Avatars and Permits smart contracts

Team Vitality

Independent security assessment report

inference

Report version: 1.0 / date: 26.10.2022

inference

Table of contents

Version / Date	Description
1.0 / 26.10.2022	Final version

Summary

Inference AG was engaged by Team Vitality to perform an independent security assessment of their Avatars and Permits smart contracts.

Inference AG performed the security assessment based on the agreed scope, following our approach and activities as outlined in the “Project overview” chapter between 14th September 2022 and 22th September 2022. Feedback from Team Vitality 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 one critical and three low rated security issues. Additionally, different observations were also made which, if resolved appropriately, may improve the quality of Team Vitality's Avatars and Permits smart contracts.

Based on our activities in our follow-up assessment, we can confirm that the critical rated security issue has been resolved. Additionally, the FA2 non-compliance issue has been partly resolved.

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-VAP-001: Gas exhaustion in "user permits"	low / open
S-VAP-002: FA2 non-compliance	low / partly resolved
S-VAP-003: TZIP-017 non-compliance	low / open
S-VAP-004: Arbitrary token transfers when using gasless transfers	critical / closed
Observations	
O-VAP-001: Proprietary event logging solution	- / open
O-VAP-002: Documentation	- / open
O-VAP-003: Sending tez to contracts	- / open
O-VAP-004: Testing	- / partly resolved

Project overview

Scope

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

- Avatars
  - Contract written in Archetype: https://gist.githubusercontent.com/theblockchaincoder/c45e842db95ad5af81e29ed5cfee95a7/raw/7e1680bcf41e6c3aba24069d0a86d4d3b48a24af/avatars.arl
  - Compiled contract in Michelson on Tezos ghostnet: KT19jGPB4dMktdx7QU6y1Z9F2vPbzEakJob9
- Permits
  - Contract written in Archetype: https://gist.githubusercontent.com/theblockchaincoder/c45e842db95ad5af81e29ed5cfee95a7/raw/7e1680bcf41e6c3aba24069d0a86d4d3b48a24af/permits.arl
  - Compiled contract in Michelson on Tezos ghostnet: KT1Hsk7pciZL7ywSmosUTAdSywpivxfFtR1X

Our reassessment considered:

- Avatars
  - Contract written in Archetype: https://gist.githubusercontent.com/theblockchaincoder/c45e842db95ad5af81e29ed5cfee95a7/raw/14c80cbddc938cde9d53ee1d7955fa0e819a635d/avatars.arl
- Permits
  - Contract written in Archetype: https://gist.githubusercontent.com/theblockchaincoder/c45e842db95ad5af81e29ed5cfee95a7/raw/14c80cbddc938cde9d53ee1d7955fa0e819a635d/permits.arl

The Michelson code for the following Tezos mainnet contracts were also considered and assessed:

- KT1G5ph5ybHBbAy2hEd5RbPTuGEQuWXMWsBB for Avatars
- KT1PZgwg6HPsQmghvpZjUQZRL3TocuRRTP2K for Permits
- KT19ij2bHXkhMALzoTZCG88FWgAHRR21247v for “event logging sink”

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.
- Any potential adversarial activities conducted by the administrator of the contract or operational errors by administrators were out of scope.
- Key management of associated secret keys has not been assessed.

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 beens listed in appendix Adversarial scenarios. These were identified together with the Team Vitality 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
- Source code review of the deployed smart contract in Michelson

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 changes in the deployed smart contracts in Michelson
- Reassessing security issues and observation from initial assessment in case they are claimed to be resolved

Security issues

S-VAP-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 Avatars smart contract would no longer work. However, regular FA2 transfers in Avatars smart contract are still possible.
Severity - Low

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

Comment from Team Vitality:
Due to the project topology, with social login signup and free drop to empty wallets at 95%, transfers are initiated by marketplace and verified dApps that don’t allow this scenario. Considering the planning, we do not want to take any changes on this one, with an almost null criticity and incidence probability

Results from follow-up assessment:
No changes. Status remains “open”.

S-VAP-002: FA2 non-compliance

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

- The smart contract fails if a transfer of zero ("0") tokens is submitted (TZIP-012)
- The balance_of entrypoints does not fail with error message "FA2_TOKEN_UNDEFINED" if a balance of a non-existing token is requested (TZIP-012).

This poses a risk that requested transfers are failing in certain situations/solutions interacting with the Avatars smart contract.

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

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

Comment from Team Vitality:
Due to the project topology, with social login signup and free drop to empty wallets at 95%, transfers are initiated by marketplace and verified dApps that don't allow this scenario. Considering the planning, we do not want to take any changes on this one, with an almost null criticity and incidence probability.

Results from follow-up assessment:
Zero token amount transfers are now allowed. The other reported points have not been addressed. Status updated from "open" to "partly resolved".

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

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

- The Permits smart contract does not fail, if permit is already existing (TZIP-017)
- The entrypoint “set_expiry” in the Permits smart contract has a different format (TZIP-017)
- The entrypoint “transfer_gasless in the Avatars smart contract has 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 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 Team Vitality:
Due to the low criticality and incidence probability, and planning, we consider it not necessary to take action on this one.

Results from follow-up assessment:
No changes. Status remains “open”.

S-VAP-004: Arbitrary token transfers when using gasless transfers

The entrypoint “transfer_gasless” does not check whether the signer of token transfer is the owner of the tokens to be transferred.

Probability - High.
Impact - High.
Severity - Critical.

Recommendation:
We recommend implementing a check to ensure the signer of the transfers is allowed to transfer the corresponding tokens.

Comment from Team Vitality:
Fixed.

Results from follow-up assessment:
A check has been added. Status updated from “open” to “closed”.

Observations

O-VAP-001: Proprietary event logging solution

The Avatars smart contract has implemented a proprietary event logging solution. Events to be logged are sent as regular transactions to a specific smart contract address (event logging sink).

Recommendation:
We recommend adapting the Tezos native event logging solution instead of using a proprietary event logging solution. Relying on the Tezos native event logging solution should provide broader, better, and long-term tool support.

The Tezos native event logging solution is introduced with the Kathmandu protocol upgrade, which is expected to be activated on Tezos mainnet in the evening / night of the 23th September 2022. Furthermore, the Tezos native event logging solution also requires less gas than the proprietary event logging solution.

Additional information:

- https://tezos.gitlab.io/kathmandu/event.html?highlight=event%20logging

Comment from Team Vitality:
The event logging system is from the Archetype language creators, it is maintained, well tested and used within the tezos ecosystem.

Results from follow-up assessment:
No changes. Status remains “open”.

O-VAP-002: Documentation

Documentation for the smart contracts is missing.

Recommendation:
We recommend adding documentation/specification which at least provides general information about:

- Description what an entrypoint is for and what it should do
- Meaning and types of input parameters / arguments
- Pre conditions (e.g. who can call the entrypoint)
- Post conditions
- Returned operations (transfers, contract origination, etc.)
- Use case and workflow for users to use the smart contract even if the front-end website is not available.

Furthermore, we recommend providing documentation specific for these contracts in order to appropriately document the following:

- Advise users on risks connected with using permits (e.g. risk of guessing registered permits)
- 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 Team Vitality:
Documentation is so far internal and might be added on github in a later state.

Results from follow-up assessment:
No changes. Status remains “open”.

O-VAP-003: 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 both smart contracts in scope.

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

Comment from Team Vitality:
Due to the project topology, with social login signup and free drop to empty wallets at 95%, transfers are initiated by marketplace and verified dApps that don't allow this scenario.

Results from follow-up assessment:
No changes. Status remains "open".

O-VAP-004: Testing

No test cases have been provided to be reviewed in this security assessment.

This observation affects both smart contracts in scope.

Recommendation:
We recommend covering all entrypoints and code with appropriate test cases. Test cases should also include test cases expecting to fail in order to verify whether the contract is correctly working (e.g. access-restricted entrypoints or failing conditions).

Comment from Team Vitality:
Test results will be shared with Inference.

Results from follow-up assessment:
Testing has been performed and results have been shared with Inference. However, we have not assessed the test cases, since they have not been shared and whether they are appropriately defined. Status updated from "open" to "partly resolved".

Disclaimer

This security assessment report ("Report") by Inference AG ("Inference") is solely intended for Team Vitality ("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 Team Vitality 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 too transfers executed 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