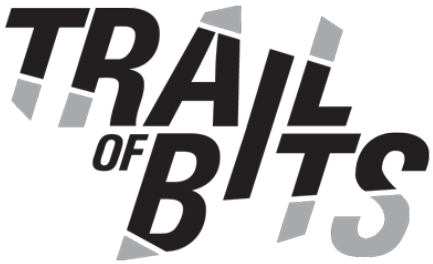

CasperLabs Highway Protocol

Security Assessment

December 1, 2020

Prepared For:
Andreas Fackler | *CasperLabs*
andreas@casperlabs.io

Medha Parlikar | *CasperLabs*
medha@casperlabs.io

Prepared By:
Fredrik Dahlgren | *Trail of Bits*
fredrik.dahlgren@trailofbits.com

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

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

Will Song | *Trail of Bits*
will.song@trailofbits.com

Changelog:

December 1, 2020:	Initial draft
January 8, 2021:	Finalized draft with fix log

Executive Summary

From November 16 to December 1, 2020, Trail of Bits performed an assessment of CasperLabs' Highway consensus protocol. Trail of Bits performed this assessment working from commit hash 40a573e of the `casper-node` repository.

Trail of Bits engineers used the first week to familiarize themselves with the provided protocol paper for Highway, deriving security properties to verify within the codebase. Preliminary manual review was conducted against the provided implementation in search of issues regarding improper authentication, insufficient data validation, denial of service, and undefined behavior. A non-exhaustive list of initial concerns included verification of signatures against creators for wire units, validating endorsement logic, review of leader selection, and improperly handled `unwrap` calls that may lead to panics. This resulted in one preliminary finding (TOB-CLHW-001) and several documentation recommendations detailed within Appendix B.

In the final week, Trail of Bits validated remaining concerns regarding malicious validator banning/slashing mechanisms, state transitions, resource exhaustion attacks, and triaging concerns from the previous week. This resulted in two additional findings (TOB-CLHW-002, TOB-CLHW-003), documentation recommendations, and a code quality recommendation listed within Appendix C.

Overall, Trail of Bits' investigation into the Highway protocol did not produce high severity results and showed proper use of security hygiene. Moving forward, we recommend improving the protocol paper for clarity and employing the use of fuzz tests to discover any potential deep-rooted data validation issues in the implementation. Additionally, we recommend testing denial of service attacks against the network and individual nodes in practice, as time did not permit us to conduct these tests in depth during the course of the assessment.

Project Dashboard

Application Summary

Name	Highway Protocol
Version	40a573e
Type	Rust, PDF
Platforms	UNIX

Engagement Summary

Dates	November 16 - December 1, 2020
Method	Whitebox
Consultants Engaged	4
Level of Effort	4 person-weeks

Vulnerability Summary

Total Low-Severity Issues	2	■ ■
Total Informational-Severity Issues	1	■
Total	3	

Category Breakdown

Data Validation	1	■
Denial of Service	2	■ ■
Total	3	

Engagement Goals

The engagement was scoped to provide a security assessment of the Highway protocol paper and corresponding Rust implementation within the `casper-node` repository.

Specifically, we sought to answer the following questions:

- Will misbehaving validators be punished?
- Is the system prone to a resource exhaustion attack?
- Are units appropriately validated before being appended to state?
- Are signed wire units appropriately signed/verified?
- Can an attacker game the system such that they are selected as the next leader?
- Is LNC validation appropriately implemented? Is the buffer mechanism for LNC validators appropriate?

Coverage

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

- Differential review of the protocol paper against the implementation revealed some inconsistencies (TOB-CLHW-001, Appendix B).
- Analysis of the signature scheme used for wire units ensured the signature was validated against the creator's keypair.
- Partial coverage of leader selection did not yield any immediate concern. There was emphasis on ensuring that an attacker could not control leader selection. We considered if an attacker could control or influence the weights map or random seed, they may have been able to influence leader selection, however we did not determine an avenue for such an attack.
- Review of the data validation performed on units prior to any state changes generally yielded positive results, although we made recommendations regarding the handling of invalid vertices (TOB-CLHW-002).
- General review of code correctness did not yield any findings. Initial concerns involved suspected potential panics which could be triggered by failed calls prior to an `unwrap` operation.
- Employment of static analysis tooling did not reveal any concerns.
- A review of rate limiting capabilities with respect to units led to a broad concern for denial of service attacks via non-unit type messages (TOB-CLHW-003).

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

❒ **Implement LNC validation** as specified within the Highway protocol paper. TOB-CLHW-001

❒ **Consider rate limiting or otherwise penalizing the sender and disconnecting from them in the event of invalid vertices being sent.** TOB-CLHW-002

❒ **Consider flagging users who have sent invalid messages to be restricted at the receiving socket** before pushing subsequent messages into the queue. TOB-CLWHW-003

Long Term

❒ **Review all code locations regarding failed data validation to deduce if the sender should be rate limited or otherwise penalized**. This will disincentivize peers from further malicious acts.

Findings Summary

#	Title	Type	Severity
1	LNC validation not implemented	Data Validation	Informational
2	Peers are not punished for invalid vertices	Denial of Service	Low
3	Insufficient rate limiting mechanism	Denial of Service	Low

1. LNC validation not implemented

Severity: Informational | Difficulty: Undetermined
Type: Data Validation | Finding ID: TOB-CLHW-001
Target: `node/src/components/consensus/highway_core/state.rs`

Description
The current iteration of the Highway protocol does not support LNC validation as it is defined in the protocol paper.

Verification of LNC is important for units in order to ensure that the amount of equivocation votes contained in any message downset is limited. This in turn helps protect against malicious behavior such as equivocation bombs.

The impact of this finding has not been determined during the course of this assessment. As such, there may be alternate mitigations in place to prevent any such attacks.

Recommendation
Implement LNC validation as specified within the Highway protocol paper.

2. Peers are not punished for sending invalid vertices

Severity: Low | Difficulty: Medium
Type: Denial of Service | Finding ID: TOB-CLHW-002
Target: `node/src/components/consensus/era_supervisor.rs`

Description
The Highway protocol considers messages within the network as vertices within a graph. When processing incoming vertices with the `add_vertices` function, validation is performed upon initial receipt and after initial pre-validation. Unfortunately, if validation fails, there is no meaningful repercussion for the sender which sent the invalid message. Similarly, failed pre-validation does not seem to penalize the sender either, even by way of rate limiting.

In the event of an attacker sending malicious messages, it appears that the receiving node will attempt to validate each malicious message, providing a potential avenue for denial of service attacks:

```
                    match self.highway.validate_vertex(pvv) {
[...]
                        Err((pvv, err)) => {
                            info!(?pvv, ?err, "invalid vertex");
                            // drop all the vertices that might have depended on this one
                            self.drop_dependent_vertices(vec![pvv.inner().id()]);
                            // TODO: Disconnect from senders!
                        }
```

Figure 2.1: node/src/components/consensus/protocols/highway.rs#L259-L285

In the figure above, we note that an invalid vertex does not result in any penalization and that a disconnect should occur in a future iteration of this implementation. However, it may be more prudent to consider flagging and rate limiting the sender of this message.

Rate limiting occurs within `state.rs` 's `validate_unit`, but only applies to unit-type vertices. The severity for this finding is low, as this type of attack would not be propagated throughout the network and any spam attacks would only affect a single node. Disconnecting from a node will rotate the connection to another peer which is less likely to be part of any coordinated attack effort.

Exploit Scenario
Bob is a Casper node operator. Alice, an attacker, notes that the Highway protocol implementation does not rate limit or penalize senders of invalid vertices. Alice decides to send a large number of invalid messages to Bob. As a result, Bob's node is forced to perform validation on each, logging messages for each invalid vertex, leading to resource exhaustion.

Recommendation
Short term, disconnect from the sender and consider rate limiting them for sending invalid messages.

Long term, review all code locations regarding failed data validation to deduce if the sender should be rate limited or otherwise penalized. This will disincentivize peers from further malicious acts.

3. Insufficient rate limiting mechanism

Severity: Low | Difficulty: Medium
Type: Denial of Service | Finding ID: TOB-CLHW-003
Target: `node/src/components/consensus/era_supervisor.rs`

Description
Currently, the rate limiting mechanism used to prevent spam targets Unit-type vertices during validation stages. However, it does not account for other vertex types.

Additionally, it may be more valuable to rate limit messages closer to the socket. This would prevent a user from spamming the socket with other message types, which may be effective due to the fact that the incoming message queue could be flooded and cause an out-of-memory exception.

The severity for this finding is low, as this type of attack would not be propagated throughout the network and any spam attacks would only affect a single node. Disconnecting from a node upon detection of an invalid message, as initially planned, will rotate the connection to another peer which is less likely to be part of any coordinated attack effort.

Exploit Scenario
Bob is a Casper node operator. Alice, an attacker, notes that the Highway protocol implementation does not rate limit messages besides units per round. Alice decides to send a large amount of invalid non-unit messages to Bob. As a result, Bob's node is forced to process each, leading to a resource exhaustion.

Recommendation
Short term, consider flagging users who have sent invalid messages to be restricted at the receiving socket before pushing subsequent messages into the queue.

Long term, review all code locations regarding failed data validation to deduce if the sender should be rate limited or otherwise penalized. This will disincentivize peers from further malicious acts.

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. Recommendations for Documentation

CasperLabs provided extensive documentation for their overall system, including a protocol paper detailing the Highway consensus protocol. Overall, Trail of Bits found the documentation helpful in providing background information for the technology throughout the course of the assessment. However, due to the complexity of the protocol, the provided documentation can often be difficult to parse. In response, we provide some recommendations for clarifying the Highway paper.

- There are a few components of the proof Lemma 1 that should be explained with more detail. For example, the justification for some inequalities and logical progressions are omitted that should give more clarity if included.
- The definition of the GHOST rule should be more formalized. Specifically, the GHOST rule is invoked by passing in a tree of blocks and opinion function as input, but the description does not explicitly state how to use this opinion function.
- The description of the endorsement strategy does not discuss when or if validators should ever return to relaxed mode after entering cautious mode. The section on eras indicates that validators should return to relaxed mode at the beginning of every era. The description of the endorsement strategy should mention that validators should only exit cautious mode upon the completion of the era.
- Near the end of section 3.6.1, the following is stated for units violating LNC: “Thus it is important to suitably modify the unit reception strategy so that units not satisfying LNC are kept in a separate buffer instead of being discarded right away.” However, there are not further instructions for how to process these units after they are placed in this buffer.
- The $\texttt{opinion}_u$ and `vote` terms are defined circularly on page 10 (v on the left hand side in the definition of $\texttt{opinion}_u$ is a validator, but *v* on the right hand side is a unit). That is, $\texttt{opinion}_u$ is defined in terms of `vote`, which is defined in terms of $\texttt{opinion}_u$.
- The sets C_i in Lemma 4 only contains witness units, and they are shown to be a (|H|,k)-summit. However, this could be explained with more detail; specifically, it is not shown why they satisfy the convexity condition for summits.
- There is a mismatch between the code and protocol paper for the definition of an endorsed unit. The paper calls a unit endorsed if `n/2` validators endorse it. The implementation uses `sum(weight(v); v endorses the unit) >= (2/3) * total_weight`.

C. Recommendations for Code Quality

Overall, Trail of Bits found the implementation of the Highway protocol reviewed during this assessment to perform generally appropriate data validation in areas of concern. Although many of our suspicions did not lead to security findings, we derived recommendations which we believe will improve the security posture of the product as it undergoes further development.

- Calls to the `find_in_swimlane` and `wire_unit` functions are followed by a subsequent `unwrap` call. In the event that either of these function calls returns `None` or an `Err`, the subsequent `unwrap` call will result in a panic.
    - Although these cases are non-problematic in the current implementation, future changes to the codebase may overlook an edge case which makes this reachable in the future. (node/src/components/consensus/highway_core/state.rs#L631-L632)
        - In the former case, `find_in_swimlane` if the `unit.seq_number` for the unit with the provided `hash` is less than the provided `seq_number`, this will result in the abovementioned panic. (node/src/components/consensus/highway_core/state.rs#L653)
        - The latter case with the `wire_unit` function may fail in the event that the unit with the provided `hash` does not exist within the state. (node/src/components/consensus/highway_core/state.rs#L431)
    - As a result, we would suggest adding appropriate error handling for the return values from these called functions before performing the `unwrap` call.

D. Fix Log

During the week of December 14th, Trail of Bits reviewed CasperLabs' fixes and mitigations to address issues TOB-CLHW-001 through TOB-CLHW-003. Additionally, changes to the protocol paper and implementation were made to alleviate concerns detailed in Appendix B and Appendix C.

The results of this fix review can be observed below:

ID	Title	Severity	Status
01	LNC validation not implemented	Informational	Fixed
02	Peers are not punished for sending invalid vertices	Low	Partially Fixed
03	Insufficient rate limiting mechanism	Low	Partially Fixed
B	Recommendations for Documentation	N/A	Fixed
C	Recommendations for Code Quality	N/A	Fixed

For additional information, please refer to the detailed fix log.

Detailed Fix Log

Finding 1: LNC validation not implemented
Fixed. In later iterations of the codebase, CasperLabs has introduced the implementation of LNC validation and enabled it in the following pull requests: #534, #586, #609. A partial review of these changes did not reveal any concerns.

Finding 2: Peers are not punished for sending invalid vertices
Partially fixed. This issue is not immediately addressed with a rate limiting mechanism as suggested, however alternate mitigations have been applied. Specifically, a mitigation was added to limit the size of large evidence vertices.

The risk associated with this issue is noted as very low in practice.

Finding 3: Insufficient rate limiting mechanism
Partially fixed. Similar to Finding 2, this issue is not immediately addressed with a rate limiting mechanism as suggested, however a size limit on evidence vertices has been imposed as a mitigating factor against resource exhaustion attacks.

The risk associated with this issue is noted as very low in practice.

Appendix B: Recommendations for Documentation
Fixed. The CasperLabs team has adjusted the relevant protocol paper to address all concerns mentioned within this appendix.

Appendix C: Recommendations for Code Quality
Fixed. CasperLabs indicated such an issue would violate an important invariant of the system that should not occur, for which there is no ideal remediation scenario. In order to prevent such a violation, CasperLabs has indicated they have increased documentation surrounding such invariants within critical portions of the code, as done in pull request #622.