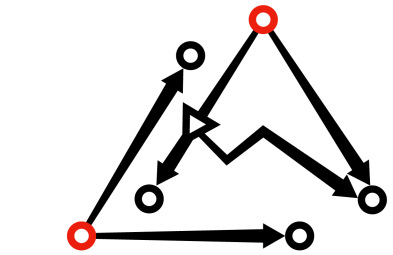

Least Authority

PRIVACY MATTERS

Wrap Protocol Smart Contracts

Security Audit Report

Tezos Foundation

Final Report Version: 24 May 2021

Table of Contents

This audit makes no statements or warranties and is for discussion purposes only.

Overview

Background

Tezos Foundation has requested that Least Authority perform a security audit of the Wrap Protocol Smart Contracts. Wrap is a bridge that allows users to wrap ERC20 tokens into FA2 tokens on the Tezos Blockchain.

Project Dates

- **March 8 - April 2**: Code review *(Completed)*
- **April 8**: Delivery of Initial Audit Report *(Completed)*
- **May 20 - 21:** Verification *(Completed)*
- **May 24:** Delivery of Final Audit Report *(Completed)*

Review Team

- Anna Kaplan, Cryptography Researcher and Engineer
- Mirco Richter, Cryptography Researcher and Engineer
- Nathan Ginnever, Security Researcher and Engineer

Coverage

Target Code and Revision

For this audit, we performed research, investigation, and review of the Wrap Protocol Smart Contracts followed by issue reporting, along with mitigation and remediation instructions outlined in this report.

The following code repositories are considered in-scope for the review:

- Tezos Wrap Contracts: https://github.com/bender-labs/wrap-tz-contracts
- ETH Wrap Contracts: https://github.com/bender-labs/wrap-eth-contract

Specifically, we examined the Git revisions for our initial review:

Tezos Wrap Contracts: `1e972173e672b360825e3fa08c4fca08373c1c98`

ETH Wrap Contracts: `A916b1b3c07e290df14842afa57f6eeaee4591dd`

For the verification, we examined the Git revision:

Tezos Wrap Contracts: `162fb81c69107ab49ea918a08045bbb267644e56`

ETH Wrap Contracts: `72540a361b763b72265b79efb9ae86377a215fb1`

For the review, these repositories were cloned for use during the audit and for reference in this report:

https://github.com/LeastAuthority/wrap-tz-contracts

https://github.com/LeastAuthority/wrap-eth-contracts

This audit makes no statements or warranties and is for discussion purposes only.

All file references in this document use Unix-style paths relative to the project's root directory.

In addition, any dependency and third party code, unless specifically mentioned as in-scope, were considered out of scope for this review.

Supporting Documentation

The following documentation was available to the review team:

- Wrap Protocol Whitepaper: https://github.com/bender-labs/docs/blob/main/Wrap%20Protocol%20-%20Whitepaper.pdf
- Tezos Contracts Wiki: https://github.com/bender-labs/wrap-tz-contracts/wiki
- R. Belchior, A. Vasconcelos,S. Guerreiro, M. Correia, 2021, "A Survey on Blockchain Interoperability: Past, Present, and Future Trends." r arXiv:2005.14282v3 2021[BVG+21]
- FA2 Standard (TZIP-12): https://gitlab.com/tzip/tzip/-/blob/master/proposals/tzip-12/tzip-12.md

Areas of Concern

Our investigation focused on the following areas:

- Correctness of the implementation;
- Adversarial actions and other attacks on the smart contracts;
- Potential misuse and gaming of the smart contracts;
- Attacks that impacts funds, such as the draining or the manipulation of funds;
- Mismanagement of funds via transactions;
- Denial of Service (DoS) and security exploits that would impact the smart contracts intended use or disrupt the execution;
- Vulnerabilities in the smart contracts code;
- Protection against malicious attacks and other ways to exploit smart contracts;
- Inappropriate permissions and excess authority;
- Data privacy, data leaking, and information integrity; and
- Anything else as identified during the initial analysis phase.

Findings

General Comments

The Wrap Protocol implements a bridge between the Ethereum and Tezos blockchains by wrapping and unwrapping ERC-20 tokens into FA2 tokens. Specifically, it implements a *Federated Two-Way Pegs* approach [BVG+C21] to interchain interoperability where a trusted third-party group of signers (i.e. the quorum of validators) is responsible for locking and unlocking funds using simple multi-signature smart contracts on both blockchains.

System Design

The Wrap Protocol contracts consist of two repositories representing the Ethereum and Tezos system components: `wrap-eth-contracts` and `wrap-tz-contracts`, respectively. The smart contracts in `wrap-eth-contracts` are written in Solidity and provide the functionality of the protocol on the Ethereum side of the bridge. The smart contracts in `wrap-tz-contracts` are written in Ligo and provide the quorum, minter, and FA2 token functionality of the Wrap Protocol on the Tezos blockchain.

This audit makes no statements or warranties and is for discussion purposes only.

The wrapping and unwrapping of transactions in `wrap-eth-contracts` is administered through `WrapMultisig.sol`, a deposit smart contract that implements a Gnosis Safe Contracts based multi-signature wallet that allows the quorum to sign the wrapping and unwrapping transactions on the Ethereum blockchain. We carefully examined the implementations for wrapping and unwrapping of ERC-20 tokens and the multi-signature wallet and found them to be correct.

In examining the `wrap-tz-contracts`, we checked that each entry point in the smart contracts functions as intended, as defined in the corresponding specifications available in the project documentation, and found that they are implemented according to the specifications. We investigated and tested for edge cases, including gas cost and data overflow exploits, to which contract to contract interactions are particularly susceptible. In addition, we performed a meta analysis of the entire system and considered possible attack vectors in which the quorum could be compromised through financial incentives.

In analyzing Wrap Protocol's tolerance to signature replay attacks on both the Ethereum and Tezos blockchains, we did not identify any areas of concern. We examined the logic and accuracy of fee ratios and calculations, reviewed permissions and the implemented administration control mechanisms, and verified the correct implementation of the minting and burning functionality of the smart contracts.

Quorum's Centralized Authority

`wrap-tz-contracts` includes a simple multi-signature smart contract, the quorum, which implements the method for calling critical functions in the minter smart contracts, including minting and adding entry points. The minter smart contracts contains the wrapper logic for both wrapping and unwrapping entry points, in addition to providing the logic for collecting wrapping and unwrapping fees. Wrapped tokens are organized in the `FA2/multi-asset` and the `FA2/nft` smart contracts, which are customizations and adaptations of the FA2 Contracts Implementation by TQ Tezos and implement the FA2 standard (TZIP-12). We found the implementation to be correct in its adherence to the specification.

The quorum is an administered smart contract that provides the administrator with the sole authority to change the quorum. Given that a single quorum smart contract can arbitrarily manage a large number of `minter` smart contracts, the quorum has the potential to control a significant amount of value. This design decision introduces a centralization of authority that may potentially result in considerable security vulnerabilities in the context of a malicious administrator.

We recommend that the Bender Labs team explore options to create a more decentralized `quorum`. There are existing mechanisms to incentivize the quorum to behave, however, it would take a minimal amount of collusion to compromise the protocol, which may result in a loss of user funds. We suggest exploring an alternative approach to the quorum, such as utilizing a consensus protocol that will help ensure that signers of the release of tokens on both sides of the bridge are trusted (Suggestion 7).

High Gas Costs

In the event that a signature broadcast service is created to ease the burden on users so that users are not required to wrap and unwrap tokens, an unwrapping process that consumes gas must be performed by the service operators. As a result, an imbalance in gas fees is present as the Tezos fees are currently less expensive and the gas costs of the Ethereum and Tezos blockchains (or any two blockchains seeking interoperability) will be asymmetric. Furthermore, griefing may be introduced in the swap of Tezos tokens to Ethereum tokens if the fees were subsidized for the users of the bridge. This can be crafted into an attack against any third-party organization that is providing the service of signature broadcast for unwrapping on the Ethereum blockchain. As a result, we recommend the Bender Labs team explore opportunities to reduce gas costs for users by creating a service where the quorum subsidizes the costs, enhancing the user experience and creating reasonable rate limits to avoid exploitation (Suggestion 6).

This audit makes no statements or warranties and is for discussion purposes only.

Code Quality

In our review of the Wrap Protocol smart contracts, we found the code contained in `wrap-eth-contracts` and `wrap-tz-contracts` to be well written and organized. Given that Tezos is still a rapidly developing ecosystem with ongoing research and development, certain tools (e.g. linters, static analysis, etc.) have not been established for use in security review and analysis. We recommend that development teams in the Tezos ecosystem continue to contribute to the standardization of best practices and subsequently adopt programming standards, tools, and best practices into the development lifecycle as the ecosystem matures. Doing so will help to further reduce the risk of unexpected vulnerabilities.

In the `wrap-eth-contracts`, the Solidity code adhered to basic linting rules, facilitating an easier review and understanding of the code. The Bender Labs team has adapted the Gnosis Smart Contracts, a well known and audited standard, to implement safe transfer, safe math, and domain separation in accordance with best practices.

Tests

In examining test coverage, we found that both `wrap-tz-contracts` and `wrap-eth-contracts` include test coverage for the intended scenarios and common failure cases. However, the existing tests are not exhaustive in that they do not comprehensively capture all failure scenarios, which would aid in identifying potential edge cases and errors. In addition, executing tests for failure cases can help determine if error conditions operate and catch problems as expected. We recommend expanding test coverage to include all error cases for both `wrap-eth-contracts` and `wrap-tz-contracts` (Suggestion 2).

Documentation

In our review of the Wrap Protocol, we found the documentation provided for `wrap-tz-contracts` to be sufficient, providing helpful definitions and explanations of the properties of the smart contracts, in addition to informative diagrams that describe the general system design, functionality, and interactions.

However, `wrap-eth-contracts` contains no documentation, which is a hindrance to understanding the intended functionality of the system. We recommend creating documentation in order to facilitate more efficient understanding of the intended functionality of the system, which will enable users and security researchers to better identify vulnerabilities and make more informed security decisions. In addition, this will supplement developers' understanding of the implementation and minimize the potential for unintended errors (Suggestion 1).

Code Comments

We found minimal code comments stating the intended functionality within `wrap-eth-contracts`. As a result, we recommend creating code comments following NatSpec guidelines to help increase visibility into the intended functionality of the code, which is beneficial for users, security researchers, and the overall security of the system. While `wrap-tz-contracts` contain no comments, the code base is sufficient without their inclusion for users familiar with LIGO languages (Suggestion 1).

Scope

We found that the scope of the audit to be sufficient for both components of the Wrap Protocol Smart Contracts. `wrap-eth-contracts` and `wrap-tz-contracts` are independent, self-contained, and do not rely on external dependencies.

This audit makes no statements or warranties and is for discussion purposes only.

We recommend that any updates to the smart contract system be followed up with a security audit. Changes to the existing protocol and implementation (e.g. the decentralization of the quorum and provision of a consensus protocol recommended in Suggestion 7) would likely introduce new security and economic considerations. As a result, we suggest that such changes to the system be thoroughly analyzed and reviewed by an independent team for potential security vulnerabilities.

Specific Issues & Suggestions

We list the issues and suggestions found during the review, in the order we reported them. In most cases, remediation of an issue is preferable, but mitigation is suggested as another option for cases where a trade-off could be required.

ISSUE / SUGGESTION	STATUS
Issue A: Fee Distribution Bug [Found by Wrap Protocol]	Resolved
Suggestion 1: Improve Documentation	Partially Resolved
Suggestion 2: Increase Test Coverage	Resolved
Suggestion 3: Consider a Two-Step Process to Update the Administrator in FA2	Resolved
Suggestion 4: Update Solidity Compiler Version	Resolved
Suggestion 5: Use Standardized Elliptical Curve Digital Signature Algorithm Library	Resolved
Suggestion 6: Explore Opportunities to Reduce Gas Costs for Users	Unresolved
Suggestion 7: Expand the Protocol For a More Decentralized Quorum Replacement	Unresolved

Issue A: Fee Distribution Bug [Identified by Wrap Protocol]

Location

`ligo/minter/fees.mligo#L48`

Synopsis

In the method `generate_tx_destitinations` in the minter contract, there is a fold on the token list to generate the transfer and update the internal ledger accordingly.

However, the folder function was updating the ledger on the ledger instance in the closure context, instead of doing it on the accumulated ledger given as a parameter as intended. As a result, only the balance of the last token visited was updated.

Impact

A signer could have claimed its reward several times, until the actual minter balance in the FA2 token smart contract was depleted.

This audit makes no statements or warranties and is for discussion purposes only.

Remediation

This issue could have been avoided with a less naive unit test (i.e. testing for several tokens at once instead of testing for only one token), which was the case for other folds elsewhere in the smart contract.

Additionally, a linter could have detected that the accumulator was not used in the folder function. However, such tools do not currently exist in the Tezos ecosystem.

The issue can thus be resolved by fixing the coding error.

Status

The Bender Labs team has corrected this simple coding bug, thus resolving this issue.

Verification

Resolved.

Suggestions

Suggestion 1: Improve Documentation

Location

`wrap-eth-contracts`

Wrap Protocol Whitepaper

Synopsis

Project Documentation

`wrap-eth-contracts` contains no documentation, which limits the knowledge and understanding of the intended functionality of the system for both users and reviewers.

Furthermore, the Wrap Protocol Whitepaper is missing details from the Wiki Documentation and lacks technical details for the Ethereum related areas of the system.

Project documentation enables users and security researchers to better identify vulnerabilities, make more informed security decisions, and facilitates a more efficient understanding of the intended functionality of the system. As a result, robust documentation minimizes the potential for unintended errors and can prevent misunderstandings in the future development process.

Code Comments

We found minimal code comments stating the intended functionality within `wrap-eth-contracts`, which help increase visibility into the intended functionality of the code which is beneficial for users, security researchers, and the overall security of the system.

Mitigation

Project Documentation

We recommend creating documentation for `wrap-eth-contracts`, providing helpful definitions and explanations of the properties of the smart contracts, in addition to informative diagrams that describe the general system design, functionality, and interactions.

This audit makes no statements or warranties and is for discussion purposes only.

We also recommend that the Wrap Protocol Whitepaper be updated to include information from the Wiki, in addition to information on the Ethereum related areas of the system.

Code Comments

We recommend creating code comments following NatSpec guidelines.

Status

Project Documentation

The Bender Labs team added documentation to the Ethereum repository Wiki. However, the Whitepaper has not yet been updated and, as a result, this component is partially resolved.

Code Comments

The Bender Labs team added code comments to `wrap-eth-contracts`.

Verification

Project Documentation

Partially Resolved.

Code Comments

Resolved.

Suggestion 2: Increase Test Coverage

Location

`wrap-tz-contracts`

`wrap-eth-contracts`

Synopsis

Test coverage for both `wrap-tz-contracts` and `wrap-eth-contracts` includes success cases and common failure cases. However, the existing tests are not exhaustive in that they do not comprehensively capture all failure scenarios, which may lead to potentially missing edge cases and errors. For example, in `wrap-eth-contracts` tests for the behavior of the Multi-Signature management when a 0 threshold is supplied are not present. Executing tests for failure cases can help determine if error conditions operate and catch problems as expected.

Mitigation

We recommend expanding test coverage to include all error cases for both `wrap-eth-contracts` and `wrap-tz-contracts`.

Status

The Bender Labs team improved test coverage for both `wrap-tz-contracts` and `wrap-eth-contracts`. Additionally, more extensive failure case scenarios are covered through new tests in `wrap-eth-contracts`.

Verification

Resolved.

This audit makes no statements or warranties and is for discussion purposes only.

Suggestion 3: Consider a Two-Step Process to Update the Administrator in FA2

Location

`ligo/fa2`

Synopsis

While it might be advantageous to disable the administrator of the quorum and the minter smart contracts, there are no safeguards against typos in the `set_administrator` entry point of the associated FA2 smart contract. If the administrator is changed to an invalid or unknown address, large parts of the contract become essentially useless. The `set_administrator` entry point in the FA2 smart contract does not double check the validity of the new administrator. If a typo or uncontrolled address is inserted, the full functionality of the contract is permanently lost. The Least Authority team understands that extended features such as address validity checks are gas-expensive, however, it might be beneficial nonetheless at a critical point such as this one.

Mitigation

We recommend the Wrap Protocol consider creating a two-step process that would first propose an update to the administrator of the contract, followed by a second transaction that has to be sent from the new administrator's address. Once the transaction is confirmed, the proposed address is accepted as the new administrator. This would catch any updates to incorrect or malformed state.

Status

The Bender Labs team implemented the recommended two-step update process for the minter smart contract administrator.

Verification

Resolved.

Suggestion 4: Update Solidity Compiler Version

Location

`wrap-eth-contracts`

Synopsis

The Solidity compiler version used in `wrap-eth-contracts` is between v0.6.0 and v0.7.0 which is considered to be out of date at this time. An outdated compiler version does not incorporate newer compiler fixes and updates.

Mitigation

We recommend the Bender Labs team update the compiler version in `wrap-eth-contracts` to v0.7.0 and as high as v0.8.0.

Status

The Bender Labs team updated the compiler version in accordance with the recommended mitigation.

Verification

Resolved.

This audit makes no statements or warranties and is for discussion purposes only.

Suggestion 5: Use Standardized Elliptical Curve Digital Signature Algorithm Library

Location

`main/contracts/WrapMultisig.sol#L152-L201`

Synopsis

The `wrap-eth-contracts` use a custom ECDSA signature recovery method that is based on outdated implementations. While the implementation does not appear to be insecure, there are standard libraries available that should be implemented instead. Customization introduces the potential for implementation errors that may result in serious vulnerabilities as opposed to the use of standardized, trusted, and audited alternatives.

Mitigation

We recommend the Bender Labs team use the OpenZeppelin library instead of the custom ECDSA signature recovery method that is based on outdated implementations.

Status

The Bender Labs team implemented the OpenZeppelin library for signature recovery.

Verification

Resolved.

Suggestion 6: Explore Opportunities to Reduce Gas Costs for Users

Location

`main/contracts/WrapMultisig.sol#L115`

Synopsis

In the event that a signature broadcast service is created to ease the burden on users so that users are not required to wrap and unwrap tokens, an unwrapping process that consumes gas must be performed by the service operators. The ability for any individual to collect signatures and broadcast them on-chain shifts the burden to the user base. This also creates a useability issue for those that want to use the bridge by putting the burden of cost on the user. Additional fees associated with the $WRAP token will further exacerbate user costs. The cost burden on users is the data needed to be placed on-chain in the signatures, transfers, and any computation to check signatures in `execTransaction()`.

The gas costs of the Ethereum and Tezos chains (or any two chains seeking interoperability) will be asymmetric. This imbalance in gas fees is present in Tezos as the fees are currently less expensive than gas fees on Ethereum. In the event that fees are subsidized for the users of the bridge, this may cause griefing in the swap of Tezos tokens to Ethereum tokens. Furthermore, this can be crafted into an attack against any third-party organization that is providing the service of signature broadcast for unwrapping on the Ethereum chain. If this service is not provided, then users will experience a minimum value that may be transferred to Tezos, with the knowledge that the cost to move back to Ethereum will make the bridging uneconomical.

The usability of the bridge for users will be impacted by gas costs. If a service is created, funds of any third-party providing this service for users may be lost to grief attacks if not handled carefully. Previously, a third party was providing a service to help users broadcast quorum signatures for the unwrapping of

This audit makes no statements or warranties and is for discussion purposes only.

Ethereum tokens, which has been an issue in the xDai bridge and is likely to occur in practice if a service is created.

Mitigation

We recommend the Bender Labs team explore opportunities to reduce gas costs for users by creating a service for users where the quorum subsidizes the costs, which will improve the user experience and provide reasonable rate limits to avoid exploitation.

Status

The Bender Labs team provided the following response:

> *"The two alternative designs were:*
>
> 1. *Have Bender Labs or another entity pay for gas fees for all users*
> 2. *Have the Signer's Quorum pay for gas fees for all users*
>
> *Both alternatives are not sustainable as they put a limitless financial burden on one or several entities. For both these alternatives, if gas costs become too high in aggregate (which is the case if a lot of users use Wrap - ultimately our goal) then the entities have no choice but to stop operating. We make the case that having users pay for gas is the only sustainable and scalable option for Wrap Protocol."*

We acknowledge the Bender Labs team's response, however, we recommend reconsidering the possibility of providing reasonable rate limits to avoid exploitation.

Verification

Unresolved.

Suggestion 7: Expand the Protocol For a More Decentralized Quorum Replacement

Synopsis

The Tezos Wrap Protocol uses a federated quorum of signers (i.e. an `n` out of `m` multisignature approach) as the consensus model for cross chain bridging of Ethereum tokens. This is an inherently centralized approach for ensuring that the signers will release Ethereum tokens and mint Tezos tokens correctly. A minimal amount of collusion is required in order to compromise this protocol and may cause harm to any users that use the bridge.

In particular, the impact would be significant in the event that significant value is bridged, resulting in a loss of all user funds by theft from the quorum. At least `n` out of the full `m` quorum members would need to collude, however, this is highly feasible as only `n` out of `m` key holders need to make a choice to behave arbitrarily. An existing mechanism is in place to collect fees as a quorum member. The Wrap Protocol Whitepaper states that this mechanism, along with the addition of a bonded stake, will incentivise the quorum to behave. The amount of fees collected would need to be larger than the Total Value Locked (TVL). There is no code to place a stake or bond on the quorum in the current implementation.

Mitigation

Some projects (e.g Polkadot) have created a consensus protocol similar to Bitcoin, that ensures that the signers of the release of tokens behave correctly on both blockchains of the bridge. We acknowledge that this is an expensive option for bridging, as maintaining a blockchain to ensure security would require significant resources. However, we recommend that this approach be considered.

This audit makes no statements or warranties and is for discussion purposes only.

We recommend that the Bender Labs team consider decentralized options and implement one that is best suited to their overall strategy.

Status

The Bender Labs team provided the following response:

> *"In the current implementation of Wrap Protocol, the Signers Quorum is made up of 5 entities which are clearly identified and active in the Tezos community. The Quorum relies on a 3-of-5 multisignature governance, which means that users have to put some trust in the hands of the Quorum.*
>
> *The current quorum members are: Bender Labs, Baking Bad, Bake N Rolls, Blockscale, Madfish. We are working on a more decentralized design for the Quorum, allowing anyone to become a Quorum member provided that some conditions are respected."*

We acknowledge the Bender Labs team's stated intent and ongoing efforts to develop a more decentralized design for the Quorum, allowing anyone to become a Quorum member, provided that some conditions are met. However, at present, the security of the Wrap Protocol is based on a centralized trust assumption in the honesty of five entities, which is not a decentralized approach.

We also acknowledge that the implementation of properly decentralized interchain bridges can be a difficult and time consuming effort. Thus, we recommend that the Bender Labs team continue to prioritize the decentralization of the Quorum and, once implemented, have the updated design assessed and verified by an independent security auditing team.

Verification

Unresolved.

This audit makes no statements or warranties and is for discussion purposes only.

About Least Authority

We believe that people have a fundamental right to privacy and that the use of secure solutions enables people to more freely use the Internet and other connected technologies. We provide security consulting services to help others make their solutions more resistant to unauthorized access to data and unintended manipulation of the system. We support teams from the design phase through the production launch and after.

The Least Authority team has skills for reviewing code in C, C++, Python, Haskell, Rust, Node.js, Solidity, Go, and JavaScript for common security vulnerabilities and specific attack vectors. The team has reviewed implementations of cryptographic protocols and distributed system architecture, including in cryptocurrency, blockchains, payments, and smart contracts. Additionally, the team can utilize various tools to scan code and networks and build custom tools as necessary.

Least Authority was formed in 2011 to create and further empower freedom-compatible technologies. We moved the company to Berlin in 2016 and continue to expand our efforts. Although we are a small team, we believe that we can have a significant impact on the world by being transparent and open about the work we do.

For more information about our security consulting, please visit https://leastauthority.com/security-consulting/.

Our Methodology

We like to work with a transparent process and make our reviews a collaborative effort. The goals of our security audits are to improve the quality of systems we review and aim for sufficient remediation to help protect users. The following is the methodology we use in our security audit process.

Manual Code Review

In manually reviewing all of the code, we look for any potential issues with code logic, error handling, protocol and header parsing, cryptographic errors, and random number generators. We also watch for areas where more defensive programming could reduce the risk of future mistakes and speed up future audits. Although our primary focus is on the in-scope code, we examine dependency code and behavior when it is relevant to a particular line of investigation.

Vulnerability Analysis

Our audit techniques included manual code analysis, user interface interaction, and whitebox penetration testing. We look at the project's web site to get a high level understanding of what functionality the software under review provides. We then meet with the developers to gain an appreciation of their vision of the software. We install and use the relevant software, exploring the user interactions and roles. While we do this, we brainstorm threat models and attack surfaces. We read design documentation, review other audit results, search for similar projects, examine source code dependencies, skim open issue tickets, and generally investigate details other than the implementation. We hypothesize what vulnerabilities may be present, creating Issue entries, and for each we follow the following Issue Investigation and Remediation process.

Documenting Results

We follow a conservative, transparent process for analyzing potential security vulnerabilities and seeing them through successful remediation. Whenever a potential issue is discovered, we immediately create

an Issue entry for it in this document, even though we have not yet verified the feasibility and impact of the issue. This process is conservative because we document our suspicions early even if they are later shown to not represent exploitable vulnerabilities. We generally follow a process of first documenting the suspicion with unresolved questions, then confirming the issue through code analysis, live experimentation, or automated tests. Code analysis is the most tentative, and we strive to provide test code, log captures, or screenshots demonstrating our confirmation. After this we analyze the feasibility of an attack in a live system.

Suggested Solutions

We search for immediate mitigations that live deployments can take, and finally we suggest the requirements for remediation engineering for future releases. The mitigation and remediation recommendations should be scrutinized by the developers and deployment engineers, and successful mitigation and remediation is an ongoing collaborative process after we deliver our report, and before the details are made public.

Responsible Disclosure

Before our report or any details about our findings and suggested solutions are made public, we like to work with your team to find reasonable outcomes that can be addressed as soon as possible without an overly negative impact on pre-existing plans. Although the handling of issues must be done on a case-by-case basis, we always like to agree on a timeline for resolution that balances the impact on the users and the needs of your project team. We take this agreed timeline into account before publishing any reports to avoid the necessity for full disclosure.

This audit makes no statements or warranties and is for discussion purposes only.