Archetype FA2 smart contract templates

Tezos Foundation

Independent security assessment report

inference

Report version: 1.0 / date: 09.01.2023

inference

Table of contents

Version / Date	Description
1.0 / 09.01.2023	Final version

Summary

Inference AG was engaged by Tezos Foundation to perform an independent security assessment of the Archetype FA2 smart contract templates.

Inference AG performed the security assessment based on the agreed scope, following our approach and activities as outlined in the “Project overview” chapter between 24th October 2022 and 5th January 2023. Feedback from the team behind Archetype was received during the course of the assessment and Inference performed a follow-up assessment.

Based on our scope and our performed activities, our security assessment revealed 1 specification issue rated with a high security rating and 3 low rated security issues. Additionally, different observations were also made which, if resolved appropriately, may improve the quality of the Archetype FA2 smart contract templates.

Based on the activities in our follow-up assessment, we were able to state that all of our reported security issues and observations have been resolved by the Archetype team.

Overview on issues and observations

Details for each reported issue or observation can be obtained from the "Security issues" and "Observations" sections.

Row header	Severity / Status
Issues	
S-AFA-001: Gas exhaustion in "user permits"	low / closed
S-AFA-002: FA2 non-compliance	low / closed
S-AFA-003: TZIP-017 non-compliance	low / closed
S-AFA-004: Burn functionality available to token owners	high / closed
Observations	
O-AFA-001: Documentation	- / closed
O-AFA-002: Sending tez to contracts	- / closed
O-AFA-003: tokenId in fungible version	- / closed
O-AFA-004: minting tokens in multi asset version	- / closed

Project overview

Scope

The scope of the security assessment was the following sets of smart contracts:

- FA2 - fungible version: contracts/fa2_fungible.arl
- FA2 - multi asset version: contracts/fa2_multi.arl
- FA2 - NFT version: contracts/fa2_nft.arl
- Permits: contracts/permits.arl

Our initial assessment considered commit "0a28886abed7e00546109bf49e0071ce1a86d8f1" in the source code repository https://github.com/completium/archetype-fa2/

We also considered the documentation available under https://archetype-lang.org/.

Our reassessment considered commit "bedc9e70081426d642cea83174cde5088ba3cf25" and the following compiled smart contracts in Michelson[1] stored at https://github.com/InferenceAG/misc/tree/8f16b49199791583bfad777f9316de58f2c29cf4/fa2-archetype:

- FA2 - fungible version filename: fa2_fungible.tz
- FA2 - multi asset version filename: fa2_multi.tz
- FA2 - NFT version filename: fa2_nft.tz
- Permits filename: permits.tz

Scope limitations

Our security assessment is based on the following key assumptions and scope limitations:

- Any potential adversarial activities conducted by the administrator of the contract or operational errors by administrators were out of scope.
- Content of metadata and token metadata was out of scope. For instance: any off-chain views have not been assessed.
- Assumption that only a handful of consumers will be registered in any permit smart contract.

[1] Compiled using completium-cli version 0.4.63 and archetype version 1.4.0

Methodology

Inference's methodology for smart contract security assessments on Tezos is a combination of a source code review of the smart contract source code in the high-level language (e.g. Ligo or SmartPy), and the resulting compiled code in Michelson. This approach provides additional assurance that the compiler producing the Michelson code is correct, and does not introduce any security issues. Furthermore, this approach fosters a better understanding for smart contract users of what has been reviewed and what has been effectively deployed on-chain.

In order to ensure a high quality in our security assessments, Inference is using subject matter experts having a high adversarial scenario mindset to spot potential issues in smart contracts under review. Additionally, we apply checklists derived from good practices and commonly known issues based on the Tezos smart contract assessment checklist to document our work and to ensure good issue coverage.

Furthermore, Inference maintains regular communications with the smart contract development team to ensure a correct understanding of the smart contract solution and environment, but also to make teams aware of any observations as soon as possible.

Inference's internal quality assurance procedures ensure that results of security assessments are challenged for completeness and appropriateness by a second independent expert.

Objectives

The objectives are the identification of security issues with regards to the assessed smart contracts and their conceptual design and specification. The security assessment also focuses on adversarial scenarios on specific use cases which have been listed in appendix Adversarial scenarios. These were identified together with the Archetype FA2 smart contract template developers and checked during our security assessment.

Activities

Our security assessment activities for the defined scope were:

- Source code review of smart contract code in Archetype

We applied the checklist for smart contract security assessments on Tezos, version 1.1 obtained from https://github.com/InferenceAG/TezosSecurityAssessmentChecklist. We applied the following security checklist tables:

- System / Platform
- Storage
- Gas issues and efficiency
- Code issues
- Transaction
- Entrypoint
- Admin / Operator functions
- Other topics & test cases

Our activities for the follow-up assessment were:

- Source code review of changes in the smart contract code in Archetype
- Source code review of the compiled smart contracts in Michelson
- Reassessing security issues and observation from initial assessment in case they are claimed to be resolved

Security issues

S-AFA-001: Gas exhaustion in “user permits”

Permits associated with a user are stored in a “map” data type structure.

This poses a risk: should the stored permits grow too large, the map with the user’s permits can no longer be successfully loaded when a corresponding entrypoint is executed since loading of the map with all of the user’s permits may consume all available gas.

The map is loaded in the following entrypoints of the permits smart contract: permit, consume, check, and set_expiry.

Probability - Low, since this requires a large amount of permits.
Impact - Low: The permit feature would no longer work for the user who has registered too many permits. Thus, transfers with permits and gasless transfers in the Archetype FA2 smart contracts within the scope of this assessment would no longer work. However, regular FA2 transfers in the Archetype FA2 smart contracts within the scope of this assessment are still possible.
Severity - Low

Recommendation:
We recommend limiting the number of permits per users and enforcing this limit within the permits smart contract.

Comment from the Archetype team:
Fixed by restricting the maximum size of the permit map.

Results from follow-up assessment:
Fixed. Status updated from “open” to “closed”.

S-AFA-002: FA2 non-compliance

The FA2 smart contract templates do not fully adhere to the FA2 standard (TZIP-012):

1. The multi asset version always returns zero ("0") as the balance for any token identifier which is bigger than 0 (TZIP-012) when calling the entrypoint "balance_of".
2. The entrypoint "transfer" in the multi asset version does not fail if a token does not exist (TZIP-012).
3. The NFT version fails if a transfer of zero ("0") tokens is submitted (TZIP-012).
4. The NFT version does not fail with error message "FA2_TOKEN_UNDEFINED" if a balance of a non-existing token is requested (TZIP-012) using the entrypoint "balance_of".
5. The single, multi, and NFT versions are throwing the error message "CALLER_NOT_OWNER" instead of "FA2_NOT_OWNER"[2].

This poses a risk that requested transfers fail in certain situations or applications interacting with the Archetype FA2 smart contracts.

Probability - Low.
Impact - Low.
Severity - Low.

Recommendation:
We recommend analysing the situation and potentially adapting the Archetype smart contract templates in order to be compliant with the FA2 standard (TZIP-012).

Comment from the Archetype team:
All points fixed.

Results from follow-up assessment:
Fixed. Status updated from "open" to "closed".

[2] The error message "FA2_NOT_OWNER" is not explicitly a MUST according to the TZIP-012. However, we recommend using this error message instead of "CALLER_NOT_OWNER".

S-AFA-003: TZIP-017 non-compliance

The smart contracts do not fully adhere to the permit specification (TZIP-017):

1. The permits smart contract does not fail if a permit already exists (TZIP-017).
2. The entrypoint “set_expiry” in the permits smart contract has a different format (TZIP-017)
3. The entrypoints “transfer_gasless” in the fungible, multi asset, and NFT versions of Archetype smart contract templates have a different name (TZIP-017) and different parameter types (TZIP-017).

This poses a risk that a system relying on the permit interface may not work or work incorrectly.

Probability - Low.
Impact - Low.
Severity - Low.

Recommendation:
We recommend analysing the situation and potentially adapting the smart contracts in order to be compliant with the permit interface standard (TZIP-017).

Comment from the Archetype team:
All points fixed.
Regarding #3, permit_transfer entrypoint is added to the different versions. The “transfer_gasless” entrypoint is kept for legacy reasons.

Results from follow-up assessment:
Fixed. Status updated from “open” to “closed”.

S-AFA-004: Burn functionality available to token owners

The Archetype documentation defines that the burn functionality of the fungible, multi asset, and NFT versions of the Archetype FA2 smart contracts is only available to the smart contract owner.

However, the smart contract code for the burn entrypoints of these versions can be called by the token owner only.

This poses a risk that token owners are destroying their own tokens, which is not in line with the Archetype documentation/specification.

Probability - High.
Impact - Low. However, it depends on the type of FA2 version and tokens.
Severity - High.

Recommendation:
We recommend analysing the situation and potentially adapting the smart contracts and/or the documentation.

Comment from the Archetype team:
Documentation updated.

Results from follow-up assessment:
Fixed. Documentation updated to “burn entrypoint callable by token owner only”. Status updated from “open” to “closed”.

Observations

O-AFA-001: Documentation

Documentation for the smart contracts describing specific risks when adopting Archetype FA2 smart contract templates is missing.

Recommendation:
We recommend providing documentation specific to the contracts within scope in order to appropriately document the following issues:

- Advise users on risks connected with using permits (e.g. risk of guessing registered permits or resubmission of permits, in case they are not executed in order).
- Advise admins/developers about potential problems (for instance, potential gas exhaustion if too many “consumers” are registered or too much token metadata has to be stored on chain).

Comment from the Archetype team:
Documentation updated.

Results from follow-up assessment:
Fixed. Status updated from “open” to “closed”.

O-AFA-002: Sending tez to contracts

Entrypoints which do not expect any tez to be sent do not reject any sent tez. Thus, if tez are mistakenly sent with a transaction to the smart contract then the sent tez are locked in the smart contract since there is no withdrawal function available.

This observation affects all smart contracts in scope.

Recommendation:
We recommend rejecting any sent tez in the case where tez are not expected to be sent to an entrypoint. This should be implemented at the very least for non privileged entrypoints.

Comment from the Archetype team:
Check added.

Results from follow-up assessment:
Fixed. Checked is added at least to all non-privileged entrypoints. Status updated from “open” to “closed”.

O-AFA-003: tokenId in fungible version

The Archetype smart contract template for the fungible version is a single asset FA2 smart contract. Thus, there is only one token with tokenId “0” managed by the fungible version.

However we noted that:

1. The entrypoint “set_token_metadata” requires as a parameter the token identifier and allows defining token metadata for other tokens by the smart contract owner.
2. There exists an entrypoint “update_operators_for_all”, which does not really make sense in a single asset FA smart contract. This functionality is already provided by the standard FA2 entrypoint “update_operators”.

This observation affects only the FA2 smart contract template for the fungible version.

Recommendation:
We recommend

1) removing the token identifier parameter from the entrypoint “set_token_metadata” in the FA2 smart contract template for the fungible version.
2) removing the entrypoint “update_operators_for_all” including it instead in the corresponding big map data structure “operators_for_all”.

Comment from the Archetype team:

1.) Changed.
2.) Removed.

Results from follow-up assessment:
Fixed. Status updated from “open” to “closed”.

O-AFA-004: minting tokens in multi asset version

The Archetype smart contract template for the multi asset version also allows minting tokens for tokens which are not yet defined.

Recommendation:
We recommend only allowing the minting of tokens for defined tokens. A token should first be defined by adding the corresponding token metadata using the entrypoint “set_token_metadata”. If a token does not exist the entrypoint “mint” should fail with the error message “FA2_TOKEN_UNDEFINED”.

Comment from the Archetype team:
Code updated.

Results from follow-up assessment:
Fixed. Status updated from “open” to “closed”.

Disclaimer

This security assessment report ("Report") by Inference AG ("Inference") is solely intended for Tezos Foundation ("Client") with respect to the Report's purpose as agreed by the Client. The Report may not be relied upon by any other party than the Client and may only be distributed to a third party or published with the Client's consent. If the Report is published or distributed by the Client or Inference (with the Client's approval) then it is for information purposes only and Inference does not accept or assume any responsibility or liability for any other purpose or to any other party.

Security assessments of a software or technology cannot uncover all existing vulnerabilities. Even an assessment in which no weaknesses are found is not a guarantee of a secure system. Generally, code assessments enable the discovery of vulnerabilities that were overlooked during development and show areas where additional security measures are necessary. Within the Client's defined time frame and engagement, Inference has performed an assessment in order to discover as many vulnerabilities of the technology or software analysed as possible. The focus of the Report's security assessment was limited to the general items and code parts defined by the Client. The assessment shall reduce risks for the Client but in no way claims any guarantee of security or functionality of the technology or software that Inference agreed to assess. As a result, the Report does not provide any warranty or guarantee regarding the defect-free or vulnerability-free nature of the technology or software analysed.

In addition, the Report only addresses the issues of the system and software at the time the Report was produced. The Client should be aware that blockchain technology and cryptographic assets present a high level of ongoing risk. Given the fact that inherent limitations, errors or failures in any software development process and software product exist, it is possible that even major failures or malfunctions remain undetected by the Report. Inference did not assess the underlying third party infrastructure which adds further risks. Inference relied on the correct performance and execution of the included third party technology itself.

Appendix

Adversarial scenarios

The following adversarial scenarios have been identified together with Archetype FA2 smart contract template developers and checked during our security assessment.

Scenario	Impact rating & descriptive	Assessment result
Permit Submission of a permit allowing you to make a transfer on behalf of somebody else.	High: Loss of assets. Loss of trust/reputation.	**Ok** Nothing identified.
Permit Guessing permits in order to execute a transfer earlier than intended or not at all.	Medium: Unwanted or transfers executed too early.	**Note** Transfer permits can be guessed**.**
Permit Preventing a permit from being registered.	Low: No transfers via permit possible	**Note** Permit counter is increased with every submitted permit or gasless transfer. Thus, a permit or gasless transfer may not be registered or executed if the counter has already been increased in the meantime.
Permit A single permit can be used multiple times.	High: Loss of assets. Loss of trust/reputation.	**Ok** Not possible, since the permit is removed once used.
Permit Using the permit feature for other functions than transfers.	Unknown: Depends on the function.	**Ok** Permit feature can only be used for transfers.

Risk rating definition for smart contracts

Severities are quantified with two dimensions, roughly defined as follows, whereas the examples have to be regarded as indication only:

Probability of occurrence / materialisation of an issue
(bullets for a category are linked with each other with “and/or” condition.)

- Low:
  - A trusted / privileged role is required.
  - Contract may end up in the issue if other conditions, which are also unlikely to happen, are required.
- Medium:
  - A specific role or contract state is required to trigger the issue.
  - Contract may end up in the issue if another condition is fulfilled as well.
- High:
  - Anybody can trigger the issue.
  - Contract’s state will over the short or long term end up in the issue.

Impact:
(bullets for a category are linked with each other with “and/or” condition.)

- Low:
  - Non-compliance with TZIP standards
  - Unclear error messages
  - Confusing structures
- Medium:
  - A minor amount of assets can be withdrawn or destroyed.
- High:
  - Not inline with the specification
  - A non-minor amount of assets can be withdrawn or destroyed.
  - Entire or part of the contract becomes unusable.

Severity:

	Low impact	Medium impact	High impact
High probability	**High**	**Critical**	**Critical**
Medium probability	**Medium**	**High**	**Critical**
Low probability	**Low**	**Medium**	**High**

Glossary

Term	Description
Archetype	High level smart contract language. Website: https://archetype-lang.org/
Ligo	High level smart contract language. Website: https://ligolang.org/
Origination	Deployment of a smart contract
SmartPy	High level smart contract language. Website: https://smartpy.io/
TZIP	Tezos Improvement Proposal