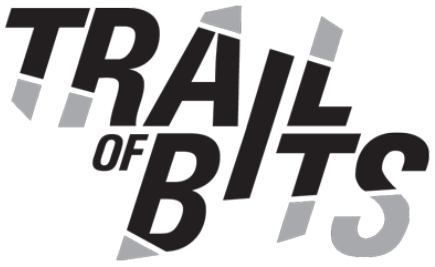

Symbol

Security Assessment

27th July, 2020

Prepared For:
Iain Wilson | *NEM Group*
info@nem.group

Dave Hodgson | *NEM Group*
info@nem.group

David Mansell | *NEM Group*
info@nem.group

Prepared By:
Dominik Czarnota | *Trail of Bits*
dominik.czarnota@trailofbits.com

Jim Miller | *Trail of Bits*
jim.miller@trailofbits.com

David Pokora | *Trail of Bits*
david.pokora@trailofbits.com

Changelog:

Jul 27, 2020:	Initial report draft
Aug 7, 2020:	Updated issue #11
Aug 31, 2020:	Added Appendix D. Fuzzing catapult-server
Sep 18, 2020:	Updates made to Appendix D
Sep 24, 2020:	Added Appendix E. Fix Log
Oct 30, 2020:	Review of amendments detailed in Appendix F
Nov 25, 2020:	Review of amendments detailed in Appendix G
Dec 7, 2020:	Added Appendix H and extended executive summary

Executive Summary

From June 13 through June 24, 2020, NEM Group engaged Trail of Bits to review the security of the Symbol network's node and REST gateway repositories. Trail of Bits conducted this assessment over the course of four-person weeks with two engineers working from the provided repositories.

In the first week of the assessment, we employed static analysis techniques such as cppcheck, scan-build, CodeQL, and CLion code inspection while gaining a deeper understanding of the codebase through manual review. Preliminary manual review included potential generic classes of issues such as memory corruption, use of cryptography, data validation, auditing/logging, configuration, and more. This led to the discovery of six findings ranging from informational to high severity.

In the final week, we conducted a deeper analysis of the Symbol network, identifying critical components within the system and pursuing testing against those targets. This led to the discovery of several new issues ranging from informational to low severity. A partial review of data flow and chain state validation revealed insufficient data validation issues related to maximum packet sizes (TOB-SYM-007) and inflationary mosaic supply (TOB-SYM-008) within the catapult server, and incorrect receipt types defined within the REST gateway (TOB-SYM-011). Further review of authentication schemes and access controls revealed additional concerns regarding default permissions used with boost filesystem APIs (TOB-SYM-009).

Finally, a review of cryptography led to the discovery of a potential padding oracle attack (TOB-SYM-010). Trail of Bits engineers also began pursuing dynamic testing via `libFuzzer` and various engines multiplexed via `DeepState`. Unfortunately, when we wished to employ dynamic analysis, the fuzzing targets were often at a level of depth within the system that required additional overhead for setup.

Overall, we believe the Symbol network repositories reviewed show positive consideration for common classes of security vulnerabilities. As a result, low-hanging fruit seemed less prevalent throughout the codebase, and static analysis techniques reflected few true-positive concerns. Trail of Bits believes there may be latent data validation issues deeper within the system that could be exposed through increased dynamic testing. We recommend integrating a robust fuzzing harness such as `AFL` to test security properties in-depth, while employing static analysis tools in the CI/CD build pipeline to uncover newly introduced vulnerabilities and strengthen the system's security posture.

Update September 24, 2020: Trail of Bits reviewed fixes implemented for the issues presented in this report, amendments to voting key structure, and additional changes made to certain repositories. See the results from those reviews in Appendix E: Fix Log, Appendix F: Amendments

to Voting Key Structure, *Appendix G: Additional Symbol changes review*, *and H: Fix log for issues from Appendix G*.

Project Dashboard

Application Summary

Name	Symbol
Version	`catapult-server` commit: f84eb88727 `catapult-rest` commit: 25d62f2393
Type	C++, JavaScript
Platforms	Windows, Linux

Engagement Summary

Dates	July 13–July 24, 2020
Method	Whitebox
Consultants Engaged	2
Level of Effort	4 person-weeks

Vulnerability Summary

Total High-Severity Issues	1	■
Total Medium-Severity Issues	1	■
Total Low-Severity Issues	3	■ ■ ■
Total Informational-Severity Issues	3	■ ■ ■
Total Undetermined-Severity Issues	3	■ ■ ■
Total	11	■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■

Category Breakdown

Access Controls	2	■ ■
Configuration	2	■ ■
Cryptography	2	■ ■
Data Validation	2	■ ■ ■
Denial of Service	1	■
Undefined Behavior	2	■ ■
Total	11	■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■

Engagement Goals

This engagement was scoped to provide a security assessment of the previously mentioned Symbol network components and their communications, namely the `catapult-server` and `catapult-rest` repositories.

Specifically, we sought to answer the following questions:

- Are there standard compiler and platform mitigations that could be leveraged to improve the security posture of Symbol nodes?
- Do the provided repositories have code correctness issues that could lead to memory corruption, arbitrary code execution, etc.?
- Are appropriate access controls set for sensitive information stored at rest?
- Do the target components make appropriate use of encryption to secure sensitive information?
- Is the default configuration for the server overly permissive?
- Are there any insufficient data validation issues that may lead to a denial-of-service attack? Could they be leveraged against the consensus model to take over the network?
- Does logging result in sensitive data exposure? Could the logging system be leveraged to perform a resource exhaustion attack?

Coverage

This section highlights some of the analysis coverage we achieved based on our high-level engagement goals.

- Review of currently configured compiler mitigations revealed the need for additional mitigations that would help prevent an attack against the server. (TOB-SYM-001).
- In analyzing the codebase for generic classes of vulnerabilities such as memory corruption, use-after-frees, arbitrary code execution, and more, we discovered an issue dereferencing a potentially invalid pointer (TOB-SYM-002) and a lack of overflow checks (TOB-SYM-008).
- While investigating the use of cryptography in the system, we discovered a potentially cryptographically insecure RNG (TOB-SYM-004) and a potential padding oracle attack (TOB-SYM-010). We also found some positive reinforcements, including appropriate wrapping of dependencies like OpenSSL and proper implementation of EdDSA.
- Review of various file operations revealed permission-based concerns (TOB-SYM-005, TOB-SYM-006, TOB-SYM-009).

- Default server configurations related to authentication were found to be sufficient. For example, by default, only machine-local connections are allowed, and the maximum ban list size is fairly large.
- Based on our review of improper data validation, we are concerned about `ConfigurationBags` verification (TOB-SYM-003), maximum packet sizes potentially leading to denial of service (TOB-SYM-007), and parsing of an incorrect receipt type in the REST gateway (TOB-SYM-011).
- A partial review of logging did not indicate that sensitive information such as key material could be leaked.

Recommendations Summary

This section aggregates all the recommendations made during the engagement. Short-term recommendations address the immediate causes of issues. Long-term recommendations pertain to the development process and long-term design goals.

Short term

❑ **Enable security mitigations for the `catapult-server` build using the compiler and linker flags described in the table below.** While the compiler used in the `catapult-service-bootstrap` build enables some of the mitigations by default, enabling them explicitly ensures the mitigations are there in case the compiler used has different defaults. TOB-SYM-001

❑ **Refactor the method so the `empty()` check ensures at least one item exists in the container before calling `back()` and dereferencing the underlying data.** TOB-SYM-002

❑ **Validate the exact configuration bag size to prevent issues resulting from unexpected size changes.** TOB-SYM-003

❑ **Replace the internal randomness provider for the `HighEntropyRandomGenerator` with one that explicitly provides cryptographically secure randomness, such as those provided by OpenSSL and LibSodium.** TOB-SYM-004

❑ **Add the `O_CLOEXEC` flag to the flags used for opening files in the `RawFile` class by default and drop the flag explicitly when needed.** This will prevent leaking opened file descriptors into child processes if the node spawns child processes. TOB-SYM-005

❑ **Explicitly set the permissions of the `symbol-cli.config.json` file to `0o600` in the `symbol-cli` when creating or saving the file.** This will prevent users on the same machine from accessing someone else's account if they access the config file and are able to brute-force the password. TOB-SYM-006

❑ **Set lower maximum default packet sizes in the `PacketPayloadBuilder` and `ServerPacketHandlers` classes' constructors.** This will help prevent unwanted denial-of-service scenarios even if an attacker is able to construct big enough payloads to cause them. TOB-SYM-007

❑ **Don't use `boost::filesystem::create_directory` or `create_directories` functions for creating directories in `catapult-server`.** Instead, use a method that allows you to set permissions of the created directories explicitly and appropriately. This will prevent

insecure configurations that may lead to unexpected behavior and allow access from other system users in less securely configured systems. TOB-SYM-009

❑ **Include an HMAC alongside the AES-CBC encryption to ensure that ciphertexts are not altered by an adversary and to prevent potential padding oracle attacks.** TOB-SYM-010

❑ **Fix the duplicate `ReceiptType` value in the `catapult-rest` codebase.** TOB-SYM-011

Long term

❑ **Enable the security mitigations for all binaries used and add scanning for security mitigations into the CI/CD pipeline to ensure that certain options are always enabled.** This will make it harder to exploit potential bugs found in dependencies used by the `catapult-server`. TOB-SYM-001

❑ **Review the possible return values of such functions to ensure the expected data will always be returned or otherwise handled accordingly.** TOB-SYM-002

❑ **Be mindful of the importance of using libraries that may resolve to implementations that do not satisfy requirements when using a given platform or compiler.** TOB-SYM-004

❑ **Check the permissions of the `symbol-cli.config.json` file when reading it in the `symbol-cli` and warn the user if the file permissions are overly broad.** This will help prevent situations in which the user accidentally sets permissions that are too broad for the file and never changes them back. TOB-SYM-006

❑ **Add overflow checks that would either error out the program or log an error to `BaseValue` operators, `ChainScore` operators, the `MosaicEntrySupplyMixin::increaseSupply` function, and others.** If those checks are unwanted due to performance issues, enable them only during tests and debug builds. This will help catch bugs earlier and prevent unwanted overflows that could cause financial risks. TOB-SYM-008

❑ **Consider clearing plaintext buffers upon any authentication or decryption failure to prevent invalid plaintexts from being used.** TOB-SYM-010

Findings Summary

#	Title	Type	Severity
1	Missing compiler mitigations	Configuration	Low
2	Undefined behavior dereferencing `std::list.back()` on an empty container	Undefined Behavior	Undetermined
3	Current `ConfigurationBags` verification may lead to bugs	Data Validation	Informational
4	High-entropy RNG does not guarantee high entropy	Cryptography	Medium
5	Use `O_CLOEXEC` flag by default when opening files on Linux	Configuration	Informational
6	The `symbol-cli` saves the config file as readable for others	Access Controls	High
7	Maximum packet size of 4GB may lead to denial-of-service attacks	Denial of Service	Undetermined
8	Lack of overflow checks	Data Validation	Informational
9	The `boost::filesystem::create_directory` defaults to 0777 permissions	Access Controls	Low
10	Potential padding oracle attack in `AesCbcDecrypt`	Cryptography	Undetermined
11	Incorrect `ReceiptType` in `catapult-rest`	Undefined Behavior	Low

1. Missing compiler mitigations

Severity: Low
Type: Configuration
Target: `catapult-server`

Difficulty: Undetermined
Finding ID: TOB-SYM-001

Description

The `catapult-server` build does not enable all modern compiler security mitigations. This makes it easier for an attacker who finds a low-level vulnerability to exploit a bug and gain control over the process.

Trail of Bits analyzed the `catapult-server` binary and its dependencies from the 0.9.5.1 version built from the [`catapult-service-bootstrap` repository](). We analysed both `cmake` files and checked the resulting binaries with the [`checksec`]() tool version 2.2.3 (aea5f9d) and with [`checksec.rs`]() version 0.0.6 (`d4da9ad`). Figure TOB-SYM-001.1 shows the `checksec.sh` output. All of the dependencies used lack "FULL RELRO" and some of them are missing fortify source and stack canaries. Also, while not detected by `checksec.sh`, the binaries are missing the stack clash protection, CFI, and SafeStack mitigations.

```
→ checksec checksec --dir=./catapult-binaries/
RELRO           STACK CANARY      NX           PIE           RPATH      RUNPATH    Symbols          FORTIFY Fortified      Fortifiable   Filename
Full RELRO      Canary found      NX enabled   PIE enabled   RW-RPATH   No RUNPATH   6215) Symbols    Yes   2              8             ./catapult-binaries/bin/catapult.recovery
Full RELRO      Canary found      NX enabled   PIE enabled   RW-RPATH   No RUNPATH   5567) Symbols    Yes   2              8             ./catapult-binaries/bin/catapult.broker
Full RELRO      Canary found      NX enabled   PIE enabled   RW-RPATH   No RUNPATH   8332) Symbols    Yes   3              10            ./catapult-binaries/bin/catapult.server
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   4049) Symbols    Yes   2              8             ./catapult-binaries/lib/libextension.syncsource.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   642) Symbols     No    0              1             ./catapult-binaries/lib/libcatapult.mongo.plugins.accountlink.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   2183) Symbols    Yes   1              5             ./catapult-binaries/lib/libcatapult.plugins.hashcache.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   1184) Symbols    No    0              3             ./catapult-binaries/lib/libcatapult.mongo.plugins.lockhash.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   1245) Symbols    No    0              4             ./catapult-binaries/lib/libcatapult.mongo.plugins.metadata.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   3724) Symbols    Yes   2              8             ./catapult-binaries/lib/libextension.pluginhandlers.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   2899) Symbols    Yes   1              5             ./catapult-binaries/lib/libcatapult.plugins.multisig.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   4000) Symbols    Yes   2              8             ./catapult-binaries/lib/libextension.zeromq.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   3396) Symbols    Yes   1              5             ./catapult-binaries/lib/libcatapult.plugins.locksecret.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   1483) Symbols    No    0              4             ./catapult-binaries/lib/libcatapult.mongo.plugins.namespace.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   3997) Symbols    Yes   2              8             ./catapult-binaries/lib/libextension.networkheight.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   3876) Symbols    Yes   2              8             ./catapult-binaries/lib/libextension.hashcache.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   3870) Symbols    Yes   2              8             ./catapult-binaries/lib/libextension.filespooling.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   7104) Symbols    Yes   3              10            ./catapult-binaries/lib/libextension.packetserver.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   1230) Symbols    No    0              3             ./catapult-binaries/lib/libcatapult.mongo.plugins.locksecret.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   7456) Symbols    Yes   3              10            ./catapult-binaries/lib/libextension.timesync.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   5206) Symbols    Yes   2              8             ./catapult-binaries/lib/libextension.mongo.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   3529) Symbols    Yes   1              5             ./catapult-binaries/lib/libcatapult.plugins.mosaic.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   1199) Symbols    No    0              4             ./catapult-binaries/lib/libcatapult.mongo.plugins.restrictionaccount.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   1654) Symbols    Yes   1              5             ./catapult-binaries/lib/libcatapult.plugins.aggregate.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   5666) Symbols    Yes   2              8             ./catapult-binaries/lib/libextension.harvesting.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   8649) Symbols    Yes   3              10            ./catapult-binaries/lib/libextension.partialtransaction.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   3283) Symbols    Yes   1              5             ./catapult-binaries/lib/libcatapult.plugins.lockhash.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   10261) Symbols   Yes   3              10            ./catapult-binaries/lib/libextension.sync.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   2761) Symbols    Yes   1              5             ./catapult-binaries/lib/libcatapult.plugins.metadata.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   3232) Symbols    Yes   1              5             ./catapult-binaries/lib/libcatapult.plugins.restrictionaccount.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   1141) Symbols    No    0              3             ./catapult-binaries/lib/libcatapult.mongo.plugins.multisig.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   4315) Symbols    Yes   1              5             ./catapult-binaries/lib/libcatapult.plugins.coresystem.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   2449) Symbols    Yes   1              5             ./catapult-binaries/lib/libcatapult.plugins.accountlink.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   1575) Symbols    Yes   1              5             ./catapult-binaries/lib/libcatapult.plugins.signature.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   3086) Symbols    Yes   1              5             ./catapult-binaries/lib/libcatapult.plugins.restrictionmosaic.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   3990) Symbols    Yes   1              5             ./catapult-binaries/lib/libcatapult.plugins.namespace.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   614) Symbols     No    0              1             ./catapult-binaries/lib/libcatapult.mongo.plugins.transfer.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   1195) Symbols    No    0              3             ./catapult-binaries/lib/libcatapult.mongo.plugins.restrictionmosaic.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   3866) Symbols    Yes   2              8             ./catapult-binaries/lib/libextension.transactionsink.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   1278) Symbols    No    0              3             ./catapult-binaries/lib/libcatapult.mongo.plugins.mosaic.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   2100) Symbols    Yes   1              6             ./catapult-binaries/lib/libcatapult.plugins.transfer.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   3824) Symbols    Yes   2              8             ./catapult-binaries/lib/libextension.addressextraction.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   4239) Symbols    Yes   2              8             ./catapult-binaries/lib/libextension.diagnostics.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   3771) Symbols    Yes   2              8             ./catapult-binaries/lib/libextension.unbondedpruning.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   6866) Symbols    Yes   3              10            ./catapult-binaries/lib/libextension.nodediscovery.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   RW-RUNPATH   645) Symbols     No    0              1             ./catapult-binaries/lib/libcatapult.mongo.plugins.aggregate.so
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   No RUNPATH   631) Symbols     Yes   3              9             ./catapult-binaries/deps/libbson-1.0.so.0.0.0
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   No RUNPATH   2772) Symbols    Yes   5              14            ./catapult-binaries/deps/libzmq.so.5.2.2
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   No RUNPATH   1434) Symbols    No    0              3             ./catapult-binaries/deps/libboost_program_options.so.1.71.0
Partial RELRO   No canary found   NX enabled   DSO           No RPATH   No RUNPATH   48) Symbols      No    0              0             ./catapult-binaries/deps/libboost_system.so.1.71.0
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   No RUNPATH   13054) Symbols   Yes   7              20            ./catapult-binaries/deps/librocksdb.so.6.6.4
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   No RUNPATH   570) Symbols     No    0              2             ./catapult-binaries/deps/libbsoncxx.so.3.4.0
Partial RELRO   No canary found   NX enabled   DSO           No RPATH   No RUNPATH   54) Symbols      No    0              0             ./catapult-binaries/deps/libboost_atomic.so.1.71.0
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   No RUNPATH   226) Symbols     No    0              2             ./catapult-binaries/deps/libboost_chrono.so.1.71.0
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   No RUNPATH   1784) Symbols    Yes   4              11            ./catapult-binaries/deps/libboost_regex.so.1.71.0
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   No RUNPATH   284) Symbols     No    0              2             ./catapult-binaries/deps/libboost_date_time.so.1.71.0
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   No RUNPATH   3711) Symbols    Yes   4              11            ./catapult-binaries/deps/libboost_log.so.1.71.0
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   No RUNPATH   2279) Symbols    Yes   4              12            ./catapult-binaries/deps/libmongoc-1.0.so.0.0.0
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   No RUNPATH   No Symbols       Yes   4              7             ./catapult-binaries/deps/libgflags.so.2.2.1
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   No RUNPATH   No Symbols       No    0              3             ./catapult-binaries/deps/libsnappy.so.1.1.7
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   No RUNPATH   2321) Symbols    No    0              2             ./catapult-binaries/deps/libmongocxx.so.3.4.0
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   No RUNPATH   750) Symbols     No    0              3             ./catapult-binaries/deps/libboost_thread.so.1.71.0
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   No RUNPATH   2875) Symbols    Yes   1              5             ./catapult-binaries/deps/libboost_log_setup.so.1.71.0
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   No RUNPATH   438) Symbols     No    0              7             ./catapult-binaries/deps/libboost_filesystem.so.1.71.0
Partial RELRO   Canary found      NX enabled   DSO           No RPATH   No RUNPATH   185) Symbols     No    0              2             ./catapult-binaries/deps/libboost_timer.so.1.71.0
→ checksec
```

Figure TOB-SYM-001.1: `checksec` *output. Note that the yellow DSO (dynamic shared object) is ok, as dynamic libraries are always built with PIC (position-independent code), which is the same as PIE (position-independent executable).*

Modern compilers support a number of exploit mitigations, including:

- NX (non-executable data)
- PIE (position-independent code for ASLR)
- stack canaries (for buffer overflow detection)
- RELRO (for hardening data sections)
- FORTIFY_SOURCE (for additional buffer overflow detection and format string protection)
- stack clash protection (for detecting when stack pointer clashes with other memory region)
- CFI (control flow integrity)
- SafeStack (for further stack-overflow protection)

By default, compilers do not enable many of these mitigations. For a detailed description of these exploit mitigation technologies, see Appendix B.

Recommendation
Short term, enable security mitigations for the `catapult-server` build using the compiler and linker flags described in the table below. While the compiler used in the `catapult-service-bootstrap` build enables some of the mitigations by default, enabling them explicitly ensures the mitigations are there in case the compiler used has different defaults.

For additional assurance, consider verifying if ASLR is enabled during program startup by checking if the value stored in the `/proc/sys/kernel/randomize_va_space` file is 2, and error out if it is lower.

Long term, enable the security mitigations for all binaries used and add scanning for security mitigations into the CI/CD pipeline to ensure that certain options are always enabled. This will make it harder to exploit potential bugs found in dependencies used by the `catapult-server`.

GCC flag	What it enables
`-z noexecstack`	NX bit
`-Wl,-z,relro,-z,now`	Full RELRO (read-only segments after relocation and disables lazy bindings)
`-fstack-protector-all`	Adds stack canaries for all the functions. (Note: This might make the program

or (less secure) `-fstack-protector-strong --param ssp-buffer-size=4`	slower. To protect only functions that have buffers, use the latter version.)
`-fPIE -pie`	PIE (needs ASLR enabled when launching a program).

`-D_FORTIFY_SOURCE=2 -O2`
 	
or (less secure)
 	
`-D_FORTIFY_SOURCE=1 -O1`
	FORTIFY_SOURCE protections.
	(Note that it requires an appropriate optimization flag (`-O1` or `-O2`).
	The latter version (`-D_FORTIFY_SOURCE=1 -O1`) is less secure as it will enable only compile-time protections, while the former will also add runtime checks.)

`-fstack-clash-protection`	Adds checks to functions that may allocate a lot of memory on the stack to ensure the new stack pointer (and stack frame) do not overlap with another memory region such as heap.
`-fsanitize=cfi -fvisibility=hidden -flto`	Enables control flow integrity checks that help prevent program control flow hijacking (Clang/LLVM only).
`-fsanitize=safe-stack`	Enables `SafeStack` which splits stack frames of certain functions into the safe stack and the unsafe stack, to make it harder to hijack the program's control flow (Clang/LLVM only).
`-Wall -Wextra -Wpedantic -Wshadow -Wconversion -Wformat-security`	Compile-time checks and warnings.

References

- Debian hardening recommendations
- GCC `man` page
- LD `man` page (see -z keywords)

2. Undefined behavior dereferencing `std::list.back()` on an empty container

Severity: Undetermined | Difficulty: Undetermined
Type: Undefined Behavior | Finding ID: TOB-SYM-002
Target: `catapult-server/src/catapult/ionet/NodeInteractionsContainer.cpp`

Description
The `NodeInteractionsContainer` has an operation to add a `NodeInteractionBucket` to a `std::list`, but it dereferences `std::list.back()` prior to checking `std::list.empty()`, which may produce undefined behavior.

```
	void NodeInteractionsContainer::addInteraction(Timestamp timestamp, const
consumer<NodeInteractionsBucket&>& consumer) {
		auto bucketAge = utils::TimeSpan::FromDifference(timestamp,
m_buckets.back().CreationTime);
		if (m_buckets.empty() || BucketDuration() <= bucketAge)
			m_buckets.push_back(NodeInteractionsBucket(timestamp));

		consumer(m_buckets.back());
	}
```

Figure 2.1: *Undefined behavior dereferencing* `std::list.back()` *on an empty container (catapult-server/src/catapult/ionet/NodeInteractionsContainer.cpp#L71-L77).*

The C++ reference noted below states: “Calling this function on an empty container causes undefined behavior.” For example, depending on the platform, this could produce a denial of service by accessing protected/non-existent memory regions.

The reachability of this code path or likelihood of reproduction was not determined, so this issue is marked undetermined severity.

Exploit Scenario
Bob is an operator of a public Symbol network node who expects uninterrupted service due to the perceived nature of Symbol’s security and decentralization. However, due to this bug, Bob’s node encounters a crash. This may also affect other operators like Bob and their ability to perform consensus, which may have dire consequences for network governance.

Recommendation
Short term, refactor the method so the `empty()` check ensures at least one item exists in the container prior to calling `back()` and dereferencing the underlying data.

Long term, review the possible return values of such functions to ensure the expected data will always be returned or otherwise handled accordingly.

References
- `list::back` - C++ Reference

3. Current `ConfigurationBags` verification may lead to bugs

Severity: Informational | Difficulty: Low
Type: Data Validation | Finding ID: TOB-SYM-003
Target: `catapult-server/src/catapult/utils/ConfigurationUtils.cpp`

Description
The `catapult-server` uses the `VerifyBagSizeLte` function to verify the size of the loaded configuration bag. This function only throws an error if the configuration bag is bigger than expected. Therefore, since adding a new parameter will still pass the verification without changing the expected size, any changes to the number of loaded parameters might lead to an unwanted situation.

```
void VerifyBagSizeLte(const ConfigurationBag& bag, size_t expectedSize) {
    if (bag.size() > expectedSize)
        CATAPULT_THROW_INVALID_ARGUMENT_1("configuration bag contains too many properties",
bag.size());
}
```

Figure TOB-SYM-003.1: The `VerifyBagSizeLte` *function (catapult-server/blob/v0.9.6.3/src/catapult/utils/ConfigurationUtils.cpp#L38-L41).*

Recommendation
Short term, validate the exact configuration bag size to prevent issues resulting from unexpected size changes.

4. High-entropy RNG does not guarantee high entropy

Severity: Medium | Difficulty: Low
Type: Cryptography | Finding ID: TOB-SYM-004
Target: `catapult-server/src/catapult/utils/RandomGenerator.cpp`

Description
The `HighEntropyRandomGenerator` is used as a source of cryptographically secure randomness. Unfortunately, this is based on `std::random_device`, which is not guaranteed to be high-entropy or cryptographically secure.

Without explicit hardening of standard libraries, different compilers or target platforms may not offer a cryptographically secure source of randomness via `std::random_device`. The provider may even be deterministic.

Exploit Scenario
Bob is a node operator who wishes to run Symbol network nodes on a given platform. An attacker, Eve, knows that `std::random_device` is not cryptographically secure on the platform Bob's nodes are running on, so she may be able to deduce Bob's private keys more easily, leaving Bob's funds at risk.

Recommendation
Short term, replace the internal randomness provider for the `HighEntropyRandomGenerator` with one that explicitly provides cryptographically secure randomness, such as those provided by OpenSSL and LibSodium.

Long term, be mindful of the importance of using libraries that may resolve to implementations that do not satisfy requirements when using a given platform or compiler.

References
- Is `std::random_device` cryptographic[ally] secure?
- Everything You Never Wanted to Know about C++'s `random_device`: It Might Actually Be Deterministic

5. Use O_CLOEXEC flag by default when opening files on Linux

Severity: Informational
Type: Configuration
Target: `catapult-server/src/catapult/io/RawFile.cpp`

Difficulty: Undetermined
Finding ID: TOB-SYM-005

Description
The `RawFile` wrapper used for opening files does not use the `O_CLOEXEC` flag by default. Enabling this flag allows an opened file to close when a process calls `execve` and thereby prevents unexpected leaks of file descriptors to child processes.

```
namespace catapult { namespace io {
    namespace {
        // (...)
#ifdef _MSC_VER
        // (...)
#else
        // (...)
        constexpr auto Flag_Read_Only = O_RDONLY;
        constexpr auto Flag_Read_Write = O_RDWR;
        constexpr auto New_File_Create_Truncate_Flags = O_CREAT | O_TRUNC;
        constexpr auto New_File_Create_Flags = O_CREAT;

        FileOperationResult<int> nemOpen(int& fd, const char* name, OpenMode mode, LockMode
lockMode) {
            int flags = mode == OpenMode::Read_Only ? Flag_Read_Only : Flag_Read_Write;
            int createFlag = mode == OpenMode::Read_Write
                ? New_File_Create_Truncate_Flags
                : (mode == OpenMode::Read_Append ? New_File_Create_Flags : 0);
            int lockingFlags = LockMode::File == lockMode
                ? ((flags & Flag_Read_Write) ? File_Locking_Exclusive :
File_Locking_Shared_Read)
                : File_Locking_None;

            return open(fd, name, File_Binary_Flag | flags | createFlag, lockingFlags,
New_File_Permissions);
        }
```

Figure TOB-SYM-005.1: File flags and their usage (catapult-server/blob/v0.9.6.3/src/catapult/io/RawFile.cpp#L134-L137 and #L230-L240).

Recommendation
Short term, add the `O_CLOEXEC` flag to the flags used for opening files in the `RawFile` class by default and drop the flag explicitly when needed. This will prevent leaking opened file descriptors into child processes if the node spawns child processes.

6. The `symbol-cli` saves the config file as readable for others

Severity: High
Type: Access Controls
Target: `symbol-cli`

Difficulty: Low
Finding ID: TOB-SYM-006

Description
Importing a profile into `symbol-cli` creates the `symbol-cli.config.json` file. This file is created with the `fs.writeFileSync` function (Figure TOB-SYM-006.1), which by default uses the `0o666` file permissions. This allows all users to read the created config file.

```
  /**
   * Save profiles from JSON.
   * @param {JSON} profiles
   */
  private saveProfiles(profiles: ProfileRecord) {
      fs.writeFileSync(this.filePath, JSON.stringify(profiles), 'utf-8');
  }
```

Figure TOB-SYM-006.1: The `saveProfiles` *function that saves the* `symbol-cli.config.json` *file with permissions that are too broad (symbol-cli/src/respositories/profile.repository.ts#L203-L209).*

Exploit Scenario
Alice imports her account into `symbol-cli`. Eve, who has access to Alice's home directory, copies her `symbol-cli.config.json` file and brute-forces their password. Eve then steals Alice's mosaics.

Recommendation
Short term, explicitly set the permissions of the `symbol-cli.config.json` file to `0o600` in the `symbol-cli` when creating or saving the file. This will prevent users on the same machine from accessing someone else's account if they access the config file and are able to brute-force the password.

Long term, check the permissions of the `symbol-cli.config.json` file when reading it in the `symbol-cli` and warn the user if the file permissions are too broad. This will help prevent situations in which the user accidentally sets permissions that are too broad for the file and never changes them back.

7. Maximum packet size of 4GB may lead to denial-of-service attacks

Severity: Undetermined | Difficulty: High
Type: Denial of Service | Finding ID: TOB-SYM-007
Target: `catapult-server/../PacketPayloadBuilder.h and PacketHandlers.h`

Description

The `PacketPayloadBuilder` and `ServerPacketHandlers` classes' constructors set the `maxPacketDataSize` argument's default value to a maximum 32-bit unsigned integer value, which is more than 4GB (Figures TOB-SYM-007.1-2). Allowing such big packet sizes may lead to denial-of-service attacks in which an attacker would fill the node's memory with huge packets.

```
/// Packet payload builder for creating payloads composed of heterogeneous data.
class PacketPayloadBuilder {
public:
    /// Creates builder for a packet with the specified \a type.
    explicit PacketPayloadBuilder(PacketType type) : PacketPayloadBuilder(type,
std::numeric_limits<uint32_t>::max())
    {}

    /// Creates builder for a packet with the specified \a type and max packet data size (\a
maxPacketDataSize).
    PacketPayloadBuilder(PacketType type, uint32_t maxPacketDataSize)
            : m_maxPacketDataSize(maxPacketDataSize)
            , m_payload(type)
            , m_hasError(false)
    {}
```

Figure TOB-SYM-007.1: Setting the default `maxPacketDataSize` *in* `PacketPayloadBuilder` *class' constructor (catapult-server/src/catapult/ionet/PacketPayloadBuilder.h#L28-L40).*

```
/// Collection of packet handlers where there is at most one handler per packet type.
class ServerPacketHandlers {
// (...)
public:
     /// Creates packet handlers with a max packet data size (\a maxPacketDataSize).
    explicit ServerPacketHandlers(uint32_t maxPacketDataSize =
std::numeric_limits<uint32_t>::max());
```

Figure TOB-SYM-007.2: Setting the default `maxPacketDataSize` *in* `ServerPacketHandlers` *class' constructor (catapult-server/src/catapult/ionet/PacketHandlers.h#L68-L79).*

Recommendation

Short term, set lower maximum default packet sizes in the `PacketPayloadBuilder` and `ServerPacketHandlers` classes' constructors. This will help prevent unwanted denial-of-service scenarios even if an attacker is able to construct big enough payloads to cause them.

8. Lack of overflow checks

Severity: Informational
Type: Data Validation
Target: `catapult-server`

Difficulty: Undetermined
Finding ID: TOB-SYM-008

Description
The `catapult-server` provides wrappers for various value types and simplifies their usage with arithmetic operator overloads. However, many of those operations are not checked against integer overflows and may lead to unexpected behavior if an overflow does occur.

While we did not find any cases where such an overflow could occur, the `BaseValue` operators (Figure TOB-SYM-008.1), `ChainScore` operators (Figure TOB-SYM-008.2), `MosaicEntrySupplyMixin::increaseSupply` function (Figure TOB-SYM-008.3), and others could use the `CheckedAdd` utility function (Figure TOB-SYM-008.4) and either error out or log an error if an overflow occurs.

```
constexpr BaseValue operator+(BaseValue rhs) const {
    return BaseValue(this->unwrap() + rhs.unwrap());
}

/// Subtracts \a rhs from this value and returns a new value.
constexpr BaseValue operator-(BaseValue rhs) const {
    return BaseValue(this->unwrap() - rhs.unwrap());
}
```

Figure TOB-SYM-008.1: The `BaseValue` *operators (catapult-server/src/catapult/utils/BaseValue.h#L109-L116).*

```
/// Adds \a rhs to this chain score.
ChainScore& operator+=(const ChainScore& rhs) {
    m_score += rhs.m_score;
    return *this;
}

/// Subtracts \a rhs from this chain score.
ChainScore& operator-=(const ChainScore& rhs) {
    m_score -= rhs.m_score;
    return *this;
}
```

Figure TOB-SYM-008.2: The `ChainScore` *operators (catapult-server/src/catapult/model/ChainScore.h#L65-L75).*

```
void MosaicEntrySupplyMixin::increaseSupply(Amount delta) {
    m_supply = m_supply + delta;
}
```

Figure TOB-SYM-008.3: The `MosaicEntrySupplyMixin::increaseSupply` *function (catapult-server/plugins/txes/mosaic/src/state/MosaicEntry.cpp#L31-L40).*

```
/// Adds \a delta to \a value if and only if there is no overflow.
template<typename T>
bool CheckedAdd(T& value, T delta) {
    if (value > std::numeric_limits<T>::max() - delta)
        return false;

    value += delta;
    return true;
}
```

Figure TOB-SYM-008.4: The `CheckedAdd` *function (catapult-server/src/catapult/utils/IntegerMath.h#L28-L36).*

Recommendation

Long term, add overflow checks that would either error out the program or log an error to `BaseValue` operators, `ChainScore` operators, the `MosaicEntrySupplyMixin::increaseSupply` function, and others. If those checks are unwanted due to performance issues, enable them only during tests and debug builds. This will help catch bugs earlier and prevent unwanted overflows that could cause financial risks.

9. The `boost::filesystem::create_directory` defaults to `0777` permissions

Severity: Low
Type: Access Controls
Target: `catapult-server`

Difficulty: High
Finding ID: TOB-SYM-009

Description
The `boost::filesystem::create_directory` function used across the `catapult-server` codebase creates a directory through the `mkdir` syscall passing in `0777` permissions. While this is usually further limited by the default `umask` setting set on a given Linux system, this might allow incorrect permissions for the created directories, and an attacker who has an account on the same system would be able to create files and directories in there.

Exploit Scenario
Alice sets up her machine with a `umask` setting that's too broad and hosts a Symbol node. Eve, who has an account on the same machine as Alice, accesses directories created by Alice's Symbol node and alters the behavior of that node.

Recommendation
Short term, don't use `boost::filesystem::create_directory` or `create_directories` functions for creating directories in `catapult-server`. Instead, use a method that allows you to set permissions of the created directories explicitly and appropriately. This will prevent insecure configurations that may lead to unexpected behavior and allow access from other system users in less securely configured systems.

10. Potential padding oracle attack in `AesCbcDecrypt`

Severity: Undetermined | Difficulty: Undetermined
Type: Cryptography | Finding ID: TOB-SYM-010
Target: `catapult-server/src/catapult/crypto/AesCbcDecrypt.cpp`

Description
The `catapult-server` uses an elliptic curve integrated encryption scheme (ECIES) in its `crypto` library. Specifically, `Ed25519` is used to derive a shared secret, which acts as an input into a key derivation function, which is then used as a symmetric key for AES-CBC. However, AES-CBC is paired with PKCS #7 padding and is not authenticated with a message authentication code (MAC), which could make a padding oracle attack possible.

```
bool TryAesCbcDecrypt(const SharedKey& key, const RawBuffer& input, std::vector<uint8_t>&
output) {
	AesInitializationVector initializationVector;
	if (input.Size < initializationVector.size())
		return false;

	output.resize(input.Size - initializationVector.size());
	if (0 != output.size() % Aes_Pkcs7_Padding_Size)
		return false;

	std::memcpy(initializationVector.data(), input.pData, initializationVector.size());

	auto outputSize = static_cast<int>(output.size());

	OpensslCipherContext cipherContext;
	cipherContext.dispatch(EVP_DecryptInit_ex, EVP_aes_256_cbc(), nullptr, key.data(),
initializationVector.data());
	cipherContext.dispatch(EVP_DecryptUpdate, output.data(), &outputSize, input.pData +
initializationVector.size(), outputSize);
	if (!cipherContext.tryDispatch(EVP_DecryptFinal_ex, output.data() + outputSize,
&outputSize))
		return false;

	// drop PKCS#7 padding
	if (!DropPadding(output))
		return false;

	return true;
}
```

Figure TOB-SYM-010.1: AES-CBC used without a MAC (catapult-server/src/catapult/crypto/AesCbcDecrypt.cpp#L58-L82).

The `catapult-server` uses ECIES inside of the `registerServices` function for its `HarvestingServiceRegistrar`. If an attacker were able to repeatedly alter and transmit ciphertexts, this function could be turned into a padding oracle, allowing an attacker to recover the underlying plaintexts.

It is unclear exactly how much access an adversary would have to the ciphertexts, and thus it's unclear if this issue is currently exploitable.

Exploit Scenario
An attacker, Eve, is able to alter and send multiple ciphertexts to be decrypted inside of the `registerServices` function. Eve observes enough decryption failures to learn the underlying plaintexts of those ciphertexts.

Recommendation
Short term, include an HMAC alongside the AES-CBC encryption to ensure that ciphertexts are not altered by an adversary and to prevent potential padding oracle attacks.

Long term, consider clearing plaintext buffers upon any authentication or decryption failure to prevent invalid plaintexts from being used.

11. Incorrect `ReceiptType` in `catapult-rest`

Severity: Low | Difficulty: Undetermined
Type: Undefined Behavior | Finding ID: TOB-SYM-011
Target: `catapult-rest/catapult-sdk/src/plugins/receipts.js`

Description

The `ReceiptType` defined in `catapult-rest` contains a duplicate value (Figure TOB-SYM-011.1) which is inconsistent with the `ReceiptType` defined in `catapult-server` (Figure TOB-SYM-011.2).

```
const ReceiptType = {
        1: 'receipts.balanceTransfer',
        2: 'receipts.balanceChange',
        3: 'receipts.balanceChange',
        4: 'receipts.artifactExpiry',
        5: 'receipts.inflation'
};
```

Figure TOB-SYM-011.1: `ReceiptType` *in* `catapult-rest` *(catapult-rest/catapult-sdk/src/plugins/receipts.js#L24-L30).*

```
/// Enumeration of basic receipt types.
/// \note BasicReceiptType is used as highest nibble of receipt type.
enum class BasicReceiptType : uint8_t {
        /// Some other receipt type.
        Other = 0x0,

        /// Balance transfer.
        BalanceTransfer = 0x1,

        /// Balance credit.
        BalanceCredit = 0x2,

        /// Balance debit.
        BalanceDebit = 0x3,

        /// Artifact expiry receipt.
        ArtifactExpiry = 0x4,

        /// Inflation.
        Inflation = 0x5,

        /// Aggregate receipt.
        Aggregate = 0xE,

        /// Alias resolution.
        AliasResolution = 0xF
};
```

Figure TOB-SYM-011.2: `ReceiptType` *in* `catapult-server` *(catapult-server/src/catapult/model/ReceiptType.h#L29-L55).*

Recommendation

Short term, fix the duplicate `ReceiptType` value in the `catapult-rest` codebase.

Trail of Bits discussed this issue with the Symbol team and agreed that this issue does not require the proposed change. The `BalanceCredit` and `BalanceDebit` receipt types are included in the `balanceChange` type and don't need separate representations. Due to that, we only recommend documenting this in the code.

A. Vulnerability Classifications

Vulnerability Classes	
Class	**Description**
Access Controls	Related to authorization of users and assessment of rights
Auditing and Logging	Related to auditing of actions or logging of problems
Authentication	Related to the identification of users
Configuration	Related to security configurations of servers, devices or software
Cryptography	Related to protecting the privacy or integrity of data
Data Exposure	Related to unintended exposure of sensitive information
Data Validation	Related to improper reliance on the structure or values of data
Denial of Service	Related to causing system failure
Error Reporting	Related to the reporting of error conditions in a secure fashion
Patching	Related to keeping software up to date
Session Management	Related to the identification of authenticated users
Timing	Related to race conditions, locking or order of operations
Undefined Behavior	Related to undefined behavior triggered by the program

Severity Categories	
Severity	**Description**
Informational	The issue does not pose an immediate risk, but is relevant to security best practices or Defense in Depth
Undetermined	The extent of the risk was not determined during this engagement
Low	The risk is relatively small or is not a risk the customer has indicated is important
Medium	Individual user's information is at risk, exploitation would be bad for client's reputation, moderate financial impact, possible legal implications for client

High	Large numbers of users, very bad for client's reputation, or serious legal or financial implications

Difficulty Levels	
Difficulty	**Description**
Undetermined	The difficulty of exploit was not determined during this engagement
Low	Commonly exploited, public tools exist or can be scripted that exploit this flaw
Medium	Attackers must write an exploit, or need an in-depth knowledge of a complex system
High	The attacker must have privileged insider access to the system, may need to know extremely complex technical details or must discover other weaknesses in order to exploit this issue

B. Compiler Mitigations

NX makes the data sections (including the stack and heap) of the program non-executable. This makes it more difficult for an attacker to execute shellcode. Attackers normally use return-oriented programming (ROP) to bypass NX. Forcing attackers to use ROP makes exploits less reliable across different builds of a program. This mitigation is enabled by default.

Stack canaries (also known as stack cookies) make buffer overflow vulnerabilities more difficult to exploit. A stack canary is a global, randomly-generated value that is copied to the stack between the stack variables and stack metadata in a function's prologue. When a function returns, the canary on the stack is checked against the global value. The program exits if there's a mismatch. This makes it more difficult for an exploit to overwrite the return address on the stack. Depending on the circumstances, attackers may bypass this mitigation by leaking the cookie with a separate information leak vulnerability or by brute-forcing the cookie byte-by-byte.

ASLR (address-space layout randomization) randomizes where each section of the program is placed in memory. This makes it more difficult for an attacker to write reliable exploits, primarily by making it more difficult to jump to ROP gadgets. ASLR requires cooperation from both the system and the compiler. In order to support ASLR fully, a program must be compiled as a position-independent executable (PIE). Most of the Linux distributions have ASLR enabled. This can be checked by reading the value stored in the `/proc/sys/kernel/randomize_va_space` file: 0 means that ASLR is disabled, 1 means it is partially enabled (fewer bits of the addresses are randomized), and 2 means it is fully enabled. This file is writable, and an admin can disable or enable this mitigation.
ASLR may be bypassed if an attacker has an information leak in the program.

RELRO (relocations read-only) is a mitigation technique to harden the data sections of an ELF process. It has three modes of operation: disabled, partial, and full. When a program uses a function from a dynamically loaded library, this function address is stored in the GOT.PLT section (Global Offset Table for Procedure Linkage Table). When RELRO is disabled, the function addresses in GOT.PLT point to a dynamic resolver function that resolves the given function address when it is called for the first time. In this case, the memory where the address is stored is both readable and writable. Because of that, an attacker who has control over the process control flow can change the entry of a given function in GOT.PLT to point to any other executable address. For example, they can change the `puts` function's GOT.PLT entry to point to a `system` function. Then, if the program calls `puts("bin/sh")`, a `system("/bin/sh")` would be called instead. When RELRO is fully enabled, the dynamic resolver resolves all of the addresses on program startup and changes the permissions of data sections (and therefore GOT.PLT) to read-only.

FORTIFY_SOURCE is a `glibc`-specific feature that enables a series of mitigations primarily aimed at preventing buffer overflows. With a `FORTIFY_SOURCE` level of 1, `glibc` will add compile-time warnings when potentially unsafe calls to common `libc` functions (e.g., `memcpy` and `strcpy`) are made. With a `FORTIFY_SOURCE` level of 2, `glibc` will add more stringent run-time checks to these functions. Additionally, `glibc` will enable a number of lesser-known mitigations. For example, it will disallow the use of a `%n` format specifier in format strings that aren't located in read-only memory pages. This prevents overwriting data (and gaining code execution) with format string vulnerabilities.

Stack clash protection mitigates a "stack clash vulnerability," where a program's stack memory region grows so much that it overlaps with another memory region. This bug makes the program confuse two different memory addresses (stack and, e.g., heap) so that some of their data overlap, leading to denial of service or control flow hijacking. The stack clash protection mitigation works by adding explicit memory probing to functions that allocate a lot of stack memory, so that the function's stack allocation will never make the stack pointer jump over the stack memory guard page, which is located before the stack.

Control Flow Integrity (CFI) (https://clang.llvm.org/docs/ControlFlowIntegrity.html) mitigates various control-flow hijack attempts and so makes it harder to exploit vulnerabilities like use-after-free or use exploitation techniques such as ROP. This mitigation is currently implemented only in Clang/LLVM.

SafeStack (https://clang.llvm.org/docs/SafeStack.html) makes it harder to exploit stack-based buffer overflows as it separates the program stack into the safe stack (which holds saved registers and return addresses) and unsafe stack (which stores everything else). This mitigation is Clang/LLVM only.

C. Fuzzing `catapult-server`

During the assessment, Trail of Bits attempted to integrate `libFuzzer`, an in-process, coverage-guided, evolutionary fuzzing engine integrated into Clang. However, due to build issues, most likely related to compiling some libraries with GCC and the final executable with Clang (Figure C.1), DeepState was integrated instead, and its dumb fuzzing engine (the `--fuzz` flag) was used. DeepState provides an interface for symbolic execution and fuzzing engines through a unit–test-like structure.

Figures C.2-5 show the developed `cmake` files and the fuzzing harness that performs a round-robin serialization and deserialization of a `TransactionInfo` structure. The harness needs to be improved to account for finding more paths in the program; currently, with dumb fuzzing, it usually don't pass through the loop that extracts addresses and crashes there instead due to insufficient buffer length. Figure C.6 shows a traceback from such a crash. Alternatively, instead of refactoring the harness to find more paths, AFL could be used to find new paths and more interesting crashes as it de-duplicates non-unique crashes.

```
root@7dbcdd073699:/host/build# ninja tests.fuzz.initial
[1/1] Linking CXX executable bin/tests.fuzz.initial
FAILED: bin/tests.fuzz.initial
: && /usr/bin/clang++-9 -stdlib=libc++             -Weverything          -Werror
       -Wno-c++98-compat            -Wno-c++98-compat-pedantic
-Wno-disabled-macro-expansion               -Wno-padded           -Wno-switch-enum
       -Wno-weak-vtables -fvisibility=hidden -fsanitize=fuzzer -Wl,--disable-new-dtags
src/catapult/version/nix/CMakeFiles/catapult.version.nix.dir/what_version.cpp.o -o
bin/tests.fuzz.initial
-Wl,-rpath,"\$ORIGIN:\$ORIGIN/../deps:\$ORIGIN/../lib:/root/boost-build-1.71.0/lib"
/root/boost-build-1.71.0/lib/libboost_atomic.so
/root/boost-build-1.71.0/lib/libboost_system.so
/root/boost-build-1.71.0/lib/libboost_date_time.so
/root/boost-build-1.71.0/lib/libboost_regex.so
/root/boost-build-1.71.0/lib/libboost_timer.so
/root/boost-build-1.71.0/lib/libboost_chrono.so
/root/boost-build-1.71.0/lib/libboost_log.so
/root/boost-build-1.71.0/lib/libboost_thread.so  -lpthread
/root/boost-build-1.71.0/lib/libboost_filesystem.so
/root/boost-build-1.71.0/lib/libboost_program_options.so
/root/boost-build-1.71.0/lib/libboost_log_setup.so && :
/usr/bin/ld:
/usr/lib/llvm-9/lib/clang/9.0.0/lib/linux/libclang_rt.fuzzer-x86_64.a(FuzzerDataFlowTrace.cp
p.o): undefined reference to symbol
'_ZNSt14basic_ifstreamIcSt11char_traitsIcEEC1ERKNSt7__cxx1112basic_stringIcS1_SaIcEEESt13_Io
s_Openmode@@GLIBCXX_3.4.21'
//usr/lib/x86_64-linux-gnu/libstdc++.so.6: error adding symbols: DSO missing from command
line
clang: error: linker command failed with exit code 1 (use -v to see invocation)
ninja: build stopped: subcommand failed.
```

Figure C.1: Build issues when compiling an example `libfuzzer` *fuzzing harness.*

```
#include <cstdint>
#include <deepstate/DeepState.hpp>

#include "catapult/io/TransactionInfoSerializer.h"
#include "tests/test/core/mocks/MockMemoryStream.h"
#include "catapult/model/Address.h"

using namespace deepstate;
using namespace catapult;

TEST(TInfo, RoundTripSerialization) {
    constexpr const size_t SIZE = 2*sizeof(model::TransactionInfo);
    uint8_t* raw_data = reinterpret_cast<uint8_t*>(DeepState_Malloc(SIZE));
    // Prepare stream for serialization
    std::vector<uint8_t> data{raw_data, raw_data+SIZE};

    mocks::MockMemoryStream stream(data);

    // Deserialize stream into TransactionInfo
    model::TransactionInfo tinfo;
    ReadTransactionInfo(stream, tinfo);

    // Prepare stream for serialization output
    std::vector<uint8_t> buffer;
    mocks::MockMemoryStream stream2(buffer);

    // Serialize tinfo into bytes
    WriteTransactionInfo(tinfo, stream2);

    ASSERT_EQ(buffer.size(), SIZE) << "buffer size is " << buffer.size() << " expected: " <<
SIZE;
    ASSERT_EQ(memcmp(&data[0], &buffer[0], SIZE), 0) << "Data not same";
}
```

Figure C.2: An example fuzzing harness (`tests/deepstate/initial/InitialTests.cpp`*).*

```
function(catapult_deepstate_executable TARGET_NAME)
        find_package(GTest REQUIRED)
        catapult_executable(${TARGET_NAME} ${ARGN})
        add_test(NAME ${TARGET_NAME} WORKING_DIRECTORY ${CMAKE_BINARY_DIR} COMMAND
${TARGET_NAME})
endfunction()

function(catapult_deepstate_executable_target TARGET_NAME TEST_DEPENDENCY_NAME)
        catapult_deepstate_executable(${TARGET_NAME} ${ARGN})

        catapult_set_test_compiler_options()
        # Prevent compilation error (see https://github.com/trailofbits/deepstate/issues/357)
        set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -Wno-error" PARENT_SCOPE)

# Uncomment for compiling with DeepState and libfuzzer; consider also -fsanitize=address
#       target_link_libraries(${TARGET_NAME} -fsanitize=fuzzer -ldeepstate -ldeepstate_LF
${TEST_DEPENDENCY_NAME})
        target_link_libraries(${TARGET_NAME} -ldeepstate ${TEST_DEPENDENCY_NAME})
        catapult_target(${TARGET_NAME})
endfunction()
```

Figure C.3: Addition to `CMakeGlobalSettings.cmake` *to use DeepState.*

```
cmake_minimum_required(VERSION 3.14)
project(tests.deepstate)

add_subdirectory(initial)
```

Figure C.4: The `tests/deepstate/CMakeLists.txt` *file.*

```
cmake_minimum_required(VERSION 3.14)

catapult_deepstate_executable_target(tests.deepstate.initial tests.catapult.test.core)
```

Figure C.5: The `tests/deepstate/initial/CMakeLists.txt` *file.*

```
(gdb) bt
#0  boost::log::v2_mt_posix::basic_record_ostream<char>::get_record (this=0x669fd8)
   at /root/boost-build-1.71.0/include/boost/log/sources/record_ostream.hpp:161
#1  0x000000000040fc94 in
boost::log::v2_mt_posix::aux::record_pump<catapult::utils::log::catapult_logger>::~record_pu
mp (this=0x7fffffffde10)
   at /root/boost-build-1.71.0/include/boost/log/sources/record_ostream.hpp:529
#2  0x00000000004102e9 in catapult::io::BufferInputStreamAdapter<std::__1::vector<unsigned
char, std::__1::allocator<unsigned char> > >::read (
   this=0x7fffffffe2c8, buffer=...) at ../src/catapult/io/BufferInputStreamAdapter.h:59
#3  0x0000000000419fa3 in catapult::io::ReadTransactionInfo (inputStream=...,
transactionInfo=...)
   at ../src/catapult/io/TransactionInfoSerializer.cpp:51
#4  0x000000000040590d in DeepState_Test_TInfo_RoundTripSerialization () at
../tests/deepstate/initial/InitialTests.cpp:28
#5  0x0000000000405879 in DeepState_Run_TInfo_RoundTripSerialization () at
../tests/deepstate/initial/InitialTests.cpp:12
#6  0x000000000040bae3 in DeepState_RunTestNoFork.isra.0 () at
../src/catapult/io/BufferInputStreamAdapter.h:31
#7  0x000000000040bc1c in DeepState_ForkAndRunTest () at
../src/catapult/io/BufferInputStreamAdapter.h:31
#8  0x000000000040bd71 in DeepState_RunSavedTestCase () at
../src/catapult/io/BufferInputStreamAdapter.h:31
#9  0x00000000004053e6 in main ()
```

Figure C.5: Traceback from a crash from running the fuzzing harness from Figure C.2.

D. Previous security testing report fixes

As a secondary goal of our engagement, Trail of Bits was asked to assess remediations in response to the issues highlighted in a report completed by a previous security testing agency.

Due to time constraints, we were unable to test the effectiveness of all the fixes applied by NEM Group. Here are the fixes we had sufficient time to review:

- **#2: Adhoc key derivation scheme allows for the selection of weak keys:** This issue is resolved since HKDF key derivation is now used in `SharedKey.cpp`.
- **#4: Bulk signature verification uses insufficiently random coefficients:** This issue should now be resolved because bulk verification no longer happens at the given target.
- **#5: Custom implementation of Ed25519-SHA3-512:** This is no longer relevant since the Ed25519 implementation included seems to be a standardized one with no custom SHA3-512 implementation. There are notes indicating the move to Ed25519 with standard SHA512 derivation.
- **#8: Protocol content is not confidential:** This concern should be remediated by this commit, which implemented TLS 1.3. An inspection of catapult-service-bootstrap traffic (2dff3fb) does not reveal plaintext data.

Additionally, Trail of Bits engineers discussed the design of a finality protocol with NEM Group. We did not find any issues in the agreed-upon finality proposal, and the finality design is now in Release Code, which should resolve the following issues:

- **#20: Block finality is only (seemingly) guaranteed by the maximum rollback limit:** Concerns regarding this issue should be inherently remediated with the finality roadmap specified by Trail of Bits engineers.
- **#22: Nothing inhibits harvesting on multiple chains (Nothing at Stake):** A finality protocol that maintains a single, finalized chain is being designed and implemented, and does not allow parties to vote for multiple chains in each round. Additionally, they can potentially penalize parties if they do give multiple votes.

E. Fix Log

NEM Group addressed issues TOB-SYM-001 through TOB-SYM-011 in their codebase as a result of our assessment. Each of the fixes was verified by the audit team except TOB-SYM-008, which was partially verified.

ID	Title	Severity	Status
01	Missing compiler mitigations	Low	Fixed
02	Undefined behavior dereferencing `std::list.back()` on an empty container	Undetermined	Fixed
03	Current `ConfigurationBags` verification may lead to bugs	Informational	Fixed
04	High-entropy RNG does not guarantee high entropy	Medium	Fixed
05	Use `O_CLOEXEC` flag by default when opening files on Linux	Informational	Fixed
06	The `symbol-cli` saves the config file as readable for others	High	Fixed
07	Maximum packet size of 4GB may lead to denial-of-service attacks	Undetermined	Fixed
08	Lack of overflow checks	Informational	Undetermined
09	The `boost::filesystem::create_directory` defaults to `0777` permissions	Low	Fixed
10	Potential padding oracle attack in `AesCbcDecrypt`	Undetermined	Fixed
11	Incorrect `ReceiptType` in `catapult-rest`	Low	Unnecessary

Detailed Fix Log

Finding 1: Missing compiler mitigations
Fixed. The appropriate compiler mitigations have been added to CMakeGlobalSettings.cmake for release builds.

Finding 2: Undefined behavior dereferencing `std::list.back()` on an empty container
Fixed. NEM Group has added a check to verify if the bucket is empty prior to dereferencing via the `back()` call. If the bucket is empty, a new item will be added, ensuring it is non-empty before dereferencing.

Finding 3: Current `ConfigurationBags` verification may lead to bugs
Fixed. An explicit size check was added to ensure the expected size is strictly equal to the provided bag size.

Finding 4: High-entropy RNG does not guarantee high entropy
Fixed. The introduction of the SecureRandomGenerator alleviates this issue as it uses OpenSSL's random provider under the hood.

Finding 5: Use `O_CLOEXEC` flag by default when opening files on Linux
Fixed. The `O_CLOEXEC` flag has been added to a definition, which is used during the `open()` operation.

Finding 6: The `symbol-cli` saves the config file as readable for others
Fixed. The affected file has been updated to set appropriate permissions after writing the file. This includes appropriate error handling if the permissions cannot be set, i.e., if the file is removed between writing and setting permissions.

Finding 7: Maximum packet size of 4GB may lead to denial-of-service attacks
Fixed. The maximum packet size has been changed to 100MB. This has not been evaluated in practice but it should reduce DoS attack severity.

Finding 8: Lack of overflow checks
Undetermined. Appropriate overflow checks have been implemented for the second and third cases within the issue descriptio;, however, the first case that deals with the `utils::BaseValue` class was repurposed to support many value types. Our fix review has not determined whether every use of this object is secure.

Finding 9: The `boost::filesystem::create_directory` defaults to `0777` permissions
Fixed. This has been fixed by implementing a `CreateDirectory` wrapper function that calls the `mkdir` function explicitly passing the `0700` permissions.

Finding 10: Potential padding oracle attack in `AesCbcDecrypt`
Fixed. `AesCbcDecrypt.cpp` has been replaced with `AesDecrypt.cpp`, which makes use of AES-GCM, which inherently includes a message authentication code (MAC).

Finding 11: Incorrect `ReceiptType` in `catapult-rest`
Not fixed, because repair of the issue is unnecessary. These naming conventions may be ambiguous, but do not introduce a security vulnerability. Please refer to the recommendations section of the issue.

F. Amendments to Voting Key Structure

Between October 28 and October 29th, 2020, Trail of Bits reviewed amendments made to the voting key structure. Engineers performed this review working from commit hash b7b780c. At a high level, changes included a transition from the three-level voting key structure to a two-level voting key structure.

These changes align with changes to the specification derived from discussions between NEM Group and Trail of Bits (as mentioned briefly in Appendix D). The changes largely simplified this portion of the finality process by removing the use of on-the-fly keys and instead opting to sign everything in a given epoch with a specific epoch-tied key. Changes to this process included removing `batchId` and `dilution` variables, and instead using the `keyId` derived from epoch as an alternative. As a result, the "top signature" level was removed, and the focus shifted to verification in root and bottom signatures.

Our review confirmed the implementation matched the agreed-upon specification and ensured general code correctness throughout. Trail of Bits did not identify any issues during this amendment review.

G. Additional Symbol changes review

Between November 23 and November 25th, 2020, Trail of Bits reviewed changes made to the Symbol repositories. An engineer performed this review working from the versions specified by NEMtech, listed in the table below. A non-exhaustive list of updates reviewed included block changes related to importance, as well as the addition of various endpoints and epoch adjustment code. Additionally, we ran a set of Symbol nodes based on the symbol-bootstrap setup with Docker in order to confirm that nodes properly validate other nodes they connect to.

Repository	Changes reviewed from	Audited version
catapult-server (main branch)	c6f2ffd3a801	8add85f1bd7
catapult-rest (dev branch)	5c6ec9586a5	ed20bcc7cc4
symbol-sdk-typescript-javascript (dev branch)	8916ad9486	2764846fe9f
symbol-sdk-java (dev branch)	2274c70718	60c49f2118c

During the review, we identified five bugs or code quality issues. We describe those findings below, along with recommendations on how to resolve them.

1. The `endpoint.Host` and `metadata.Name` are truncated to a length of 255 (through the `GetPackedSize` function and a later `memcpy`), during `Node` objects serialization in the `PackNode` function. In practice, this issue does not expose much risk, as a node would fail to connect to a truncated (unexpected) address due to TLS peer verification. However, because this issue means that endpoint hosts longer than 255 bytes are broken, we would recommend disallowing setting hostnames or names longer than 255 bytes and validating that the appropriate sizes don't exceed the limits in the `UnpackNode` function.

2. The parseServerDuration implementation differs between `symbol-sdk-java` and `symbol-sdk-typescript-javascript`. It seems that the TS/JS SDK implementation is missing a loop, so it can't parse complex durations such as "10h:10m", which are supported in the Java SDK. Additionally, the Java SDK implementation allows for "10h:10h" formats, which are most likely unexpected. We recommend:

- Fixing the `parseServerDuration` function in the TS/JS SDK implementation so it properly handles complex duration strings (such as "10h:10m").
- Fixing both implementations to disallow incorrect or unexpected inputs such as "10h:10h" or "10h:10m:10h".
- Making the tests between the two implementations consistent, so they check against similar cases.

- Changing the Java SDK function name from `parserServerDuration` to `parseServerDuration`, so that it matches the TS/JS naming.

3. Move the block type calculation to a separate function instead of performing the same calculation in multiple places. The following code paths calculate the block type:

- catapult-server/plugins/coresystem/src/validators/BlockTypeValidator.cpp#L33
- catapult-server/extensions/harvesting/src/Harvester.cpp#L77
- catapult-server/tests/int/node/stress/test/BlockChainBuilder.cpp#L169

4. While out of scope, the `TransactionStatusEnum` in the `symbol-openapi` project is missing some error codes. For example, the `Failure_Chain_*` errors are missing. We recommend investigating this issue and considering adding tests to validate this data or generating it from the source.

5. There is a typo "`Coutn`", instead of "`Count`" in Java SDK's `FinalizationStage` enum.

H. Fix log for issues from Appendix G

NEM Group addressed issues listed in Appendix G in their codebase as a result of our assessment. Each of the fixes was verified by an audit team member.

1. The `endpoint.Host` and `metadata.Name` are truncated to a length of 255 (through the `GetPackedSize` function and a later `memcpy`) during `Node` objects serialization in the `PackNode` function.
Fixed. The length of `endpoint.Host` and `metadata.Name` is now validated when a Node object is created, ensuring the fields will not get truncated during serialization.

2. The parseServerDuration implementation differs between `symbol-sdk-java` and `symbol-sdk-typescript-javascript`.
Fixed. The implementations (`symbol-sdk-java`, `symbol-sdk-typescript-javascript`) should now accept the same duration formats.

3. Move the block type calculation to a separate function instead of performing the same calculation in multiple places.
Fixed. The code has been refactored and a `CalculateBlockTypeFromHeight` function has been introduced.

4. While out of scope, the `TransactionStatusEnum` in the `symbol-openapi` project is missing some error codes.
Fixed. The error codes were added in `symbol-openapi#250`.

5. There is a typo "`Coutn`", instead of "`Count`" in Java SDK's `FinalizationStage` enum.
Fixed. The typo was fixed in a recent commit.