NJM Studios

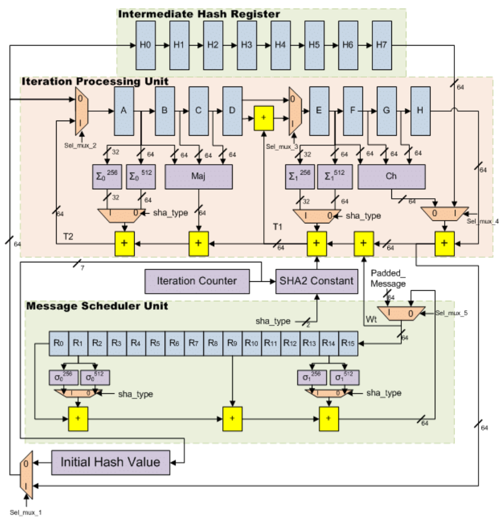
Product End-User Information
3rd Edition

NJM Prototype Interplatform Hashing

November 8th, 2022

Product Overview

The NJM 2 Encryption standard builds on the Secure Hash Algorithms released by the US NSA in 1993. More specifically, it utilizes a more recently published hash, SHAKE256, released in 2015. At a low level, NJM slightly modifies the SHAKE256 hash to use 51 standard operations per block instead of 64, but also adds intermediary padding that occurs pseudorandomly between corestep [REDACTED].



Top Level Security Introductions

Beyond the hash algorithm itself, NJM standard pipelines require a stream of data to serve as a key, as opposed to intaking raw bits. This stream of data functions as a circular linked-list with a dynamic point of entry, declared by the initial signed author. If an attempting signer fails to authenticate with a stop in the circulation, the next correct endpoint (as well as various “trap” endpoints) are recycled and dumped. This information is updated circularly before the attempting signer receives a failure callback. The starting point may or may not update at this time.

Backups/Rewrites

This technology is intended to be implemented between multiple servers to heavily decrease the chance of the key being obtained by a malicious actor. However, if a server in the chain drops, the rest of the chain is expected to update dynamically.

506 - Variant Also Negotiates

If a server/endpoint fails, its response is translated into a 506 error, “Variant also Negotiates”. In an NJM key/keychain, exactly 5 bytes are reserved to inform the previous endpoint of what to do in case of failure. This information is decrypted using [REDACTED].

Mass Failure

If more than 29.9% of the servers/endpoints in a chain fail, the chain will break permanently and demand a rewrite from the initial author. This permanently locks all data previously referenced by the key. When the initial author recycles the key and instantiates new servers/endpoints, the data can be recovered using a backup key.

Backup Key

On the standard pipeline, feeding into a functioning chain AND passing a backup key to the final endpoint will revitalize all previously broken chains for exactly 1 call, and then prematurely delete them before the call finishes. The data will be transmitted without any buffer or callbacks to prevent malicious actors from abusing the backup key. If the backup key is used again before the first call data is returned, the first call will be canceled and the chain will be erased early. If the backup key is passed to any endpoint other than the final endpoint, the chain will be erased with no callback.

Notice: An actor with access to a backup key has the ability to permanently delete all data on a chain, but not the ability to read it.