03/02/2022 Incident analysis

General Description

On March 02, 2022 (epoch 284), a series of cascade failures in the automated scoring mechanism happened that caused almost all validators to get a score of zero, that in turn caused an emergency unstake of almost 6M SOL of Marinade stake.
At no point there were funds at risk.

Since no funds were at risk, we decided to address issues in our own off-chain scoring code and then to restake to more than 450 validators, as we normally would according to the delegation formula during epoch 285.

Technical Description

The incident was triggered by a cascade of events and failures leading to the unstake:

1. Solana changed the "getConfirmedBlock" JSON API to "getBlock" (planned deprecation).

2. GenesysGo, that runs the RPC servers that Stakeview.app queries, upgraded to the version of RPC software that implemented that change.

3. As a result, Stakeview.app automated fetch of credits-earned data was getting nothing because it was querying a now-deprecated API, and as a result it was calculating zero credits for all validators, resulting in reported 0% APY for everyone. That was solved on the Stakeview.app side, but not before March 2, 1:11 UTC when the scoring bot grabbed the data.

4. The scoring bot uses Stakeview.app APY information, and there was a “check” to emergency-unstake validators with 0% APY. (Note: We removed the dependency by using our own information about current epoch credits to determine if the validator was down for most of the current epoch. We still query the stakeview.app APY data because it is useful, but only for informational purposes)

5. Our bot read 0% APY from stakeview.app epoch-report, that led to preparing an emergency-unstake for all validators with 0% APY. When reading stakeview.app information, we control for several error conditions: no-file, invalid json, null, undefined, no APY data on specific validators, etc. But there was not any specific control for getting a valid json with all validators and 0.0% APY for everyone.

6. Final cascading error. We had in place an "emergency-brake" mechanism to stop the emergency-unstake automation if the unstakes were too many. That automation emergency-brake code failed on our side.

7. Most emergency unstakes were executed

Impact:

1. All validators got an APY of 0%, and the score process decided to emergency-unstake (that's the programmed action)
    - We incorrectly used 3rd party data (APY) to program emergency-unstake actions.
    - On March 2, 1:11 UTC, the APY returned was 0% for all validators
    - While we have sanity checks for the total count of APY records reported, we have no sanity checks on the distribution of the values.

2. Secondary Problem: Emergency Brake failure: We had an emergency brake in place to avoid unstaking too many validators at once by the automated process. However, the emergency brake did not evaluate the data correctly and failed to stop the process. The previously performed tests for Emergency Brake did not reveal the bug due to the nature of the test dataset.

- The brake is a bash expression comparing two numbers, and the correct way of doing that is (($A < $B)) or [[$A -lt $B]], incorrect way is [[$A < $B]] . The correct ways compare A and B as if A and B were numbers, while the incorrect compare as if those variables were strings. When setting the brake's threshold to 60, you want the total count of emergency unstakes to be less than 60: tests performed were in the line of: does it behave as expected with 0, 1, 10, 59 and does it let the unstakes through? does it behave as expected with 60, 69, 90 and does it stop the execution? well if you try it with 455, then the "incorrect" expression [[450 < 60]] evaluates as true, because 450 is alphabetically less than 60
  Lessons: when possible, prefer languages with strong typing (rust?) or languages that at least try to look like they have some sort of typings (TS?).. sadly, bash is the go-to lang for CI/CD, automated processes etc.

Steps taken to prevent similar situations in the future:
- Emergency brake has been fixed.
- Sanity check for the total score returned by the processing was added.
- We are no longer using stakeview.app to control emergency unstakes (we only use it for informative purposes now)

3. TVL reports: This large emergency-unstake operation led to mistakenly reporting a drop in the Marinade's "total value locked" displayed on our website, Defillama, and other sources. The TVL report has now been updated to account even for *yet to be staked SOL* under Marinade's control.

4. The result of the incident is similar to performing a "full-rebalance" of most of Marinade stake. The cost of such full-rebalance, considering 6 mil SOL unstaked and re-staked in the following epoch, is estimated as approximately 0.044% of the amount rebalanced, or 2640 SOL (one epoch of rewards). The rebalance could prevent the price of mSOL to raise 0.0004 at the end of this epoch. We're committed to bear the full cost of the incident as described and we're analyzing the best mechanism to make mSOL token holders whole.