A Taxonomy of operational security considerations for manufacturer installed keys and Trust Anchors

Michael Richardson
<mcr@sandelman.ca>

https://www.sandelman.ca/SSW/talk/2022-t2trg-taxonomy-installed-keys

“The future is already here — it's just not very evenly distributed.”

Usually attributed to William Gibson

History of this draft

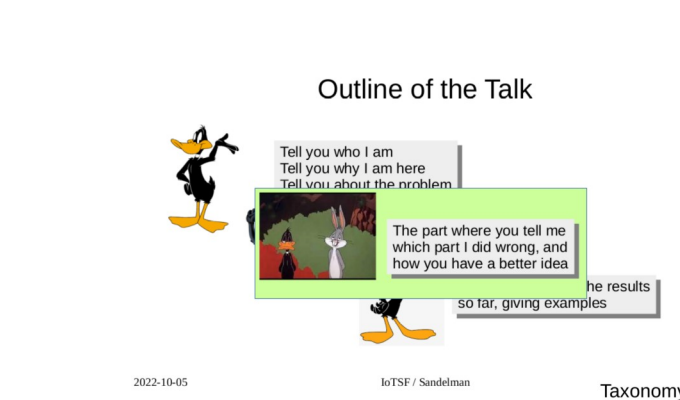
- ANIMA documents
  - masa-considerations
  - registrar-considerations
  - common contents,
- Went to SECDISPATCH in virtual interim meeting in July 2022, at IETF108
  - https://datatracker.ietf.org/meeting/108/session/secdispatch

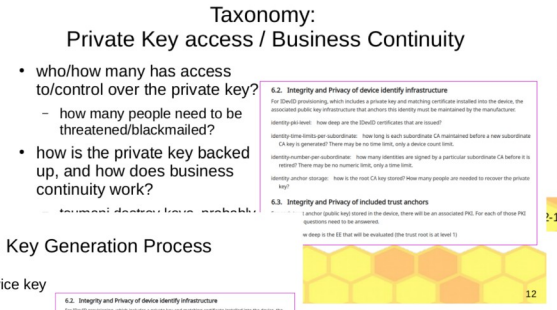
- Presentations at T2TRG in 2021.

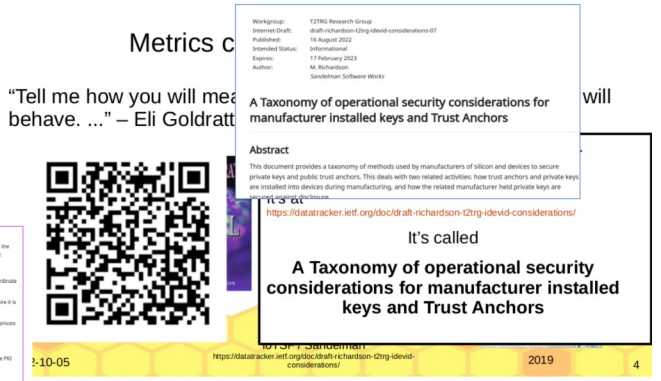
- Many presentations done privately at PKI, SecureElement and TEE vendors in fall 2020, winter 2021
- Invitation to bring work to T2TRG in late 2020, at

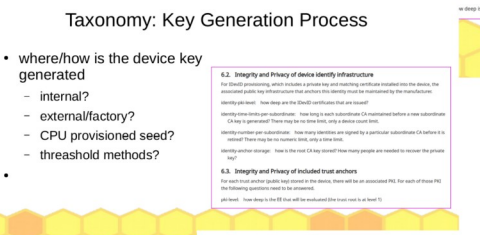
- Presentations and feedback at IoTSF 2022, TPM.DEV 2022, RIPE85 IoT WG
- Review responses

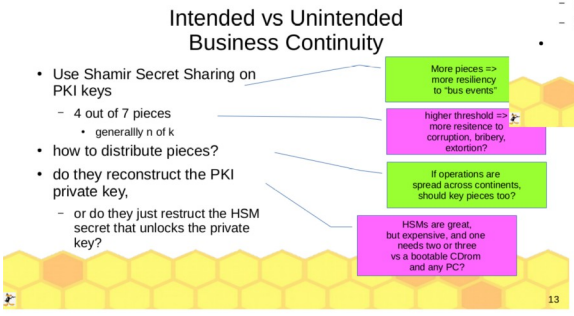
Highlights from the talks

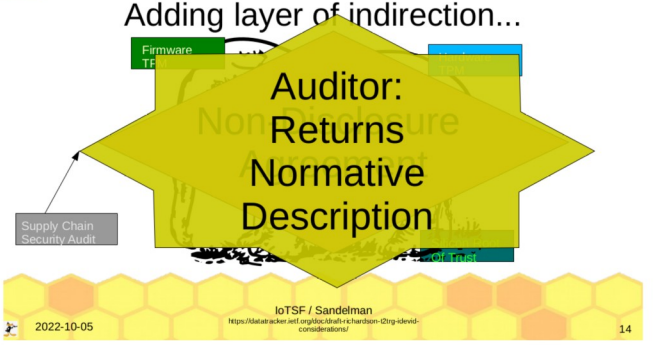










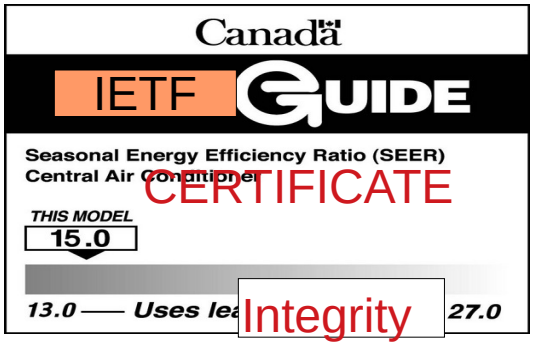


Non-Goals of the document

1) Not aiming to write an ISO27001-ish evaluation process.
2) Not going to make any recommendation as to an appropriate solution for a given risk situation.

Goals of this document

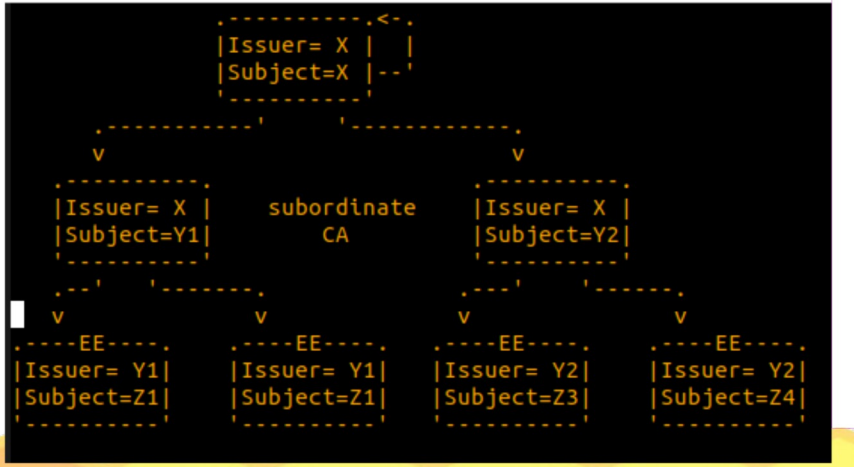
- Enumerate the reasonable, and maybe some less reasonable ways to provision and secure keys, and give them **names**.
- Not just the most secure way, because it is not always worth it.



admin:password

Taxonomy: Public Key Infrastructure Depth

- self-signed certificate is a PKI of level “one”
  - **not** counting from zero
-



6.2. Integrity and Privacy of device identify infrastructure

For IDevID provisioning, which includes a private key and matching certificate installed into the device, the associated public key infrastructure that anchors this identity must be maintained by the manufacturer.

identity-pki-level: how deep are the IDevID certificates that are issued?

identity-time-limits-per-subordinate: how long is each subordinate CA maintained before a new subordinate CA key is generated? There may be no time limit, only a device count limit.

identity-number-per-subordinate: how many identities are signed by a particular subordinate CA before it is retired? There may be no numeric limit, only a time limit.

identity-anchor-storage: how is the root CA key stored? How many people are needed to recover the private key?

6.3. Integrity and Privacy of included trust anchors

For each trust anchor (public key) stored in the device, there will be an associated PKI. For each of those PKI the following questions need to be answered.

pki-level: how deep is the EE that will be evaluated (the trust root is at level 1)

Taxonomy: Key Generation Process

- where/how is the device key generated
  - internal?
  - external/factory?
  - CPU provisioned seed?
  - threashold methods?
-

4.1.2. Key Generation process

4.1.2.1. On-device private key generation

Generating the key on-device has the advantage that the private key never leaves the device. The disadvantage is that the device may not have a verified random number generator. [factoringrsa] is an example of a successful attack on this scenario.

4.1.2.2. Off-device private key generation

Generating the key off-device has the advantage that the randomness of the private key can be better analyzed. As the private key is available to the manufacturing infrastructure, the authenticity of the public key is well known ahead of time.

If the device does not come with a serial number in silicon, then one should be assigned and placed into a

4.1.2.3. Key setup based on 256 bit secret seed

A hybrid of the previous two methods leverages a symmetric key that is often provided by a silicon vendor to OEM manufacturers.

Each CPU (or a Trusted Execution Environment [I-D.ietf-teep-architecture], or a TPM) is provisioned at fabrication time with a unique, secret seed, usually at least 256 bits in size.

This value is revealed to the OEM board manufacturer only via a secure channel. Upon first boot, the system (probably within a TEE, or within a TPM) will generate a key pair using the seed to initialize a Pseudo-Random-Number-Generator (PRNG). The OEM, in a separate system, will initialize the same PRNG and generate the same key pair. The OEM then derives the public key part, signs it and turns it into a certificate. The private part is then destroyed, ideally never stored or seen by anyone. The certificate (being public information) is placed into a database, in some cases it is loaded by the device as its IDevID certificate, in other cases, it is retrieved

lacks good name

Intended vs Unintended Business Continuity

- Use Shamir Secret Sharing on PKI keys
  - 4 out of 7 pieces
    - generallly n of k
- how to distribute pieces?
- do they reconstruct the PKI private key,
  - or do they just reconstruct the HSM secret that unlocks the private key?

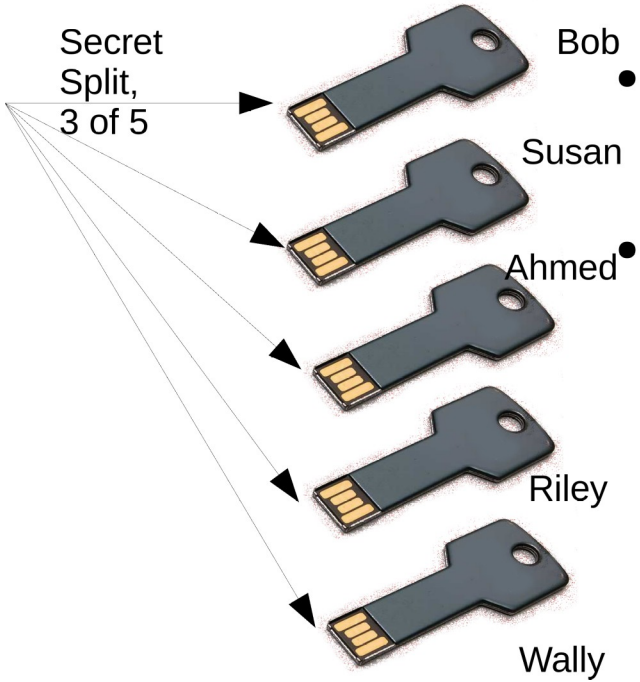
More pieces => more resiliency to “bus events”

higher threshold => more resitence to corruption, bribery, extortion?

If operations are spread across continents, should key pieces too?

HSMs are great, but expensive, and one needs two or three vs a bootable CDrom and any PC?

Example PKI Architecture: IDevID



- Hardware Security Module (HSM) to hold root key.
- Intermediate CA has online key

Next Steps

- Get document adopted
- Some word-smithing and maybe some bikeshedding over some of the terminology/definitions
- Promote the terms more widely
  - Within the IETF and IRTF and also outside
  - Listen to feedback, look for awkward usages
- Publish in late 2023

Questions!