Security Assessment

Manifold Reference

CertiK Assessed on Jul 3rd, 2023

CertiK Assessed on Jul 3rd, 2023

Manifold Reference

The security assessment was prepared by CertiK, the leader in Web3.0 security.

Executive Summary

TYPES	ECOSYSTEM	METHODS
DeFi	EVM Compatible	Manual Review, Static Analysis

LANGUAGE	TIMELINE	KEY COMPONENTS
Solidity	Delivered on 07/03/2023	N/A

CODEBASE

https://github.com/erc6551/reference/tree/main/src

View All in Codebase Page

COMMITS

1a5b0054577436cac920a52816e297f804961aa2

d79a49e99fee1154a2023bd164850c5d0c34c208

f2e98f272e4133b161e5617896578e26189c8e1e

View All in Codebase Page

Vulnerability Summary

6	2	0	0	4	0
Total Findings	Resolved	Mitigated	Partially Resolved	Acknowledged	Declined

0 Critical			Critical risks are those that impact the safe functioning of a platform and must be addressed before launch. Users should not invest in any project with outstanding critical risks.
2 Major	2 Acknowledged		Major risks can include centralization issues and logical errors. Under specific circumstances, these major risks can lead to loss of funds and/or control of the project.
0 Medium			Medium risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform.
2 Minor	2 Acknowledged		Minor risks can be any of the above, but on a smaller scale. They generally do not compromise the overall integrity of the project, but they may be less efficient than other solutions.
2 Informational	2 Resolved		Informational errors are often recommendations to improve the style of the code or certain operations to fall within industry best practices. They usually do not affect the overall functioning of the code.

TABLE OF CONTENTS | MANIFOLD REFERENCE

CODEBASE | MANIFOLD REFERENCE

Repository

https://github.com/erc6551/reference/tree/main/src

Commit

1a5b0054577436cac920a52816e297f804961aa2

d79a49e99fee1154a2023bd164850c5d0c34c208

f2e98f272e4133b161e5617896578e26189c8e1e

AUDIT SCOPE | MANIFOLD REFERENCE

12 files audited ● 2 files with Acknowledged findings ● 3 files with Resolved findings ● 7 files without findings

ID	File	SHA256 Checksum
SER	src/examples/simple/SimpleERC6551Account.sol	4168ffa60155d9c176c21e7f79089068c702e16bd8619c7304ca105d62060c6e
ERU	src/examples/upgradeable/ERC6551AccountUpgradeable.sol	add6f688aed75cca3552d642fad6e67074863667f0c75013ae648a887ea8ebc8
ERP	src/examples/upgradeable/ERC6551AccountProxy.sol	7d836187bebe12fece5b21897a948a428929ace11fe20ba61e27e9651da64971
ERA	src/lib/ERC6551AccountLib.sol	0259c4f0de1917e529fc684ac5bc6510406da696ae45df8760b5d57eded693db
ERC	src/ERC6551Registry.sol	e7b7fbd6b39e1b3fa0b75bf4ce73cb06b8448a33c15f55e14a153c0d480cfff3
ERB	src/lib/ERC6551BytecodeLib.sol	d1da34d949913257eb0d5605160c87937399c6ed9c61b9353b0217177c3497d3
SEC	examples/simple/SimpleERC6551Account.sol	84833081f2e3290a72b693e227bc5a746bb975b55fbfa9acc308f1f3767c6662
ECA	examples/upgradeable/ERC6551AccountProxy.sol	36cbe4b441837fc44f90252362b783428daa16869bb541ee06d547cd726ece75
ECU	examples/upgradeable/ERC6551AccountUpgradeable.sol	add6f688aed75cca3552d642fad6e67074863667f0c75013ae648a887ea8ebc8
ERL	lib/ERC6551AccountLib.sol	7de886dcc7ac02c4236ea355700bf783bda589e199a9e160b186fed156692ed6
ECB	lib/ERC6551BytecodeLib.sol	d1da34d949913257eb0d5605160c87937399c6ed9c61b9353b0217177c3497d3
ERR	ERC6551Registry.sol	723318466d847e9d360f86ec04ce11bdf5ef9d490b6ac241d328fc0a444cfa76

APPROACH & METHODS | MANIFOLD REFERENCE

This report has been prepared for Manifold to discover issues and vulnerabilities in the source code of the Manifold Reference project as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Manual Review and Static Analysis techniques.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

The security assessment resulted in findings that ranged from critical to informational. We recommend addressing these findings to ensure a high level of security standards and industry practices. We suggest recommendations that could better serve the project from the security perspective:

- Testing the smart contracts against both common and uncommon attack vectors;
- Enhance general coding practices for better structures of source codes;
- Add enough unit tests to cover the possible use cases;
- Provide more comments per each function for readability, especially contracts that are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

FINDINGS | MANIFOLD REFERENCE

6	0	2	0	2	2
Total Findings	Critical	Major	Medium	Minor	Informational

This report has been prepared to discover issues and vulnerabilities for Manifold Reference. Through this audit, we have uncovered 6 issues ranging from different severity levels. Utilizing the techniques of Manual Review & Static Analysis to complement rigorous manual code reviews, we discovered the following findings:

ID	Title	Category	Severity	Status
ERU-01	**Centralized Control Of Contract Upgrade**	**Centralization**	**Major**	**Acknowledged**
GLOBAL-01	**Centralization Related Risks**	**Centralization**	**Major**	**Acknowledged**
EXA-01	Potential Fraud Risk Associated With The ERC721 Token Contract Code	Volatile Code	Minor	Acknowledged
EXA-02	Third-Party Dependency Usage	Design Issue	Minor	Acknowledged
ERA-01	Misuse Of `extcodecopy()` In `salt()` And `token()` Functions	Coding Issue	Informational	Resolved
SRC-01	Missing Zero Address Validation	Volatile Code	Informational	Resolved

ERU-01 | CENTRALIZED CONTROL OF CONTRACT UPGRADE

Category	Severity	Location	Status
Centralization	**Major**	**src/examples/upgradeable/ERC6551AccountUpgradeable.sol (v1)**	**Acknowledged**

Description

The `ERC6551AccountUpgradeable` is an upgradeable contract, the owner can upgrade the contract without the community's commitment. If an attacker compromises the account, they can change the implementation of the contract and drain tokens from the contract.

It is important to note that in practical use cases, the owners of upgradeable contracts are typically **NOT** the client. Instead, they should be the users themselves.

Recommendation

The risk describes the current project design and potentially makes iterations to improve in the security operation and level of decentralization, which in most cases cannot be resolved entirely at the present stage. We advise the client to carefully manage the privileged account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., multisignature wallets. Indicatively, here are some feasible suggestions that would also mitigate the potential risk at a different level in terms of short-term, long-term and permanent:

Short Term:

Timelock and Multi sign (⅔, ⅗) combination *mitigate* by delaying the sensitive operation and avoiding a single point of key management failure.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key compromised;
  AND
- A medium/blog link for sharing the timelock contract and multi-signers addresses information with the public audience.

Long Term:

Timelock and DAO, the combination, *mitigate* by applying decentralization and transparency.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.
  AND
- A medium/blog link for sharing the timelock contract, multi-signers addresses, and DAO information with the public audience.

Permanent:

Renouncing the ownership or removing the function can be considered *fully resolved*.

- Renounce the ownership and never claim back the privileged roles.
  OR
- Remove the risky functionality.

Alleviation

The client has acknowledged the finding.

GLOBAL-01 | CENTRALIZATION RELATED RISKS

Category	Severity	Location	Status
Centralization	**Major**		**Acknowledged**

Description

In the contracts `SimpleERC6551Account` and `ERC6551AccountUpgradeable`, the role `owner()` has authority over the `executeCall()` function, which can perform low-level calls. Any compromise to the `owner()` account may allow the hacker to take advantage of the above authorities.

It is important to note that in practical use cases, the `owner() is typically **NOT** the client. Instead, they should be the users themselves.

Recommendation

The risk describes the current project design and potentially makes iterations to improve in the security operation and level of decentralization, which in most cases cannot be resolved entirely at the present stage. We recommend carefully managing the privileged account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., multi-signature wallets.

Indicatively, here are some feasible suggestions that would also mitigate the potential risk at a different level in terms of short-term, long-term and permanent:

Short Term:

Timelock and Multi sign (⅔, ⅗) combination *mitigate* by delaying the sensitive operation and avoiding a single point of key management failure.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key compromised;
  AND
- A medium/blog link for sharing the timelock contract and multi-signers addresses information with the public audience.

Long Term:

Timelock and DAO, the combination, *mitigate* by applying decentralization and transparency.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Introduction of a DAO/governance/voting module to increase transparency and user involvement;
  AND
- A medium/blog link for sharing the timelock contract, multi-signers addresses, and DAO information with the public audience.

Permanent:

Renouncing the ownership or removing the function can be considered *fully resolved*.

- Renounce the ownership and never claim back the privileged roles;
  OR
- Remove the risky functionality.

Alleviation

The client has acknowledged this finding.

EXA-01 POTENTIAL FRAUD RISK ASSOCIATED WITH THE ERC721 TOKEN CONTRACT CODE

Category	Severity	Location	Status
Volatile Code	Minor	src/examples/simple/SimpleERC6551Account.sol (v1); src/examples/upgradeable/ERC6551AccountUpgradeable.sol (v1)	Acknowledged

Description

The existing contract design relies on the accuracy and reliability of the ERC721 token contract. A potential vulnerability could arise if a malicious actor were to supply a fraudulent token. For instance, they might manipulate the `ownerOf()` function in the ERC721 token contract to perpetually return their own address. This means that even when the `safeTransfer()` function is invoked and the `to` address is designated to another account, the actual ownership of the token remains unchanged. This loophole could allow the malicious actor to maintain control over the token and any associated assets within the token-bound account, leading to potential scams and asset losses for other users.

Scenario

Consider the following scenario:

- Alice owns an ERC-721 token X, which owns token bound account Y.
- Bob offers to purchase token X via a decentralized marketplace, assuming he will receive the account Y along with the token.
- The marketplace invokes the `safeTransferFrom()` function and sets the `to` address to Bob's account.
- Due to the token contract's `ownerOf()` function consistently returning Alice's account address, Alice remains the true owner.
- Consequently, Bob cannot exert control over account Y.

Recommendation

This issue underscores the importance of conducting thorough audits and evaluations of any third-party contracts that your contract interacts with, including ERC721 token contracts. It is also recommended to consider implementing additional safeguards within the contract to detect and mitigate such potential exploits.

Alleviation

[Client]: Issue acknowledged. The security model of ERC-6551 assumes that token contracts conform to the ERC-721 specification and accurately report token ownership. Users should only use token bound accounts owned by tokens whose contract implementations are trusted. No changes will be made for the current version.

EXA-02 | THIRD-PARTY DEPENDENCY USAGE

Category	Severity	Location	Status
Design Issue	Minor	src/examples/simple/SimpleERC6551Account.sol (v1); src/examples/upgradeable/ERC6551AccountUpgradeable.sol (v1)	Acknowledged

Description

The contracts are serving as the underlying entities to interact with one or more third party protocols. The scope of the audit treats third party entities as black boxes and assume their functional correctness. However, in the real world, third parties can be compromised and this may lead to lost or stolen assets. In addition, upgrades of third parties can possibly create severe impacts, such as increasing fees of third parties, migrating to new LP pools, etc.

```
                address contractAddress,
```

- The contract `ERC6551AccountUpgradeable` interacts with third party contract with `IERC721` interface via `contractAddress`.

```
        (uint256 chainId, address tokenContract, uint256 tokenId) = this.token();
```

- The contract `SimpleERC6551Account` interacts with third party contract with `IERC721` interface via `tokenContract`.

Recommendation

The auditors understood that the business logic requires interaction with third parties. It is recommended for the team to constantly monitor the statuses of third parties to mitigate the side effects when unexpected activities are observed.

Alleviation

[Client]: Issue acknowledged. The security model of ERC-6551 assumes that token contracts conform to the ERC-721 specification and accurately report token ownership. No changes will be made for the current version.

ERA-01 MISUSE OF `extcodecopy()` IN `salt()` AND `token()` FUNCTIONS

Category	Severity	Location	Status
Coding Issue	Informational	src/lib/ERC6551AccountLib.sol (v1): 42, 53	Resolved

Description

The `salt()` function uses `extcodecopy()` with `0x4d` as the last parameter. This is intended to copy `0x20` bytes from code at position `0x2d` into the variable `footer`. However, as per the Ethereum Virtual Machine (EVM) Opcodes documentation, the last parameter of `extcodecopy()` should represent the length of the section to be copied rather than the "end position + 1". This confusion also exists in the `token()` function.

```
48                                                    function salt()
                                                   49
                                                      internal view returns
                                                      (uint256) {
        bytes memory footer = new bytes(0x20);

        assembly {
            // copy 0x20 bytes from beginning of footer
            extcodecopy(address(), add(footer, 0x20), 0x2d, 0x4d)
        }

        return abi.decode(footer, (uint256));
    }
```

Recommendation

Although the function is working as intended due to the length of footer being `0x20` and the last parameter `0x4d` being larger than `0x20`, it's recommended to correct this for the sake of accuracy and readability. Updating the last parameter of `extcodecopy()` to accurately represent the length of the code to be copied can prevent potential misunderstandings or errors.

Alleviation

The client has resolved this issue in the commit https://github.com/erc6551/reference/tree/d79a49e99fee1154a2023bd164850c5d0c34c208/src/lib.

SRC-01 | MISSING ZERO ADDRESS VALIDATION

Category	Severity	Location	Status
Volatile Code	Informational	src/ERC6551Registry.sol (v1): 21; src/examples/upgradeable/ERC6551AccountProxy.sol (v1): 11	Resolved

Description

The following addresses are not validated before being used, potentially allowing the implementation of a proxy contract to be the zero address.

```
11              defaultImplementation = _defaultImplementation;
```

- `_defaultImplementation` is not zero-checked before being used.

```
21              implementation,
```

- `implementation` is not zero-checked before being used.

Recommendation

It is recommended to add a zero-check for the passed-in address value to prevent unexpected errors.

Alleviation

The client has resolved the issue in the commit https://github.com/erc6551/reference/commit/f2e98f272e4133b161e5617896578e26189c8e1e.

APPENDIX | MANIFOLD REFERENCE

Finding Categories

Categories	Description
Coding Issue	Coding Issue findings are about general code quality including, but not limited to, coding mistakes, compile errors, and performance issues.
Volatile Code	Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases and may result in vulnerabilities.
Centralization	Centralization findings detail the design choices of designating privileged roles or other centralized controls over the code.
Design Issue	Design Issue findings indicate general issues at the design level beyond program logic that are not covered by other finding categories.

Checksum Calculation Method

The "Checksum" field in the "Audit Scope" section is calculated as the SHA-256 (Secure Hash Algorithm 2 with digest size of 256 bits) digest of the content of each file hosted in the listed source repository under the specified commit.

The result is hexadecimal encoded and is the same as the output of the Linux "sha256sum" command against the target file.

DISCLAIMER | CERTIK

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to you ("Customer" or the "Company") in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes, nor may copies be delivered to any other person other than the Company, without CertiK's prior written consent in each instance.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is 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 assessment. This report does 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.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing 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.

The assessment services provided by CertiK is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.

ALL SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF ARE PROVIDED "AS IS" AND "AS AVAILABLE" AND WITH ALL FAULTS AND DEFECTS WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED UNDER APPLICABLE LAW, CERTIK HEREBY DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS. WITHOUT LIMITING THE FOREGOING, CERTIK SPECIFICALLY DISCLAIMS ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AND ALL WARRANTIES ARISING FROM COURSE OF DEALING, USAGE, OR TRADE PRACTICE. WITHOUT LIMITING THE FOREGOING, CERTIK MAKES NO WARRANTY OF ANY KIND THAT THE SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF, WILL MEET CUSTOMER'S OR ANY OTHER PERSON'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULT, BE COMPATIBLE OR WORK WITH ANY SOFTWARE, SYSTEM, OR OTHER SERVICES, OR BE SECURE, ACCURATE, COMPLETE, FREE OF HARMFUL CODE, OR ERROR-FREE. WITHOUT LIMITATION TO THE FOREGOING, CERTIK PROVIDES NO WARRANTY OR

UNDERTAKING, AND MAKES NO REPRESENTATION OF ANY KIND THAT THE SERVICE WILL MEET CUSTOMER'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULTS, BE COMPATIBLE OR WORK WITH ANY OTHER SOFTWARE, APPLICATIONS, SYSTEMS OR SERVICES, OPERATE WITHOUT INTERRUPTION, MEET ANY PERFORMANCE OR RELIABILITY STANDARDS OR BE ERROR FREE OR THAT ANY ERRORS OR DEFECTS CAN OR WILL BE CORRECTED.

WITHOUT LIMITING THE FOREGOING, NEITHER CERTIK NOR ANY OF CERTIK'S AGENTS MAKES ANY REPRESENTATION OR WARRANTY OF ANY KIND, EXPRESS OR IMPLIED AS TO THE ACCURACY, RELIABILITY, OR CURRENCY OF ANY INFORMATION OR CONTENT PROVIDED THROUGH THE SERVICE. CERTIK WILL ASSUME NO LIABILITY OR RESPONSIBILITY FOR (I) ANY ERRORS, MISTAKES, OR INACCURACIES OF CONTENT AND MATERIALS OR FOR ANY LOSS OR DAMAGE OF ANY KIND INCURRED AS A RESULT OF THE USE OF ANY CONTENT, OR (II) ANY PERSONAL INJURY OR PROPERTY DAMAGE, OF ANY NATURE WHATSOEVER, RESULTING FROM CUSTOMER'S ACCESS TO OR USE OF THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS.

ALL THIRD-PARTY MATERIALS ARE PROVIDED "AS IS" AND ANY REPRESENTATION OR WARRANTY OF OR CONCERNING ANY THIRD-PARTY MATERIALS IS STRICTLY BETWEEN CUSTOMER AND THE THIRD-PARTY OWNER OR DISTRIBUTOR OF THE THIRD-PARTY MATERIALS.

THE SERVICES, ASSESSMENT REPORT, AND ANY OTHER MATERIALS HEREUNDER ARE SOLELY PROVIDED TO CUSTOMER AND MAY NOT BE RELIED ON BY ANY OTHER PERSON OR FOR ANY PURPOSE NOT SPECIFICALLY IDENTIFIED IN THIS AGREEMENT, NOR MAY COPIES BE DELIVERED TO, ANY OTHER PERSON WITHOUT CERTIK'S PRIOR WRITTEN CONSENT IN EACH INSTANCE.

NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS.

THE REPRESENTATIONS AND WARRANTIES OF CERTIK CONTAINED IN THIS AGREEMENT ARE SOLELY FOR THE BENEFIT OF CUSTOMER. ACCORDINGLY, NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH REPRESENTATIONS AND WARRANTIES AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH REPRESENTATIONS OR WARRANTIES OR ANY MATTER SUBJECT TO OR RESULTING IN INDEMNIFICATION UNDER THIS AGREEMENT OR OTHERWISE.

FOR AVOIDANCE OF DOUBT, THE SERVICES, INCLUDING ANY ASSOCIATED ASSESSMENT REPORTS OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

CertiK | Securing the Web3 World

Founded in 2017 by leading academics in the field of Computer Science from both Yale and Columbia University, CertiK is a leading blockchain security company that serves to verify the security and correctness of smart contracts and blockchain-based protocols. Through the utilization of our world-class technical expertise, alongside our proprietary, innovative tech, we're able to support the success of our clients with best-in-class security, all whilst realizing our overarching vision; provable trust for all throughout all facets of blockchain.