Acala

Security Assessment

16th February 2021

Prepared For:
Ruitao Su | *Acala*
ruitao@laminar.one

Bette Chen | *Acala*
bette@laminar.one

Bryan Chen | *Acala*
bryan@laminar.one

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

Michael Colburn | *Trail of Bits*
michael.colburn@trailofbits.com

Changelog:

January 27, 2021	Draft report delivered
February 16, 2021	Final report delivered

Executive Summary

From January 4 through January 22, 2021, Acala engaged Trail of Bits to review the security of Acala Network, a Polkadot blockchain parachain. Trail of Bits conducted this assessment over the course of six person-weeks with two engineers working from the provided repositories.

During the first week, we focused on gaining an understanding of the codebase and the protocols used within the Acala Network. This consisted of reviewing the documentation, performing static analysis against the Acala repositories, and prioritizing high-impact areas within the source code for further investigation. In the second week, we focused our review on the Homa and Honzon protocols. We also investigated the heavily used Substrate and ORML libraries and modified them in order to log additional information useful for testing. The final week was dedicated to finishing investigations into the Homa and Honzon protocols and reviewing the incentive pools and authority code. Additionally, we analyzed all JavaScript code for potential web security issues.

Our review resulted in 11 findings ranging from Low to Informational in severity. The three low severity findings are about the inability of transferring MAX amount of ACA tokens (TOB-ACA-004), insecure configuration for running Acala docker container (TOB-ACA-001) and missing of security-related HTTP headers in Acala-dapp (TOB-ACA-008).

In addition to the findings, we documented the Homa protocol properties we tested in Appendix B and listed the steps for testing a local Acala node using its development accounts (Alice, Bob, etc.) with Acala-dapp and Polkadot UI in Appendix D. We also included Docker containers recommendations in Appendix C and documented some non-security issues in the Appendix E: Code Quality Recommendations.

Overall, Acala is a complex distributed system whose security heavily depends on the chosen parameters and protocols. However, Acala lacks proper documentation on many aspects of its system, which results in misunderstanding and weak implementation. Multiple features, such as the Polkadot bridge, governance, and reputation, are yet to be implemented, leading to major modifications after the audit conclusion. We recommend finishing the implementation, then extending and centralizing documentation so it describes the chosen parameters and algorithms in detail and then holding another security assessments to make sure the security expectations of the system hold.

Project Dashboard

Application Summary

Name	Acala

Version	`Acala (da5aeeafd0)`
	`Acala.js (7f2f4c930d)`
	`Acala-dapp (bdb240968)`
	`Apps (eea2c1baa0)`

Type	Rust, JavaScript
Platforms	Linux

Engagement Summary

Dates	January 4 through January 22, 2021
Method	Whitebox
Consultants Engaged	2
Level of Effort	6 person-weeks

Vulnerability Summary

Total Low-Severity Issues	4	■ ■ ■ ■
Total Informational-Severity Issues	6	■ ■ ■ ■ ■ ■
Total Undetermined-Severity Issues	1	■
Total	11	

Category Breakdown

Configuration	3	■ ■ ■
Data Validation	2	■ ■
Denial of Service	1	■
Documentation	2	■ ■
Patching	1	■
Undefined Behavior	2	■ ■
Total	11	

Engagement Goals

The engagement was scoped to provide a security assessment of the Acala Network parachain.

Specifically, we sought to answer the following questions:

- Are there any design flaws in the Acala system architecture?
- Is it possible to influence the staking protocol in a malicious way, e.g., to steal funds?
- Are there appropriate access controls applied to executed extrinsics?
- Are the parameters appropriately validated? Are their bounds limited properly to prevent the possibility of exhausting the storage?
- Does any unsafe arithmetic occur? If so, can overflows, underflows, or rounding errors result?
- Are there events recorded when meaningful actions happen?

Coverage

Acala. We manually reviewed the code and dynamically ran some scenarios on a local test network. We investigated the chain configuration and certain aspects of the Acala system like the Homa and Honzon protocols mentioned below.

Acala-dapp, Acala/apps, Acala.js. We used static analysis tools such as Webstorm code inspection. We did not focus our review on those projects. We used Acala-dapp to test a local build of the Acala chain. We tried working with the Acala/apps but we were unable to build its Docker image locally, so we used the Polkadot UI one instead (by loading types defined by the `acala-network/type-definitions==0.6.2-4` package), or, used Acala-dapp.

Homa Protocol. We manually reviewed the code and documentation for the staking liquidity protocol and extracted system properties related to the core functionality of the protocol, minting LDOT, various redemption mechanisms and token withdrawal. We investigated each of these properties and documented our results in Appendix B.

Honzon Protocol. We manually reviewed this stablecoin protocol with a focus on proper accounting in token issuance, CDP management as well as the collateral auction process when delinquent users are liquidated.

Incentive pools. We manually reviewed the various incentive pools on the Acala chain. We looked for any flaws in the accumulation logic as well as the reward claiming process. As the incentive mechanisms were not yet documented we could not fully verify their correctness.

Authority. We manually reviewed the governance module to ensure the various types of governance actions adhered to the timelines in the brief specification provided.

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

❑ **Add the "`--cap-drop=ALL --security-opt=no-new-privileges:true`" flags to all `docker run` invocations listed in the Acala documentation for running a node in a Docker container.** This will increase the security of containers run by the users. TOB-ACA-001

❑ **Either remove the `sudo` functionality from the Acala chain specification or, document how it will be disabled or removed after the network goes live, similarly to what the Polkadot did during its launch.** TOB-ACA-002

❑ **Evaluate the change of `liquidAmountToBurn` value in the `getStakingAmountInRedeemByFreeUnbounded` and `getStakingAmountInClaimUnbonding` functions in the `acala.js` codebase.** TOB-ACA-003

❑ **Fix the inability of transferring "MAX" ACA tokens amount to another account through Acala-dapp.** This can be done by adding a special "transfer of all ACA tokens" extrinsic that would first deduct the fees and then perform the transfer of the remaining tokens, and using that extrinsic when sending a "MAX" ACA tokens transfer in the Acala-dapp UI. TOB-ACA-004

❑ **Update the Substrate dependency used in Acala to the latest version as its newer version doesn't contain the chaostests framework that contains security vulnerabilities in its dependencies.** TOB-ACA-005

❑ **Add steps for testing Acala and Acala-dapp locally with blockchain test/development accounts (Alice, Bob, etc.) into Acala's README or/and Development Guide.** This will help developers to bootstrap a working environment and test features faster. TOB-ACA-006

❑ **Fix the CSRF issue in `Acala/apps` which was reported in `polkadot.js/apps#4465`.** This will prevent an attacker from sending a malicious link to its victim that will change the RPC endpoint URL. If possible, send a patch that fixes this issue to the upstream repository. TOB-ACA-007

❑ **Add the missing security-related HTTP headers to Acala-dapp.** Prevent the site from being rendered in iframes with `X-Frame-Options: DENY` header and implement the CSP policy and validate it with a CSP Evaluator. This will help mitigate the effects of attacks such as clickjacking or XSS. TOB-ACA-008

❑ **Change the way small values are displayed in the Acala-dapp.** Allow the user to specify the amount of decimal places to be displayed and inform the user if they have a small, non-zero value of a given token which is not currently displayed. TOB-ACA-009

❑ **Fix the unresponsiveness of Acala-dapp when a small number is provided into the "Receive" form of the "Swap" functionality.** TOB-ACA-010

❑ **Expand and unify the documentation to better cover governance, incentives, and gas fees.** TOB-ACA-011

Long term

❑ **Consider using a multi-stage Docker image build for Acala docker image and adding only the necessary binaries to the resulting image.** This will decrease the attack surface for an attacker who gains code execution in the container and should also result in smaller Docker images. Additionally, review the Appendix C: Docker Recommendations for further Docker guidance. TOB-ACA-001

❑ **Add tests to ensure the "MAX" ACA and other tokens transfer succeeds.** This will prevent similar bugs from appearing if the code changes. All financial transactions should be tested with both use cases and misuse cases. TOB-ACA-004

❑ **Implement dependency checks for Acala's dependencies (and their dependencies) as part of the CI/CD pipeline of application development.** Do not allow builds to continue with any outdated dependencies. TOB-ACA-005

❑ **Keep track of the `polkadot.js/apps` changes and update the `Acala/apps` repository regularly.** This will help prevent similar issues to occur in the future. TOB-ACA-007

❑ **Track the further developments of CSP and similar web browser features that help with mitigating security risk.** As new protections are developed, ensure they are adopted as quickly as possible. TOB-ACA-008

❑ **Add tests to Acala-dapp to see if it allows the user to see a small amount of tokens.** TOB-ACA-009

❑ **Add functional tests against forms in Acala-dapp to see if they correctly handle small values or malicious inputs.** TOB-ACA-010

❑ **Regularly review public documentation to ensure it is kept up to date with the current system.** TOB-ACA-011

Findings Summary

#	Title	Type	Severity
1	Insecure configuration for running Acala node in a Docker container	Configuration	Low
2	Sudo is enabled on the Acala chain	Configuration	Informational
3	Changed but unused liquidAmountToBurn value	Undefined Behavior	Informational
4	Transferring "max" ACA tokens through Acala-dapp fails and only burns the fees	Data Validation	Low
5	The Substrate dependency "chaostests" contain out of date dependencies that have security vulnerabilities	Patching	Informational
6	Lack of proper development guidance on using Acala-dapp with Acala	Documentation	Informational
7	CSRF in `Acala/apps` settings which allows changing the RPC endpoint URL	Data Validation	Informational
8	Missing security-related HTTP headers in the Acala-dapp application	Configuration	Low
9	Small amounts are not displayed in Acala-dapp or are displayed in a scientific notation	Undefined Behavior	Informational
10	Providing too small value renders Acala-dapp unresponsive	Denial of Service	Informational
11	Documentation is incomplete	Documentation	Informational

1. Insecure configuration for running Acala node in a Docker container

Severity: Low
Difficulty: High
Type: Configuration
Finding ID: TOB-ACA-001
Target: `documentation for running node in a Docker container`

Description

The Acala documentation describes how to launch an Acala node in a Docker container, from the `acala/acala-node` docker images. While those images run the node process as a non-root user, the documented way to run them does not prevent the user to gain more privileges. This may allow an attacker who gets code execution access within the container to escalate their privileges in certain scenarios.

The first docker invocation can be found at the integration-guide/node-management documentation page:

```
      docker run -p 9944:9944 acala/acala-node:0.6.2 --name
"calling_home_from_a_docker_container" --rpc-external --ws-external
```

While the second at the network-maintainers/mandala-maintainers-guide page:

```
      docker run -d --restart=always -p 30333:30333 -p 9933:9933 -p 9944:9944
-v node-data:/acala/data acala/acala-node:latest --chain mandala
--base-path=/acala/data/01-001 --ws-port 9944 --rpc-port 9933 --port 30333
--ws-external --rpc-external --ws-max-connections 1000 --rpc-cors=all
--unsafe-ws-external --unsafe-rpc-external --pruning=archive --name "Name of
Telemetry"
```

Both of those invocations do not set the "No New Privileges" flag and do not drop all Linux capabilities. Those features prevent the user from gaining more privileges through e.g. suid binaries and from gaining more Linux capabilities. The current value of those settings can be inspected by reading the `/proc/$PID/status` file as shown in Figures 1.1 and 1.2, where the second figure shows those settings with security flags applied to the `docker run` invocation.

```
$ docker run --rm -it --entrypoint bash acala/acala-node:latest
acala@d7720fb273ce:/$ cat /proc/$$/status | egrep 'Cap|NoNewPrivs|Seccomp'
CapInh: 00000000a80425fb
CapPrm: 0000000000000000
CapEff: 0000000000000000
CapBnd: 00000000a80425fb
CapAmb: 0000000000000000
NoNewPrivs:     0
Seccomp:        2
```

Figure 1.1: Inspecting the parent process (here: `bash`*) Linux capabilities sets, the "No New Privileges" flag and its Seccomp status. While the process has no effective Linux capabilities (*`CapEff`*) currently, it may gain them through e.g. suid binaries.*

```
$ docker run --rm -it --cap-drop=ALL --security-opt=no-new-privileges:true --entrypoint bash
acala/acala-node:latest
acala@aeec10d88e66:/$ cat /proc/$$/status | egrep 'Cap|NoNewPrivs|Seccomp'
CapInh: 0000000000000000
CapPrm: 0000000000000000
CapEff: 0000000000000000
CapBnd: 0000000000000000
CapAmb: 0000000000000000
NoNewPrivs:     1
Seccomp:        2
```

Figure 1.2: Inspecting the process status after passing in `docker run` *flags which drops all Linux capabilities and sets the "No New Privileges" flag.*

Exploit Scenario

Alice runs an Acala node in a Docker container. Eve finds a bug that allows her to run arbitrary code on Alice's node. She then uses the bug and runs a suid binary to gain more privileges in the container to attack the host system.

Currently, the only suid binaries present in the `acala/acala-node:latest` (`93c46b70e208`) image are `/bin/mount`, `/bin/umount`, `/bin/su` and `/usr/lib/openssh/ssh-keysign`. Analyzing these binaries was outside the scope of this assessment.

Recommendations

Short term, add the "`--cap-drop=ALL --security-opt=no-new-privileges:true`" flags to all `docker run` invocations listed in the Acala documentation for running a node in a Docker container. This will increase the security of containers run by the users.

Long term, consider using a multi-stage Docker image build for Acala docker image and adding only the necessary binaries to the resulting image. This will decrease the attack surface for an attacker who gains code execution in the container and should also result in smaller Docker images. Additionally, review the Appendix C: Docker Recommendations for further Docker guidance.

2. Sudo is enabled on the Acala chain

Severity: Informational | Difficulty: Medium
Type: Configuration | Finding ID: TOB-ACA-002
Target: `Acala/bin/acala/service/src/chain_spec/acala.rs#L164`

Description
The Acala chain spec uses the sudo module (Figure 2.1), which allows for robust testing but should not exist in a production network. The existence of a `sudo` functionality, or, a lack of clear documentation of a sudo removal step may decrease the users' trust in the Acala Network.

```
        acala_runtime::GenesisConfig {
                // (...)
                pallet_sudo: Some(SudoConfig { key: root_key.clone() }),
```

Figure 2.1: The sudo pallet enabled in Acala chain specification (`Acala/bin/acala/service/src/chain_spec/acala.rs#L164`).

Recommendation
Short term, either remove the `sudo` functionality from the Acala chain specification or, document how it will be disabled or removed after the network goes live, similarly to what the Polkadot did during its launch.

3. Changed but unused liquidAmountToBurn value

Severity: Informational
Difficulty: High
Type: Undefined Behavior
Finding ID: TOB-ACA-003
Target: `acala.js/packages/sdk-homa/src/staking-pool.ts`

Description

There are two places in the `acala.js` codebase where a variable of `liquidAmountToBurn` is introduced, used, changed but unused further in the flow (Figures 3.1-2). Those cases should be further investigated as they could be real bugs.

```
public getStakingAmountInRedeemByFreeUnbonded(/* (...) */): { /* (...) */ } {
  // (...)
  let liquidAmountToBurn = amount;
  let demandStakingAmount = liquidExchangeRate.times(liquidAmountToBurn);
  // (...)
  if (!demandStakingAmount.isZero() && !availableFreePool.isZero()) {
    if (demandStakingAmount.isGreaterThan(availableFreePool)) {
      const ratio = FixedPointNumber.fromRational(availableFreePool, demandStakingAmount);

      liquidAmountToBurn = ratio.times(liquidAmountToBurn);
      demandStakingAmount = availableFreePool;
    }
  // (...) - the liquidAmountToBurn value is unused
}
```

Figure 3.1: The `getStakingAmountInRedeemByFreeUnbonded` *function* (`acala.js/packages/sdk-homa/src/staking-pool.ts#L165-L179`).

```
public getStakingAmountInClaimUnbonding(/* (...) */): { /* (...) */ } {
  // (...)
  let liquidAmountToBurn = amount;
  let demandStakingAmount = liquidExchangeRate.times(liquidAmountToBurn);
  // (...)
  if (!demandStakingAmount.isZero() && !availableUnclaimedUnbonding.isZero()) {
    if (demandStakingAmount.isGreaterThan(availableUnclaimedUnbonding)) {
      const ratio = FixedPointNumber.fromRational(availableUnclaimedUnbonding,
demandStakingAmount);

      liquidAmountToBurn = ratio.times(liquidAmountToBurn);
      demandStakingAmount = availableUnclaimedUnbonding;
    }
    // (...) - the liquidAmountToBurn value is unused
}
```

Figure 3.2: The `getStakingAmountInClaimUnbonding` *function* (`acala.js/packages/sdk-homa/src/staking-pool.ts#L275-L289`).

Recommendation

Short term, evaluate the change of `liquidAmountToBurn` value in the `getStakingAmountInRedeemByFreeUnbounded` and `getStakingAmountInClaimUnbonding` functions in the `acala.js` codebase.

4. Transferring "max" ACA tokens through Acala-dapp fails and only burns the fees

Severity: Low	Difficulty: Low
Type: Data Validation	Finding ID: TOB-ACA-004
Target: `Acala / Acala-dapp`	

Description

The Acala-dapp UI provides a way of transferring all tokens to a given account via the "MAX" button in its "Transfer" dialog (Figure 4.1). However, trying to perform a transfer of ACA tokens with the "MAX" amount always fails with a "balances.InsufficientBalance" result (Figure 4.2) and the sender ends up losing the transfer fees.

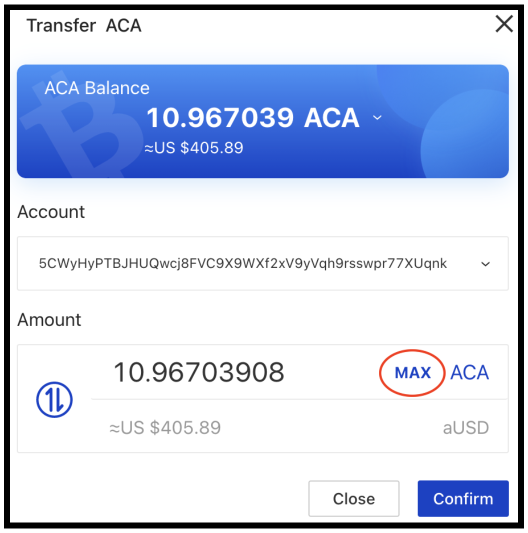


Figure 4.1: The "Transfer" dialog with the "MAX" button.

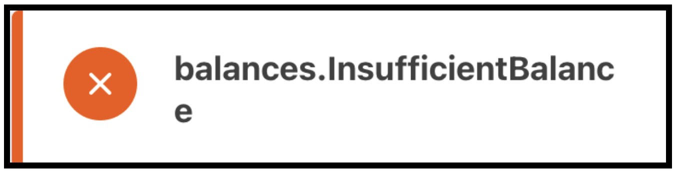

balances.InsufficientBalanc
e

Figure 4.2: The "`balances.InsufficientBalance`" error displayed in the top right corner of Acala-dapp when attempting to transfer "MAX" ACA tokens amount.

Exploit Scenario

Alice wants to transfer all of her ACA tokens into another account. She uses the "MAX" transfer option through the Acala-dapp application. The transfer fails and Alice ends up with the transfer fees deducted from her ACA tokens balance.

Recommendations

Short term, fix the inability of transferring "MAX" ACA tokens amount to another account through Acala-dapp. This can be done by adding a special "transfer of all ACA tokens" extrinsic that would first deduct the fees and then perform the transfer of the remaining tokens, and using that extrinsic when sending a "MAX" ACA tokens transfer in the Acala-dapp UI.

Long term, add tests to ensure the "MAX" ACA and other tokens transfer succeeds. This will prevent similar bugs from appearing if the code changes. All financial transactions should be tested with both use cases and misuse cases.

5. The Substrate dependency "chaostests" contain out of date dependencies that have security vulnerabilities

Severity: Informational | Difficulty: Undetermined
Type: Patching | Finding ID: TOB-ACA-005
Target: `Acala's dependency substrate/.maintain/chaostests dependencies`

Description
The Substrate dependency version `6b600cdeb` used by Acala uses two outdated dependencies that contain security vulnerabilities in its testing framework called "chaostests".

The outdated dependencies are:
- Node-fetch version 1.7.3, with a DoS vulnerability (CVE-2020-15168), fixed in 2.6.1.
- Node-forge version 0.8.5, with a prototype pollution vulnerability (CVE-2020-7720), fixed in 0.10.0.

The severity of this finding is set to Informational as we believe this issue does not impact the security of the Acala blockchain since the vulnerable code is not used by Acala code.

Recommendations
Short term, update the Substrate dependency used in Acala to the latest version as its newer version doesn't contain the chaostests framework that contains security vulnerabilities in its dependencies.

Long term, implement dependency checks for Acala's dependencies (and their dependencies) as part of the CI/CD pipeline of application development. Do not allow builds to continue with any outdated dependencies.

6. Lack of proper development guidance on using Acala-dapp with Acala

Severity: Informational
Difficulty: Undetermined
Type: Documentation
Finding ID: TOB-ACA-006
Target: `Acala development documentation`

Description
The Acala Development Guide lacks proper documentation on how to use a local build of Acala-dapp and Acala node to test or develop the Acala blockchain. It is also not clear how to match the versions of the two, as the git commits that change corresponding dependencies versions are not having clear commit messages.

In Appendix D we describe steps for testing Acala and Acala-dapp with its test/development accounts (Alice, Bob, etc.) that could be added to the development guide.

Recommendations
Short term, add steps for testing Acala and Acala-dapp locally with blockchain test/development accounts (Alice, Bob, etc.) into Acala's README or/and Development Guide. This will help developers to bootstrap a working environment and test features faster.

7. CSRF in `Acala/apps` settings which allows changing the RPC endpoint URL

Severity: Informational
Difficulty: Medium
Type: Data Validation
Finding ID: TOB-ACA-007
Target: `Acala/apps`

Description
There is a Cross-Site Request Forgery (CSRF) vulnerability in the settings page of the `polkadot.js/apps` project on which the `Acala/apps` project is based upon. This allows an attacker to send a malicious link to the victim, which will change the victim's RPC endpoint URL.

The issue has been reported to the upstream repository in `polkadot.js/apps#4465`.

Recommendations
Short term, fix the CSRF issue in `Acala/apps` which was reported in `polkadot.js/apps#4465`. This will prevent an attacker from sending a malicious link to its victim that will change the RPC endpoint URL. If possible, send a patch that fixes this issue to the upstream repository.

Long term, keep track of the `polkadot.js/apps` changes and update the `Acala/apps` repository regularly. This will help prevent similar issues to occur in the future.

8. Missing security-related HTTP headers in the Acala-dapp application

Severity: Low
Difficulty: High
Type: Configuration
Finding ID: TOB-ACA-008
Target: `Acala/acala-dapp`

Description
The responses from the Acala-dapp website are missing the following security-related HTTP headers:

- Content-Security-Policy. The lack of this header could allow for an attacker to exploit XSS vulnerabilities that a CSP might otherwise mitigate.
- X-Frame-Options that can prevent the site from being rendered e.g. in an iframe and mitigates clickjacking attacks. Currently, if the site is rendered in an iframe, it won't work properly e.g. with the Polkadot{.js} extension as it doesn't work in iframes. The extension has an issue for it in polkadot-js/extension#531 and if it gets fixed, it will make Acala-dapp less secure due to lack of the `X-Frame-Options` setting.

Exploit Scenario
Eve finds a bug in Acala-dapp which allows her to trigger an XSS attack on Alice's browser when she executes an action. Eve then prepares a custom-crafted XSS payload. Due to a lack of CSP header, the Alice's browser executes Eve's attack and Eve executes actions on behalf of Alice's account.

Recommendation
Short term, add the missing security-related HTTP headers to Acala-dapp. Prevent the site from being rendered in iframes with `X-Frame-Options: DENY` header and implement the CSP policy and validate it with a CSP Evaluator. This will help mitigate the effects of attacks such as clickjacking or XSS.

Long term, track the further developments of CSP and similar web browser features that help with mitigating security risk. As new protections are developed, ensure they are adopted as quickly as possible.

References
- Mozilla developer docs: Content Security Policy
- CSP evaluator from Google

9. Small amounts are not displayed in Acala-dapp or are displayed in a scientific notation

Severity: Informational
Type: Undefined Behavior
Target: `Acala/acala-dapp`

Difficulty: Low
Finding ID: TOB-ACA-009

Description
The Acala-dapp presents small amounts of values in a scientific notation (Figures 9.1-2) and if a value is too small, e.g. 0.00000001 it may not be presented at all (Figure 9.2). This behavior may be confusing for users who don't understand the rules of displaying the values present in the system.

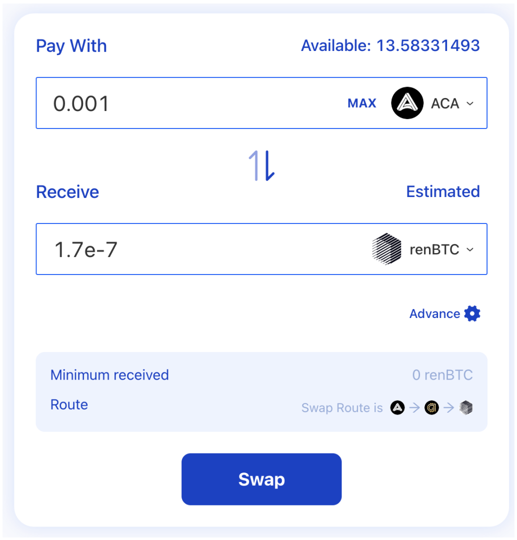


Figure 9.1: Small amount is displayed in a scientific notation.

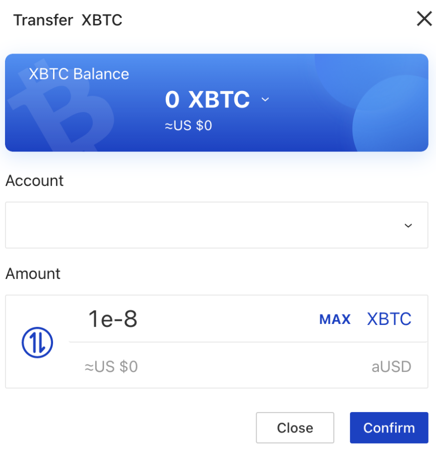


Figure 9.2: While the account has 0.00000001 XBTC which can be seen by trying to transfer "MAX" XBTC tokens, the UI does not have an option to display this value at all.

Recommendations
Short term, change the way small values are displayed in the Acala-dapp. Allow the user to specify the amount of decimal places to be displayed and inform the user if they have a small, non-zero value of a given token which is not currently displayed.

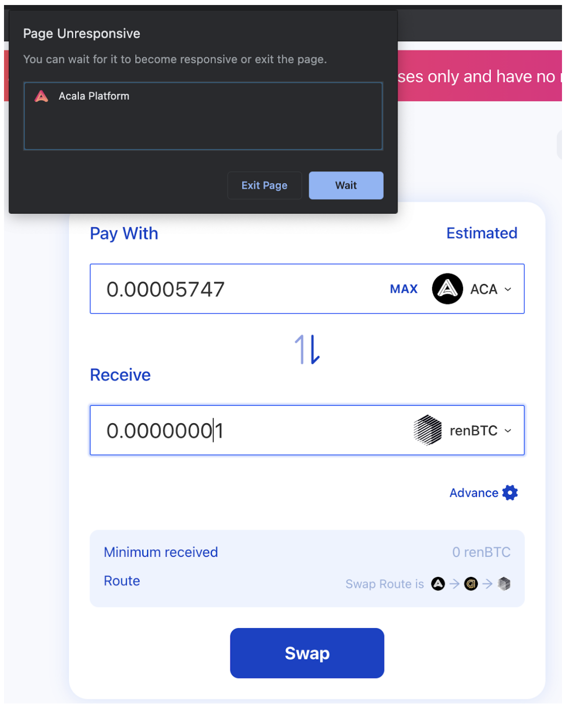
Long term, add tests to Acala-dapp to see if it allows the user to see a small amount of tokens.

10. Providing too small value renders Acala-dapp unresponsive

Severity: Informational
Difficulty: Low
Type: Denial of Service
Finding ID: TOB-ACA-010
Target: `Acala/acala-dapp`

Description
Passing in a too small value into the "Receive" form field in the "Swap" tab renders the Acala-dapp application unresponsive. This prevents the user from performing an action which is valid within the system.

This can be reproduced by going to the "Swap" tab, choosing e.g. the XBTC currency and pasting in the "0.000000001" (or smaller) value into the "Receive" form as shown in Figure 10.1.



Page Unresponsive

You can wait for it to become responsive or exit the page.

Acala Platform

ses only and have no

Pay With: 0.00005747 ACA (Estimated)

Receive: 0.00000001 renBTC

Minimum received: 0 renBTC

Route: Swap Route is

Figure 10.1: Providing a too small value into the "Receive" field caused the page to hang.

Recommendations
Short term, fix the unresponsiveness of Acala-dapp when a small number is provided into the "Receive" form of the "Swap" functionality.

Long term, add functional tests against forms in Acala-dapp to see if they correctly handle small values or malicious inputs.

11. Documentation is incomplete

Severity: Informational
Difficulty: High
Type: Documentation
Finding ID: TOB-ACA-011
Target: `documentation`

Description
Several areas of the Acala system would benefit from more thorough documentation. Comprehensive documentation of the system's behaviour is important for informing users of how the system operates, onboarding new employees, as well as verifying correctness of functionality.

The following areas were identified that would benefit from improved documentation:

- Governance processes, which are described both in the wiki and in a hackmd note but neither is comprehensive and there are some points of conflict.
- The various network incentives managed by the `incentives` module.
- Gas fees.

Exploit Scenario
Bob joins Acala as a new developer. Unaware of the expected behavior of a particular module, he modifies system parameters as part of a pull request and introduces unexpected behavior into the system.

Recommendation
Short term, expand and unify the documentation to better cover governance, incentives, and gas fees.

Long term, regularly review public documentation to ensure it is kept up to date with the current system.

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
Testing	Related to test methodology or test coverage
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. Homa Properties

The table below lists properties of the Homa Tokenized Liquidity Staking protocol gathered and tested during the assessment.

Property	**Status**
Minting	
The user receives the correct amount of LDOT	Minting uses `liquid_exchange_rate` to determine the correct amount of LDOT to issue.
The system takes the correct amount of DOT	Minting calls the DOT `transfer` function to transfer the amount passed to `mint`.
Zero mints are handled properly	The function immediately returns if the amount to mint is zero.
The user can't mint an amount greater than their balance	Minting attempts to transfer the desired amount of tokens and the underlying token implementation ensures there is sufficient balance for the transfer.
"Free" Unbonding (immediate) (unbonding is dispatched through the `homa/redeem` extrinsic)	
The protocol handles unbonds for greater than the amount of liquid tokens	Free unbonding takes into account how many tokens are liquid and scales the actual amount to redeem appropriately.
Immediate unbonding doesn't allow malicious validators to avoid slashing	The delay due to unbonding LDOT, the fee mechanism and the limited quantity of free unbonded DOT available should act as a disincentive to malicious validators self-nominating with LDOT.
The correct fee amount is taken	The function first determines what percentage of the desired quantity of tokens to be returned are available and then calculates the fee based on the remaining amount.
The correct amount of LDOT is burned	After determining how many tokens to burn, the system burns the tokens via the underlying token implementation.

Zero unbonds are handled properly	The function returns immediately if the amount to mint is zero.
The user can't unbond an amount greater than their balance	Unbonding attempts to `withdraw` (burn) the desired amount of tokens and the underlying token implementation ensures there is sufficient balance for the transfer.
"Claim" Unbonding (targeted era) (unbonding is dispatched through the `homa/redeem` extrinsic)	
An appropriate amount of DOT is queued to unbond by the target era	The system tracks how much future unbonding DOT has already been claimed and calculates the maximum proportion of the desired amount that can be redeemed. Any additional unbonding is handled by the `rebalance` function.
The target era is in the correct range (i.e., in the future, but less the normal unbonding window)	The target era is checked to ensure it is in the future but not beyond the regular unbonding period.
The correct fee amount is taken	The fees are subtracted from the demand staking amount.
The correct amount of LDOT is burned	After determining how many tokens to burn, the system burns the tokens via the underlying token implementation.
Zero unbonds are handled properly	The function returns immediately if the amount to mint is zero.
The user can't unbond an amount greater than their balance	Unbonding attempts to `withdraw` (burn) the desired amount of tokens and the underlying token implementation ensures there is sufficient balance for the transfer.
Unbonding (natural) (unbonding is dispatched through the `homa/redeem` extrinsic)	
An appropriate amount of DOT is queued to unbond	The system tracks how much DOT is unbonding in the future and adds the requested amount to the total. The actual unbonding is scheduled by the `rebalance` function in the next era.
Zero unbonds are handled properly	The function returns immediately if the amount to mint is zero.

The correct amount of LDOT is burned	After determining how many tokens to burn, the system burns the tokens via the underlying token implementation.
The user can't unbond an amount greater than their balance	Unbonding attempts to `withdraw` (burn) the desired amount of tokens and the underlying token implementation ensures there is sufficient balance for the transfer.
Redemption	
The correct amount of DOT is returned to the user	The system iterates over all matured unbondings to calculate the sum total amount of DOT to return to the user.
The user is not able to withdraw DOT when they have no unbonded tokens available.	The redemption calculation amount is initialized to zero. Successful redemptions are removed from the set of unbondings, preventing a double withdrawal.
Rebalancing	
The protocol adjusts downward when the free balance is above the max unbonded ratio	The system calculates the difference between the current ratio and the max ratio and triggers bondings as necessary.
The protocol adjusts upward when the free balance is below the minimum unbonded ratio	The system calculates the difference between the target ratio and the current ratio and triggers unbondings as necessary.
Elections	
The user can update their set of nominees	Users can invoke the `nominate` extrinsic to provide a new set of nominees for the next era or `chill` to clear their nominees without unbonding.
The user's vote weight is proportional to their locked balance	The vote weight for a particular nominee by a particular user is equal to the user's amount of bonded LDOT.
The user can increase their locked tokens to increase their vote share	A user can invoke the bond extrinsic to increase their amount of bonded LDOT.
The system is able to handle ties	Ties are handled by the underlying sort functionality.

C. Docker Recommendations

This appendix provides general recommendations regarding the use of Docker. We recommend using the steps listed in "Basic security" and "Limiting container privileges" and avoiding the options present in the "Avoid these options" paragraph. At the end we also describe features that Docker containers are based upon and list additional references to learn about those topics.

Basic security

- Do not add users to the `docker` group. Being in the `docker` group allows users to escalate their privileges to root without authentication.
- Do not run containers as a root user. If user namespaces are not used, the root user within the container is the real root user on the host. Instead, create another user within the Docker image and set the container user with the `USER` instruction in the image's Dockerfile specification. Alternatively, the user and group can also be set by passing in the `--user $UID:$GID` flag to the `docker run` command.
- Do not use the `--privileged` flag. Using this flag allows the process within the container to access all host resources and so hijack the machine.
- Do not mount the Docker daemon socket (usually `/var/run/docker.sock`) into the container. Accessing the Docker daemon socket allows spawning a privileged container to "escape the container" and access any host resources.
- Be careful when mounting volumes from special filesystems such as `/proc` or `/sys` into a container as this may allow gaining information about the host machine or escalating privileges, if the container would have write access into the mounted paths.

Limiting container privileges

- Pass the `--cap-drop=all` flag to the `docker run` command to drop all Linux capabilities and enable only those the process within the container needs by using the `--cap-add=...` flag. Although, be careful here, as adding capabilities may allow the process to escalate its privileges and "escape" the container.
- Pass the `--security-opt=no-new-privileges:true` flag to the `docker run` command to prevent the processes from gaining more privileges.
- Limit resources given for the contained process to prevent potential DoS scenarios.
- Do not use root (uid=0 or gid=0) in the container if it is not needed (use `USER ...` in Dockerfile, or docker run `--user $UID:$GID ...`).
- (Optional) Use user namespaces to limit the user and group ids available in the container to only those that would be mapped from the host to the container.
- (Optional) Adjust the Seccomp and AppArmor profiles to limit access the process can perform even more.

Avoid these options

- `--privileged` - this flag "removes ALL security".
- `--cap-add=all` - adds all Linux capabilities.
- `--security-opt apparmor=unconfined` - disables AppArmor.
- `--security-opt seccomp=unconfined` - disables Seccomp
- `--device-cgroup-rule='a *:* rwm'` - allows accessing all devices (see docs).
- `--pid=host` - use host pid namespace.
- `--uts=host` - use host uts namespace.
- `--network=host` - use host network namespace, which allows accessing all network interfaces available on the host.

Linux features the Docker containers security is based upon

Feature	Description
namespaces	A feature that allows to isolate or limit the view (and so use) of a global system resource. There are various namespaces, each wrapping different resources: pid, network, mount, uts, ipc, user and cgroup. As an example, if a process creates a new PID namespace, it will see itself as PID=1 and won't be able to send signals to processes created in its parent namespace. The namespaces a process belongs to can be seen by listing the `/proc/$PID/ns/` directory (each namespace has its own ID), or, by using the `lsns` tool.
control groups	A mechanism to group processes/tasks into hierarchical groups and allows to meter or limit resources within those groups such as memory, CPU, I/O or network. The cgroups a process belongs to can be read from the `/proc/$PID/cgroup` file. The whole cgroup hierarchy can be seen from the `/sys/fs/cgroup/<cgroup controller or hierarchy>/` directories, assuming the cgroup controllers are mounted there (can be seen with the `mount | grep cgroup` command). There are also two versions of cgroups: cgroups v1 and cgroups v2, though, both of them can and often are used at the same time.
Linux capabilities	A feature that splits root privileges into "capabilities". Although this setting is more related to what a privileged user can do, there are different process capability sets and some of them are used for calculating the effective capabilities, e.g. after running a suid binary. Given all this, dropping all Linux capabilities from all capability sets

	helps prevent gaining more privileges for a process through e.g. suid binaries. The process Linux capability sets can be seen for a given process by reading the `CapInh`, `CapPrm`, `CapEff`, `CapBnd` and `CapAmb` values (which corresponds to (inherited, permitted, effective, bound and ambient capability sets) present in the `/proc/$PID/status` file. Those values can be decoded into meaningful capabilities names with the `capsh --decode=$VALUE` tool.
No New Privileges flag	Enabling this flag for a process prevents it from the user who launched the process from gaining more privileges through e.g. suid binaries.
Seccomp BPF syscall filtering	Seccomp BPF allows to limit syscalls and filter their arguments for a given process by writing and a "BPF program" that is later run in the kernel. Since the seccomp interface may not be convenient for users, we recommend using a sandboxing tool such as nsjail which allows specifying seccomp rules in a nice way. The Docker default Seccomp policy can be found here.
AppArmor LSM	AppArmor is a Linux Security Module that allows to limit access to certain resources via Mandatory Access Control. AppArmor profiles are loaded into the kernel. A profile can be in either "complain" or "enforce" mode. In the "complain" mode violation attempts are only logged into syslog and in "enforce" mode such attempts are blocked. To see which profiles are loaded into the kernel use the `aa-status` tool. To see if a given process works under the rules of an AppArmor profile read the `/proc/$PID/attr/current` file. If AppArmor is not enabled for the process, this file contains a value of "`unconfined`". Otherwise, it will return the name of the policy and its mode, e.g. "`docker-default (enforce)`". The Docker AppArmor profile template can be found here, or, in its generated form here.

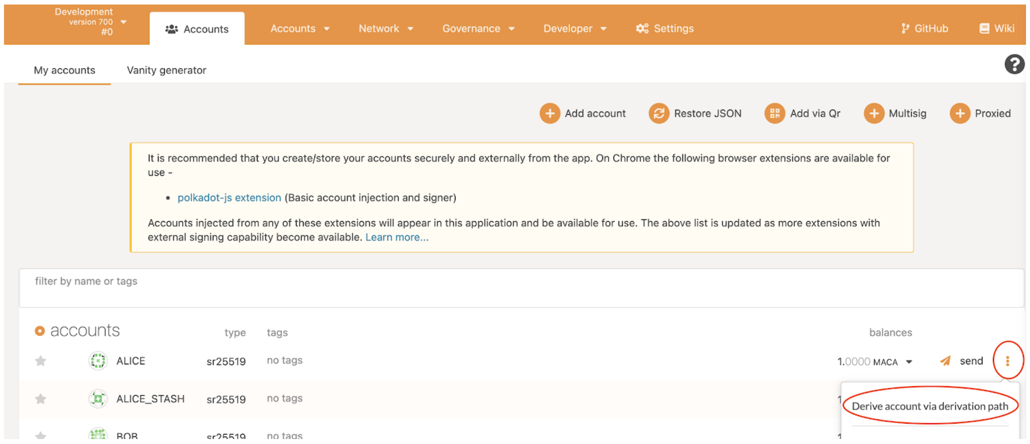
Additional references

- "Understanding Docker container escapes" - our blog post that breaks down a container escape technique and shows what constraints needs to be met for the technique to work.
- "Namespaces in operation, part 1: namespaces overview" - a LWN article consisting of 7 parts which is an overview of Linux namespaces feature.
- "False Boundaries and Arbitrary Code Execution" - an old, but thorough post about Linux capabilities and possible privilege escalations through them.

D. Using test accounts on a local Acala node with a local Acala-dapp build

This appendix describes the steps to be taken in order to test Acala node with a local Acala-dapp build and e.g. play with the blockchain test accounts (Alice, Bob, etc.). Those steps can be adapted as steps for developers to develop Acala blockchain locally, as described in TOB-ACA-006.

1. Build and run Acala node:
    a. Set a default compiler, similarly as it's done in Acala's Dockerfile:
    ```
    rustup default nightly-2020-11-16
    rustup target add wasm32-unknown-unknown --toolchain nightly-2020-11-16
    ```
    b. Build and run Acala:
    ```
    make init && make run
    ```
2. Find a corresponding Acala-dapp version. This step is not straightforward. The two versions should use the same types definitions. E.g. Acala-dapp bdb2409 corresponds to Acala da5aeeaf.
3. Build and run Acala-dapp:
    ```
    yarn install && yarn run start:dapp
    ```
4. Now, to use test accounts (Alice, Bob, etc.) in Acala-dapp, they have to be imported to a polkadot.js wallet extension. To do this, one can do:
    a. Launch a Polkadot UI docker container:
    ```
    docker run --rm -it --name polkadot-ui -e WS_URL=ws://localhost:9944 -p 80:80 jacogr/polkadot-js-apps:latest
    ```
    b. Go to the Polkadot UI's account tab and import all of the accounts into JSON files by using the "Derive account via derivation path" option as shown below. Leave the "derivation path" empty and fill in the account's password.

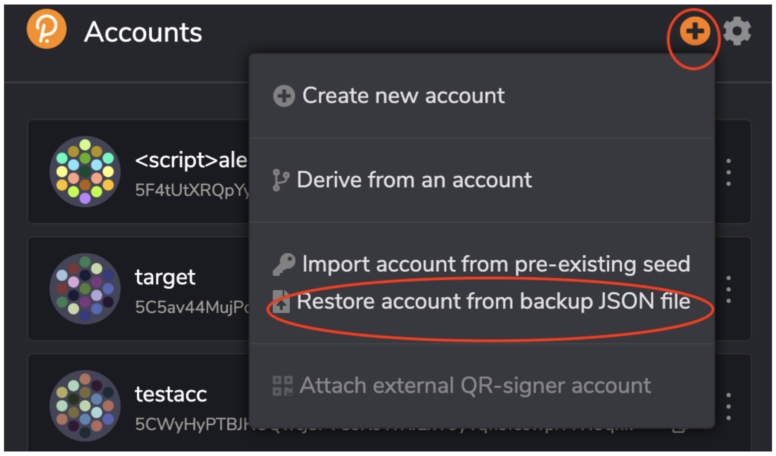


It is recommended that you create/store your accounts securely and externally from the app. On Chrome the following browser extensions are available for use -

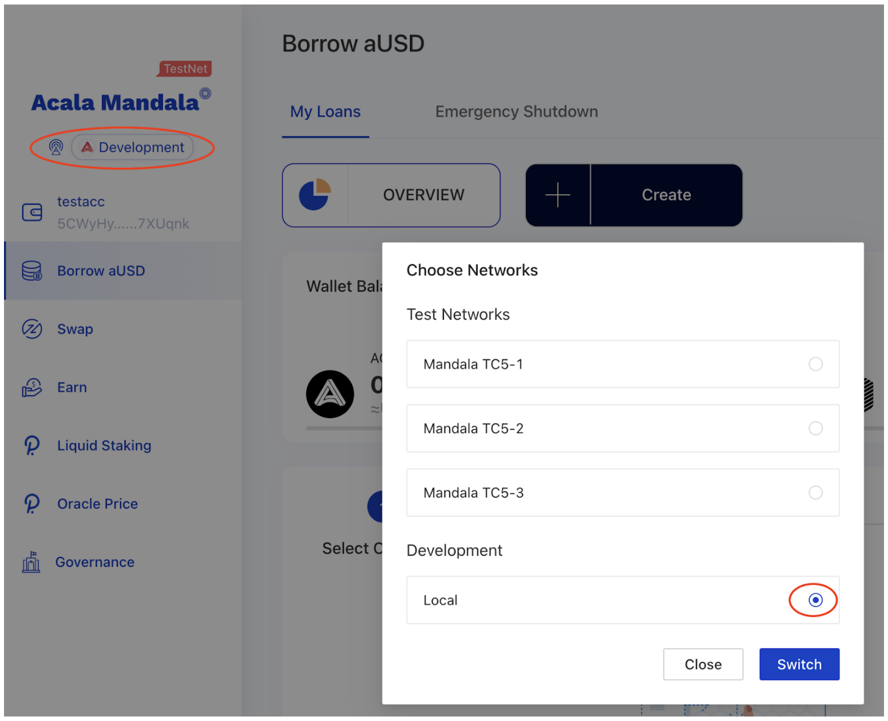
- polkadot-js extension (Basic account injection and signer)

Accounts injected from any of these extensions will appear in this application and be available for use. The above list is updated as more extensions with external signing capability become available. Learn more...

    c. Export the imported testing accounts JSON files into the polkadot.js wallet extension by using its "Restore account from backup JSON file" option, as shown below.


Accounts
Create new account
Derive from an account
Import account from pre-existing seed
Restore account from backup JSON file
Attach external QR-signer account
<script>ale
5F4tUtXRQpYy
target
5C5av44MujPc
testacc
5CWyHyPTBJH

5. Now, change the network in the Acala-dapp by using the button below the "Acala Mandala" heading, as shown below and synchronize polkadot.js wallet extension with the Acala-dapp instance. From now on, the test accounts can be used in the Acala-dapp to play with local instances of Acala blockchain and Acala-dapp.



6. [Optional] When needed to purge chain information, use the `make restart` command in Acala.

E. Code Quality Recommendations

The following recommendations are not associated with specific vulnerabilities. However, they enhance code readability and may prevent the introduction of vulnerabilities in the future.

Acala:

- Fix the typo in the `query_qtorage_deposit_per_byte` function name to `query_storage_deposit_per_byte`.
- Consider changing the `AccountNotFound` event name to `EvmAccountNotFound`. This may help users realise which account the event is related to.
- Fix the "polkdaot" typos in the comments in the `rebalance` function.
- Consider refactoring the acala, kurara and mandala chain specifications so it is easier to track the differences between them.

Acala-dapp:

- Consider using a proper number of days instead of 365 when calculating or displaying Annual Percentage Rate (APR) in the `StakingOverview` and in the `calcStableFeeAPR` function.

Documentation:

- Fix the `</br>` tags in the Acala Runtime Events documentation page.