Public Comments About the NIST IR 8214C ipd
"NIST First Call for Multi-Party Threshold Schemes" (initial public draft)

Frank W. Sudia[1]

April 10, 2023

To: nistir-8214c-comments@nist.gov

Re: Public Comments on NIST IR 8214C ipd (Call for Threshold)

2. Requirements and Recommendations for Submissions (Section 4)

As we embark on threshold standardization, it is well to recall the reasons threshold systems were initially developed, which included the requirement that the implemented system be able to pass Internal, External (i.e., Big X), and Federal Reserve or DoD Audits, remind ourselves to perform adequate risk assessments, and to manage the level of reliance to be placed on a given system or key, based on those risk assessments. No algorithm, no matter how clever, can function outside of some physical implementation, including associated message protocols, while being subject to audit, control methodologies, operational risk, and reliance management.

Recall the key info-sec principles of confidentiality, integrity, availability, and authenticity (CIAA), each of which can be defeated in various ways. Any responsible development team seeking to get their app into production will need to comply with a tall stack of app-development and security standards, and convince their Auditors to let them go live. How each mathematical quantity will be generated, stored, managed, used, verified, backed up, restored, where, when and by whom, in order to achieve CIAA objectives, must be explained and documented.

In addition to verifying the physical implementation, the Auditors will also ask you to prove that the system has the Abstract properties you say it has, and is immune from tampering by malicious adversaries. Is this really a 3 out of 5 system, and can we be assured there are no secret quorums or secret shares, let alone that one party somehow has the entire key, etc?

On the policy / marketing level, teams should recall how unpopular PKI has been, with experts decrying its risks, privacy groups deploring any central ID system, and government agencies who wish to "see through" everything, for whom every day there's no functioning PKI is another good day. To minimize such policy headwinds, non-centralized uses, such as *the critical need to more reliably protect Bitcoin wallets*, should be prioritized.

3. Technical Requirements (Section 5)

Turning to technical matters, in many cases the going-in Availability issue of how can we plausibly backup an important key in case of system failure, might just as well be addressed by a non-threshold (e.g., 3 out of 3) scheme using *High School Algebra*, as proposed in the pioneer US Patent 5,825,880 (expired). The eventual standard should include such non-threshold split key systems, as a limiting case, which may be simpler to implement and audit, yet could suffice for many users.

Another feature larger users may find attractive would be a well-defined option in the protocol to generate sub-splits, wherein each key share can be "split" (or co-generated) again, to one or more successive tiers, as a further security and backup/recovery risk reduction feature.

[1]Independent researcher. No conflicts of interest.

All such protocols should be future-proofed by including variable algorithm IDs and key lengths.

As remarked in this Request For Comments, "On the other extreme, a proposed protocol must not allow the major safety properties of interest to be trivially broken in case of adaptive corruptions, as in the classical example of a protocol that delegates all capabilities to a small quorum that is difficult to guess in advance, but whose overall corruption (by an adaptive adversary) would be disastrous." Page 22, lines 912-915. Not to mention that such a system violates Audit principles, defrauds anyone who participates in or relies on it, creates financial, legal, and reputational risks, and may constitute a felony of tampering with an official system, warranting jail terms.

The enumeration of Adversaries must be expanded to include not only corrupt protocol participants or malicious intermediaries, but also corrupt sponsor personnel. Any persons or groups designing, setting up, or effectuating key generation operations may be in a position to mis-design or mis-configure the process, either through their own math skills, exploits from criminal groups, or with the assistance of prior clients or employers (some with preeminent math skills) who they are still secretly working for. The notion that “you” will somehow be able to verify all this yourself is most likely false, since in reality some developer or consultant will be the one doing it.

Hence in order to convince your many Audit teams that your system has the properties you say it has, and that adversaries (including any double-agents among your staff or consultants) have not altered them, it will be preferable if its critical properties can be verified, e.g., via a test suite of protocols, viewable by all participants, and logged onto some blockchain, which demonstrate them.

For example, you could start by challenging all participants to create all partial signatures, to assure that no signatures with less that the stated quorum of signers are valid, including proof (possibly zero knowledge) that each partial signature was indeed made by the key shares that each participant supposedly has. Such proofs could be run again each time the system is modified to add, delete, or replace a key share, to assure that the claimed properties (or any new set of properties) still hold, and (perhaps for a nominal fee) upon demand of any participant.

As a worst-case scenario, proof that the conditions held *during a given signing operation* might be block-chained and appended to the signature, as a further proof of its genuineness.

Many variations of such proofs might arise, but I will not develop any of them here. (Nor am I planning to research, develop, or patent any further ideas related to this topic.)

Many thanks to NIST for initiating this effort, and best of luck to the proposing teams.

9. Other Comments

Some Related Patents (<u>all expired</u>):

1. Sudia et al, US Patent 5,825,880, Multi-Step Digital Signature Method and System, 10-20-1998 (the *High School Algebra* version)
2. Brickell et al, US Patent 5,867,578, Adaptive Multi-Step Digital Signature System and Method of Operation Thereof, 2-2-1999 (the advanced math version)
3. Asay et al, US Patent 5,903,882, Reliance server for electronic transaction system, 5-11-99 (part 2 of the spec, written by me, may be more readable)
4. Sudia et al, US Patent 6,209,091, Multi-Step Digital Signature Method and System, 3-27-01 (further claims re delegation)