

A decorative graphic in the top-left corner consisting of a series of glowing blue dots arranged in a circular, concentric pattern, resembling a fingerprint or a data visualization.

Papers GmbH

Tezos Implementation Review

Document Name:	report_85589_Crypto_Currency_Protocol_Review_Tezos_v1.0.docx
Version:	v1.0
Project Number:	85589
Date of Delivery:	January 24th, 2019
Author:	Lukasz Dykcik, Compass Security Schweiz AG
Classification:	STRICTLY CONFIDENTIAL

Table of Contents

1 OVERVIEW.....	3
1.1 To the Reader.....	3
1.2 Document Structure.....	3
2 OVERALL STATEMENT.....	4
2.1 Goals and Methodology.....	4
2.2 Results.....	4
2.3 Disclaimer.....	4
3 VULNERABILITIES AND REMEDIATION.....	5
3.1 Tezos Tests.....	5
4 TEZOS TESTS.....	7
4.1 Test #1.....	7
4.2 Test #2.....	10
4.3 Test #3.....	13
4.4 Test #4.....	14
4.5 Test #5.....	16
4.6 Test #6.....	18
4.7 Test #7.....	19
4.8 Test #8.....	20
4.9 Test #9.....	21
4.10 Test #10.....	26
4.11 Test #11.....	27
5 APPENDIX.....	29
5.1 Compass Weaknesses Rating.....	29
5.1.1 What the rating IS NOT.....	29
5.1.2 What to do with the weaknesses table.....	29
5.1.3 Examples.....	29
5.1.4 Tests with result "INFO" and N/A.....	29
5.2 Recheck Coloring.....	30

1 Overview

1.1 To the Reader

This document is geared towards project teams, development personnel and other individuals concerned with the security issues of the usage of Tezos cryptocurrency on the AirGap Vault mobile application. The purpose of this document is to summarize the results of the tests performed on the existing security systems using technical terminology. The points pertaining to security issues are listed in chapter 3.

1.2 Document Structure

Chapter	Content
1	Document overview
2	Overall statement explaining the outcome of the security tests
3	A list of the detected weaknesses as well as suggestions for improvement
4	Protocol of the performed security tests
5	Appendix

2 Overall Statement

At the turn of the year 2019 to 2020, Compass Security tested the implementation of the Tezos protocol during a 3-person-day timespan. No critical vulnerabilities have been found, only a few medium-rated security deficiencies. In order to achieve a high security standard, it is nevertheless recommended to address and mitigate the discovered issues.

2.1 Goals and Methodology

The goal of the project was to find vulnerabilities in the implementation of the Tezos protocol, in particular in the newly added possibility in the application to transfer tokens present on the Tezos blockchain. The focus of the tests was put on attack vectors that may result in the user of the application losing his funds.

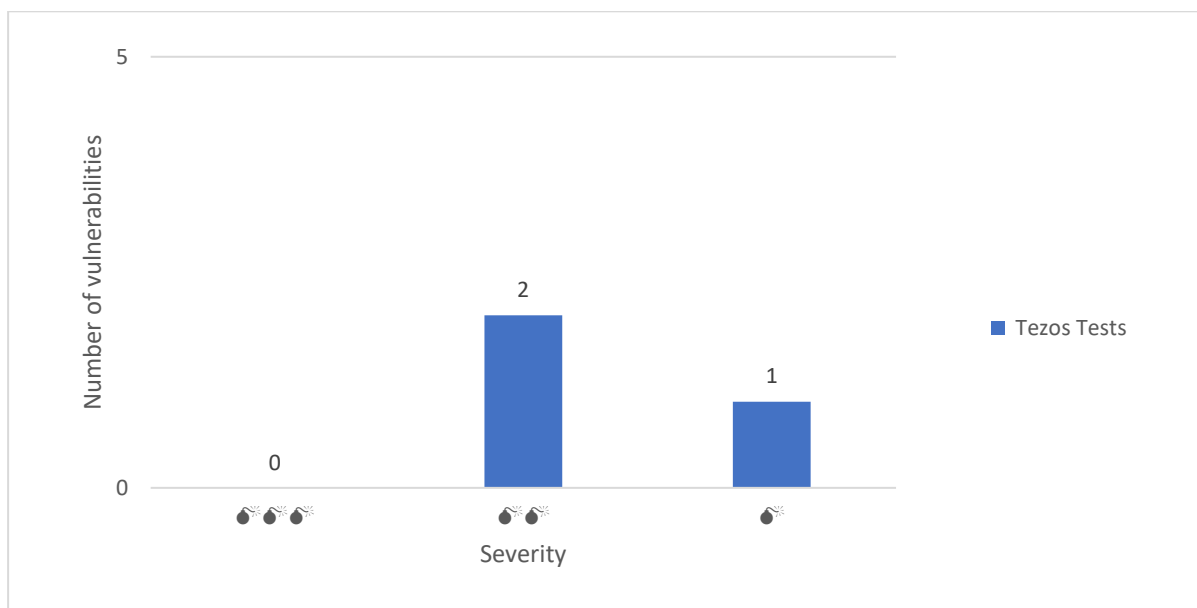
The tests were performed on the Android application v3.0.0 compiled on 20th December 2019. For the tests, the Tezos Babylonnet blockchain was used. During the tests, access to the source code of the application was granted.

2.2 Results

During the tests, no attack vector that would allow an attacker to steal money from a careful user of the AirGap Vault application was found. The application accepts only a single type of potentially untrusted input, that is the serialized transaction to approve. For the Tezos protocol, the serialized transaction contains forged Tezos transaction data. The application unforges the data and presents it in a readable form to the user. Because the data the user sees are derived directly from the forged transaction, the attack surface for an attacker is highly limited since all his malicious inputs have to be first flawlessly unforged.

Nevertheless, a few noncritical security issues were identified. It was possible to create a transaction spending XTZ disguised as a transaction moving Bitcoin tokens. It was also possible to come up with a transaction that is shown as a Bitcoin token transfer when in fact other tokens are spent. A user of the application had no means to notice that the transaction he signs does in fact something different than what is shown in the application. However, it is worth noting, that although it was possible to trick the user into signing a transaction he did not intend to sign, considering the current prices of crypto assets, these attack vectors would not enable an attacker to commit a fraud.

The following diagram gives an overview over the identified vulnerabilities and their severity.



Compass Security recommends addressing all issues listed in the vulnerability table in section 3 according to their rating. Vulnerabilities with a high severity should be addressed as soon as possible. Medium- and low-rated vulnerabilities can be mitigated in the medium term.

2.3 Disclaimer



This statement is applicable to the application as tested during the project. The application may have undergone changes since.

3 Vulnerabilities and Remediation

The tables in this chapter summarize the security issues found during the security review. A definition for each column is given here:

No.	Reference	Weakness	Threat	Remediation	Rating	Comment
Each issue is consecutively numbered.	Reference to the corresponding test case in the following chapters.	Explains the vulnerability identified during the analysis.	Explains what could happen if the weakness is exploited.	Recommendation on how to correct the vulnerability.	Compass rating of the weakness and the corresponding threat:  : Low  : Medium  : High INFO : Not security relevant issue <i>See section 5.1 for detailed description.</i>	

3.1 Tezos Tests

No.	Reference	Weakness	Threat	Remediation	Rating	Comment
1.	4 #1 4 #9	Tezos Transaction Shown as Token Transfer It is possible to create a transaction where an XTZ transfer is shown to the user as a transfer of Bitcoin tokens on the Tezos blockchain.	A user can sign a transaction being convinced he approves transfer of Bitcoin tokens whereas he actually spends XTZ.	The currency of operations shown to the user should always match the actual operations. In particular, the shown currency should not be derived from other fields than the content of the forged transaction.		
2.	4 #4	Token Contract Address Unverified The application accepts any contract address but the user is shown the same information disregarding what tokens he transfers.	A user may by accident sign an operation transferring different tokens although the Bitcoin token transfer is shown in the application.	The application should verify whether the destination contract address matches the expectations. It should also be verified that the operation parameters invoke the expected actions.		

No.	Reference	Weakness	Threat	Remediation	Rating	Comment
3.	4 #3	<p>Misleading Transaction Details</p> <p>The extra transaction details shown to the user do not always represent the actual content of the unforged transaction. In case of a Bitcoin token transfer, the shown destination address, as well as the amount, do not match the real data.</p>	<p>An advanced user who understands details of operations on Tezos blockchain cannot trust the extra transaction details shown in the application. This can undermine the user's trust to the whole application.</p>	<p>In the extra transaction details the user should be able to see the actual unforged transaction data.</p>	<p>●</p>	
4.	4 #8	<p>Nonhexadecimal Characters in Forged Transaction Representation Allowed</p> <p>The application accepts nonhexadecimal characters in the hexadecimal representation of a Tezos forged transaction.</p>	<p>This vulnerability does not constitute a direct threat as a forged transaction with nonhexadecimal characters is unambiguously parsed. The information shown to the user represent exactly the data used to create a signed transaction.</p>	<p>In order to improve the application's resistance against attacks requiring usage of unexpected characters, make sure that only hexadecimal characters are accepted in strings expected to contain hexadecimal characters only.</p>	<p>INFO</p>	

4 Tezos Tests

4.1 Test #1

No.	Description of Test	Expected Result	Actual Result	PASS FAIL
1.	Is it possible to create an XTZ transaction whereas the user sees it as a TZBTC transaction?	No.	Yes, it is possible to trick the user to sign a TZBTC transaction although in fact the transaction transfers XTZ	FAIL

Details #1

The following data is forged, it is a standard transaction sending 1.23 XTZ to

`tz1d75oB6T4zUMexzkr5WscGktZ1NsslJrT7:`

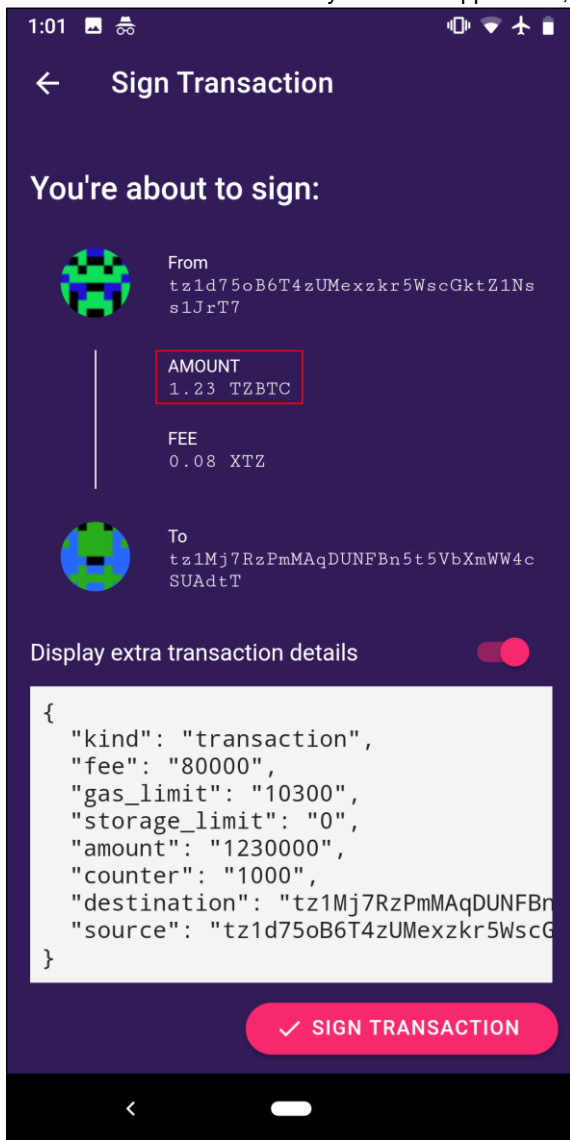
```
{ branch: 'BLty3agyzaeRaNKjEoFCuraBcW2ihvfwSaQQ1v8ntaE8z47Z8YJ',
  contents:
  [ { kind: 'transaction',
    fee: '80000',
    gas_limit: '10300',
    storage_limit: '0',
    amount: '1230000',
    counter: '1000',
    destination: 'tz1Mj7RzPmMAqDUNFBn5t5VbXmWW4cSUAdtT',
    source: 'tz1d75oB6T4zUMexzkr5WscGktZ1NsslJrT7' } ] }
```

The forged transaction is put into the array representing the transaction that will be encoded and sent to the Vault application.

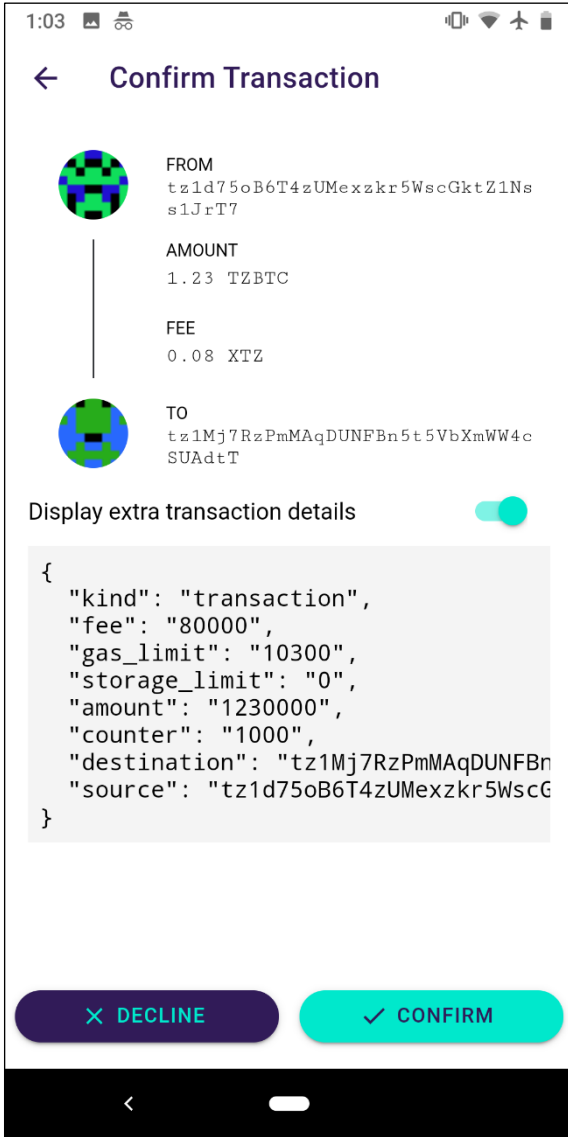
Note that the type is changed to `xtz-btc`:

```
[b'1', b'0', b'xtz-btc',
 [ [b'9bf4f76db480718ffdb1839cc14f6e2c71404482c84186d5fd7f387fe5a6bd7a6c00bf97f5f1dbfd6ada0cf986d0a812f1bf0a572abc80f104e807bc5000b0894b000016e64994c2ddb293695b63e4cade029d3c8b5e300'],
 b'444e1f4ab90c304a5ac003d367747aab63815f583ff2330ce159d12c1ecceba1', b'airgap-wallet://?d='] ] ]
```

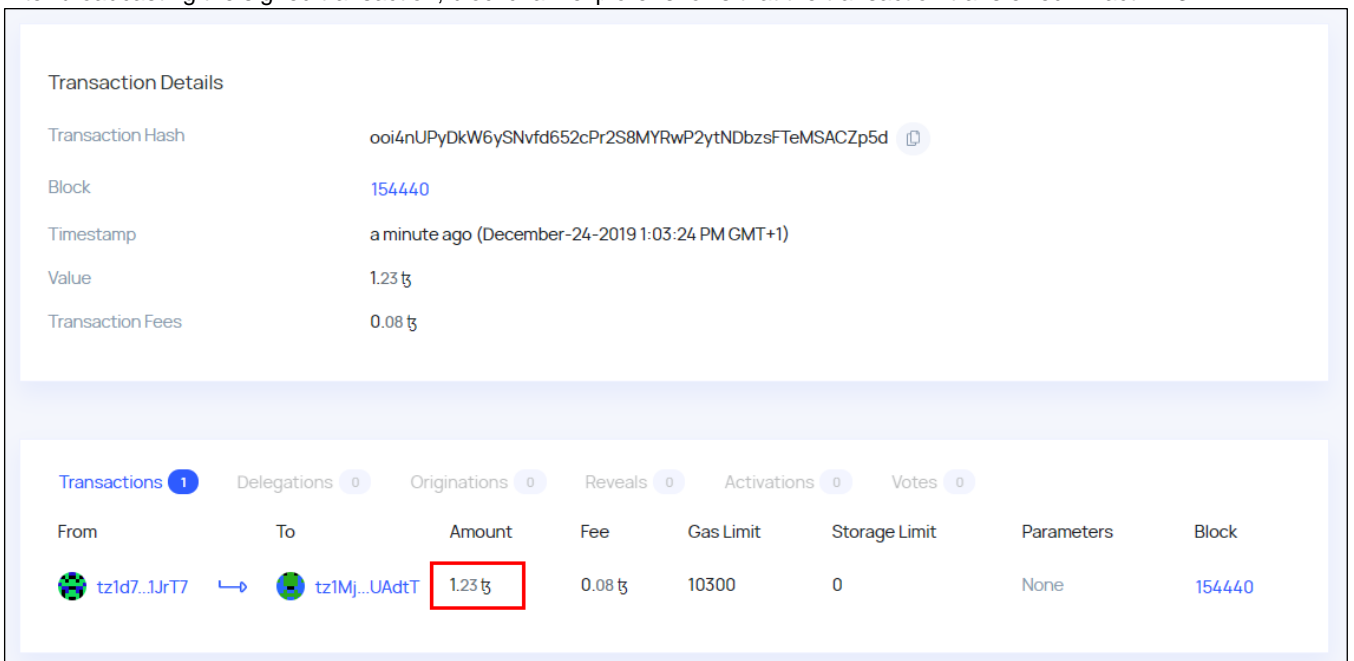
While the above data is read by the Vault application, the user sees transaction value in TZBTC



The transaction could be successfully signed, in the Wallet application it is still shown as TZBTC:



After broadcasting the signed transaction, blockchain explorer shows that the transaction transferred in fact 1.23 XTZ:



4.2 Test #2

No.	Description of Test	Expected Result	Actual Result	PASS FAIL
2.	Is it possible to create an TZBTC transaction whereas the user sees it as an XTZ transaction?	No.	No, such transaction cannot be parsed nor signed.	PASS

Details #2

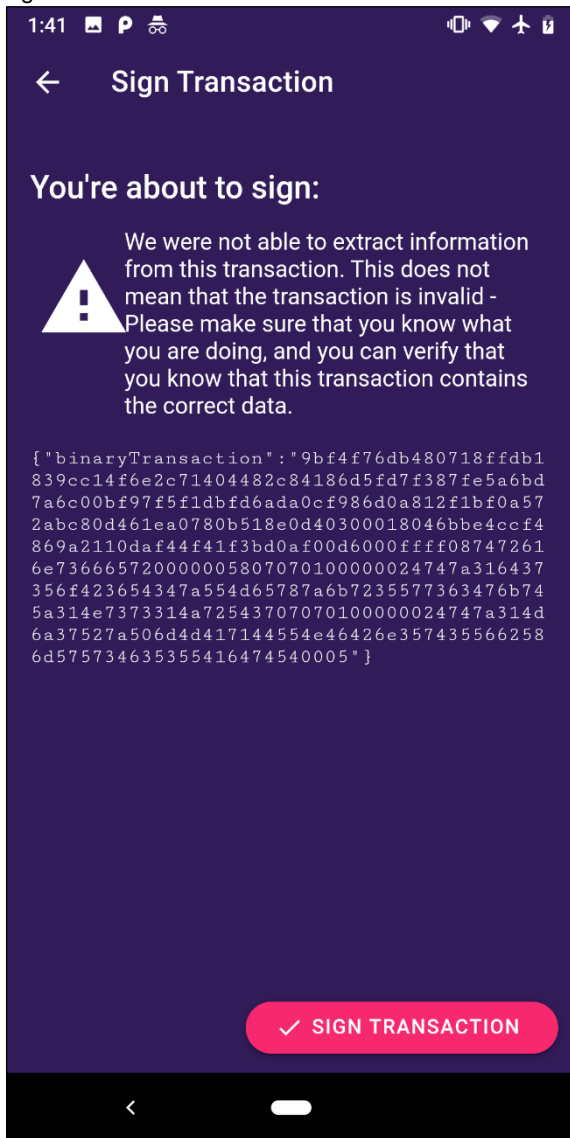
The following transaction is forged:

```
{ branch: 'BLty3agyzAeRaNKjEoFCuraBcW2ihvfwSaQQ1v8ntaE8z47Z8YJ',
  contents:
  [ { kind: 'transaction',
    source: 'tz1d75oB6T4zUMexzkr5WscGktZ1NsslJrT7',
    destination: 'KT1LH2o12xVRwTpJMZ6QJG74Fox8gE9QieFd',
    amount: '0',
    fee: '1600000',
    gas_limit: '400000',
    storage_limit: '60000',
    counter: '1002',
    parameters:
    { entrypoint: 'transfer',
      value:
      '{ "prim": "Pair", "args": [ { "string": "tz1d75oB6T4zUMexzkr5WscGktZ1NsslJrT7" }, {
"prim": "Pair", "args": [ { "string": "tz1Mj7RzPmMAqDUNFBn5t5VbXmWW4cSUAdtT" }, { "int":
"11" } ] } ] } } ] } ] }
```

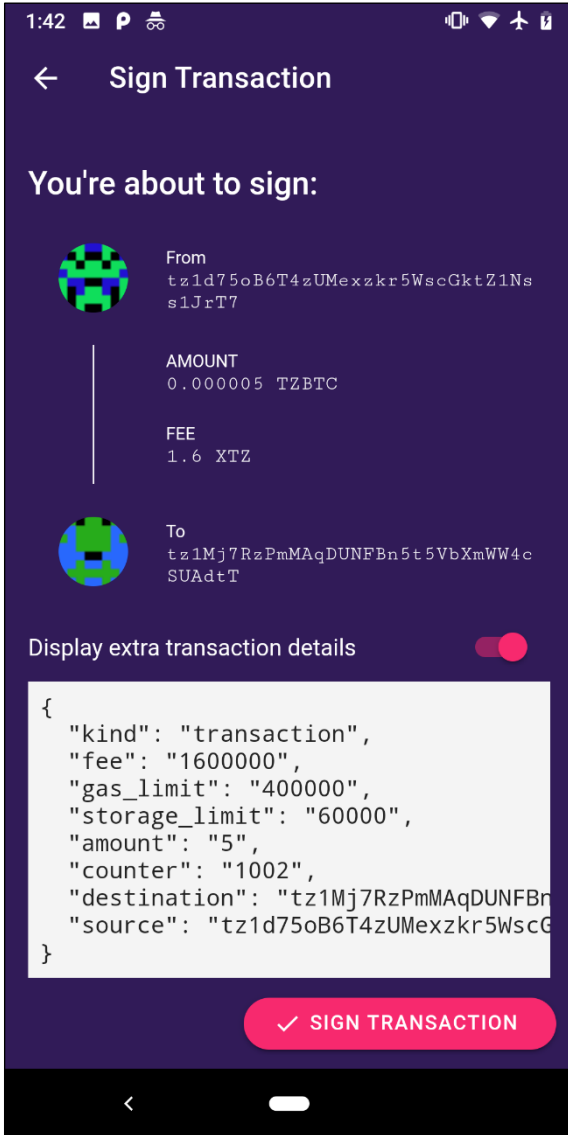
Putting the forged transaction with XTZ type:

```
[b'1', b'0', b'xtz',
[[b'9bf4f76db480718ffdb1839cc14f6e2c71404482c84186d5fd7f387fe5a6bd7a6c00bf97f5f1dbfd6ada0cf9
86d0a812f1bf0a572abc80d461ea0780b518e0d40300018046bbe4ccf4869a2110daf44f41f3bd0af00d600ffff
087472616e736665720000005807070100000024747a316437356f423654347a554d65787a6b723577363476b74
5a314e7373314a72543707070100000024747a314d6a37527a506d4d417144554e46426e3574355662586d575734
635355416474540005'], b'444e1f4ab90c304a5ac003d367747aab63815f583ff2330ce159d12c1ecceba1',
b'airgap-wallet://?d=']]
```

After reading in the Vault application, the following error message is shown and the transaction cannot be signed although the sign transaction button is visible:



Whereas, if the same transaction is marked as xtz-btc, it can be parsed and signed:

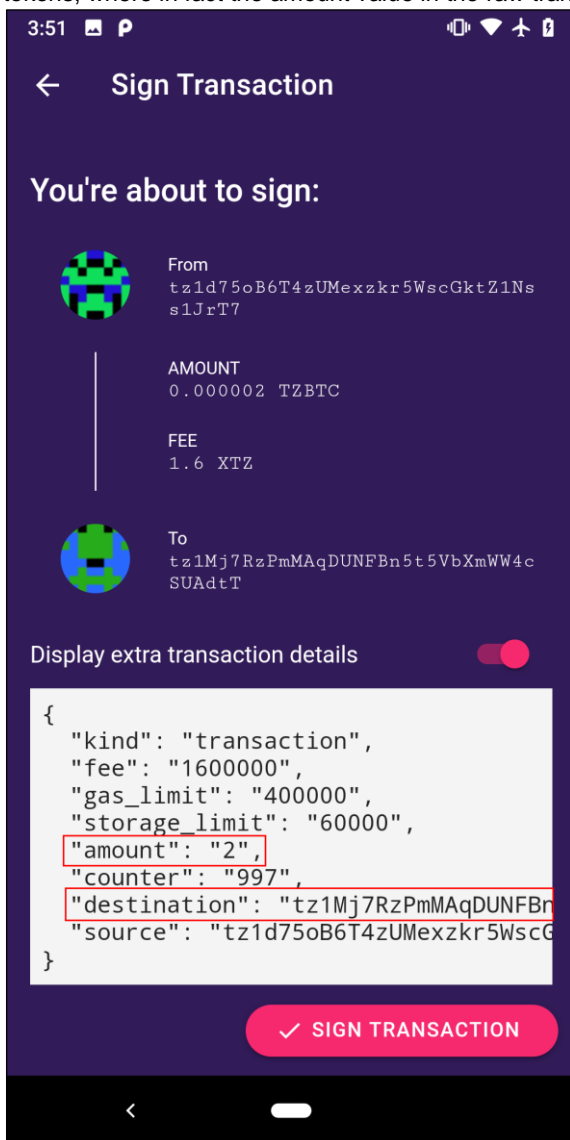


4.3 Test #3

No.	Description of Test	Expected Result	Actual Result	PASS FAIL
3.	Can the user see raw transaction details to get additional information about the transaction he is going to sign?	Yes.	Yes, however, the details do not always represent a raw unforged data. In case of TZBTC transactions, the destination address and the amount refer to the transferred tokens and not to the information of the transaction.	FAIL

Details #3

The user can see details of the transaction he is about to sign. However, in case of token transfers, the destination address is set to the token recipient and not the recipient of the transaction. The shown amount specifies the number of transferred tokens, where in fact the amount value in the raw transaction is 0:



4.4 Test #4

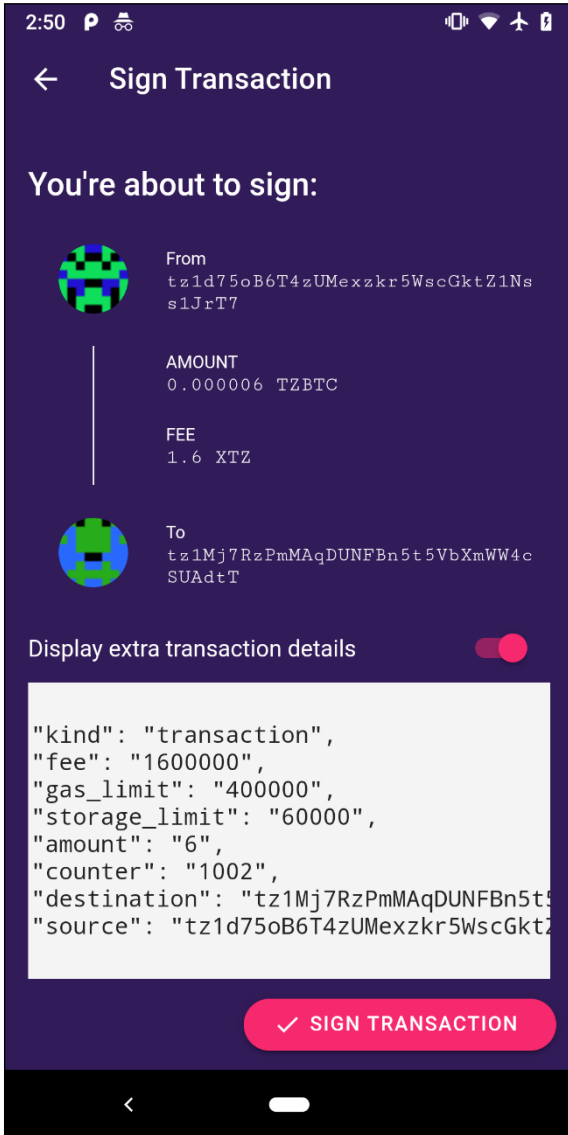
No.	Description of Test	Expected Result	Actual Result	PASS FAIL
4.	Is it possible to specify another contract address and convince the user that he sends TZBTC although in fact other tokens are transferred?	No.	Yes, it is possible to specify arbitrary contract address and the user has no possibility to verify what address is present in the transaction he will sign.	FAIL

Details #4

The destination of the transaction specifies another contract:

```
{ branch: 'BLCvADECsvgD7g3E1auCNGCjuU5DfQmMBmMFhpfcVYBs9Ldpvr7',
  contents:
  [ { kind: 'transaction',
    source: 'tz1d75oB6T4zUMexzkr5WscGktZ1Nss1JrT7',
    destination: 'KT1LH2o12xVRwTpJMZ6QJG74Fox8gE9QieFd',
    amount: '0',
    fee: '1600000',
    gas_limit: '400000',
    storage_limit: '60000',
    counter: '1002',
    parameters:
    { entrypoint: 'transfer',
      value:
      '{ "prim": "Pair", "args": [ { "string": "tz1d75oB6T4zUMexzkr5WscGktZ1Nss1JrT7" }, {
"prim": "Pair", "args": [ { "string": "tz1Mj7RzPmMAqDUNFBn5t5VbXmWW4cSUAdtT" }, { "int": "6"
} ] ] }' } ] ] }
```

While signing, the user does not see where the transaction will be sent to, thus the user cannot verify whether the tokens he sends are TZBTC or other tokens:



4.5 Test #5

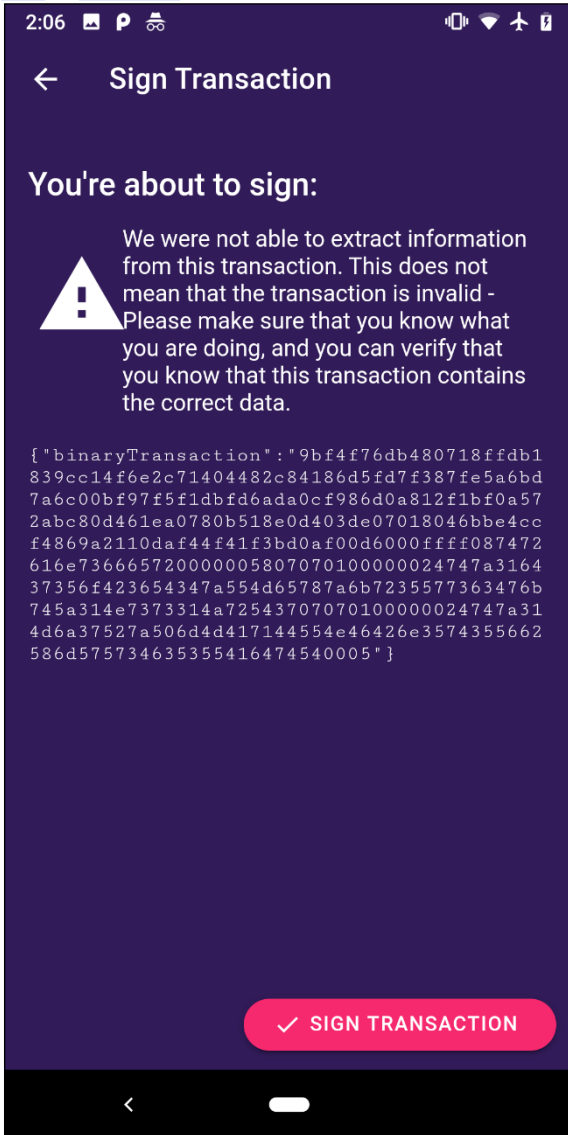
No.	Description of Test	Expected Result	Actual Result	PASS FAIL
5.	Is it possible to send XTZ in the same operation as TZBTC?	If the user is able to see what he will sign, then such transactions may be possible.	No, if there is a transaction containing an amount higher than 0 and some parameters, it causes an error and cannot be signed.	PASS

Details #5

The following transaction was forged. The transaction sends XTZ to `KT1LH2o12xVRwTpJMZ6QJG74Fox8gE9QieFd` and transfers tokens to `tz1Mj7RzPmMAqDUNFBn5t5VbXmWW4cSUAdtT`:

```
{ branch: 'BLty3agyzAeRaNKjEoFCuraBcW2ihvfwSaQQ1v8ntaE8z47Z8YJ',
  contents:
    [ { kind: 'transaction',
      source: 'tz1d75oB6T4zUMexzkr5WscGktZ1Nss1JrT7',
      destination: 'KT1LH2o12xVRwTpJMZ6QJG74Fox8gE9QieFd',
      amount: '990',
      fee: '1600000',
      gas_limit: '400000',
      storage_limit: '60000',
      counter: '1002',
      parameters:
        { entrypoint: 'transfer',
          value:
            '{ "prim": "Pair", "args": [ { "string": "tz1d75oB6T4zUMexzkr5WscGktZ1Nss1JrT7" }, {
"prim": "Pair", "args": [ { "string": "tz1Mj7RzPmMAqDUNFBn5t5VbXmWW4cSUAdtT" }, { "int":
"11" } ] } ] }' } ] }
```


However, the above transaction cannot be correctly parsed by the Vault application. Disregarding whether its type is set to `xtz` or `xtz-btc`, the following error appears:

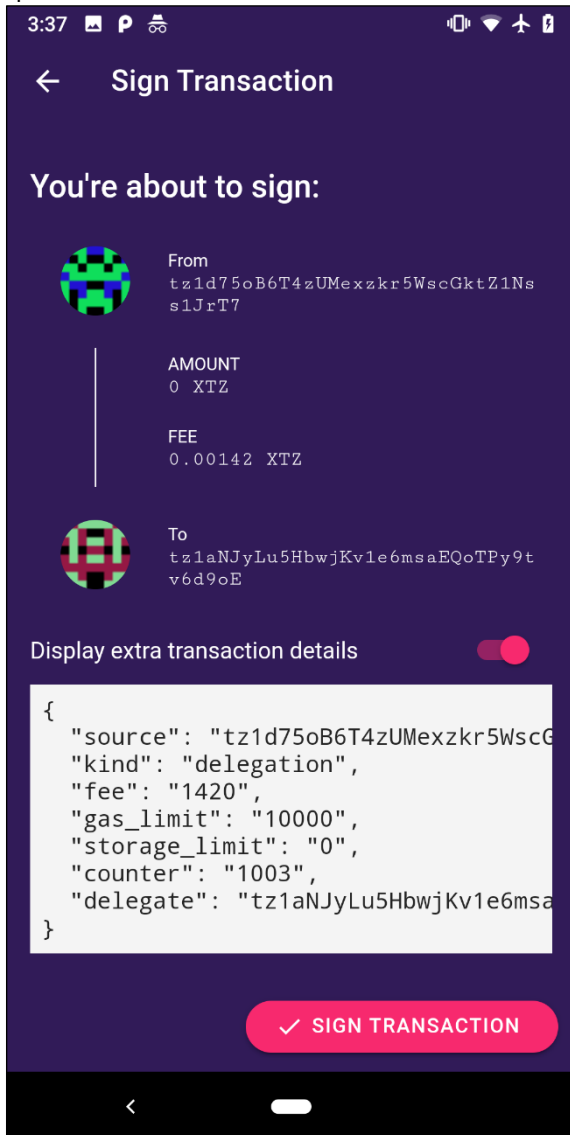


4.6 Test #6

No.	Description of Test	Expected Result	Actual Result	PASS FAIL
6.	Is it possible to delegate funds and also transfer money to the baker in the same operation?	If the user is able to see what he will sign, then such transactions may be possible.	No, it is not possible to forge a delegation transaction that also transfers XTZ.	PASS

Details #6

In the delegate transaction, the shown amount is always equal to 0. As it is not possible to forge a delegate transaction with the amount field (because of the Tezos specifications), it is not possible to delegate funds and transfer coins in the same operation:

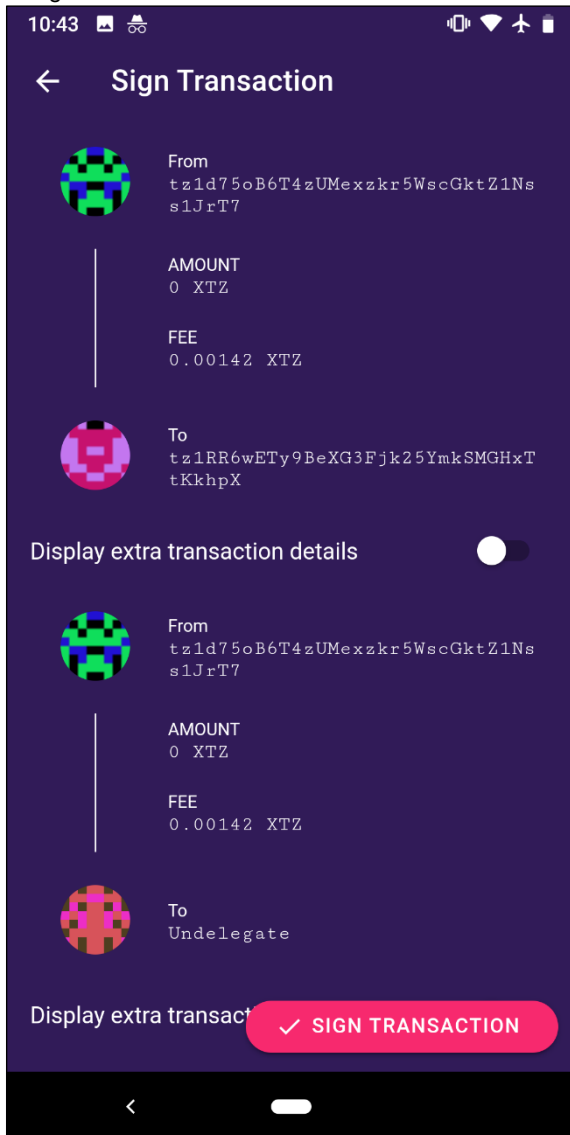


4.7 Test #7

No.	Description of Test	Expected Result	Actual Result	PASS FAIL
7.	Is it possible to delegate or undelegated funds without the user noticing it?	No.	No, all operations are shown to the user.	PASS

Details #7

Two operations are shown to the user. The difference between delegate and undelegated operation is the presence of the delegation address.



4.8 Test #8

No.	Description of Test	Expected Result	Actual Result	PASS FAIL
8.	Is it possible to specify non-hexadecimal characters in place of the forged transaction data?	No, the forged transaction is represented as a hexadecimal string so other characters should not be allowed.	If nonhexadecimal characters are present in the forged transaction, no error is thrown. The value is interpreted despite presence of unexpected characters, however, the data that will be signed is exactly the same data that is shown to the user.	INFO

Details #8

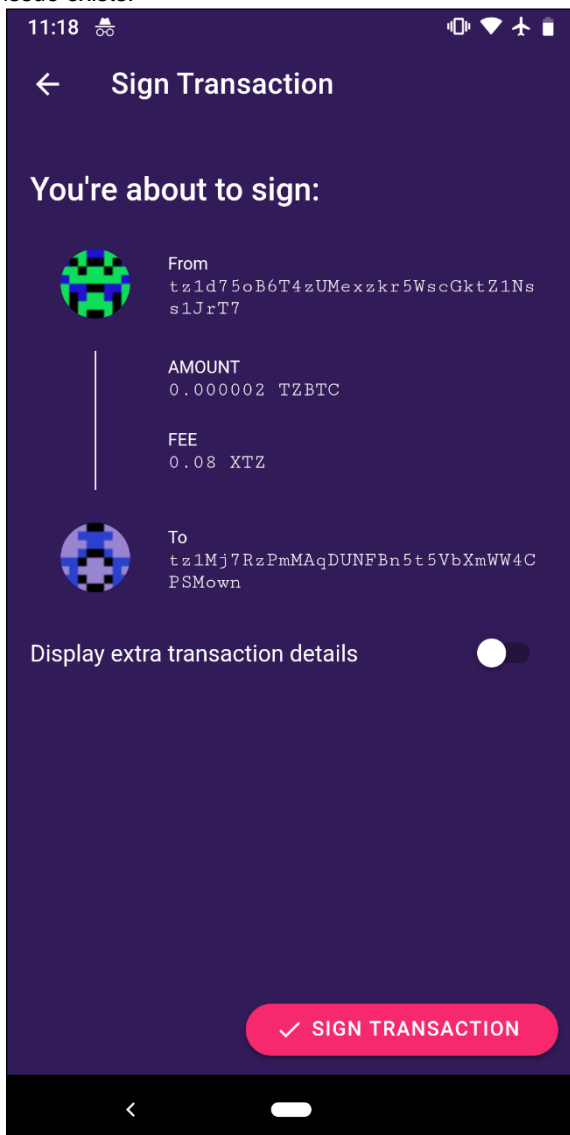
The following string is sent as a forged transaction:

```
4637b7c88ca8a39eae68c92067394a959588fb9874bebc1961fafcbb727d10a96c00bf97f5f1dbfd6ada0cf986d0a812f1bf0a572abc80f104ee07bc500002000016e64994c2dabd293695b63e4cade029d3c8b5e00
```

It is interpreted in the application as:

```
4637b7c88ca8a39eae68c92067394a959588fb9874bebc1961fafcbb727d10a96c00bf97f5f1dbfd6ada0cf986d0a812f1bf0a572abc80f104ee07bc500002000016e64994c2dabd293695b63e4cade029d3c8b50e00
```

The recipient address shown to the user is the same address that will be present in the signed transaction, thus no security issue exists:



4.9 Test #9

No.	Description of Test	Expected Result	Actual Result	PASS FAIL
9.	Is it possible to have a TZBTC transfer operation and XTZ transfer operation in the same transaction?	If the user is able to see what he will sign, then such transactions may be possible.	It is possible to have multiple types of operations as long as no operation is appended after the token transfer operation, otherwise an error occurs. In cases without an error the user sees both transactions but it is not clear which transaction transfers tokens and which XTZ.	FAIL

Details #9

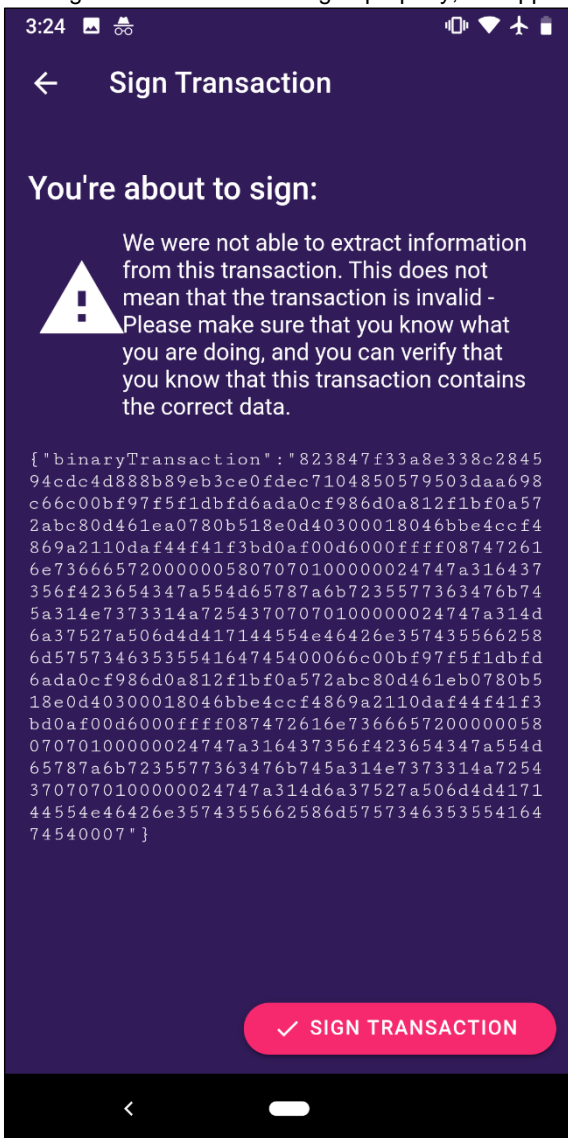
The following two token transfer operations are forged using a request to a node:

```
{ branch: 'BLhddJyKpodtACVZwTE1Lv8nvHVWVWAXbJW9rAxaUJ3JxQbnZgxM',
  contents:
  [ { kind: 'transaction',
    source: 'tz1d75oB6T4zUMexzkr5WscGktZ1Nss1JrT7',
    destination: 'KT1LH2o12xVRwTpJMZ6QJG74Fox8gE9QieFd',
    amount: '0',
    fee: '1600000',
    gas_limit: '400000',
    storage_limit: '60000',
    counter: '1002',
    parameters:
    { entrypoint: 'transfer',
      value:
      '{ "prim": "Pair", "args": [ { "string": "tz1d75oB6T4zUMexzkr5WscGktZ1Nss1JrT7" }, {
"prim": "Pair", "args": [ { "string": "tz1Mj7RzPmMAqDUNFBn5t5VbXmWW4cSUAdtT" }, { "int": "6"
} ] ] }' } } },
    { kind: 'transaction',
    source: 'tz1d75oB6T4zUMexzkr5WscGktZ1Nss1JrT7',
    destination: 'KT1LH2o12xVRwTpJMZ6QJG74Fox8gE9QieFd',
    amount: '0',
    fee: '1600000',
    gas_limit: '400000',
    storage_limit: '60000',
    counter: '1003',
    parameters:
    { entrypoint: 'transfer',
      value:
      '{ "prim": "Pair", "args": [ { "string": "tz1d75oB6T4zUMexzkr5WscGktZ1Nss1JrT7" }, {
"prim": "Pair", "args": [ { "string": "tz1Mj7RzPmMAqDUNFBn5t5VbXmWW4cSUAdtT" }, { "int": "7"
} ] ] }' } } } ] }
```

Forged transaction:

```
823847f33a8e338c284594cdc4d888b89eb3ce0fdec7104850579503daa698c66c00bf97f5f1dbfd6ada0cf986d0
a812f1bf0a572abc80d461ea0780b518e0d40300018046bbe4ccf4869a2110daf44f41f3bd0af00d6000ffff0874
72616e736665720000005807070100000024747a316437356f423654347a554d65787a6b7235577363476b745a31
4e7373314a72543707070100000024747a314d6a37527a506d4d417144554e46426e3574355662586d5757346353
554164745400066c00bf97f5f1dbfd6ada0cf986d0a812f1bf0a572abc80d461eb0780b518e0d40300018046bbe4
ccf4869a2110daf44f41f3bd0af00d6000ffff087472616e736665720000005807070100000024747a316437356f
423654347a554d65787a6b7235577363476b745a314e7373314a72543707070100000024747a314d6a37527a506d
4d417144554e46426e3574355662586d575734635355416474540007
```

Although the transaction is forged properly, the application cannot parse multiple token transfer operations:



A transaction with a Tezos transfer followed by token transfer is forged:

```
{ branch: 'BLhddJyKpodtACVZwTE1Lv8nvHVWWAXbJW9rAxaUJ3JxQbnZgxM',
  contents:
  [ {
    kind: 'transaction',
    source: 'tz1d75oB6T4zUMexzkr5WscGktZ1Nss1JrT7',
    destination: 'tz1Mj7RzPmMAqDUNFBn5t5VbXmWW4cSUAdtT',
    amount: '1230',
    fee: '80000',
    gas_limit: '10300',
    storage_limit: '0',
    counter: '1002'
  }, {
    kind: 'transaction',
    source: 'tz1d75oB6T4zUMexzkr5WscGktZ1Nss1JrT7',
    destination: 'KT1LH2o12xVRwTpJMZ6QJG74Fox8gE9QieFd',
    amount: '0',
    fee: '1600000',
    gas_limit: '400000',
    storage_limit: '60000',
    counter: '1003',
    parameters: { entrypoint: 'transfer',
      value: { "prim": "Pair", "args": [ { "string": "tz1d75oB6T4zUMexzkr5WscGktZ1Nss1JrT7"
    }, { "prim": "Pair", "args": [ { "string": "tz1Mj7RzPmMAqDUNFBn5t5VbXmWW4cSUAdtT" }, {
      "int": "7" } ] } ] } } ] } }
```

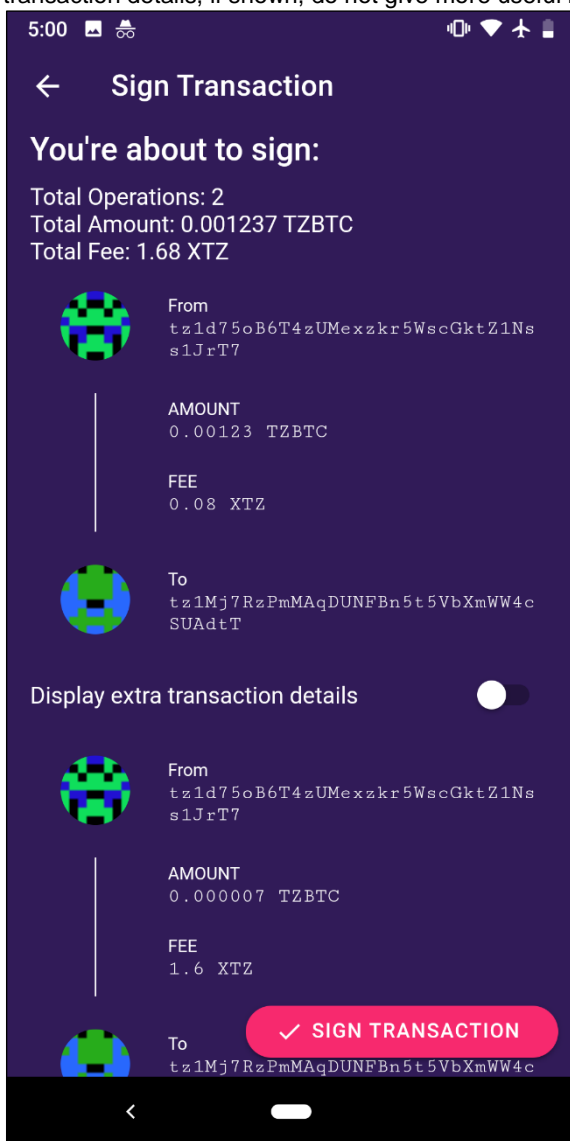
Forged transaction:

```
823847f33a8e338c284594cdc4d888b89eb3ce0fdec7104850579503daa698c66c00bf97f5f1dbfd6ada0cf986d0
a812f1bf0a572abc80f104ea07bc5000ce09000016e64994c2ddb293695b63e4cade029d3c8b5e3006c00bf97f5
f1dbfd6ada0cf986d0a812f1bf0a572abc80d461eb0780b518e0d40300018046bbe4ccf4869a2110daf44f41f3bd
0af00d6000ffff087472616e73666572000005807070100000024747a316437356f423654347a554d65787a6b72
35577363476b745a314e7373314a72543707070100000024747a314d6a37527a506d4d417144554e46426e357435
5662586d575734635355416474540007
```

Payload that will be sent to the application after encoding:

```
[b'1', b'0', b'xtz-btc',
[[b'823847f33a8e338c284594cdc4d888b89eb3ce0fdec7104850579503daa698c66c00bf97f5f1dbfd6ada0cf9
86d0a812f1bf0a572abc80f104ea07bc5000ce09000016e64994c2ddb293695b63e4cade029d3c8b5e3006c00bf
97f5f1dbfd6ada0cf986d0a812f1bf0a572abc80d461eb0780b518e0d40300018046bbe4ccf4869a2110daf44f41
f3bd0af00d6000ffff087472616e73666572000005807070100000024747a316437356f423654347a554d65787a
6b7235577363476b745a314e7373314a72543707070100000024747a314d6a37527a506d4d417144554e46426e35
74355662586d575734635355416474540007'],
b'444e1f4ab90c304a5ac003d367747aab63815f583ff2330ce159d12c1ecceba1', b'airgap-
wallet://?d=']]
```

The application shows XTZ transfer as if it was a TZBTC transfer. The user cannot distinguish which one is which. The extra transaction details, if shown, do not give more useful information as the contract address is not shown.



Interestingly, two operations where there is an operation after the token transfer operation causes an error in the application. An example transaction:

```
{ branch: 'BLhddJyKpodtACVZwTElLv8nvHVWwAXbJW9rAxaUJ3JxQbnZgxM',
  contents:
    [ {
      kind: 'transaction',
      source: 'tz1d75oB6T4zUMexzkr5WscGktZ1Nss1JrT7',
```

```

destination: 'KT1LH2o12xVRwTpJMZ6QJG74Fox8gE9QieFd',
amount: '0',
fee: '1600000',
gas_limit: '400000',
storage_limit: '60000',
counter: '1002',
parameters: { entrypoint: 'transfer',
  value: { "prim": "Pair", "args": [ { "string": "tz1d75oB6T4zUMexzkr5WscGktZ1Nss1JrT7"
}, { "prim": "Pair", "args": [ { "string": "tz1Mj7RzPmMAqDUNFBn5t5VbXmWW4cSUAdtT" }, {
"int": "7" } ] } ] } }
}, {
  kind: 'transaction',
  source: 'tz1d75oB6T4zUMexzkr5WscGktZ1Nss1JrT7',
  destination: 'tz1Mj7RzPmMAqDUNFBn5t5VbXmWW4cSUAdtT',
  amount: '1230',
  fee: '80000',
  gas_limit: '10300',
  storage_limit: '0',
  counter: '1003'
} ] }

```

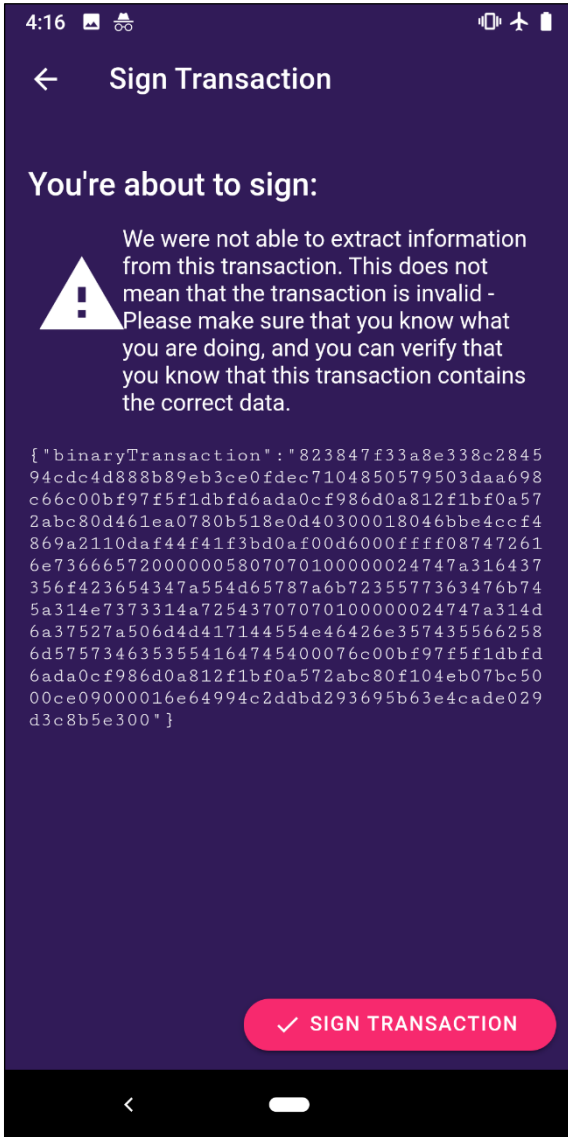
Forged transaction:

```

823847f33a8e338c284594cdc4d888b89eb3ce0fdec7104850579503daa698c66c00bf97f5f1dbfd6ada0cf986d0
a812f1bf0a572abc80d461ea0780b518e0d40300018046bbe4ccf4869a2110daf44f41f3bd0af00d6000ffff0874
72616e736665720000005807070100000024747a316437356f423654347a554d65787a6b7235577363476b745a31
4e7373314a72543707070100000024747a314d6a37527a506d4d417144554e46426e3574355662586d5757346353
554164745400076c00bf97f5f1dbfd6ada0cf986d0a812f1bf0a572abc80f104eb07bc5000ce09000016e64994c2
ddbd293695b63e4cade029d3c8b5e300

```


The application is unable to understand that transaction. Disregarding whether the type of the transaction is set to "xtz" or "xtz-btc", an error occurs:

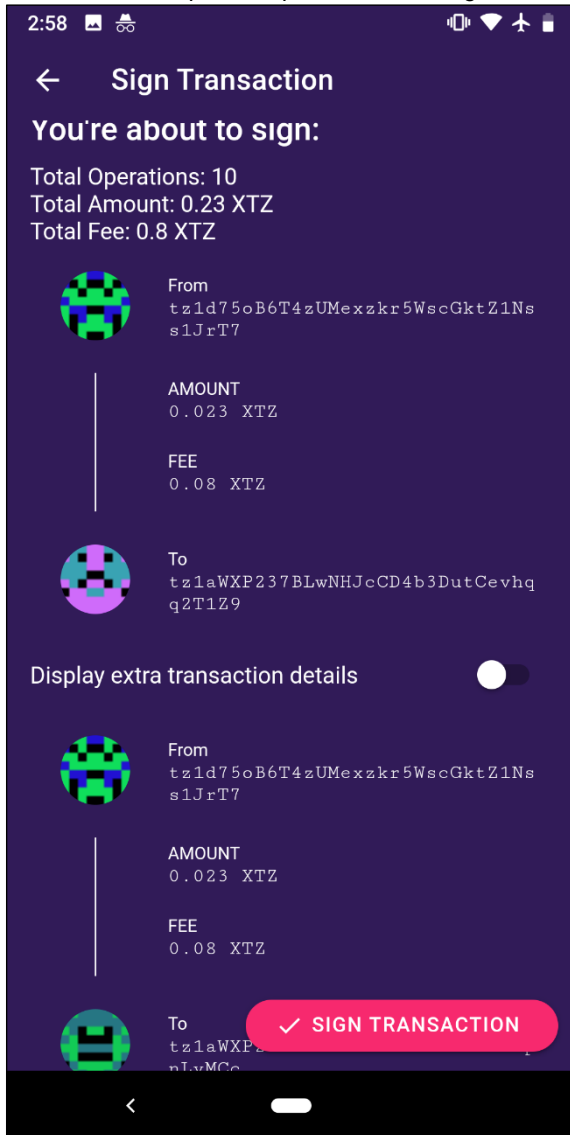


4.10 Test #10

No.	Description of Test	Expected Result	Actual Result	PASS FAIL
10.	If many operations are present in a single transaction, are all operations legibly shown to the user?	Yes.	As expected.	PASS

Details #10

It was tried with up to 10 operations in a single transaction. All operations are properly presented:



4.11 Test #11

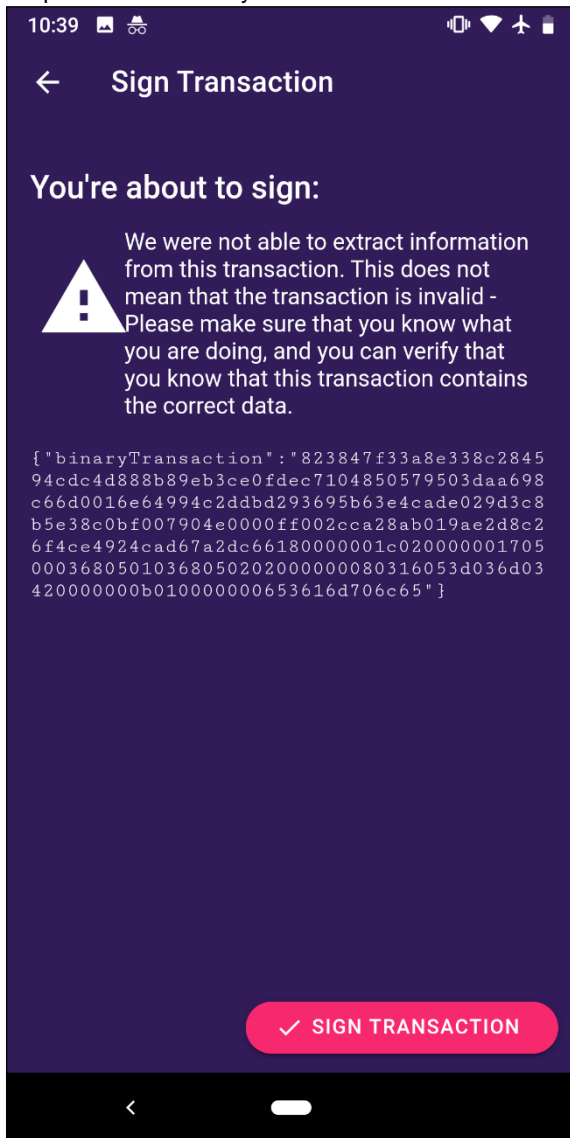
No.	Description of Test	Expected Result	Actual Result	PASS FAIL
11.	Is it possible to hide other operation types in a transaction that the user may sign?	No.	Other operation types are not supported and an exception is thrown when a transaction with such an operation is scanned.	PASS

Details #11

The following transaction containing origination operation is successfully forged:

```
{ branch: 'BLhddJyKpodtACVZwTE1Lv8nvHVWWAXbJW9rAxaUJ3JxQbnZgxM',
  contents:
    [ { source: 'tz1Mj7RzPmMAqDUNFBn5t5VbXmWW4cSUAdtT',
      kind: 'origination',
      fee: '1420',
      gas_limit: '10000',
      storage_limit: '0',
      balance: '0',
      counter: '1008',
      script: {code: [ { "prim": "parameter", "args": [ { "prim": "string" } ] }, { "prim":
"storage", "args": [ { "prim": "string" } ] }, { "prim": "code", "args": [ [ { "prim": "CAR"
}, { "prim": "NIL", "args": [ { "prim": "operation" } ] }, { "prim": "PAIR" } ] ] }, storage: {
"string": "Sample" } },
      delegate: 'tz1PirboZKFVqkfeE45hVLpkpXaZtLk3mqC17' } ] ] }
```

The application does not accept that transaction as a valid one. Similarly, transactions containing other types of operations are not parsed successfully:



5 Appendix

5.1 Compass Weaknesses Rating

Please read this section to understand the Compass weaknesses rating.

5.1.1 What the rating IS NOT

It IS NOT a risk rating. The motivation and opportunity of threat agents as well as the financial impact is not taken into consideration as it cannot be determined by Compass Security.




All vulnerabilities are rated independent from other security controls that might be in place. Examples are:

- If Compass performs tests in the Intranet, border protection is not taken into consideration. We assume that the place we are testing from is hostile.
- If assessing systems in the Intranet, other systems in the Intranet that are not assessed are not taken into consideration for the rating.

5.1.2 What to do with the weaknesses table

- The customer should carefully review the weaknesses table and assess the risk based on the business impact. The final risk rating does not necessarily need to match the initial Compass rating.
- This internal rating should enable the customer to decide how the risk should be treated (e.g. mitigate, accept, avoid or transfer). The decision should be driven by the risk appetite of the company.
- A risk mitigation plan should be developed to schedule and prioritize the remediation of the individual weaknesses.

5.1.3 Examples

Rating	Severity	Examples
 High	<ul style="list-style-type: none"> ▪ Exploitation is easy and leads to high privileges and/or affects many users. ▪ System can be controlled with little effort ▪ High impact if vulnerability is disclosed <p>Fix should be implemented with highest priority. Keep in mind that an issue within a back-end system might not pose the same threat as one in an Internet-facing service.</p>	<ul style="list-style-type: none"> ▪ SQL Injection or Cross-Site Scripting (XSS) ▪ Privilege escalation vulnerabilities ▪ Remote shell vulnerabilities ▪ Authorization bypass vulnerabilities ▪ Default accounts with high privileges ▪ Security filter bypass ▪ Weak encryption ciphers or protocols ▪ Phone in surveillance mode ▪ XML External Entity (XXE)
 Medium	<ul style="list-style-type: none"> ▪ Exploitation can lead to higher privileges if combined with other weaknesses ▪ Exploitation requires significant effort <p>Fix should be implemented in a reasonable time.</p>	<ul style="list-style-type: none"> ▪ Exposed management interfaces ▪ Caching of sensitive data ▪ Denial-of-Service conditions ▪ Insecure cookie settings ▪ Disclosure of usernames, email-addresses ▪ Large attack surface due to open ports
 Low	<ul style="list-style-type: none"> ▪ Abuse does not lead to higher privileges ▪ Information disclosure vulnerabilities <p>Can be solved in the long term.</p>	<ul style="list-style-type: none"> ▪ Disclosure of product and version (banners) ▪ Default pages and samples ▪ DNS zone transfer ▪ DNS reverse lookups
INFO	Just an informational point without security relevant implications.	<ul style="list-style-type: none"> ▪ Usability and performance issues ▪ Developer and staging bugs ▪ Clean-up notes

5.1.4 Tests with result "INFO" and N/A

- All tests with the result "INFO" will be listed in the weaknesses table
- All tests with the result "N/A" will NOT be listed in the weaknesses table

5.2 Recheck Coloring

The following color code is used for pointing out, whether a previously identified vulnerability is solved, partly solved, not solved, no recheck conducted or if new vulnerabilities have been found.

Lavender	Red	Yellow	Green	Gray
A new vulnerability was found.	Vulnerability still exists.	Vulnerability was partially eliminated.	Vulnerability was eliminated.	No recheck conducted.