



Composable Finance – Democracy

Substrate Pallet Security
Audit

Prepared by: Halborn

Date of Engagement: September 19th, 2022 – September 26th, 2022

Visit: Halborn.com

DOCUMENT REVISION HISTORY	3
CONTACTS	4
1 EXECUTIVE OVERVIEW	5
1.1 INTRODUCTION	6
1.2 AUDIT SUMMARY	6
1.3 TEST APPROACH & METHODOLOGY	6
RISK METHODOLOGY	7
1.4 SCOPE	9
2 ASSESSMENT SUMMARY & FINDINGS OVERVIEW	10
3 FINDINGS & TECH DETAILS	11
3.1 (HAL-01) POTENTIAL UNEXPECTED BEHAVIOUR CAUSED BY BIG PREIMAGE - INFORMATIONAL	13
Description	13
Code Location	13
Risk Level	14
Recommendation	14
Remediation Plan	15
3.2 (HAL-02) USAGE OF ROOT ORIGIN - INFORMATIONAL	15
Description	15
Code Location	15
Risk Level	15
Recommendation	16
Remediation Plan	16
4 AUTOMATED TESTING	16
4.1 CARGO AUDIT	18

Description	18
Results	18

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	09/25/2022	Michal Bajor
0.2	Draft Review	09/26/2022	Timur Guvenkaya
0.3	Draft Review	09/26/2022	Gabi Urrutia
1.0	Remediation Plan	11/24/2022	Michal Bajor
1.1	Remediation Plan Review	11/24/2022	Timur Guvenkaya
1.2	Remediation Plan Review	11/24/2022	Piotr Cielas
1.3	Remediation Plan Review	11/24/2022	Gabi Urrutia

CONTACTS

CONTACT	COMPANY	EMAIL
Rob Behnke	Halborn	Rob.Behnke@halborn.com
Steven Walbroehl	Halborn	Steven.Walbroehl@halborn.com
Gabi Urrutia	Halborn	Gabi.Urrutia@halborn.com
Piotr Cielas	Halborn	Piotr.Cielas@halborn.com
Timur Guvenkaya	Halborn	Timur.Guvenkaya@halborn.com
Michal Bajor	Halborn	Michal.Bajor@halborn.com



EXECUTIVE OVERVIEW



1.1 INTRODUCTION

Composable Finance engaged Halborn to conduct a security audit on the Democracy pallet beginning on September 19th, 2022 and ending on September 26th, 2022. The security assessment was scoped to the pallet provided to the Halborn team.

1.2 AUDIT SUMMARY

The team at Halborn was provided two weeks for the engagement and assigned a full-time security engineer to audit the security of the Democracy pallet. The security engineer is a blockchain and smart-contract security expert with advanced penetration testing, smart-contract hacking, and deep knowledge of multiple blockchain protocols.

The purpose of this audit is to:

- Ensure that Substrate pallet's functions operate as intended
- Identify potential security issues with the Substrate pallet

In summary, Halborn identified some security risks that were acknowledged by the Composable Finance team.

1.3 TEST APPROACH & METHODOLOGY

Halborn performed a combination of manual and automated security testing to balance efficiency, timeliness, practicality, and accuracy regarding the scope of the Composable Substrate pallets. While manual testing is recommended to uncover flaws in logic, process, and implementation; automated testing techniques help enhance coverage of the code and can quickly identify items that do not follow security best practices. The following phases and associated tools were used throughout the term of the audit:

- Research into the architecture, purpose, and use of the platform.
- Smart contract manual code review and walkthrough to identify any logic issue.
- Mapping out possible attack vectors
- Thorough assessment of safety and usage of critical Rust variables and functions in scope that could lead to arithmetic vulnerabilities.
- On chain testing of core functions(`polkadot.js`).
- Active Fuzz testing {`cargo-fuzz`, `honggfuzz`}
- Scanning dependencies for known vulnerabilities (`cargo audit`).

RISK METHODOLOGY:

Vulnerabilities or issues observed by Halborn are ranked based on the risk assessment methodology by measuring the **LIKELIHOOD** of a security incident and the **IMPACT** should an incident occur. This framework works for communicating the characteristics and impacts of technology vulnerabilities. The quantitative model ensures repeatable and accurate measurement while enabling users to see the underlying vulnerability characteristics that were used to generate the Risk scores. For every vulnerability, a risk level will be calculated on a scale of 5 to 1 with 5 being the highest likelihood or impact.

RISK SCALE - LIKELIHOOD

- 5 - Almost certain an incident will occur.
- 4 - High probability of an incident occurring.
- 3 - Potential of a security incident in the long term.
- 2 - Low probability of an incident occurring.
- 1 - Very unlikely issue will cause an incident.

RISK SCALE - IMPACT

- 5 - May cause devastating and unrecoverable impact or loss.
- 4 - May cause a significant level of impact or loss.
- 3 - May cause a partial impact or loss to many.

- 2 - May cause temporary impact or loss.
- 1 - May cause minimal or un-noticeable impact.

The risk level is then calculated using a sum of these two values, creating a value of 10 to 1 with 10 being the highest level of security risk.

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
----------	------	--------	-----	---------------

- 10 - CRITICAL
- 9 - 8 - HIGH
- 7 - 6 - MEDIUM
- 5 - 4 - LOW
- 3 - 1 - VERY LOW AND INFORMATIONAL

1.4 SCOPE

The review was scoped to the `democracy` pallet in `frame` directory in the `ComposableFi/substrate` repository.

Commit IDs used for the engagement:

- `76e3033cd1ebe9487757381fd2f34f3fb54caa93`

2. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	0	0	0	2

LIKELIHOOD

IMPACT

(HAL-01) (HAL-02)				

SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
POTENTIAL UNEXPECTED BEHAVIOUR CAUSED BY BIG PREIMAGE	Informational	ACKNOWLEDGED
USAGE OF ROOT ORIGIN	Informational	ACKNOWLEDGED



FINDINGS & TECH DETAILS



3.1 (HAL-01) POTENTIAL UNEXPECTED BEHAVIOUR CAUSED BY BIG PREIMAGE – INFORMATIONAL

Description:

The `democracy` pallet requires a `preimage` to be submitted prior to submitting a proposal. The `preimage` itself is constructed by hashing the encoded proposal and matching it with its corresponding asset ID. User can submit anything as an encoded proposal, not necessarily a valid one. The user submitting the preimage via `note_preimage` function needs to transfer some balance, which depends on the size of the `encoded_proposal`. The encoded proposal is represented as a `Vec` of bytes in the code. When the number of tokens owed for this proposal is calculated, the length of the `encoded_proposal` is casted to `u32` type. This will work as expected for proposals of size not exceeding 4 GB; however, it will break if the user manages to send bigger proposals.

Code Location:

Listing 1: `substrate/frame/democracy/src/lib.rs`

```

1950 fn note_preimage_inner(who: T::AccountId, encoded_proposal: Vec<u8
    ↳ >) -> DispatchResult {
1951     let proposal_hash = T::Hashing::hash(&encoded_proposal[..]);
1952     ensure!(!<Preimages<T, I>>::contains_key(&proposal_hash),
    ↳ Error::<T, I>::DuplicatePreimage);
1953
1954     let deposit = <BalanceOf<T, I>>::from(encoded_proposal.len()
    ↳ as u32)
1955         .saturating_mul(T::PreimageByteDeposit::get());
1956     T::Currency::reserve(&who, deposit)?;
1957
1958     let now = <frame_system::Pallet<T>>::block_number();
1959     let a = PreimageStatus::Available {
1960         data: encoded_proposal,
1961         provider: who.clone(),
1962         deposit,

```

```
1963         since: now,  
1964         expiry: None,  
1965     };  
1966     <Preimages<T, I>>::insert(proposal_hash, a);  
1967  
1968     Self::deposit_event(Event::<T, I>::PreimageNoted {  
1969         ↪ proposal_hash, who, deposit });  
1969  
1970     Ok(())  
1971 }
```

Risk Level:

Likelihood - 1

Impact - 1

Recommendation:

It is recommended to implement a verification mechanism which will first check the size of `encoded_proposal` and will not perform any further computation if that size exceeds a predefined limit.

Remediation Plan:

ACKNOWLEDGED: The **Composable Finance team** acknowledged this issue.

3.2 (HAL-02) USAGE OF ROOT ORIGIN – INFORMATIONAL

Description:

The **democracy** pallet is using the root origin in multiple function. Using such origin does not align with the blockchain paradigm of decentralization.

Code Location:

An exemplary function utilizing a root origin:

Listing 2: substrate/frame/democracy/src/lib.rs (Line 956)

```

955 pub fn cancel_queued(origin: OriginFor<T>, which: ReferendumIndex)
    ↳ -> DispatchResult {
956     T::EnsureRoot::ensure_origin(origin)?;
957     T::Scheduler::cancel_named((T::DemocracyId::get(), which).
    ↳ encode())
958     .map_err(|_| Error::<T, I>::ProposalMissing)?;
959     Ok(())
960 }
```

Risk Level:

Likelihood - 1

Impact - 1

Recommendation:

It is recommended to not use a root origin and introduce an appropriate committee for handling administrative operations.

Remediation Plan:

ACKNOWLEDGED: The **Composable Finance team** acknowledged this issue.



AUTOMATED TESTING



4.1 CARGO AUDIT

Description:

Halborn used automated security scanners to assist with detection of well-known security issues and vulnerabilities. Among the tools used was `cargo audit`, a security scanner for vulnerabilities reported to the RustSec Advisory Database. All vulnerabilities published in <https://crates.io> are stored in a repository named The RustSec Advisory Database. `cargo audit` is a human-readable version of the advisory database which performs a scanning on Cargo.lock. Security Detections are only in scope. All vulnerabilities shown here were already disclosed in the above report. However, to better assist the developers maintaining this code, the auditors are including the output with the dependencies tree, and this is included in the cargo audit output to better know the dependencies affected by unmaintained and vulnerable crates.

Results:

Crate: `owning_ref`

Version: `0.4.1`

Title: Multiple soundness issues in `owning_ref`

Date: 2022-01-26

ID: RUSTSEC-2022-0040

URL: <https://rustsec.org/advisories/RUSTSEC-2022-0040>

Solution: No safe upgrade is available!

Crate: `rocksdb`

Version: `0.18.0`

Title: Out-of-bounds read when opening multiple column families with TTL

Date: 2022-05-11

ID: RUSTSEC-2022-0046

URL: <https://rustsec.org/advisories/RUSTSEC-2022-0046>

Solution: Upgrade to `>=0.19.0`

Crate: `time`

Version: `0.1.44`

Title: Potential segfault in the time crate
 Date: 2020-11-18
 ID: RUSTSEC-2020-0071
 URL: <https://rustsec.org/advisories/RUSTSEC-2020-0071>
 Solution: Upgrade to >=0.2.23

Crate: ansi_term
 Version: 0.12.1
 Warning: unmaintained
 Title: ansi_term is Unmaintained
 Date: 2021-08-18
 ID: RUSTSEC-2021-0139
 URL: <https://rustsec.org/advisories/RUSTSEC-2021-0139>

Crate: serde_cbor
 Version: 0.11.2
 Warning: unmaintained
 Title: serde_cbor is unmaintained
 Date: 2021-08-15
 ID: RUSTSEC-2021-0127
 URL: <https://rustsec.org/advisories/RUSTSEC-2021-0127>

Crate: sp-version
 Version: 5.0.0
 Warning: yanked



THANK YOU FOR CHOOSING

 **HALBORN**

