Swipe

Security Assessment

September 22nd, 2020

By :
Camden Smallwood @ CertiK
camden.smallwood@certik.org

Alex Papageorgiou @ CertiK
alex.papageorgiou@certik.org

DISCLAIMER

CertiK reports are not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. These reports are not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts CertiK to perform a security review.

CertiK Reports do not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

CertiK Reports should not be used in any way to make decisions around investment or involvement with any particular project. These reports in no way provide investment advice, nor should be leveraged as investment advice of any sort.

CertiK Reports represent an extensive auditing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. CertiK's position is that each company and individual are responsible for their own due diligence and continuous security. CertiK's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

What is a CertiK report?

- A document describing in detail an in depth analysis of a particular piece(s) of source code provided to CertiK by a Client.
- An organized collection of testing results, analysis and inferences made about the structure, implementation and overall best practices of a particular piece of source code.
- Representation that a Client of CertiK has indeed completed a round of auditing with the intention to increase the quality of the company/product's IT infrastructure and or source code.

SUMMARIES

Project Summary

Project Name	Swipe
Description	Swipe has designed and implemented Governance and Timelock smart contracts as the basis for the voting functionality in their SXP token staking system.
Platform	Ethereum; Solidity, Yul
Codebase	GitHub Repository
Commits	1. ce376c7d141df0e47a6031626b958e515309053c 2. b735346945fd363c7afb01c5e4c74bd65fde1cec

Audit Summary

Delivery Date	Sep. 22, 2020
Method of Audit	Static Analysis, Manual Review
Consultants Engaged	2
Timeline	Aug. 31, 2020 - Sep. 2 2020

Vulnerability Summary

Total Issues	10
Total Critical	0
Total Major	0
Total Minor	5
Total Informational	5

FINDINGS

ID	Title	Type	Severity
SGT-01	Multiple Solidity versions used	Language Specific	Informational
SGT-02	Duplicate require statements	Language Specific	Informational
SGT-03	Duplicate functionality	Language Specific	Informational
SGT-04	Duplicate require statements	Language Specific	Informational
SGT-05	Incorrect requirement condition	Logic	Minor
SGT-06	Unused return value	State Change	Minor
SGT-07	Unused return value	State Change	Minor
SGT-08	Insufficient gas use	Performance	Informational
SGT-09	Unused return value	State Change	Minor
SGT-10	Unused return value	State Change	Minor

SGT-01: Multiple Solidity versions used

Type	Severity	Location
Language Specific	Informational	See List

Description:

Different Solidity compiler version requirements were used throughout the provided smart contracts:

File	Version
Governance.sol	`^0.5.16`
GovernanceEvent.sol	`^0.5.16`
GovernanceProxy.sol	`^0.5.0`
GovernanceStorage.sol	`^0.5.16`
GovernanceTimelock.sol	`^0.5.16`
GovernanceTimelockEvent.sol	`^0.5.16`
GovernanceTimelockProxy.sol	`^0.5.0`
GovernanceTimelockStorage.sol	`^0.5.16`
IGovernanceTimelock.sol	`^0.5.0`
IStaking.sol	`^0.5.0`

Recommendation:

We recommended using Solidity compiler version `0.5.16` or greater for the `GovernanceProxy.sol`, `GovernanceTimelockProxy.sol`, `IGovernanceTimelock.sol` and `IStaking.sol` files.

Alleviation:

The recommendation was not taken into account.

SGT-02: Duplicate require statements

Type	Severity	Location
Language Specific	Informational	Governance.sol
Language Specific	Informational	GovernanceTimelock.sol

Description:

Both the `Governance` and `GovernanceTimelock` contracts has multiple duplicate instances of the same requirement related to access restriction that the message sender should be the `guardian` account and the `authorizedNewGuardian` account.

Recommendation:

We recommended that this behavior could be abstracted in a new `Guardian` contract which inherits from and extends the previously-verified OpenZeppelin `Ownable` contract to encapsulate `guardian` account functionality such as non-guardian access restriction and transferring to new guardian accounts. This can be re-used across the `Governance` and `GovernanceTimelock` contracts through inheritance to improve code re-use, shorten overall code length and eliminate room for error in future refactoring.

Alleviation:

The recommendation was not taken into account.

SGT-03: Duplicate functionality

Type	Severity	Location
Language Specific	Informational	Governance.sol
Language Specific	Informational	GovernanceTimelock.sol

Description:

Both the `Governance` and `GovernanceTimelock` contracts has duplicate functions related to assuming guardianship.

Recommendation:

We recommended that this behavior could be abstracted in a new `Guardian` contract which inherits from and extends the previously-verified OpenZeppelin `Ownable` contract to encapsulate `guardian` account functionality such as non-guardian access restriction and transferring to new guardian accounts. This can be re-used across the `Governance` and `GovernanceTimelock` contracts through inheritance to improve code re-use, shorten overall code length and eliminate room for error in future refactoring.

Alleviation:

The recommendation was not taken into account.

SGT-04: Duplicate require statements

Type	Severity	Location
Language Specific	Informational	Governance.sol

Description:

The `Governance` contract has multiple duplicate instances of the same requirement that a proposal identifer is valid.

Recommendation:

We recommended creating a `validProposal` modifier to improve code re-use, shorten overall code length and eliminate room for error in future refactoring.

Alleviation:

The recommendation was not taken into account.

SGT-05: Incorrect requirement condition

Type	Severity	Location
Logic	Minor	Governance.sol, L172-L175

Description:

The `Governance.propose` function had a requirement with an incorrect condition that the sender's internal voting power should be greater than the proposal threshold instead of being at least the proposal threshold.

Recommendation:

Based on the documentation for the proposal threshold, we recommended changing the condition for the requirement in the `Governance.propose` function so that the sender's voting power must be greater than or equal to the proposal threshold.

Alleviation:

The issue was fixed in Swipe's develop branch in commit 8dfbc6131236a65506fbc754e3a69843421b652d.

SGT-06: Unused return value

Type	Severity	Location
State Change	Minor	Governance.sol, L275-L281

Description:

The `Governance.internalQueueOrRevert` function makes a call to the `_timelock` state variable's `GovernanceTimelock.queueTransaction` function and ignores the returned transaction hash bytes.

Recommendation:

We recommended determining if the transaction hash returned from the call to the timelock's `GovernanceTimelock.queueTransaction` function is necessary and considering incorporating it into the system or emitting an event.

Alleviation:

The recommendation was not taken into account.

SGT-07: Unused return value

Type	Severity	Location
State Change	Minor	Governance.sol, L292-L302

Description:

The `Governance.execute` function has a loop over the queued proposal's actions in which a call is made to the `GovernanceTimelock.executeTransaction` function for each proposal action and ignores the data returned from the call to each target.

Recommendation:

We recommended determining if the data returned from the call to the timelock's `GovernanceTimelock.executeTransaction` function is necessary and considering incorporating it into the system or emitting an event.

Alleviation:

The recommendation was not taken into account.

SGT-08: Insufficient gas use

Type	Severity	Location
Performance	Informational	Governance.sol, L334-L360

Description:

The `Governance.getProposal` function makes insufficient use of gas by copying `Proposal` field values from a storage pointer, which has a higher gas cost than copying the `Proposal` structure into memory.

Recommendation:

We recommended using a memory pointer instead of a storage pointer when retrieving the `Proposal` entry from the `_proposals` state variable.

Alleviation:

The recommendation was not taken into account.

SGT-09: Unused return value

Type	Severity	Location
State Change	Minor	Governance.sol, L558-L564

Description:

The `Governance.queueAuthorizeGuardianshipTransfer` function makes a call to the `_timelock` state variable's `GovernanceTimelock.queueTransaction` function and ignores the returned transaction hash bytes.

Recommendation:

We recommended determining if the transaction hash returned from the call to the timelock's `GovernanceTimelock.queueTransaction` function is necessary and considering incorporating it into the system or emitting an event.

Alleviation:

The recommendation was not taken into account.

SGT-10: Unused return value

Type	Severity	Location
State Change	Minor	Governance.sol, L573-L579

Description:

The `Governance.executeAuthorizeGuardianshipTransfer` function makes a call to the `_timelock` state variable's `GovernanceTimelock.executeTransaction` function and ignores the returned transaction hash bytes.

Recommendation:

Determine if the data returned from the call to the timelock's `GovernanceTimelock.executeTransaction` function is necessary and consider incorporating it into the system or emitting an event.

Alleviation:

The recommendation was not taken into account.