Smart Contract Audit

Date	November 2020

PAXOS

We conducted a security analysis of Paxos' multisig wallet contract in November 2020. This wallet is based upon Christian Lundkvist's SimpleMultiSig contract which we previously reviewed.

1 | EXECUTIVE SUMMARY

We conducted a security analysis of Paxos' multisig wallet contract in November 2020. This wallet is based upon Christian Lundkvist's SimpleMultiSig contract. We previously reviewed it. The review was done by Sergii Kravchenko and Nicholas Ward over 20 person-days, from February 15 to February 26, 2021.

Paxos's modification permits the owners to be changed once the wallet has been deployed. This report examines the impact of these changes.

This assessment was performed between November 7th and 11th, 2020. Steve Marx was the principal conductor of the engagement. The effort required was 8 hours.

1.1 | Scope

Repository	SHA-1 Hash
SimpleMultiSig.sol	80d54d79fa1ec6268ad42d01f393417edb47bdc5

2 | RECOMMENDATIONS

2.1 | Update to a more recent version of the Solidity compiler Fixed

The Solidity version was upgraded to 0.6.11 in paxosglobal/simple-multisig#8. This is the most recent version that Slither supports.

We recommend that you update to the most recent version of Solidity. You can also make small improvements to the compiler, which are separate recommendations.

2.2 | Convert DOMAIN_SEPARATOR to be immutable Fixed

This has been fixed in paxosglobal/simple-multisig#9.

Beginning with Solidity compiler version 0.6.5, state variables may be marked as immutable. These state variables need to be initialized by the contract's builder. They will behave much like constants if they are not initialized in the contract's constructor. This is a good fit to the DOMAIN_SEPARATOR. It is calculated at runtime to include contract address, but acts as a constant.

code/contracts/SimpleMultiSig.sol:L25

```
bytes32 DOMAIN_SEPARATOR;         // hash for EIP712, computed from contract address
```

2.3 | Convert the assembly call to Solidity Fixed

This has been fixed in paxosglobal/simple-multisig#9.

Starting with version 0.5.0, the Solidity `address.call()` function no longer has the padding bug described in https://github.com/ethereum/solidity/issues/2884. It is possible to remove the assembly block from `execute()` and use Solidity instead. This is a small gain for readability.

code/contracts/SimpleMultiSig.sol:L25

```
bytes32 DOMAIN_SEPARATOR;         // hash for EIP712, computed from contract address
```

code/contracts/SimpleMultiSig.sol:L79-L84

```
// If we make it here all signatures are accounted for.
// The address.call() syntax is no longer recommended, see:
// https://github.com/ethereum/solidity/issues/2884
nonce = nonce + 1;
bool success = false;
assembly { success := call(gasLimit, destination, value, add(data, 0x20), mload(data), 0, 0) }
```

2.4 | Update comments about state mutability Fixed

This has been fixed in paxosglobal/simple-multisig@3824608.

The comments accompanying the `ownersArr` and is `Owner` variables indicate that they are immutable. However, in the modified contract, both variables can be changed after deployment.

code/contracts/SimpleMultiSig.sol:L22-L23

```
mapping (address => bool) isOwner; // immutable state
address[] public ownersArr;        // immutable state
```

3 | FINDINGS

Each issue is assigned a severity:

- **Minor** problems are subjective. These are usually suggestions about best practices or readability. These issues should be addressed by code maintainers.
- **Medium** issues are objective, but they are not security vulnerabilities. These issues should be addressed, unless there are compelling reasons not to.
- Security vulnerabilities are critical issues that can't be exploited directly or require special conditions to be exploited. All of these **Major** problems should be addressed.
- Security vulnerabilities that could be exploited to cause **Critical** issues need to be addressed.

3.1 Owners cannot be removed Critical Fixed

This has been fixed in paxosglobal/simple-multisig#5, and appropriate tests have been added.

Description

`SetOwners()`'s purpose is to replace the current owners with a new group. The is `Owner` mapping is not updated. This means that any address considered to be an owner will continue to be considered an owner when signing transactions.

Recommendation

Before adding new owners to `setOwners_()` loop through the existing owners and clear their `Owner` booleans as follows:

```
for (uint256 i = 0; i < ownersArr.length; i++) {
  isOwner[ownersArr[i]] = false;
}
```

APPENDIX 1 DISCLOSURE

ConsenSys Dialigence ("CD") receives compensation from clients (the Clients) for the analysis performed in these reports (the Reports). Reports can be distributed via ConsenSys publications or other distributions.

Reports are not intended to endorse or indict any project or team. They also do not guarantee security for any project. This Report doesn't consider or have any bearing on the economics of token sales, token sales, or any other product, services, or assets. Cryptographic tokens, which are emerging technologies, carry high technical risks and uncertainties. Any Report does not provide any representation or warranty to Third-Parties in any way. This includes regarding the bug-free nature of code, any business model or proprietors, or the legal compliance of such businesses. The Reports should not be relied upon by any third party, even if it is used to make decisions about buying or selling tokens, products, services, or assets. This Report is not intended as investment advice and should not be relied on as such. It is also not an endorsement of this team or project, and is not a guarantee of absolute security. CD is not obligated to any Third-Party for publishing these Reports.

PURPOSE OF THE REPORTS Reports and analysis contained therein are only for Clients. They can be published with their permission. Our review will only cover Solidity code. We are limited to reviewing the Solidity codes we have identified as being included in this report. Solidity language is still under development. It may have flaws and risks. The review does NOT cover the compiler layer or any other areas that could pose security risks beyond Solidity. Cryptographic tokens, which are emerging technologies, carry high technical risk and uncertainty.

CD makes the Reports accessible to clients and other parties (i.e. "third parties") via its website. CD hopes that the public availability of these analyses will help the blockchain ecosystem to develop best practices in this rapidly changing area of innovation.

LINKS TO OTHER WEBSITES FROM THIS WEB site You can, via hypertext or other computer hyperlinks, gain access web sites owned by people other than ConsenSys. These hyperlinks are provided only for your convenience and are not intended to replace the owners of these web sites. ConsenSysys or CD are not responsible or liable for any content or operation of these Web sites. You also agree that ConsenSysys or CD will not be liable for any third-party Web site. Except as stated below, linking from this Web Site to another site does not mean or imply that ConsenSysys or CD endorses that site's content or its operator. It is up to you to decide whether or not you can use content from any other websites to which the Reports link. ConsenSys or CD will not be responsible for third-party software used on the Web Site. They also assume no responsibility and will have no liability to any person or entity as regards the accuracy and completeness of any result generated by such software.

TIMELINESS CONTENT. The Reports are current as of the Report's date. However, they can be modified at any time. ConsenSys or CD are the only sources of information, unless otherwise indicated.