:                 [503] {           -- noAuthRequired
 612    0:                   NULL            -- documentation indicates this is a
Boolean
        :                   }
 614    3:                 [702] {           -- origin
 618    1:                   INTEGER 0       -- KeyOrigin.GENERATED
        :                   }
 621    2:                 [703] {           -- rollbackResistant
 625    0:                   NULL            -- documentation indicates this is a
Boolean
        :                   }
 627   42:                 [704] {           -- rootOfTrust
 631   40:                   SEQUENCE {      -- verifiedBootKey
 633   32:                     OCTET STRING
        :                   19 62 B0 53 85 79 FF CE 9A C9 F5 07 C4 6A FE 3B
        :                   92 05 5B AC 71 46 46 22 83 C8 5C 50 0B E7 8D 82
 667    1:                     BOOLEAN TRUE -- deviceLocked
 670    1:                     ENUMERATED 0 -- verifiedBootState (verified)
        :                     }
        :                   }
 673    5:                 [705] {           -- osVersion
 677    3:                   INTEGER 90000  -- Android P
        :                   }
 682    5:                 [706] {           -- osPatchLevel
 686    3:                   INTEGER 201806 -- June 2018
        :                   }
 691    8:                 [710] {           -- attestationIdBrand
 695    6:                   OCTET STRING 'google'
        :                   }
 703    9:                 [711] {           -- attestationIdDevice
 707    7:                   OCTET STRING 'walleye'
```

```
        :                   }
 716    9:                 [712] {          -- attestationIdProduct
 720    7:                   OCTET STRING 'walleye'
        :                   }
 729   14:                 [713] {          -- attestationIdSerial
 733   12:                   OCTET STRING 'HT83K1A03849'
        :                   }
 747    8:                 [716] {          -- attestationIdManufacturer
 751    6:                   OCTET STRING 'Google'
        :                   }
 759    9:                 [717] {          -- attestationIdModel
 763    7:                   OCTET STRING 'Pixel 2'
        :                   }
        :                 }
        :               }
        :             }
        :           }
        :         }
        :       }
        :     }
 772   13:   SEQUENCE {
 774    9:     OBJECT IDENTIFIER sha256WithRSAEncryption (1 2 840 113549 1 1 11)
 785    0:     NULL
        :     }
 787  385:   BIT STRING
        :     05 41 B9 13 11 53 93 A2 02 62 1F 15 35 8E D9 7C
        :     A1 D5 2E ED 13 AC 24 26 B2 A1 2F EE B4 0C 4D 71
        :     DC 9F 55 EC A1 F6 64 62 F2 73 A8 7E FC 48 63 29
        :     1E F5 0D 48 F3 73 43 0C 00 E0 D4 07 86 A6 A4 38
        :     0E A8 47 0F 27 01 01 31 52 F6 62 8A 4B 80 BE 72
        :     FB 02 E7 56 84 CA CA 4D C3 6C 7C B2 BA C7 D7 9B
        :     C5 9D 90 65 4E F5 54 8F 25 CC 11 7F 8E 77 10 6A
        :     6E 9F 80 89 48 8B 1D 51 AA 3B B7 C5 24 3C 28 B1
        :             [ Another 256 bytes skipped ]
        :   }
   0 1304: SEQUENCE {
   4  768:   SEQUENCE {
   8    3:     [0] {
  10    1:       INTEGER 2
        :       }
  13   10:     INTEGER 10 34 53 32 94 08 68 79 38 72
  25   13:     SEQUENCE {
  27    9:       OBJECT IDENTIFIER
        :         sha256WithRSAEncryption (1 2 840 113549 1 1 11)
  38    0:       NULL
        :       }
  40   27:     SEQUENCE {
  42   25:       SET {
```

```
  44   23:         SEQUENCE {
  46    3:           OBJECT IDENTIFIER serialNumber (2 5 4 5)
  51   16:           PrintableString '87f4514475ba0a2b'
         :           }
         :         }
         :       }
  69   30:     SEQUENCE {
  71   13:       UTCTime 26/05/2016 17:14:51 GMT
  86   13:       UTCTime 24/05/2026 17:14:51 GMT
         :       }
 101   27:     SEQUENCE {
 103   25:       SET {
 105   23:         SEQUENCE {
 107    3:           OBJECT IDENTIFIER serialNumber (2 5 4 5)
 112   16:           PrintableString 'c6047571d8f0d17c'
         :           }
         :         }
         :       }
 130  418:     SEQUENCE {
 134   13:       SEQUENCE {
 136    9:         OBJECT IDENTIFIER rsaEncryption (1 2 840 113549 1 1 1)
 147    0:         NULL
         :         }
 149  399:       BIT STRING, encapsulates {
 154  394:         SEQUENCE {
 158  385:           INTEGER
         :             00 B3 01 0D 78 BC 06 33 25 CA D6 A7 2C EF 49 05
         :             4C C1 77 36 F2 E5 7B E8 4C 0A 87 8F 77 6A 09 45
         :             9B AC E8 72 DA E2 0E 20 3D 68 30 A5 86 26 14 77
         :             AD 7E 93 F5 1D 38 A9 DB 5B FE B2 B8 1A 7B CD 22
         :             3B 17 98 FC 1F 4F 77 2D 92 E9 DE 5F 6B 02 09 4E
         :             99 86 53 98 1C 5E 23 B6 A4 61 53 A5 FB D1 37 09
         :             DB C0 0A 40 E9 28 E6 BE E2 8E 57 94 A9 F2 13 3A
         :             11 40 D2 34 99 A6 B4 F3 99 F2 5D 4A 5D 6A 6C 4B
         :                     [ Another 257 bytes skipped ]
 547    3:           INTEGER 65537
         :           }
         :         }
         :       }
 552  221:     [3] {
 555  218:       SEQUENCE {
 558   29:         SEQUENCE {
 560    3:           OBJECT IDENTIFIER subjectKeyIdentifier (2 5 29 14)
 565   22:           OCTET STRING, encapsulates {
 567   20:             OCTET STRING
         :               7B 7B F8 43 CA 1F 0F 96 27 0F 10 6F 7D 0C 23 14
         :               72 8F 1D 80
         :             }
```

```
          :            }
 589   31:           SEQUENCE {
 591    3:             OBJECT IDENTIFIER authorityKeyIdentifier (2 5 29 35)
 596   24:             OCTET STRING, encapsulates {
 598   22:               SEQUENCE {
 600   20:                 [0]
          :                   0E 55 6F 46 F5 3B 77 67 E1 B9 73 DC 55 E6 AE EA
          :                   B4 FD 27 DD
          :                 }
          :               }
          :             }
 622   12:           SEQUENCE {
 624    3:             OBJECT IDENTIFIER basicConstraints (2 5 29 19)
 629    1:             BOOLEAN TRUE
 632    2:             OCTET STRING, encapsulates {
 634    0:               SEQUENCE {}
          :               }
          :             }
 636   14:           SEQUENCE {
 638    3:             OBJECT IDENTIFIER keyUsage (2 5 29 15)
 643    1:             BOOLEAN TRUE
 646    4:             OCTET STRING, encapsulates {
 648    2:               BIT STRING 7 unused bits
          :                 '1'B (bit 0)
          :               }
          :             }
 652   36:           SEQUENCE {
 654    3:             OBJECT IDENTIFIER nameConstraints (2 5 29 30)
 659   29:             OCTET STRING, encapsulates {
 661   27:               SEQUENCE {
 663   25:                 [0] {
 665   23:                   SEQUENCE {
 667   21:                     [2] 'invalid;email:invalid'
          :                     }
          :                   }
          :                 }
          :               }
          :             }
 690   84:           SEQUENCE {
 692    3:             OBJECT IDENTIFIER cRLDistributionPoints (2 5 29 31)
 697   77:             OCTET STRING, encapsulates {
 699   75:               SEQUENCE {
 701   73:                 SEQUENCE {
 703   71:                   [0] {
 705   69:                     [0] {
 707   67:                       [6]
          :                     'https://android.googleapis.com/attestation/crl/1'
          :                     '0345332940868793872'
```

```
         :                     }
         :                   }
         :                 }
         :               }
         :             }
         :           }
         :         }
         :       }
         :     }
 776   13:   SEQUENCE {
 778    9:     OBJECT IDENTIFIER sha256WithRSAEncryption (1 2 840 113549 1 1 11)
 789    0:     NULL
         :     }
 791  513:   BIT STRING
         :     69 13 A7 56 B3 9F E1 2B CE A2 09 89 E5 DC 03 B4
         :     B6 FF F6 1E 96 C7 62 C2 31 D1 B3 D6 1A 9E 36 CF
         :     C2 FC 0E 06 FA 0E CF B5 2D F8 19 D6 13 96 0B 56
         :     B0 EE 86 3B B1 B8 38 70 4E 57 EB D9 60 DC 58 74
         :     FE C8 EB A5 78 9F B7 19 5C F0 80 CF 29 16 6B 04
         :     3A 5D 7C 2E 5F 11 12 36 BE 46 29 45 04 41 8F B5
         :     AB C6 31 5F 23 28 0C F2 7C 48 4A F6 43 AA 50 D0
         :     53 96 1E AD 7C A3 89 96 BB 8B BF 2D 9A 0C 16 35
         :             [ Another 384 bytes skipped ]
         :   }
   0 1393: SEQUENCE {
   4  857:   SEQUENCE {
   8    3:     [0] {
  10    1:       INTEGER 2
         :       }
  13   10:     INTEGER 03 88 26 67 60 65 89 96 85 74
  25   13:     SEQUENCE {
  27    9:       OBJECT IDENTIFIER
         :         sha256WithRSAEncryption (1 2 840 113549 1 1 11)
  38    0:       NULL
         :       }
  40   27:     SEQUENCE {
  42   25:       SET {
  44   23:         SEQUENCE {
  46    3:           OBJECT IDENTIFIER serialNumber (2 5 4 5)
  51   16:           PrintableString 'f92009e853b6b045'
         :           }
         :         }
         :       }
  69   30:     SEQUENCE {
  71   13:       UTCTime 26/05/2016 17:01:32 GMT
  86   13:       UTCTime 24/05/2026 17:01:32 GMT
         :       }
 101   27:     SEQUENCE {
```

```
 103   25:       SET {
 105   23:         SEQUENCE {
 107    3:           OBJECT IDENTIFIER serialNumber (2 5 4 5)
 112   16:           PrintableString '87f4514475ba0a2b'
         :           }
         :         }
         :       }
 130  546:     SEQUENCE {
 134   13:       SEQUENCE {
 136    9:         OBJECT IDENTIFIER rsaEncryption (1 2 840 113549 1 1 1)
 147    0:         NULL
         :         }
 149  527:       BIT STRING, encapsulates {
 154  522:         SEQUENCE {
 158  513:           INTEGER
         :             00 D2 60 D6 45 85 E3 E2 23 79 5A DA 45 57 A7 D8
         :             5B AF BD 9A 37 CB FA 97 C0 65 44 9D 3A C6 47 F6
         :             0D 0B A2 74 12 CA F7 4B B9 5F FB B4 EC 5A 2B D0
         :             16 01 DE BE E2 FE D2 76 0D 75 C4 B1 6A CB 3A 67
         :             07 21 E0 D5 19 68 C8 1B 01 A2 24 02 FE AD 40 D6
         :             A7 98 16 0F A2 98 2E A7 AD 75 34 84 6F F8 CF 8A
         :             A1 0E 90 33 40 9E D0 86 26 57 71 CE FF CF 52 E1
         :             F0 F9 2B 7E 68 62 03 D8 FD FD 02 53 03 19 AC 28
         :                     [ Another 385 bytes skipped ]
 675    3:           INTEGER 65537
         :           }
         :         }
         :       }
 680  182:     [3] {
 683  179:       SEQUENCE {
 686   29:         SEQUENCE {
 688    3:           OBJECT IDENTIFIER subjectKeyIdentifier (2 5 29 14)
 693   22:           OCTET STRING, encapsulates {
 695   20:             OCTET STRING
         :               0E 55 6F 46 F5 3B 77 67 E1 B9 73 DC 55 E6 AE EA
         :               B4 FD 27 DD
         :             }
         :           }
 717   31:         SEQUENCE {
 719    3:           OBJECT IDENTIFIER authorityKeyIdentifier (2 5 29 35)
 724   24:           OCTET STRING, encapsulates {
 726   22:             SEQUENCE {
 728   20:               [0]
         :                 36 61 E1 00 7C 88 05 09 51 8B 44 6C 47 FF 1A 4C
         :                 C9 EA 4F 12
         :               }
         :             }
         :           }
```

```
 750   15:         SEQUENCE {
 752    3:           OBJECT IDENTIFIER basicConstraints (2 5 29 19)
 757    1:           BOOLEAN TRUE
 760    5:           OCTET STRING, encapsulates {
 762    3:             SEQUENCE {
 764    1:               BOOLEAN TRUE
          :               }
          :             }
          :           }
 767   14:         SEQUENCE {
 769    3:           OBJECT IDENTIFIER keyUsage (2 5 29 15)
 774    1:           BOOLEAN TRUE
 777    4:           OCTET STRING, encapsulates {
 779    2:             BIT STRING 1 unused bit
          :               '1100001'B
          :             }
          :           }
 783   80:         SEQUENCE {
 785    3:           OBJECT IDENTIFIER cRLDistributionPoints (2 5 29 31)
 790   73:           OCTET STRING, encapsulates {
 792   71:             SEQUENCE {
 794   69:               SEQUENCE {
 796   67:                 [0] {
 798   65:                   [0] {
 800   63:                     [6]
          :                   'https://android.googleapis.com/attestation/crl/E'
          :                   '8FA196314D2FA18'
          :                     }
          :                   }
          :                 }
          :               }
          :             }
          :           }
          :         }
          :       }
          :     }
 865   13:   SEQUENCE {
 867    9:     OBJECT IDENTIFIER sha256WithRSAEncryption (1 2 840 113549 1 1 11)
 878    0:     NULL
          :     }
 880  513:   BIT STRING
          :     0E 0D 71 4A 88 0A 58 53 B6 31 14 7D DA 22 31 C6
          :     06 D6 EF 3B 22 4D D7 A5 C0 3F BF C6 B4 64 A3 FB
          :     92 C2 CC 67 F4 6C 24 25 49 6E F6 CB 08 D6 A8 0D
          :     94 06 7F 8C 8C 3C B1 77 CD C2 3F C7 5E A3 85 6D
          :     F7 A5 94 13 CD 5A 5C F3 9B 0A 0D E1 82 42 F4 C9
          :     3F AD FC FB 7C AA 27 04 CC 1C 12 45 15 EB E6 70
          :     A0 6C DE 77 77 54 9B 1F 02 05 76 03 A4 FC 6C 07
```

```
         :     F4 CB BB 59 F5 CB ED 58 D8 30 9B 6E 3C F7 76 C1
         :             [ Another 384 bytes skipped ]
         :   }
   0 1376: SEQUENCE {
   4  840:   SEQUENCE {
   8    3:     [0] {
  10    1:       INTEGER 2
         :       }
  13    9:     INTEGER 00 E8 FA 19 63 14 D2 FA 18
  24   13:     SEQUENCE {
  26    9:       OBJECT IDENTIFIER
         :         sha256WithRSAEncryption (1 2 840 113549 1 1 11)
  37    0:       NULL
         :       }
  39   27:     SEQUENCE {
  41   25:       SET {
  43   23:         SEQUENCE {
  45    3:           OBJECT IDENTIFIER serialNumber (2 5 4 5)
  50   16:           PrintableString 'f92009e853b6b045'
         :           }
         :         }
         :       }
  68   30:     SEQUENCE {
  70   13:       UTCTime 26/05/2016 16:28:52 GMT
  85   13:       UTCTime 24/05/2026 16:28:52 GMT
         :       }
 100   27:     SEQUENCE {
 102   25:       SET {
 104   23:         SEQUENCE {
 106    3:           OBJECT IDENTIFIER serialNumber (2 5 4 5)
 111   16:           PrintableString 'f92009e853b6b045'
         :           }
         :         }
         :       }
 129  546:     SEQUENCE {
 133   13:       SEQUENCE {
 135    9:         OBJECT IDENTIFIER rsaEncryption (1 2 840 113549 1 1 1)
 146    0:         NULL
         :         }
 148  527:       BIT STRING, encapsulates {
 153  522:         SEQUENCE {
 157  513:           INTEGER
         :             00 AF B6 C7 82 2B B1 A7 01 EC 2B B4 2E 8B CC 54
         :             16 63 AB EF 98 2F 32 C7 7F 75 31 03 0C 97 52 4B
         :             1B 5F E8 09 FB C7 2A A9 45 1F 74 3C BD 9A 6F 13
         :             35 74 4A A5 5E 77 F6 B6 AC 35 35 EE 17 C2 5E 63
         :             95 17 DD 9C 92 E6 37 4A 53 CB FE 25 8F 8F FB B6
         :             FD 12 93 78 A2 2A 4C A9 9C 45 2D 47 A5 9F 32 01
```

```
        :             F4 41 97 CA 1C CD 7E 76 2F B2 F5 31 51 B6 FE B2
        :             FF FD 2B 6F E4 FE 5B C6 BD 9E C3 4B FE 08 23 9D
        :                     [ Another 385 bytes skipped ]
 674    3:           INTEGER 65537
        :           }
        :         }
        :       }
 679  166:     [3] {
 682  163:       SEQUENCE {
 685   29:         SEQUENCE {
 687    3:           OBJECT IDENTIFIER subjectKeyIdentifier (2 5 29 14)
 692   22:           OCTET STRING, encapsulates {
 694   20:             OCTET STRING
        :               36 61 E1 00 7C 88 05 09 51 8B 44 6C 47 FF 1A 4C
        :               C9 EA 4F 12
        :             }
        :           }
 716   31:         SEQUENCE {
 718    3:           OBJECT IDENTIFIER authorityKeyIdentifier (2 5 29 35)
 723   24:           OCTET STRING, encapsulates {
 725   22:             SEQUENCE {
 727   20:               [0]
        :                 36 61 E1 00 7C 88 05 09 51 8B 44 6C 47 FF 1A 4C
        :                 C9 EA 4F 12
        :               }
        :             }
        :           }
 749   15:         SEQUENCE {
 751    3:           OBJECT IDENTIFIER basicConstraints (2 5 29 19)
 756    1:           BOOLEAN TRUE
 759    5:           OCTET STRING, encapsulates {
 761    3:             SEQUENCE {
 763    1:               BOOLEAN TRUE
        :               }
        :             }
        :           }
 766   14:         SEQUENCE {
 768    3:           OBJECT IDENTIFIER keyUsage (2 5 29 15)
 773    1:           BOOLEAN TRUE
 776    4:           OCTET STRING, encapsulates {
 778    2:             BIT STRING 1 unused bit
        :               '1100001'B
        :             }
        :           }
 782   64:         SEQUENCE {
 784    3:           OBJECT IDENTIFIER cRLDistributionPoints (2 5 29 31)
 789   57:           OCTET STRING, encapsulates {
 791   55:             SEQUENCE {
```

```
 793   53:                SEQUENCE {
 795   51:                  [0] {
 797   49:                    [0] {
 799   47:                      [6]
         :                    'https://android.googleapis.com/attestation/crl/'
         :                      }
         :                    }
         :                  }
         :                }
         :              }
         :            }
         :          }
         :        }
         :      }
 848   13:    SEQUENCE {
 850    9:      OBJECT IDENTIFIER sha256WithRSAEncryption (1 2 840 113549 1 1 11)
 861    0:      NULL
         :      }
 863  513:    BIT STRING
         :      20 C8 C3 8D 4B DC A9 57 1B 46 8C 89 2F FF 72 AA
         :      C6 F8 44 A1 1D 41 A8 F0 73 6C C3 7D 16 D6 42 6D
         :      8E 7E 94 07 04 4C EA 39 E6 8B 07 C1 3D BF 15 03
         :      DD 5C 85 BD AF B2 C0 2D 5F 6C DB 4E FA 81 27 DF
         :      8B 04 F1 82 77 0F C4 E7 74 5B 7F CE AA 87 12 9A
         :      88 01 CE 8E 9B C0 CB 96 37 9B 4D 26 A8 2D 30 FD
         :      9C 2F 8E ED 6D C1 BE 2F 84 B6 89 E4 D9 14 25 8B
         :      14 4B BA E6 24 A1 C7 06 71 13 2E 2F 06 16 A8 84
         :              [ Another 384 bytes skipped ]
         :    }
```

7.1.2. Secure Element

The structures below are not annotated except where the difference is specific to the difference between the TEE structure shown above and artifacts emitted by StrongBox.

```
   0 5143: SEQUENCE {
   4    9:   OBJECT IDENTIFIER signedData (1 2 840 113549 1 7 2)
  15 5128:   [0] {
  19 5124:     SEQUENCE {
  23    1:       INTEGER 1
  26    0:       SET {}
  28   11:       SEQUENCE {
  30    9:         OBJECT IDENTIFIER data (1 2 840 113549 1 7 1)
         :         }
  41 5100:       [0] {
  45 1114:         SEQUENCE {
  49  834:           SEQUENCE {
```

```
  53    3:             [0] {
  55    1:               INTEGER 2
        :               }
  58    1:             INTEGER 1
  61   13:             SEQUENCE {
  63    9:               OBJECT IDENTIFIER
        :                 sha256WithRSAEncryption (1 2 840 113549 1 1 11)
  74    0:               NULL
        :               }
  76   47:             SEQUENCE {
  78   25:               SET {
  80   23:                 SEQUENCE {
  82    3:                   OBJECT IDENTIFIER serialNumber (2 5 4 5)
  87   16:                   PrintableString '90e8da3cadfc7820'
        :                   }
        :                 }
 105   18:               SET {
 107   16:                 SEQUENCE {
 109    3:                   OBJECT IDENTIFIER title (2 5 4 12)
 114    9:                   UTF8String 'StrongBox'
        :                   }
        :                 }
        :               }
 125   30:             SEQUENCE {
 127   13:               UTCTime 01/01/1970 00:00:00 GMT
 142   13:               UTCTime 23/05/2028 23:59:59 GMT
        :               }
 157   31:             SEQUENCE {
 159   29:               SET {
 161   27:                 SEQUENCE {
 163    3:                   OBJECT IDENTIFIER commonName (2 5 4 3)
 168   20:                   UTF8String 'Android Keystore Key'
        :                   }
        :                 }
        :               }
 190  290:             SEQUENCE {
 194   13:               SEQUENCE {
 196    9:                 OBJECT IDENTIFIER
        :                   rsaEncryption (1 2 840 113549 1 1 1)
 207    0:                 NULL
        :                 }
 209  271:               BIT STRING, encapsulates {
 214  266:                 SEQUENCE {
 218  257:                   INTEGER
        :                   00 DE 98 94 D5 E5 05 98 E8 FC 73 4D 26 FB 48 6A
        :                   CA 06 A0 24 FA 05 D1 D2 32 10 46 F8 50 DD 3E 0D
        :                   DF 4F 95 53 D2 CB 10 1F 00 B2 62 15 1E 21 7E 05
        :                   C6 10 AC EE 7A D8 69 F1 1F 32 C3 17 CA D7 07 BE
```

```
        :                    3B 2B 83 0F B4 9C 3D C7 13 0B 9C 59 2F 1A 38 CE
        :                    A5 1D 95 A7 3C EE 70 6A CF 41 FF 55 3F E0 9C 69
        :                    E5 A0 C1 19 EF 40 E9 40 FC 74 D3 3B 96 D9 0E C1
        :                    C3 9D 14 10 0C A6 95 19 49 88 F4 AB 74 FC 86 A6
        :                            [ Another 129 bytes skipped ]
 479    3:                   INTEGER 65537
        :                    }
        :                  }
        :                }
 484  399:             [3] {
 488  395:               SEQUENCE {
 492   14:                 SEQUENCE {
 494    3:                   OBJECT IDENTIFIER keyUsage (2 5 29 15)
 499    1:                   BOOLEAN TRUE
 502    4:                   OCTET STRING, encapsulates {
 504    2:                     BIT STRING 7 unused bits
        :                       '1'B (bit 0)
        :                     }
        :                   }
 508  375:                 SEQUENCE {
 512   10:                   OBJECT IDENTIFIER '1 3 6 1 4 1 11129 2 1 17'
 524  359:                   OCTET STRING, encapsulates {
 528  355:                     SEQUENCE {
 532    1:                       INTEGER 3
 535    1:                       ENUMERATED 2   -- attestationSecurityLevel (Stro
ngBox)
 538    1:                       INTEGER 4
 541    1:                       ENUMERATED 2   -- attestationSecurityLevel (Stro
ngBox)
 544    9:                       OCTET STRING 'challenge'
 555    0:                       OCTET STRING
        :                         Error: Object has zero length.
 557   53:                       SEQUENCE {
 559    2:                         [509] {
 563    0:                           NULL
        :                           }
 565   11:                         [701] {
 569    9:                           INTEGER 00 FF FF FF FF FF E5 99 78
        :                           }
 580   28:                         [709] {
 584   26:                           OCTET STRING, encapsulates {
 586   24:                             SEQUENCE {
 588   20:                               SET {
 590   18:                                 SEQUENCE {
 592   13:                                   OCTET STRING 'AndroidSystem'
 607    1:                                   INTEGER 1
        :                                   }
        :                                 }
 610    0:                               SET {}
        :                               }
```

```
         :                             }
         :                           }
         :                         }
 612  271:                       SEQUENCE {
 616   14:                         [1] {
 618   12:                           SET {
 620    1:                             INTEGER 0
 623    1:                             INTEGER 1
 626    1:                             INTEGER 2
 629    1:                             INTEGER 3
         :                             }
         :                           }
 632    3:                         [2] {
 634    1:                           INTEGER 1
         :                           }
 637    4:                         [3] {
 639    2:                           INTEGER 2048
         :                           }
 643    8:                         [4] {
 645    6:                           SET {
 647    1:                             INTEGER 2
 650    1:                             INTEGER 32
         :                             }
         :                           }
 653    8:                         [5] {
 655    6:                           SET {
 657    1:                             INTEGER 0
 660    1:                             INTEGER 4
         :                             }
         :                           }
 663   14:                         [6] {
 665   12:                           SET {
 667    1:                             INTEGER 2
 670    1:                             INTEGER 3
 673    1:                             INTEGER 4
 676    1:                             INTEGER 5
         :                             }
         :                           }
 679    2:                         [503] {
 683    0:                           NULL
         :                           }
 685    3:                         [702] {
 689    1:                           INTEGER 0
         :                           }
 692   76:                         [704] {
 696   74:                           SEQUENCE {
 698   32:                             OCTET STRING
         :                 61 FD A1 2B 32 ED 84 21 4A 9C F1 3D 1A FF B7 AA
```

```
         :                   80 BD 8A 26 8A 86 1E D4 BB 7A 15 17 0F 1A B0 0C
 732    1:                             BOOLEAN TRUE
 735    1:                             ENUMERATED 0
 738   32:                             OCTET STRING
         :                   77 96 C5 3D 0E 09 46 2B BA BB FB 7B 8A 65 F6 8D
         :                   EF 5C 46 88 BF 99 C4 1E 88 42 01 4D 1F 01 2D C5
         :                             }
         :                           }
 772    3:                         [705] {
 776    1:                           INTEGER 0
         :                           }
 779    5:                         [706] {
 783    3:                           INTEGER 201903
         :                           }
 788    8:                         [710] {
 792    6:                           OCTET STRING 'google'
         :                           }
 800   10:                         [711] {
 804    8:                           OCTET STRING 'blueline'
         :                           }
 814   10:                         [712] {
 818    8:                           OCTET STRING 'blueline'
         :                           }
 828   11:                         [713] {
 832    9:                           OCTET STRING '8A2X0KLUU'
         :                           }
 843    8:                         [716] {
 847    6:                           OCTET STRING 'Google'
         :                           }
 855    9:                         [717] {
 859    7:                           OCTET STRING 'Pixel 3'
         :                           }
 868    6:                         [718] {
 872    4:                           INTEGER 20180905
         :                           }
 878    5:                         [719] {
 882    3:                           INTEGER 201903
         :                           }
         :                         }
         :                       }
         :                     }
         :                   }
         :                 }
         :               }
         :             }
 887   13:           SEQUENCE {
 889    9:             OBJECT IDENTIFIER
         :               sha256WithRSAEncryption (1 2 840 113549 1 1 11)
```

```
 900    0:               NULL
        :               }
 902  257:             BIT STRING
        :               83 EA 59 8D BE 37 4A D5 C0 FC F8 FB AC 8B 72 1E
        :               A5 C2 3B 0C C0 04 1B C0 5A 18 A5 DF D4 67 1D B9
        :               08 42 4B E2 2C AC 07 0F D8 0E 24 97 56 9E 14 F2
        :               D0 AC DD 1E FC DD 68 20 11 DF 88 B8 B6 22 AD 2B
        :               DB 9C 2E 5C 3F AF 0B 8F 02 68 AA 34 4B 5E C8 75
        :               B1 1A 09 D2 19 41 24 61 65 97 2C 0D A4 78 43 A7
        :               9A 27 B2 4E 24 11 4F FF E2 D8 04 56 39 75 B2 34
        :               D8 18 C7 25 F3 3F C0 6A 37 AB 49 B6 96 51 61 72
        :                       [ Another 128 bytes skipped ]
        :             }
1163 1181:          SEQUENCE {
1167  645:            SEQUENCE {
1171    3:              [0] {
1173    1:                INTEGER 2
        :                }
1176   10:              INTEGER 17 10 24 68 40 71 02 97 78 50
1188   13:              SEQUENCE {
1190    9:                OBJECT IDENTIFIER
        :                  sha256WithRSAEncryption (1 2 840 113549 1 1 11)
1201    0:                NULL
        :                }
1203   47:              SEQUENCE {
1205   25:                SET {
1207   23:                  SEQUENCE {
1209    3:                    OBJECT IDENTIFIER serialNumber (2 5 4 5)
1214   16:                    PrintableString 'ccd18b9b608d658e'
        :                    }
        :                  }
1232   18:                SET {
1234   16:                  SEQUENCE {
1236    3:                    OBJECT IDENTIFIER title (2 5 4 12)
1241    9:                    UTF8String 'StrongBox'
        :                    }
        :                  }
        :                }
1252   30:              SEQUENCE {
1254   13:                UTCTime 25/05/2018 23:28:47 GMT
1269   13:                UTCTime 22/05/2028 23:28:47 GMT
        :                }
1284   47:              SEQUENCE {
1286   25:                SET {
1288   23:                  SEQUENCE {
1290    3:                    OBJECT IDENTIFIER serialNumber (2 5 4 5)
1295   16:                    PrintableString '90e8da3cadfc7820'
        :                    }
```

```
         :               }
1313   18:             SET {
1315   16:               SEQUENCE {
1317    3:                 OBJECT IDENTIFIER title (2 5 4 12)
1322    9:                 UTF8String 'StrongBox'
         :                 }
         :               }
         :             }
1333  290:           SEQUENCE {
1337   13:             SEQUENCE {
1339    9:               OBJECT IDENTIFIER
         :                 rsaEncryption (1 2 840 113549 1 1 1)
1350    0:               NULL
         :               }
1352  271:             BIT STRING, encapsulates {
1357  266:               SEQUENCE {
1361  257:                 INTEGER
         :                 00 A5 09 D4 09 D2 30 19 36 34 71 FD 7D 41 89 E6
         :                 2C A5 9D 10 1B 4F 40 6A B0 5F 56 34 16 E6 EB D7
         :                 F3 E9 C5 DC 20 F3 86 D1 77 19 D7 15 1F E7 EC 62
         :                 DC 0A BC 64 E9 18 52 B0 AA B8 FF 58 6A E0 0F B8
         :                 56 AF 77 D3 CE 3C DC 48 52 DD B2 86 0D 76 17 7C
         :                 FD EE B4 E6 6E 0A 08 9E 06 CA 0F EC 4B B0 7C AF
         :                 EA 82 27 A8 C9 A7 63 DA 89 F6 30 BA 3C 3A E5 C6
         :                 EF 11 06 42 8A 2E FE 19 BE F2 C7 3B 34 16 B2 E2
         :                         [ Another 129 bytes skipped ]
1622    3:                 INTEGER 65537
         :                 }
         :               }
         :             }
1627  186:           [3] {
1630  183:             SEQUENCE {
1633   29:               SEQUENCE {
1635    3:                 OBJECT IDENTIFIER subjectKeyIdentifier (2 5 29 14)
1640   22:                 OCTET STRING, encapsulates {
1642   20:                   OCTET STRING
         :                 77 A4 AD DF 1D 29 89 CA 92 E3 BA DE 27 3C 70 DF
         :                 36 03 7C 0C
         :                   }
         :                 }
1664   31:               SEQUENCE {
1666    3:                 OBJECT IDENTIFIER
         :                   authorityKeyIdentifier (2 5 29 35)
1671   24:                 OCTET STRING, encapsulates {
1673   22:                   SEQUENCE {
1675   20:                     [0]
         :                 1B 17 70 C6 97 DC 84 54 75 7C 3C 98 5C E6 1D 1D
         :                 08 59 5D 53
```

```
         :                      }
         :                    }
         :                  }
1697   15:                SEQUENCE {
1699    3:                  OBJECT IDENTIFIER basicConstraints (2 5 29 19)
1704    1:                  BOOLEAN TRUE
1707    5:                  OCTET STRING, encapsulates {
1709    3:                    SEQUENCE {
1711    1:                      BOOLEAN TRUE
         :                      }
         :                    }
         :                  }
1714   14:                SEQUENCE {
1716    3:                  OBJECT IDENTIFIER keyUsage (2 5 29 15)
1721    1:                  BOOLEAN TRUE
1724    4:                  OCTET STRING, encapsulates {
1726    2:                    BIT STRING 2 unused bits
         :                      '100000'B (bit 5)
         :                    }
         :                  }
1730   84:                SEQUENCE {
1732    3:                  OBJECT IDENTIFIER cRLDistributionPoints (2 5 29 31)
1737   77:                  OCTET STRING, encapsulates {
1739   75:                    SEQUENCE {
1741   73:                      SEQUENCE {
1743   71:                        [0] {
1745   69:                          [0] {
1747   67:                            [6]
         :                  'https://android.googleapis.com/attestation/crl/1'
         :                  '7102468407102977850'
         :                            }
         :                          }
         :                        }
         :                      }
         :                    }
         :                  }
         :                }
         :              }
         :            }
1816   13:          SEQUENCE {
1818    9:            OBJECT IDENTIFIER
         :              sha256WithRSAEncryption (1 2 840 113549 1 1 11)
1829    0:            NULL
         :            }
1831  513:          BIT STRING
         :            13 22 DA F2 92 93 CE C0 9F 70 40 C9 DA 85 6B 61
         :            6F 8F BE E0 A4 04 55 C1 63 84 61 37 F5 4B 71 6D
         :            62 AA 6F BF 6C E8 48 03 AD 28 85 21 9E 3C 1C 91
```

```
         :              48 EE 65 28 65 70 D0 BD 5B CC DB CE B1 F5 B5 C3
         :              CA 7A A9 C8 8A 68 12 8A CA 6A 85 A6 BC DA 36 E9
         :              B9 94 35 82 5B CA BC B6 9F 83 03 7F 21 6C EE 82
         :              C1 3F BD C1 41 4B DD 1A 6F 6C AF 4A 52 FC 19 19
         :              17 AC 29 0C 5E D7 57 90 D5 B1 2B 36 29 1F 45 33
         :                      [ Another 384 bytes skipped ]
         :            }
2348 1376:          SEQUENCE {
2352  840:            SEQUENCE {
2356    3:              [0] {
2358    1:                INTEGER 2
         :                }
2361    9:              INTEGER 00 E8 FA 19 63 14 D2 FA 18
2372   13:              SEQUENCE {
2374    9:                OBJECT IDENTIFIER
         :                  sha256WithRSAEncryption (1 2 840 113549 1 1 11)
2385    0:                NULL
         :                }
2387   27:              SEQUENCE {
2389   25:                SET {
2391   23:                  SEQUENCE {
2393    3:                    OBJECT IDENTIFIER serialNumber (2 5 4 5)
2398   16:                    PrintableString 'f92009e853b6b045'
         :                    }
         :                  }
         :                }
2416   30:              SEQUENCE {
2418   13:                UTCTime 26/05/2016 16:28:52 GMT
2433   13:                UTCTime 24/05/2026 16:28:52 GMT
         :                }
2448   27:              SEQUENCE {
2450   25:                SET {
2452   23:                  SEQUENCE {
2454    3:                    OBJECT IDENTIFIER serialNumber (2 5 4 5)
2459   16:                    PrintableString 'f92009e853b6b045'
         :                    }
         :                  }
         :                }
2477  546:              SEQUENCE {
2481   13:                SEQUENCE {
2483    9:                  OBJECT IDENTIFIER
         :                    rsaEncryption (1 2 840 113549 1 1 1)
2494    0:                  NULL
         :                  }
2496  527:                BIT STRING, encapsulates {
2501  522:                  SEQUENCE {
2505  513:                    INTEGER
         :                    00 AF B6 C7 82 2B B1 A7 01 EC 2B B4 2E 8B CC 54
```

```
        :                   16 63 AB EF 98 2F 32 C7 7F 75 31 03 0C 97 52 4B
        :                   1B 5F E8 09 FB C7 2A A9 45 1F 74 3C BD 9A 6F 13
        :                   35 74 4A A5 5E 77 F6 B6 AC 35 35 EE 17 C2 5E 63
        :                   95 17 DD 9C 92 E6 37 4A 53 CB FE 25 8F 8F FB B6
        :                   FD 12 93 78 A2 2A 4C A9 9C 45 2D 47 A5 9F 32 01
        :                   F4 41 97 CA 1C CD 7E 76 2F B2 F5 31 51 B6 FE B2
        :                   FF FD 2B 6F E4 FE 5B C6 BD 9E C3 4B FE 08 23 9D
        :                           [ Another 385 bytes skipped ]
3022    3:                  INTEGER 65537
        :                   }
        :                 }
        :               }
3027  166:            [3] {
3030  163:              SEQUENCE {
3033   29:                SEQUENCE {
3035    3:                  OBJECT IDENTIFIER subjectKeyIdentifier (2 5 29 14)
3040   22:                  OCTET STRING, encapsulates {
3042   20:                    OCTET STRING
        :                   36 61 E1 00 7C 88 05 09 51 8B 44 6C 47 FF 1A 4C
        :                   C9 EA 4F 12
        :                     }
        :                   }
3064   31:                SEQUENCE {
3066    3:                  OBJECT IDENTIFIER
        :                     authorityKeyIdentifier (2 5 29 35)
3071   24:                  OCTET STRING, encapsulates {
3073   22:                    SEQUENCE {
3075   20:                      [0]
        :                   36 61 E1 00 7C 88 05 09 51 8B 44 6C 47 FF 1A 4C
        :                   C9 EA 4F 12
        :                       }
        :                     }
        :                   }
3097   15:                SEQUENCE {
3099    3:                  OBJECT IDENTIFIER basicConstraints (2 5 29 19)
3104    1:                  BOOLEAN TRUE
3107    5:                  OCTET STRING, encapsulates {
3109    3:                    SEQUENCE {
3111    1:                      BOOLEAN TRUE
        :                       }
        :                     }
        :                   }
3114   14:                SEQUENCE {
3116    3:                  OBJECT IDENTIFIER keyUsage (2 5 29 15)
3121    1:                  BOOLEAN TRUE
3124    4:                  OCTET STRING, encapsulates {
3126    2:                    BIT STRING 1 unused bit
        :                       '1100001'B
```

```
         :                      }
         :                    }
3130   64:                  SEQUENCE {
3132    3:                    OBJECT IDENTIFIER cRLDistributionPoints (2 5 29 31)
3137   57:                    OCTET STRING, encapsulates {
3139   55:                      SEQUENCE {
3141   53:                        SEQUENCE {
3143   51:                          [0] {
3145   49:                            [0] {
3147   47:                              [6]
         :                    'https://android.googleapis.com/attestation/crl/'
         :                              }
         :                            }
         :                          }
         :                        }
         :                      }
         :                    }
         :                  }
         :                }
         :              }
3196   13:            SEQUENCE {
3198    9:              OBJECT IDENTIFIER
         :                sha256WithRSAEncryption (1 2 840 113549 1 1 11)
3209    0:              NULL
         :              }
3211  513:            BIT STRING
         :              20 C8 C3 8D 4B DC A9 57 1B 46 8C 89 2F FF 72 AA
         :              C6 F8 44 A1 1D 41 A8 F0 73 6C C3 7D 16 D6 42 6D
         :              8E 7E 94 07 04 4C EA 39 E6 8B 07 C1 3D BF 15 03
         :              DD 5C 85 BD AF B2 C0 2D 5F 6C DB 4E FA 81 27 DF
         :              8B 04 F1 82 77 0F C4 E7 74 5B 7F CE AA 87 12 9A
         :              88 01 CE 8E 9B C0 CB 96 37 9B 4D 26 A8 2D 30 FD
         :              9C 2F 8E ED 6D C1 BE 2F 84 B6 89 E4 D9 14 25 8B
         :              14 4B BA E6 24 A1 C7 06 71 13 2E 2F 06 16 A8 84
         :                      [ Another 384 bytes skipped ]
         :            }
3728 1413:          SEQUENCE {
3732  877:            SEQUENCE {
3736    3:              [0] {
3738    1:                INTEGER 2
         :                }
3741   10:              INTEGER 03 88 26 67 60 65 89 96 85 99
3753   13:              SEQUENCE {
3755    9:                OBJECT IDENTIFIER
         :                  sha256WithRSAEncryption (1 2 840 113549 1 1 11)
3766    0:                NULL
         :                }
3768   27:              SEQUENCE {
```

```
3770   25:             SET {
3772   23:               SEQUENCE {
3774    3:                 OBJECT IDENTIFIER serialNumber (2 5 4 5)
3779   16:                 PrintableString 'f92009e853b6b045'
         :                 }
         :               }
         :             }
3797   30:           SEQUENCE {
3799   13:             UTCTime 20/06/2018 22:47:35 GMT
3814   13:             UTCTime 17/06/2028 22:47:35 GMT
         :             }
3829   47:           SEQUENCE {
3831   25:             SET {
3833   23:               SEQUENCE {
3835    3:                 OBJECT IDENTIFIER serialNumber (2 5 4 5)
3840   16:                 PrintableString 'ccd18b9b608d658e'
         :                 }
         :               }
3858   18:             SET {
3860   16:               SEQUENCE {
3862    3:                 OBJECT IDENTIFIER title (2 5 4 12)
3867    9:                 UTF8String 'StrongBox'
         :                 }
         :               }
         :             }
3878  546:           SEQUENCE {
3882   13:             SEQUENCE {
3884    9:               OBJECT IDENTIFIER
         :                 rsaEncryption (1 2 840 113549 1 1 1)
3895    0:               NULL
         :               }
3897  527:             BIT STRING, encapsulates {
3902  522:               SEQUENCE {
3906  513:                 INTEGER
         :                 00 E8 22 0B F1 72 A6 01 63 D3 3C 44 9D DB 7A 87
         :                 D6 3D 6F 6D 92 B7 C9 4A 70 96 5D 29 7A 8E 96 3E
         :                 FE F3 10 53 B2 19 A5 BF 6E 54 AD D0 0A A2 8E 54
         :                 E0 D4 B4 2E A6 E0 D4 30 F8 5A 47 CC 09 00 56 45
         :                 BE DA 5A 84 59 90 18 CE 29 6C 8E 9E E6 90 98 BD
         :                 D4 D8 F8 38 82 90 C9 79 DB 31 D3 7A A1 CA BA 6A
         :                 8B 9D 15 91 E2 6C 41 A3 2B 25 DA 4F E4 B3 14 E5
         :                 4B EC B7 89 06 44 18 67 C1 4C 03 35 18 D8 FD 7D
         :                         [ Another 385 bytes skipped ]
4423    3:                 INTEGER 65537
         :                 }
         :               }
         :             }
4428  182:           [3] {
```

```
4431  179:                SEQUENCE {
4434   29:                  SEQUENCE {
4436    3:                    OBJECT IDENTIFIER subjectKeyIdentifier (2 5 29 14)
4441   22:                    OCTET STRING, encapsulates {
4443   20:                      OCTET STRING
         :                    1B 17 70 C6 97 DC 84 54 75 7C 3C 98 5C E6 1D 1D
         :                    08 59 5D 53
         :                      }
         :                    }
4465   31:                  SEQUENCE {
4467    3:                    OBJECT IDENTIFIER
         :                      authorityKeyIdentifier (2 5 29 35)
4472   24:                    OCTET STRING, encapsulates {
4474   22:                      SEQUENCE {
4476   20:                        [0]
         :                    36 61 E1 00 7C 88 05 09 51 8B 44 6C 47 FF 1A 4C
         :                    C9 EA 4F 12
         :                        }
         :                      }
         :                    }
4498   15:                  SEQUENCE {
4500    3:                    OBJECT IDENTIFIER basicConstraints (2 5 29 19)
4505    1:                    BOOLEAN TRUE
4508    5:                    OCTET STRING, encapsulates {
4510    3:                      SEQUENCE {
4512    1:                        BOOLEAN TRUE
         :                        }
         :                      }
         :                    }
4515   14:                  SEQUENCE {
4517    3:                    OBJECT IDENTIFIER keyUsage (2 5 29 15)
4522    1:                    BOOLEAN TRUE
4525    4:                    OCTET STRING, encapsulates {
4527    2:                      BIT STRING 2 unused bits
         :                        '100000'B (bit 5)
         :                      }
         :                    }
4531   80:                  SEQUENCE {
4533    3:                    OBJECT IDENTIFIER cRLDistributionPoints (2 5 29 31)
4538   73:                    OCTET STRING, encapsulates {
4540   71:                      SEQUENCE {
4542   69:                        SEQUENCE {
4544   67:                          [0] {
4546   65:                            [0] {
4548   63:                              [6]
         :                    'https://android.googleapis.com/attestation/crl/8'
         :                    'F6734C9FA504789'
         :                              }
```

```
         :                            }
         :                          }
         :                        }
         :                      }
         :                    }
         :                  }
         :                }
         :              }
4613   13:            SEQUENCE {
4615    9:              OBJECT IDENTIFIER
         :                sha256WithRSAEncryption (1 2 840 113549 1 1 11)
4626    0:              NULL
         :              }
4628  513:            BIT STRING
         :              9B E2 2D 8C 43 AC 8F 11 35 11 77 BD F9 32 B3 01
         :              8C E9 97 58 08 E5 C0 DD C4 CC A6 B1 4A A3 E5 D0
         :              48 A6 18 1C 8E 5C FD 35 4A A5 12 C2 1A 82 64 3E
         :              B4 CC 0C 0B 1F 5E D5 11 C0 B7 49 5B A6 E7 74 37
         :              0B 7D 99 27 84 B7 E0 34 58 28 01 CC 03 76 50 F8
         :              1A B5 3B EF CA D2 FF 7D C9 37 FE D9 F7 30 3D 31
         :              24 CA 83 FD 67 AC 38 E3 82 23 B0 70 80 48 84 D6
         :              A1 2E 18 BD 94 1F 9A 8E 82 CC 2F EB 97 AA 5B A3
         :                      [ Another 384 bytes skipped ]
         :            }
         :          }
5145    0:        SET {}
         :        }
         :      }
         :    }
```

7.2. Windows 10 TPM

The next two sections provide two views of a CSR generated via invocation of the Certificate Enrollment Manager API similar to the below:

```
CertificateRequestProperties request = new CertificateRequestProperties();
request.FriendlyName = "Self-Signed Device Certificate";

request.KeyAlgorithmName = KeyAlgorithmNames.Rsa;
request.KeyStorageProviderName = "Microsoft Smart Card Key Storage Provider";
request.UseExistingKey = true;
request.Exportable = ExportOption.NotExportable;
request.ContainerName = prj.GetContainerName();

request.Subject = subject_name;
request.KeyUsages = keyUsages;
request.SmartcardReaderName = smartCardReaderName;

string privacyCa =
    "MIIDezCCAmOgAwIBAgIBATANBgkqhkiG9w0BAQsFADBUMQswCQYDVQQGEwJVUzEY" +
    "MBYGA1UEChMPVS5TLiBHb3Zlcm5tZW50MQ0wCwYDVQQLEwRESVNBMRwwGgYDVQQD" +
    "ExNQdXJlYnJlZCBQcml2YWN5IENBMB4XDTE4MDQwMzE0NTQwMFoXDTI4MDQwMzE0" +
    "NTQwMFowVDELMAkGA1UEBhMCVVMxGDAWBgNVBAoTD1UuUy4gR292ZXJubWVudDEN" +
    "MAsGA1UECxMERElTQTEcMBoGA1UEAxMTUHVyZWJyZWQgUHJpdmFjeSBDQTCCASIw" +
    "DQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAMROV8sQ707OSvjRxoX5S6MaB0r4" +
    "r5TnM97cx0RjtSVPu3O/WG9KRQdJtG9gARKKlxqgKOPJkTfTIxvUvWwKrtL9HjYs" +
    "IC2V/otsX3JKgPepud2CTIy3I1ADU7UD0/0MGqALbn+grDTaZOSi5p6cA0eo/f0X" +
    "O7UNh5r2YWOYAhZdhIy5F9BIOZEN/7pRyvKziupf3OVTQaMjMWoiDrCQC+D0xya4" +
    "8qxU/VFy4c9BmIg7uNzkHDqdaogo1Gsj5t2y0lW37IbRo6HrZ5Dl18laIX7s7n9k" +
    "Mp7GbK4rq/1FTMvI5bBpN/Pp4syi3f+oyQbSz+FPQwfBWGLukTUzPYcDVfUCAwEA" +
    "AaNYMFYwHQYDVR0OBBYEFAFy9PrSM65GYyC0EVDPU91WJ0BXMAsGA1UdDwQEAwIC" +
    "pDAoBgNVHSUEITAfBggrBgEFBQcDAgYIKwYBBQUHAwEGCSsGAQQBgjcVJDANBgkq" +
    "hkiG9w0BAQsFAAOCAQEAG777BuS/EXmuoHiVctA0n58u4SZb6i9Jvw1gI3qIryGM" +
    "2oxDSKPr36c7R2tFmAqo4m9N97wh4xFebkkYHgZWPsp0hRFy79veE+wMCw+Z0B88" +
    "ri4a2z/oTDmW9uf3r+BaZjRKpVoaYW9eztmz6DJA3wtvEdvUE2Nq4G1V5yXIdiSU" +
    "pfVd4eyEPVNy0Yp9DZDBP9vVcd5x7VfG8rzQoaDcerwrsXJ9/WLDz76A6d2/syHN" +
    "74CRuXYGhpBb7YL1jIhgVi6Rb4Dbq3dgDIkmTqUecEknuX73Oddr/phgqMOrVWUB" +
    "1XrHJbPUuC+nuPbShhJ0vPRw13TX3deqjzTsj8XEcA==";

byte[] privacyCaBytes = Convert.FromBase64String(privacyCa);
IBuffer buffer = privacyCaBytes.AsBuffer();
request.AttestationCredentialCertificate = new Certificate(buffer); ;

csrToDiscard = await
CertificateEnrollmentManager.UserCertificateEnrollmentManager.\
CreateRequestAsync(request);
```

Attestation details are described here: https://msdn.microsoft.com/en-us/library/dn366894.aspx.

The structure is essentially a Full PKI Request as described in RFC 5272.

```
* ContentInfo
  * SignedData
     * PKIData
       * Empty controlSequence
       * One TaggestRequest
         * PKCS 10
           * Basic request details along with encrypted attestation extension
       * Empty cmsSequence
       * Empty otherMsgSequence
     * Certificates bag with two certs (one of which is revoked)
```

7.2.1. Attestation statement

This section provides an annotation attestation statement as extracted from an encrypted attestation extension. The structure of the attestation statement is defined here: https://msdn.microsoft.com/en-us/library/dn408990.aspx.

```
 600 1256:                        SEQUENCE {
 604    9:                          OBJECT IDENTIFIER '1 3 6 1 4 1 311 21 24'
 615 1241:                          SET {
 619 1237:                            OCTET STRING
         :                  4B 41 53 54 01 00 00 00 02 00 00 00 1C 00 00 00
         :                  00 00 00 00 B9 04 00 00 00 00 00 00 4B 41 44 53
         :                  02 00 00 00 18 00 00 00 A1 00 00 00 00 01 00 00
         :                  00 03 00 00 FF 54 43 47 80 17 00 22 00 0B 9A FD
         :                  AB 8A 0B E9 0B BB 3F 7F E6 B6 77 91 EF A9 15 8A
         :                  03 B2 2B 8C BE 3F EC 56 B6 30 BF 82 73 9C 00 14
         :                  13 6E 2F 14 DD AF 30 72 A6 E3 89 4D BF 7A 54 26
         :                  36 2F 10 D6 00 00 00 00 51 4F CB E5 AD 8C 8C 60
         :                  E6 C2 70 80 00 D4 2C 65 4C 6B 95 ED 95 00 22 00
         :                  0B 2B E6 2C AD 8D E8 9A 85 04 D7 F3 7B B7 4C F8
         :                  32 CD B4 F1 80 CA A6 35 B9 2C 39 87 B7 96 03 C3
         :                  A3 00 22 00 0B 6C 88 60 B2 80 E3 BE 7D 34 F2 85
         :                  DC 26 9D 1B 72 A8 0A 17 CF 31 08 F1 55 F2 9B 4E
         :                  82 C8 5B 49 7B 1A F1 4B 12 A1 C5 D1 A4 C5 A4 59
         :                  C4 0A 97 E0 88 ED 1C D3 B6 38 4A 5D 6C 27 F5 69
         :                  7D 17 AD F6 C0 03 27 09 5D 93 B5 13 EA 50 B5 05
         :                  27 7B A0 51 4D 1B 17 52 87 7D B8 A6 05 4A 4F 39
         :                  CA 36 5C A1 19 19 0B 73 B4 0E 7F D3 91 DA 91 EE
         :                  37 C6 CE 78 AF 15 21 5D EB 5E 5F 23 A7 08 E9 85
         :                  D4 6B A0 95 6D D7 E0 3A D1 92 72 B7 D4 E5 35 6A
         :                  01 B0 7D 35 D0 99 BA A1 77 35 76 75 E3 90 A8 8B
         :                  86 27 B8 3D 47 75 2D 98 D0 23 4E 09 D8 26 6B 32
         :                  3C AB AC 50 A2 E8 FF 70 21 85 C5 5E B1 F5 9C B9
         :                  6E 21 27 C7 2A CD 84 61 02 47 6A A0 E1 9A 9F AF
         :                  02 43 08 D8 BF 9F 69 14 C4 8C 80 32 2D 5C A3 60
         :                  48 F5 5E 8E 65 6B 5E B5 0E A4 ED B9 8B F9 C3 D9
```

```
       :                     A8 CE C0 64 71 F6 E3 81 F7 9D 79 E5 73 7B F3 A4
       :                     6E 65 8D 72 B4 0A 3E 5E 70 5F AB 2B 89 B9 5E 65
       :                     44 BF 44 7B FB 2E 29 39 64 36 85 63 46 62 AF 25
       :                     A5 8B 19 30 AF 50 43 50 4D 38 00 00 00 02 00 00
       :                     00 03 00 00 00 38 01 00 00 E0 00 00 00 00 00 00
       :                     00 00 00 00 00 B0 00 00 00 00 00 00 00 00 00 00
       :                     00 00 00 00 00 00 00 00 00 00 00 00 00 01 36 00
       :                     01 00 0B 00 06 00 72 00 20 9D FF CB F3 6C 38 3A
       :                     E6 99 FB 98 68 DC 6D CB 89 D7 15 38 84 BE 28 03
       :                     92 2C 12 41 58 BF AD 22 AE 00 10 00 10 08 00 00
       :                     00 00 00 01 00 9B B1 27 B7 E3 5D 0C 10 74 52 1B
       :                     60 59 96 5E B6 08 D4 76 26 17 B5 92 49 39 34 CD
       :                     A4 2D 4D C9 3E 50 05 2E D8 9E 22 37 E2 05 D2 7F
       :                     3B 3E 4D 9F E0 E0 31 52 74 A0 D5 18 BE F1 9F 79
       :                     48 D6 24 69 35 3C D4 1F 55 73 75 ED 83 D6 3A E3
       :                     63 77 A6 5B 92 97 86 13 7C 69 3B DE AA E5 0E 9A
       :                     39 CF 53 DF 4C 7A E0 3C A3 EC 29 DA 18 5F 86 E6
       :                     22 D9 2C A3 8E D8 E2 3E 80 9C 69 52 FA 1E 90 3F
       :                     BA 09 04 D0 91 6A 27 2B 44 8C FF E8 DE FF BD B9
       :                     CE DD 95 67 70 FD 94 E5 3A E6 E4 EA 01 A5 AC 4A
       :                     79 5C 88 4D 07 43 C7 C0 B8 95 3E 7C 72 90 CD 35
       :                     99 B3 32 8A C7 8C 90 63 E3 46 88 62 35 A4 5B 54
       :                     F1 E8 61 0E CF 85 B4 41 6F 06 94 B6 BA 6F 4B CE
       :                     F7 8A 18 6C 5E 9A 6B 65 C3 F5 58 ED 7D 6A 3A E6
       :                     24 B6 21 6F 8C EE 1C 21 60 9E 2F 86 22 D2 2B 8F
       :                     E0 3B 12 AC 6B F5 FF 54 C6 E8 D4 3C 2E D3 B6 8E
       :                     7A 30 36 29 3D 00 DE 00 20 13 F5 31 2B 87 50 19
       :                     D3 95 1F F2 B6 00 95 5B 0A E2 54 7A A0 CF 6A 2C
       :                     F5 4F AD 77 C6 D5 4F 52 CB 00 10 3B 41 34 BF D4
       :                     FC 8B BE 87 14 47 81 4E 5C 5C 23 73 44 AF D6 56
       :                     6F A6 6E BE E7 63 9C 43 53 C4 3C 26 33 B6 AD 75
       :                     36 AC 91 98 C1 FF E3 B2 AF E6 3F 14 C0 2E 65 D7
       :                     C1 AD F6 22 D9 59 96 B6 70 8C 30 2F DE 76 1B EB
       :                     9D 56 C1 77 F8 1D 38 5C 7D 13 9C FD 1E 3E 00 1B
       :                     5A 74 C4 8E 49 2B 0B B5 C5 0E E3 A7 2C 92 E2 96
       :                     1E 9D C8 43 02 2F 8F F8 6E 66 4A FA D8 56 57 59
       :                     48 A4 D5 B7 7F 49 52 CA FA 11 E4 AF 27 E7 64 21
       :                     76 79 9B 8A A3 1A A6 FA A1 03 3E CC CD 41 26 3C
       :                     0D 3C DC 81 21 21 DE 92 4D 2A EF 66 DE D6 77 FE
       :                     41 0C 5D 44 1A D0 C4 D7 8B EA 6D DE 01 EE 97 DB
       :                     61 0F FD 62 59 00 00 00 06 00 20 8F CD 21 69 AB
       :                     92 69 4E 0C 63 3F 1A B7 72 84 2B 82 41 BB C2 02
       :                     88 98 1F C7 AC 1E DD C1 FD DB 0E 00 20 E5 29 F5
       :                     D6 11 28 72 95 4E 8E D6 60 51 17 B7 57 E2 37 C6
       :                     E1 95 13 A9 49 FE E1 F2 04 C4 58 02 3A 00 20 AF
       :                     2C A5 69 69 9C 43 6A 21 00 6F 1C B8 A2 75 6C 98
       :                     BC 1C 76 5A 35 59 C5 FE 1C 3F 5E 72 28 A7 E7 00
       :                     20 C4 13 A8 47 B1 11 12 B1 CB DD D4 EC A4 DA AA
```

```
         :                   15 A1 85 2C 1C 3B BA 57 46 1D 25 76 05 F3 D5 AF
         :                   53 00 00 00 20 04 8E 9A 3A CE 08 58 3F 79 F3 44
         :                   FF 78 5B BE A9 F0 7A C7 FA 33 25 B3 D4 9A 21 DD
         :                   51 94 C6 58 50
         :                             }
```

The format is structured as follows:

```
typedef struct  {
   UINT32 Magic;
   UINT32 Version;
   UINT32 Platform;
   UINT32 HeaderSize;
   UINT32 cbIdBinding;
   UINT32 cbKeyAttestation;
   UINT32 cbAIKOpaque;
   BYTE idBinding[cbIdBinding];
   BYTE keyAttestation[cbKeyAttestation];
   BYTE aikOpaque[cbAIKOpaque];
 } KeyAttestationStatement;
```

```
4B 41 53 54 - Magic
01 00 00 00 - Version
02 00 00 00 - Platform
1C 00 00 00 - HeaderSize
00 00 00 00 - cbIdBinding
B9 04 00 00 - cbKeyAttestation
00 00 00 00 - cbAIKOpaque
```

The remainder is the keyAttestation, which is structured as follows:

```
typedef struct {
   UINT32 Magic;
   UINT32 Platform;
   UINT32 HeaderSize;
   UINT32 cbKeyAttest;
   UINT32 cbSignature;
   UINT32 cbKeyBlob;
   BYTE keyAttest[cbKeyAttest];
   BYTE signature[cbSignature];
   BYTE keyBlob[cbKeyBlob];
 } keyAttestation;
```

```
4B 41 44 53 - Magic
02 00 00 00 - Platform
18 00 00 00 - HeaderSize
A1 00 00 00 - cbKeyAttest (161)
00 01 00 00 - cbSignature (256)
00 03 00 00 - cbKeyBlob
```

keyAttest (161 bytes) ~~~~~~~~~~~ FF 54 43 47 80 17 00 22 00 0B 9A FD AB 8A 0B E9 0B BB 3F 7F E6 B6 77 91 EF A9 15 8A 03 B2 2B 8C BE 3F EC 56 B6 30 BF 82 73 9C 00 14 13 6E 2F 14 DD AF 30 72 A6 E3 89 4D BF 7A 54 26 36 2F 10 D6 00 00 00 00 51 4F CB E5 AD 8C 8C 60 E6 C2 70 80 00 D4 2C 65 4C 6B 95 ED 95 00 22 00 0B 2B E6 2C AD 8D E8 9A 85 04 D7 F3 7B B7 4C F8 32 CD B4 F1 80 CA A6 35 B9 2C 39 87 B7 96 03 C3 A3 00 22 00 0B 6C 88 60 B2 80 E3 BE 7D 34 F2 85 DC 26 9D 1B 72 A8 0A 17 CF 31 08 F1 55 F2 9B 4E 82 C8 5B 49 7B ~~~~~~~~~~~

The keyAttest field is of type TPMS_ATTEST. The TPMS_ATTEST structure is defined in section 10.11.8 of https://trustedcomputinggroup.org/wp-content/uploads/TPM-Rev-2.0-Part-2-Structures-00.99.pdf. ~~~~~~~~~~~ FF 54 43 47 - magic 80 17 - type (TPM_ST_ATTEST_CERTIFY) 00 22 - name - TPM2B_NAME.size (34 bytes) 00 0B 9A FD AB 8A 0B E9 0B BB - TPM2B_NAME.name 3F 7F E6 B6 77 91 EF A9 15 8A 03 B2 2B 8C BE 3F EC 56 B6 30 BF 82 73 9C

00 14 - extraData - TPM2B_DATA.size (20 bytes) 13 6E 2F 14 DD AF 30 72 A6 E3 - TPM2B_DATA.buffer 89 4D BF 7A 54 26 36 2F 10 D6

00 00 00 00 51 4F CB E5 - clockInfo - TPMS_CLOCK_INFO.clock AD 8C 8C 60 - TPMS_CLOCK_INFO.resetCount E6 C2 70 80 - TPMS_CLOCK_INFO.restartCount 00 - - TPMS_CLOCK_INFO.safe

D4 2C 65 4C 6B 95 ED 95 - firmwareVersion

00 22 - attested - TPMS_CERTIFY_INFO.name.size 00 0B 2B E6 2C AD 8D E8 9A 85 - TPM2B_NAME.name 04 D7 F3 7B B7 4C F8 32 CD B4 F1 80 CA A6 35 B9 2C 39 87 B7 96 03 C3 A3

```
00 22 - TPMS_CERTIFY_INFO.qualifiedName.size 00 0B 6C 88 60 B2 80 E3
BE 7D - TPM2B_NAME.name 34 F2 85 DC 26 9D 1B 72 A8 0A 17 CF 31 08 F1
55 F2 9B 4E 82 C8 5B 49 7B ~~~~~~~~~~~

Signature (256 bytes) - generated using the AIK private key
~~~~~~~~~~~ 1A F1 4B 12 A1 C5 D1 A4 C5 A4 59 C4 0A 97 E0 88 ED 1C D3
B6 38 4A 5D 6C 27 F5 69 7D 17 AD F6 C0 03 27 09 5D 93 B5 13 EA 50 B5
05 27 7B A0 51 4D 1B 17 52 87 7D B8 A6 05 4A 4F 39 CA 36 5C A1 19 19
0B 73 B4 0E 7F D3 91 DA 91 EE 37 C6 CE 78 AF 15 21 5D EB 5E 5F 23 A7
08 E9 85 D4 6B A0 95 6D D7 E0 3A D1 92 72 B7 D4 E5 35 6A 01 B0 7D 35
D0 99 BA A1 77 35 76 75 E3 90 A8 8B 86 27 B8 3D 47 75 2D 98 D0 23 4E
09 D8 26 6B 32 3C AB AC 50 A2 E8 FF 70 21 85 C5 5E B1 F5 9C B9 6E 21
27 C7 2A CD 84 61 02 47 6A A0 E1 9A 9F AF 02 43 08 D8 BF 9F 69 14 C4
8C 80 32 2D 5C A3 60 48 F5 5E 8E 65 6B 5E B5 0E A4 ED B9 8B F9 C3 D9
A8 CE C0 64 71 F6 E3 81 F7 9D 79 E5 73 7B F3 A4 6E 65 8D 72 B4 0A 3E
5E 70 5F AB 2B 89 B9 5E 65 44 BF 44 7B FB 2E 29 39 64 36 85 63 46 62
AF 25 A5 8B 19 30 AF ~~~~~~~~~~~
```

The remainder is the keyBlob, which is defined here: https://github.com/Microsoft/TSS.MSR/blob/master/PCPTool.v11/inc/TpmAtt.h.

7.3. Yubikey

As with the Android Keystore attestations, Yubikey attestations take the form of an X.509 certificate. As above, the certificate is presented here packaged along with an intermediate CA certificate as a certificates-only SignedData message.

The attestations below were generated using code similar to that found in the yubico-piv-tool (https://github.com/Yubico/yubico-piv-tool). Details regarding attestations are here: https://developers.yubico.com/PIV/Introduction/PIV_attestation.html

7.3.1. Yubikey 4

```
   0 1576: SEQUENCE {
   4    9:   OBJECT IDENTIFIER signedData (1 2 840 113549 1 7 2)
  15 1561:   [0] {
  19 1557:     SEQUENCE {
  23    1:       INTEGER 1
  26    0:       SET {}
  28   11:       SEQUENCE {
  30    9:         OBJECT IDENTIFIER data (1 2 840 113549 1 7 1)
         :         }
  41 1533:       [0] {
  45  742:         SEQUENCE {
  49  462:           SEQUENCE {
```

```
  53    3:            [0] {
  55    1:              INTEGER 2
        :              }
  58    9:            INTEGER 00 A4 85 22 AA 34 AF AE 4F
  69   13:            SEQUENCE {
  71    9:              OBJECT IDENTIFIER
        :                sha256WithRSAEncryption (1 2 840 113549 1 1 11)
  82    0:              NULL
        :              }
  84   43:            SEQUENCE {
  86   41:              SET {
  88   39:                SEQUENCE {
  90    3:                  OBJECT IDENTIFIER commonName (2 5 4 3)
  95   32:                  UTF8String 'Yubico PIV Root CA Serial 263751'
        :                  }
        :                }
        :              }
 129   32:            SEQUENCE {
 131   13:              UTCTime 14/03/2016 00:00:00 GMT
 146   15:              GeneralizedTime 17/04/2052 00:00:00 GMT
        :              }
 163   33:            SEQUENCE {
 165   31:              SET {
 167   29:                SEQUENCE {
 169    3:                  OBJECT IDENTIFIER commonName (2 5 4 3)
 174   22:                  UTF8String 'Yubico PIV Attestation'
        :                  }
        :                }
        :              }
 198  290:            SEQUENCE {
 202   13:              SEQUENCE {
 204    9:                OBJECT IDENTIFIER
        :                  rsaEncryption (1 2 840 113549 1 1 1)
 215    0:                NULL
        :                }
 217  271:              BIT STRING
        :                30 82 01 0A 02 82 01 01 00 AB A9 0B 16 9B EF 31
        :                CC 3E AC 18 5A 2D 45 80 75 70 C7 58 B0 6C 3F 1B
        :                59 0D 49 B9 89 E8 6F CE BB 27 6F D8 3C 60 3A 85
        :                00 EF 5C BC 40 99 3D 41 EE EA C0 81 7F 76 48 E4
        :                A9 4C BC D5 6B E1 1F 0A 60 93 C6 FE AA D2 8D 8E
        :                E2 B7 CD 8B 2B F7 9B DD 5A AB 2F CF B9 0E 54 CE
        :                EC 8D F5 5E D7 7B 91 C3 A7 56 9C DC C1 06 86 76
        :                36 44 53 FB 08 25 D8 06 B9 06 8C 81 FD 63 67 CA
        :                        [ Another 142 bytes skipped ]
        :              }
 492   21:            [3] {
 494   19:              SEQUENCE {
```

```
 496   17:                 SEQUENCE {
 498   10:                   OBJECT IDENTIFIER '1 3 6 1 4 1 41482 3 3'
 510    3:                   OCTET STRING 04 03 03
         :                   }
         :                 }
         :               }
         :             }
 515   13:           SEQUENCE {
 517    9:             OBJECT IDENTIFIER
         :               sha256WithRSAEncryption (1 2 840 113549 1 1 11)
 528    0:             NULL
         :             }
 530  257:           BIT STRING
         :             52 80 5A 6D C3 9E DF 47 A8 F1 B2 A5 9C A3 80 81
         :             3B 1D 6A EB 6A 12 62 4B 11 FD 8D 30 F1 7B FC 71
         :             10 C9 B2 08 FC D1 4E 35 7F 45 F2 10 A2 52 B9 D4
         :             B3 02 1A 01 56 07 6B FA 64 A7 08 F0 03 FB 27 A9
         :             60 8D 0D D3 AC 5A 10 CF 20 96 4E 82 BC 9D E3 37
         :             DA C1 4C 50 E1 3D 16 B4 CA F4 1B FF 08 64 C9 74
         :             4F 2A 3A 43 E0 DE 42 79 F2 13 AE 77 A1 E2 AE 6B
         :             DF 72 A5 B6 CE D7 4C 90 13 DF DE DB F2 8B 34 45
         :                     [ Another 128 bytes skipped ]
         :           }
 791  783:         SEQUENCE {
 795  503:           SEQUENCE {
 799    3:             [0] {
 801    1:               INTEGER 2
         :               }
 804   17:             INTEGER
         :               00 FE B9 AF 03 3B 0B A7 79 04 02 F5 67 AE DF 72
         :               ED
 823   13:             SEQUENCE {
 825    9:               OBJECT IDENTIFIER
         :                 sha256WithRSAEncryption (1 2 840 113549 1 1 11)
 836    0:               NULL
         :               }
 838   33:             SEQUENCE {
 840   31:               SET {
 842   29:                 SEQUENCE {
 844    3:                   OBJECT IDENTIFIER commonName (2 5 4 3)
 849   22:                   UTF8String 'Yubico PIV Attestation'
         :                   }
         :                 }
         :               }
 873   32:             SEQUENCE {
 875   13:               UTCTime 14/03/2016 00:00:00 GMT
 890   15:               GeneralizedTime 17/04/2052 00:00:00 GMT
         :               }
```

```
 907   37:             SEQUENCE {
 909   35:               SET {
 911   33:                 SEQUENCE {
 913    3:                   OBJECT IDENTIFIER commonName (2 5 4 3)
 918   26:                   UTF8String 'YubiKey PIV Attestation 9e'
         :                   }
         :                 }
         :               }
 946  290:             SEQUENCE {
 950   13:               SEQUENCE {
 952    9:                 OBJECT IDENTIFIER
         :                   rsaEncryption (1 2 840 113549 1 1 1)
 963    0:                 NULL
         :                 }
 965  271:               BIT STRING
         :                 30 82 01 0A 02 82 01 01 00 93 C4 C0 35 95 7E 26
         :                 2A 7E A5 D0 29 C4 D7 E9 39 67 22 B1 09 45 46 4D
         :                 DB A4 77 CB 0B A3 F1 D0 69 3C 24 8D A2 72 72 27
         :                 E1 7F DE CB 67 A4 1D D2 E5 43 44 6F 21 39 F8 57
         :                 34 01 0E 7E C3 81 63 63 6A 6D D7 40 20 7B AF 35
         :                 61 9C 8D C1 D1 2B 25 48 EE 52 FC F3 72 6A 74 96
         :                 01 CB 1C 1A B2 AD F9 18 96 EB 59 EF E3 3A CA BC
         :                 AA 9B 42 FE FF 60 6E 28 89 49 0D C1 B1 B0 25 AE
         :                         [ Another 142 bytes skipped ]
         :               }
1240   60:             [3] {
1242   58:               SEQUENCE {
1244   17:                 SEQUENCE {
1246   10:                   OBJECT IDENTIFIER '1 3 6 1 4 1 41482 3 3'
1258    3:                   OCTET STRING 04 03 03                       -- firmwa
re version
         :                   }
1263   19:                 SEQUENCE {
1265   10:                   OBJECT IDENTIFIER '1 3 6 1 4 1 41482 3 7'
1277    5:                   OCTET STRING 02 03 4F 9B B5        -- serial number
         :                   }
1284   16:                 SEQUENCE {
1286   10:                   OBJECT IDENTIFIER '1 3 6 1 4 1 41482 3 8'
1298    2:                   OCTET STRING 01 01                          -- PIN an
d touch policy
         :                   }
         :                 }
         :               }
         :             }
1302   13:           SEQUENCE {
1304    9:             OBJECT IDENTIFIER
         :               sha256WithRSAEncryption (1 2 840 113549 1 1 11)
1315    0:             NULL
         :             }
1317  257:           BIT STRING
```

```
         :             1F 2B B8 1C 95 A1 01 74 3F 87 27 F6 B3 A6 A9 9D
         :             11 B9 ED 68 92 B9 05 2D 22 36 51 28 23 3D B0 2F
         :             7A 17 D5 8C 0C F4 3A 68 FD 2A 34 0D 80 3C F7 8F
         :             B8 79 B0 76 E5 4D 61 94 C5 72 D6 9F 6E 26 76 5F
         :             03 94 55 40 93 5C 04 EF CC 58 41 EB 7C 86 64 23
         :             5F 23 5E 94 78 73 2E 77 8C 58 C5 45 87 22 CF BA
         :             69 06 B8 C7 06 37 10 21 8C 74 AD 08 B9 85 F2 7B
         :             99 02 4A 3E E8 96 09 D3 F4 C6 AB FA 49 68 E2 E0
         :                     [ Another 128 bytes skipped ]
         :           }
         :         }
1578    0:       SET {}
         :       }
         :     }
         :   }
```

7.3.2. Yubikey 5

```
   0 1613: SEQUENCE {
   4    9:   OBJECT IDENTIFIER signedData (1 2 840 113549 1 7 2)
  15 1598:   [0] {
  19 1594:     SEQUENCE {
  23    1:       INTEGER 1
  26    0:       SET {}
  28   11:       SEQUENCE {
  30    9:         OBJECT IDENTIFIER data (1 2 840 113549 1 7 1)
         :         }
  41 1570:       [0] {
  45  762:         SEQUENCE {
  49  482:           SEQUENCE {
  53    3:             [0] {
  55    1:               INTEGER 2
         :               }
  58    9:             INTEGER 00 86 77 17 E0 1D 19 2B 26
  69   13:             SEQUENCE {
  71    9:               OBJECT IDENTIFIER
         :                 sha256WithRSAEncryption (1 2 840 113549 1 1 11)
  82    0:               NULL
         :               }
  84   43:             SEQUENCE {
  86   41:               SET {
  88   39:                 SEQUENCE {
  90    3:                   OBJECT IDENTIFIER commonName (2 5 4 3)
  95   32:                   UTF8String 'Yubico PIV Root CA Serial 263751'
         :                   }
         :                 }
         :               }
 129   32:             SEQUENCE {
```

```
 131   13:                UTCTime 14/03/2016 00:00:00 GMT
 146   15:                GeneralizedTime 17/04/2052 00:00:00 GMT
         :                }
 163   33:              SEQUENCE {
 165   31:                SET {
 167   29:                  SEQUENCE {
 169    3:                    OBJECT IDENTIFIER commonName (2 5 4 3)
 174   22:                    UTF8String 'Yubico PIV Attestation'
         :                    }
         :                  }
         :                }
 198  290:              SEQUENCE {
 202   13:                SEQUENCE {
 204    9:                  OBJECT IDENTIFIER
         :                    rsaEncryption (1 2 840 113549 1 1 1)
 215    0:                  NULL
         :                  }
 217  271:                BIT STRING
         :                  30 82 01 0A 02 82 01 01 00 C5 5B 8D E9 B9 3C 53
         :                  69 82 88 FE DA 70 FC 5C 88 78 41 25 A2 1D 7B 84
         :                  8E 93 36 AD 67 2B 4C AB 45 BE B2 E0 D5 9C 1B A1
         :                  68 D5 6B F8 63 5C 83 CB 83 38 62 B7 64 AE 83 37
         :                  37 8E C8 60 80 E6 01 F8 75 AA AE F6 6E A7 D5 76
         :                  C5 C1 25 AD AA 9E 9D DC B5 7E E9 8E 2A B4 3F 99
         :                  0D F7 9F 20 A0 28 A0 9F B3 B1 22 5F AF 38 FB 73
         :                  46 F4 C7 93 30 DD FA D0 86 E0 C9 C6 72 99 AF FB
         :                          [ Another 142 bytes skipped ]
         :                }
 492   41:              [3] {
 494   39:                SEQUENCE {
 496   17:                  SEQUENCE {
 498   10:                    OBJECT IDENTIFIER '1 3 6 1 4 1 41482 3 3'
 510    3:                    OCTET STRING 05 01 02
         :                    }
 515   18:                  SEQUENCE {
 517    3:                    OBJECT IDENTIFIER basicConstraints (2 5 29 19)
 522    1:                    BOOLEAN TRUE
 525    8:                    OCTET STRING 30 06 01 01 FF 02 01 00
         :                    }
         :                  }
         :                }
         :              }
 535   13:            SEQUENCE {
 537    9:              OBJECT IDENTIFIER
         :                sha256WithRSAEncryption (1 2 840 113549 1 1 11)
 548    0:              NULL
         :              }
 550  257:            BIT STRING
```

```
         :              05 57 B7 BF 5A 41 74 F9 5F EC 2E D2 B8 78 26 E5
         :              EF 4F EA BF 5A 64 C9 CF 06 7F CA 8C 0A FC 1A 47
         :              1C D6 AC ED C8 5B 54 72 00 9F B8 59 AB 73 25 B2
         :              D6 02 A3 59 83 31 69 EE C1 5F 3D F2 2B 1B 22 CA
         :              B6 FC F9 FB 21 32 9E 08 F3 08 54 6D C9 26 10 42
         :              08 1D 3C B5 F0 5A B1 98 D4 68 DC 91 F1 D3 91 54
         :              7A A0 34 8B F6 65 EB 13 9F 3A 1C BF 43 C5 D1 D0
         :              33 23 C6 25 A0 4C E4 E9 AA 59 80 D8 02 1E B0 10
         :                      [ Another 128 bytes skipped ]
         :            }
 811  800:          SEQUENCE {
 815  520:            SEQUENCE {
 819    3:              [0] {
 821    1:                INTEGER 2
         :                }
 824   16:              INTEGER
         :                17 7D 2D F7 D6 6D 97 CC D6 CF 69 33 87 5B F1 5E
 842   13:              SEQUENCE {
 844    9:                OBJECT IDENTIFIER
         :                  sha256WithRSAEncryption (1 2 840 113549 1 1 11)
 855    0:                NULL
         :                }
 857   33:              SEQUENCE {
 859   31:                SET {
 861   29:                  SEQUENCE {
 863    3:                    OBJECT IDENTIFIER commonName (2 5 4 3)
 868   22:                    UTF8String 'Yubico PIV Attestation'
         :                    }
         :                  }
         :                }
 892   32:              SEQUENCE {
 894   13:                UTCTime 14/03/2016 00:00:00 GMT
 909   15:                GeneralizedTime 17/04/2052 00:00:00 GMT
         :                }
 926   37:              SEQUENCE {
 928   35:                SET {
 930   33:                  SEQUENCE {
 932    3:                    OBJECT IDENTIFIER commonName (2 5 4 3)
 937   26:                    UTF8String 'YubiKey PIV Attestation 9e'
         :                    }
         :                  }
         :                }
 965  290:              SEQUENCE {
 969   13:                SEQUENCE {
 971    9:                  OBJECT IDENTIFIER
         :                    rsaEncryption (1 2 840 113549 1 1 1)
 982    0:                  NULL
         :                  }
```

```
 984  271:               BIT STRING
         :                 30 82 01 0A 02 82 01 01 00 A9 02 2D 7A 4C 0B B1
         :                 0C 02 F9 E5 9C E5 6F 20 D1 9D F9 CE B3 B3 4D 1B
         :                 61 B0 B4 E0 3F 44 19 72 88 8B 8D 9F 86 4A 5E C7
         :                 38 F0 AF C9 28 5C D8 A2 80 C9 43 93 2D FA 39 7F
         :                 E9 39 2D 18 1B A7 A2 76 8F D4 6C D0 75 96 99 0D
         :                 06 37 9D 90 D5 71 00 6E FB 82 D1 5B 2A 7C 3B 62
         :                 9E AB 15 81 B9 AD 7F 3D 30 1C C2 4B 9D C4 D5 64
         :                 32 9A 54 D6 23 B1 65 92 A3 D7 57 E2 62 10 2B 93
         :                         [ Another 142 bytes skipped ]
         :               }
1259   78:             [3] {
1261   76:               SEQUENCE {
1263   17:                 SEQUENCE {
1265   10:                   OBJECT IDENTIFIER '1 3 6 1 4 1 41482 3 3'
1277    3:                   OCTET STRING 05 01 02                           -
- firmware version
         :                   }
1282   20:                 SEQUENCE {
1284   10:                   OBJECT IDENTIFIER '1 3 6 1 4 1 41482 3 7'
1296    6:                   OCTET STRING 02 04 00 93 6A A0     -- serial number
         :                   }
1304   16:                 SEQUENCE {
1306   10:                   OBJECT IDENTIFIER '1 3 6 1 4 1 41482 3 8'
1318    2:                   OCTET STRING 01 01                              -
- PIN and touch policy
         :                   }
1322   15:                 SEQUENCE {
1324   10:                   OBJECT IDENTIFIER '1 3 6 1 4 1 41482 3 9'
1336    1:                   OCTET STRING 02                                 -
- form factor
         :                   }
         :                 }
         :               }
         :             }
1339   13:           SEQUENCE {
1341    9:             OBJECT IDENTIFIER
         :               sha256WithRSAEncryption (1 2 840 113549 1 1 11)
1352    0:             NULL
         :             }
1354  257:           BIT STRING
         :             9F EB 7A 4C F0 7C 67 11 ED C5 84 07 C8 19 41 B2
         :             71 42 08 2B D6 CD A8 5F DC AE 79 75 6C F1 E5 4D
         :             28 95 89 69 9D C0 2E A7 D4 48 51 B0 75 FF 63 FD
         :             B8 79 93 03 EA BB 8A 67 D8 E7 EC C9 1C 8E 3F AF
         :             74 30 D4 7E 74 A4 26 50 9F D4 57 AE 23 C0 8A 63
         :             4E F3 C7 CF 5A AF 91 11 A2 6B 3B 49 24 32 26 88
         :             D8 4F 6F BE BC F0 2D A9 A2 88 B4 5F 54 AF 42 72
         :             08 74 64 57 76 5A 02 9A 9D 21 4B FD 7F 44 8F AF
         :                     [ Another 128 bytes skipped ]
         :           }
```

```
        :         }
1615    0:       SET {}
        :       }
        :     }
        :   }
```

8. Privacy Considerations.

TBD

9. Security Considerations

TBD.

10. IANA Considerations

TBD.

11. Acknowledgements

Thomas Hardjono provided the text on blockchain system. Dave Thaler suggested many small variations. Frank Xialiang suggested the scalling scenarios that might preclude a 1:1 protocol between attesters and relying parties. Henk Birkholz provided many reviews. Kathleen Moriarty provided many useful edits. Ned Smith, Anders Rundgren and Steve Hanna provided many useful pointers to TCG terms and concepts. Thomas Fossati and Shawn Willden elucidated the Android Keystore goals and limitations.

12. References

12.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

12.2. Informative References

[android_security]
Kralevich, R., "The Android Platform Security Model", n.d., <https://arxiv.org/pdf/1904.05572.pdf>.

[azureattestation]
Microsoft, ., "Azure Sphere Attestation", n.d., <https://azure.microsoft.com/enus/resources/azure-sphere-device-authentication-andattestation-service/en-us/>.

[fido] FIDO Alliance, ., "FIDO Specification Overview", n.d., <https://fidoalliance.org/specifications/>.

[fido_w3c]
W3C, ., "Web Authentication: An API for accessing Public Key Credentials Level 1", n.d., <https://www.w3.org/TR/webauthn-1/>.

[fidoattestation]
FIDO Alliance, ., "FIDO 2.0: Key Attestation", n.d., <https://fidoalliance.org/specs/fido-v2.0-ps-20150904/fido-key-attestation-v2.0-ps-20150904.html>.

[fidosignature]
FIDO Alliance, ., "FIDO 2.0: Signature Format", n.d., <https://fidoalliance.org/specs/fido-v2.0-ps-20150904/fido-signature-format-v2.0-ps-20150904.html>.

[fidotechnote]
FIDO Alliance, ., "FIDO TechNotes: The Truth about Attestation", n.d., <https://fidoalliance.org/fido-technotes-the-truth-about-attestation/>.

[I-D.birkholz-rats-tuda]
Fuchs, A., Birkholz, H., McDonald, I., and C. Bormann, "Time-Based Uni-Directional Attestation", draft-birkholz-rats-tuda-03 (work in progress), July 2020.

[I-D.fedorkow-rats-network-device-attestation]
Fedorkow, G., Voit, E., and J. Fitzgerald-McKay, "TPM-based Network Device Remote Integrity Verification", draft-fedorkow-rats-network-device-attestation-05 (work in progress), April 2020.

[I-D.gutmann-scep]
Gutmann, P., "Simple Certificate Enrolment Protocol", draft-gutmann-scep-16 (work in progress), March 2020.

[I-D.tschofenig-rats-psa-token]
Tschofenig, H., Frost, S., Brossard, M., Shaw, A., and T. Fossati, "Arm's Platform Security Architecture (PSA) Attestation Token", draft-tschofenig-rats-psa-token-05 (work in progress), March 2020.

[I-D.voit-rats-trusted-path-routing]
Voit, E., "Trusted Path Routing", draft-voit-rats-trusted-path-routing-02 (work in progress), June 2020.

[ieee802-1AR]
IEEE Standard, ., "IEEE 802.1AR Secure Device Identifier", 2009, <http://standards.ieee.org/findstds/standard/802.1AR-2009.html>.

[intelsgx]
Intel, ., "Intel(R) Software Guard Extensions: Attestation & Provisioning Services", n.d., <https://software.intel.com/en-us/sgx/attestation-services>.

[keystore]
Google, ., "Android Keystore System", n.d., <https://developer.android.com/training/articles/keystore>.

[keystore_attestation]
Google, ., "Verifying hardware-backed key pairs with Key Attestation", n.d., <https://developer.android.com/training/articles/security-key-attestation>.

[RFC4210] Adams, C., Farrell, S., Kause, T., and T. Mononen, "Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP)", RFC 4210, DOI 10.17487/RFC4210, September 2005, <https://www.rfc-editor.org/info/rfc4210>.

[RFC5209] Sangster, P., Khosravi, H., Mani, M., Narayan, K., and J. Tardo, "Network Endpoint Assessment (NEA): Overview and Requirements", RFC 5209, DOI 10.17487/RFC5209, June 2008, <https://www.rfc-editor.org/info/rfc5209>.

[RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, DOI 10.17487/RFC5652, September 2009, <https://www.rfc-editor.org/info/rfc5652>.

[RFC7030] Pritikin, M., Ed., Yee, P., Ed., and D. Harkins, Ed., "Enrollment over Secure Transport", RFC 7030, DOI 10.17487/RFC7030, October 2013, <https://www.rfc-editor.org/info/rfc7030>.

[RFC8555] Barnes, R., Hoffman-Andrews, J., McCarney, D., and J. Kasten, "Automatic Certificate Management Environment (ACME)", RFC 8555, DOI 10.17487/RFC8555, March 2019, <https://www.rfc-editor.org/info/rfc8555>.

[SP800-147B]
NIST, ., "BIOS Protection Guidelines for Servers", n.d., <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-147B.pdf>.

[SP800-155]
NIST, ., "BIOS Integrity Measurement Guidelines (Draft)", n.d., <https://csrc.nist.gov/CSRC/media/Publications/sp/800-155/draft/documents/draft-SP800-155_Dec2011.pdf>.

[tapinfomodel]
Group, T., "TCG Trusted Attestation Protocol (TAP) Information Model for TPM Families 1.2 and 2.0 and DICE Family 1.0", n.d., <https://trustedcomputinggroup.org/wp-content/uploads/TNC_TAP_Information_Model_v1.00_r0.29A_publicreview.pdf>.

[tcgglossary]
Group, T., "TCG Glossary, Version 1.1", n.d., <https://trustedcomputinggroup.org/wp-content/uploads/TCG-Glossary-V1.1-Rev-1.0.pdf>.

[tpmarchspec]
Group, T., "TPM 2.0 Mobile Reference Architecture", n.d., <https://trustedcomputinggroup.org/resource/tpm-2-0-mobile-reference-architecture-specification/>.

[windowsdefender]
Microsoft, ., "Windows Defender System Guard attestation", n.d., <https://www.microsoft.com/security/blog/2018/04/19/introducing-windows-defender-system-guard-runtime-attestation/>.

[windowshealth]
Microsoft, ., "Windows Device Health Attestation", n.d., <https://docs.microsoft.com/en-us/windowsserver/security/device-health-attestation>.

[yubikey_attestation]
Yubico, ., "PIV Attestation", n.d., <https://developers.yubico.com/PIV/Introduction/PIV_attestation.html>.

Appendix A. Changes

- created new section for target use cases
- added comments from Guy, Jessica, Henk and Ned on TCG description.

Authors' Addresses

Michael Richardson
Sandelman Software Works

Email: mcr+ietf@sandelman.ca

Carl Wallace
Red Hound Software

Email: carl@redhoundsoftware.com

Wei Pan
Huawei Technologies

Email: william.panwei@huawei.com