From:	Davidson, Michael S. (Fed)
To:	Cooper, David (Fed); Dang, Quynh H. (Fed); Apon, Daniel C. (Fed); Miller, Carl A. (Fed); Dworkin, Morris J. (Fed)
Subject:	RE: Summary of comments on stateful HBS
Date:	Tuesday, April 30, 2019 11:10:10 AM

Hi guys,

Since there were additional comments that I missed, I am adding summary notes for them here. Some commonalities (but please read the full notes below, too):

1. It sounds like we need to decide on and explain our reasoning regarding whether or not to standardize the multi-tree variants.
2. They want alignment in notation and some other areas. We can't really provide this if we just refer to the RFCs. If we write them ourselves, the Gemalto comments have many helpful suggestions.
3. We have some thinking to do regarding parameter sets.

Generally speaking, Gemalto had a lot of interesting things to say, and I don't think I can quite do it justice by summarizing here. Everyone should read that on their own, especially to inform point 2 above.

Evan Clendeling:

-There is a cryptocurrency that uses XMSS called Quantum Resistant Ledger.

Crypto4a:

-They specifically ask whether we intend to standardize the hierarchical/multi-tree variants. They believe that they should be standardized to support more heterogenous parameter sets for different use cases. [My note: have we decided this yet? If we choose not to, we should explain our reasoning; in particular, we do NOT want to enable more use cases]

-Four use cases: "1) FW signing 2) Root CA (or similar high trust scenarios which have a reduced frequency of signature demands, or whose key usage can be metered out in a controlled fashion) 3) Device configuration management schemes (e.g., MUD-based schemes) 4) Trust Anchor Management Protocol (TAMP) message signing and management"

-They would like us to provide a checklist of best practices.

Gemalto:

-They want us to use similar notation/parameter sets for both algorithms in order to make things easier for developers.

-They explicitly want the multi-tree variants standardized because those are more flexible. They also claim the multi-tree variant is safer by allowing different HSMs to use different subtrees. [My note: this can be done w/o multiple layers, but at some cost, as discussed in our last meeting]. Multi-trees are helpful for splitting signature verification into phases on devices that are very RAM constrained.

-We should discuss forward security in key generation method section.

-They would prefer we align notation for the two schemes, and have specific suggestions. They would also prefer alignment of parameter sets, and some justification for those choices. Wording should be aligned for context specifiers/domain separation, because both RFCs use different terminology here as well. Both schemes have similar sub-functions, so the descriptions of these should be aligned as well. Align choice of hash functions.

-They recommend some parameter sets that aren't included in the RFCs.

-The first hashing phase of message signing is different between the two algorithms, and they would like more flexibility.

-They want AES (and possibly other primitives) added because a hardware accelerator exists for it and can speed up performance.

-The format of context identifiers can be different depending on the algorithms used. Currently optimized for SHA-256, but this isn't the only possible hash algo to use.

ISARA 2:

-They are wondering whether the comments will be made public.

-Good for code signing/certificates.

-The LMS IETF draft has too many possible parameter sets because they define OTS sets and the LMS sets independently.

-If standardizing multi-tree variants, then we should explicitly point out that at each layer of the tree, the same parameter sets are used, even though they can differ at different levels.

-They mention fault attacks, but we already discuss this.

-Existing APIs may be insufficient to describe state. For instance, if the private key is read only, the signing API can't easily evolve the key during the signing operation.

-State synchronization across a cluster of machines is hard.

-Because representations of state aren't standardized, there is room for error in development here.

-Writing state to disk isn't guaranteed to succeed on all platforms, where the write operation may be cached. Therefore, every component of the device between CPU and storage needs a power backup. Writing state to flash memory frequently will wear it out quickly. State must be synchronized between volatile and non-volatile memory.

-They suggest that the hardware should use ECC memory to avoid data corruption/bit flips.

Regards,

Michael

From: Davidson, Michael S. (Fed)

Sent: Wednesday, April 24, 2019 3:44 PM

To: Cooper, David A. (Fed) <david.cooper@nist.gov>; Dang, Quynh (Fed) <quynh.dang@nist.gov>; Apon, Daniel C. (Fed) <daniel.apon@nist.gov>; Miller, Carl A. (Fed) <carl.miller@nist.gov>; Dworkin, Morris J. (Fed) <morris.dworkin@nist.gov>

Subject: Summary of comments on stateful HBS

Hello everyone,

Below, I've summarized the comments we received. The lowest common denominator appears to be that 1) HSMs are relevant because state management is hard, and 2) root certificates are the primary use case.

Regards,

Michael

Adam Langley on X.509:
https://mailarchive.ietf.org/arch/msg/spasm/4EP3bX2adJBCmTjBMYazAKQJFU0

-State management to prevent nonce reuse isn't currently used by CAs, so there isn't an existing parallel.

Adam Langley via email:

-Claims that organizations that are sufficiently mature to use stateful HBS will do so with or without NIST approval. [Response: But then they can't work with the USG, and USG can't use them either.]

-Operational issues such as changes in staff or processes can easily lead to misuse. Stateful HBS requires HSMs and new organizational processes.

-Believes we send the wrong message of safety by standardizing this. [Response: We should do everything in our power to discourage using this in most contexts]

Panos Kampanakis:

-Don't use for online applications, or high frequency signing.

-Makes suggestions for managing state. Claims we should discuss VM cloning, signing parallelization, and applicability for X.509/root CAs. Believes root CA use is fine.

Andreas Huelsing:

-Agrees with Panos. Adds that state management is easier in a secure execution environment like an HSM.

Mike Brown of ISARA:

-Useful for IoT and root certificates, but not web traffic.

-Requests guidance on key reuse.

Larry Marks:

-"For the guidance for b) I understand that it may be recommended that the user should switch to SHA-3 for 512-bit hashes and to use SHA-512/224 and SHA-512/256 instead of SHA-224 and SHA-256 as a compensating control to make applications more secure." [Response: not really what we had in mind with this question, I think.]