CyStack

DATA SHEET

Smart Contract Audit

Trustworthy smart contract audits for a secure blockchain

Security Challenges

Blockchain technology is designed to be secure, but there are still several challenges to ensuring the security of a blockchain. Some of these challenges include:

- **Smart contract vulnerabilities**: Smart contracts, which are self-executing contracts with the terms of the agreement written directly into code, can contain vulnerabilities that can be exploited by attackers. This is especially true if the code has not been thoroughly audited or if the contract has not been properly tested.
- **51% attack**: In a 51% attack, a group of attackers gains control of more than 50% of the computational power of a blockchain network, allowing them to manipulate the network and potentially reverse or block legitimate transactions.
- **Sybil attack**: A Sybil attack is when a malicious actor creates multiple identities or nodes in a network in order to gain control of a significant portion of the network's computational power.
- **Double-spending**: In a double-spending attack, an attacker is able to spend the same digital currency or token more than once by creating a copy of the digital asset.
- **Privacy and data leakage**: A privacy and data leakage attack is when an attacker accesses sensitive data stored on the blockchain, such as personal information or financial data.
- **Side-Channel Attack**: A side-channel attack is an attack that is exploiting some side information that is not part of the main communication channel, such as power consumption or electromagnetic radiation.

These are just a few examples of the security challenges that blockchain technology faces. The field is still evolving, and security researchers and developers are constantly working to identify and mitigate new threats.

Customer Benefits

- Identify and address vulnerabilities and potential security risks before the launching
- Build trust with stakeholders and customers
- Avoid costly security breaches and downtime in the future
- Increase performance of the blockchain network it is deployed on

Overview

A smart contract audit is a comprehensive review of a smart contract's code to ensure that it functions as intended, is secure, and is free of vulnerabilities. Smart contracts are self-executing digital contracts that are based on blockchain technology, and they are used to automate transactions and enforce the terms of an agreement. Smart contract audits are typically performed by security experts who are well-versed in blockchain technology and smart contract programming languages.

The primary goal of a smart contract audit is to identify any security vulnerabilities or weaknesses in the code that could be exploited by attackers. The audit typically involves a thorough analysis of the smart contract's design, implementation, and security measures, as well as testing and analysis to identify any potential issues. The audit may also include a review of the smart contract's documentation, interviews with developers and stakeholders, and recommendations for security, functionality, and efficiency improvement to their smart contracts.

By having their smart contracts audited by a reputable third-party auditor, developers can demonstrate to stakeholders that they take security and reliability seriously and boost their credibility and reputation within the blockchain community, which can be important for attracting investors and partners. In some cases, smart contracts must comply with regulatory and compliance requirements. Also, smart contract audits can help ensure that the code meets and complies with standard and regulatory requirements if they are mandatory.

Key Features

- Code review
- Automated testing powered by SafeChain.org
- Penetration testing
- Governance testing
- Performance testing
- Interoperability testing
- Bug Bounty program powered by WhiteHub.net

How CyStack Helps

The CyStack Audit Team is a group of highly skilled security testers who use a goal-oriented approach to testing, refined through years of experience and extensive testing. Our team members have a unique blend of app development and security testing expertise, enabling them to conduct comprehensive security evaluations that uncover potential risks for organizations. Members of this team are also regular speakers at world-known cybersecurity conferences and also talented bug hunters who discovered many critical vulnerabilities in the products and are acknowledged in the Hall of Fame of global tech giants such as IBM, HP, Microsoft, Alibaba, Sea Group, etc.

Our team has extensive experience in identifying and mitigating vulnerabilities in smart contract code, which can help prevent potential attacks on your blockchain system. We use a combination of manual code review and automated tools to ensure that your smart contracts are free from bugs, errors and any potential vulnerabilities. Our team also has a deep understanding of the various blockchain platforms, such as Ethereum, EOS, TRON and more, which means that we can provide audits for smart contracts on any blockchain platform.

Methodology

Generally, the methodology for security auditing a smart contract typically includes several steps, such as:

1. **Preparation**: This includes setting the scope of the audit, identifying the stakeholders, and gathering all relevant documentation, such as the whitepaper, smart contract code, and design documents.
2. **Threat modelling**: This step involves identifying potential threats and vulnerabilities that may affect the smart contract. This includes analyzing the smart contract's functionality, data flow, and external interactions to identify any potential attack vectors.
3. **Code review**: This step involves reviewing the smart contract code to identify any bugs, errors, or vulnerabilities. This can be done manually by an experienced developer or by using automated tools to help identify potential issues. CyStack also uses SafeChain, an automated blockchain vulnerability scanner built by our team, for this stage.
4. **Test execution**: This step involves executing the smart contract on a test network and performing various types of testing, such as unit testing, functional testing, and security testing.
5. **Reporting**: This step involves documenting the findings of the audit and providing a report that includes an overview of the audit, a list of identified issues, and recommendations for remediation.
6. **Remediation**: This step involves implementing any recommended changes to the smart contract code to fix identified issues and vulnerabilities.
7. **Retesting**: This step involves re-executing the smart contract on the test network to ensure that the identified issues have been resolved and that the smart contract is now secure.

Use Cases

- Financial Services
- Digital Identity
- Business Management
- Healthcare
- Real Estate
- Supply Chain Management
- Gaming
- Digital Marketplace
- Corporate and Governance
- Crowdfunding

During a smart contract audit, the following types of vulnerabilities are tested but not limited in:

1. **Reentrancy**: This type of vulnerability occurs when a smart contract allows an attacker to repeatedly call it and extract its value multiple times.
2. **Unchecked call return value**: This type of vulnerability occurs when a smart contract does not properly check the return value of a call to another contract, which can lead to the execution of malicious code.
3. **Unchecked user input**: This type of vulnerability occurs when a smart contract does not properly validate user input, which can lead to the execution of malicious code or the manipulation of data.
4. **Unchecked math operations**: This type of vulnerability occurs when a smart contract uses math operations that can overflow or underflow, leading to unintended results.
5. **Unchecked external calls**: This type of vulnerability occurs when a smart contract calls an external contract without properly checking the return value, which can lead to the execution of malicious code or the manipulation of data.
6. **Integer overflow and underflow**: This type of vulnerability occurs when a smart contract does not properly handle large numbers, which can lead to unintended results.
7. **Unsecured data storage**: This type of vulnerability occurs when a smart contract stores sensitive data in an unsecured manner, which can lead to data breaches.
8. **Timestamp dependence**: This type of vulnerability occurs when a smart contract is dependent on the timestamp provided by the blockchain network, which can be manipulated by an attacker.
9. **Unsecured randomness**: This type of vulnerability occurs when a smart contract uses an insecure random number generator, which can be predicted by an attacker.
10. **Access control**: This type of vulnerability occurs when a smart contract does not properly implement access control, which can allow unauthorized parties to access or manipulate data.

Supported Platforms

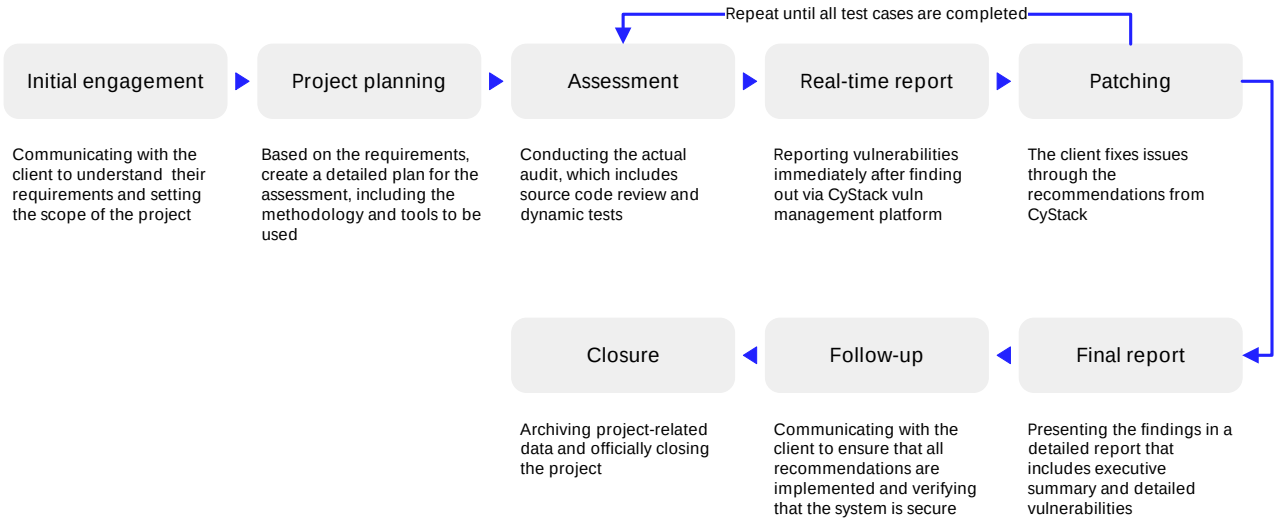
CyStack can audit smart contracts on numerous blockchain networks or chains, such as:

- **Ethereum**: The most popular blockchain network that supports smart contracts written in Solidity or Vyper, which is a python-like programming language.
- **BNB Smart Chain (BSC)**, previously Binance Smart Chain: A blockchain network that runs in parallel to the BNB Beacon Chain and supports smart contracts written in Solidity.
- **TRON**: An open-source public Ethereum-compatible blockchain platform that supports Solidity smart contracts.
- **Polygon**, formerly Matic Network: A sidechain scaling solution that runs alongside the Ethereum blockchain with smart contracts written in Solidity or Vyper.
- **Avalanche**: An open-source platform that features 3 built-in blockchains, one of which is Contract Chain (C-Chain), for launching decentralized applications (dApps) and enterprise blockchain deployments. It is EVM-compatible, and hence supports smart contracts written in Solidity.
- **Solana**: An efficient and speed-first blockchain network that is constructed with programs (smart contracts), written in Rust or C/C++.
- **NEAR**: A high-performance blockchain network that supports smart contracts written in JavaScript, Rust or AssemblyScript.
- **EOSIO**: A blockchain network that supports smart contracts written in C++.
- **NEO**: The most developer-friendly blockchain that supports smart contracts written in various programming languages, including C#, Python, Go, Java and TypeScript.
- **Algorand**: A blockchain platform that supports smart contracts written in multiple programming languages, including Python and a JavaScript-like language called Reach.
- **Aptos**: A Layer 1 blockchain with resource objects and Move programming language for smart contracts.
- **Sui**: The first permissionless Layer 1 blockchain that is written in Rust and supports smart contracts written in the Move programming language.

We also support auditing dApps and enterprise blockchains that are created and deployed with the following platforms:

- **Cosmos**: The key Layer 0 blockchain that connects different blockchains into a meta-blockchain system called interchain. Cosmos provides an SDK for building dApps and Layer 1 chains in Go.
- **Polkadot**: The first fully-sharded blockchain consists of a main chain called the Relay Chain and shards called parachains. With Parachain Development Kit (PDK), developers can build parachains written in Rust.
- **Quorum**: An open-source, permissioned blockchain protocol based on Ethereum that allows developers to deploy networks with contracts written in Solidity or Vyper.
- **Hyperledger**: An open-source collaborative effort created to advance cross-industry blockchain technologies, which provides various distributed ledger frameworks, supporting different programming languages, including Go, Python, Rust, Java, JavaScript, C++, C#, Objective-C and Swift.
- **Corda**: A permissioned peer-to-peer (P2P) distributed ledger technology (DLT) platform that is primarily used by businesses in finance-related industries to build dApps and blockchains written in Kotlin or Java.
- **Hedera Hashgraph**: A open-source public distributed ledger based on the Hashgraph algorithm, which is an alternative to blockchain. It provides SDKs supporting multiple programming languages, such as Java, JavaScript/TypeScript, Go, Rust, C++ and Swift.

Flow To Work With Clients



About CyStack

CyStack is an innovative company in the field of cybersecurity in Vietnam. We are a pioneer in building next gen security products for businesses and individuals. Our solutions focus on data protection, cyber attack prevention, and security risk management.

For more information, please call **(+84) 247 109 9656** or send an email to **contact@cystack.net** to speak to CyStack security specialist.
cystack.net