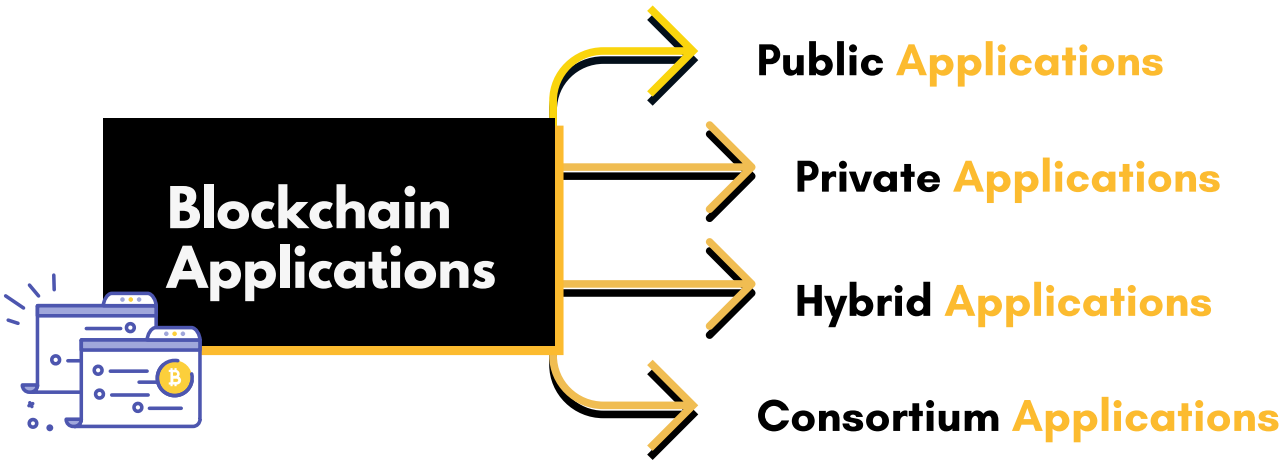
astra

Blockchain Security Testing Checklist

Blockchain is most simply defined as a decentralized, distributed ledger that records the provenance of a digital asset. By inherent design, the data on a blockchain is unable to be modified, which makes it a legitimate disruptor for industries like payments, cybersecurity and healthcare. It is an open-source tech that is currently used by many well-known companies such as British Airways, UPS, FedEx, Walmart, etc. to build blockchain-based applications for customers.

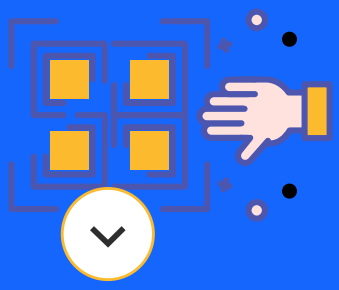


Blockchain is a vast database running on millions of devices and open to anyone. The applications of blockchain technology are limitless. According to statistics, the global blockchain market is expected to be worth $20 billion in the year 2024. Currently, 69 percent of banks are experimenting with blockchain technology to make their services more secure, seamless and transparent.

This document guides Blockchain application developers on how to attain the maximum level of protection for their app framework and the sensitive data stored within, by conducting an effective security audit. A vulnerability assessment & penetration testing checklist for Blockchain security will ensure that you don't miss any crucial area of your Blockchain apps and ensure they are developed and configured correctly with the highest level of security.

Best Practices for Blockchain Security:

- Treat your underlying blockchain solution as critical infrastructure.
- Define and enforce endorsement policies based on business contracts.
- Enforce identity and access (IAM) controls to access the blockchain solution and data.
- Define policies that ensure the right level of access to the right individual for the right use.
- Implement appropriate tokens like OAUTH, OIDC, and SAML2 to perform user authentication, verification, and authorization.
- Enforce the hardware security module (HSM) to store identity keys.
- Use privileged access management (PAM) solution for escalated actions.
- Use API security best practices to safeguard API-based transactions.
- Leverage a secrets-store for both application and privileged access.
- Adopt a data classification approach to safeguard data/information.
- Use privacy-preserving technologies for sensitive information.
- Protect applications & networks from vulnerabilities and safeguard data.
- Leverage Trusted Platform Modules (TPMs) for sensitive code execution.
- Secure communications both internally and externally over mutual or standard TLS.
- Mandate multi-factor authentication.
- Use strong cryptographic key/certificate management.
- Leverage security incident and event management (SIEM).
- Perform full-scope penetration testing and vulnerability assessment.
- Adopt and enforce an industry standards certification.
- Ensure that compliance and legal controls are in place.



Blockchain Penetration Testing Phases

Phase 1

Information Gathering and Threat Modeling

In this phase, you can understand and analyze the business and functional requirements.

This phase includes:

- **Understanding Blockchain architecture**
- **Finding threat entry points within the organization**
- **Gathering of publicly available data on potential exploits**
- **Evaluate Smart Contract Business Logic**
- **Setting objectives for conducting security testing**
- **Full test strategy designing**
- **Checking Compliance readiness**
- **Setting up the testing environment**
- **Creation of test data**

Phase 2

Testing / Discovery

In this phase, you can utilize the information gained in the first phase to perform the hands-on testing of your organization's blockchain to determine its maturity level measured against best practices and industry standards.

This phase includes:

- **API Security Testing** link
- **Functional Testing**
- **Automatic and Manual Blockchain Security Analysis**
- **Blockchain Static and Dynamic Testing**
- **Network Vulnerability Assessment**
- **Application Vulnerability Assessment**
- **Blockchain Integrity Assessment**
- **Documenting Testing DIscoveries**

Phase 3

Exploitation

In this phase, the goal is to leverage any vulnerabilities or security loopholes discovered in the second phase. This is often done manually to weed out false positives. The exploitation part is also used to exfiltrate information from the target and to maintain persistence.

This phase includes:

- **Verifying Security Weaknesses and Vulnerabilities**
- **Exploiting Security Weaknesses and Vulnerabilities**
- **Network Penetration Testing** link
- **Web Application Penetration Testing** link
- **Test against Social Engineering Attacks**
- **Review and Document Discoveries**

Blockchain Penetration Testing Checklist

- [x] Test for Unencrypted Private Data On-Chain
- [x] Test for Code With No Effects
- [x] Test for Message call with the hardcoded gas amount
- [x] Test for Unexpected Ether balance
- [x] Test for Hash Collisions With Multiple Variable Length Arguments
- [x] Testing for Presence of unused variables
- [x] Test for Right-To-Left-Override control character (U+202E)
- [x] Test for Typographical Error
- [x] Test for DoS With Block Gas Limit
- [x] Test for Arbitrary Jump with Function Type Variable
- [x] Test for Insufficient Gas Griefing
- [x] Test for Incorrect Inheritance Order
- [x] Test for Writing to Arbitrary Storage Location
- [x] Test for Requirement Violation
- [x] Test for Lack of Proper Signature Verification
- [x] Test for local Missing Protection against Signature Replay Attacks
- [x] Test for Weak Sources of Randomness from Chain Attributes
- [x] Test for Shadowing State Variables
- [x] Test for Incorrect Constructor Name
- [x] Test for Signature Malleability
- [x] Test for Blocking values as a proxy for time
- [x] Test for Authorization through tx.origin
- [x] Test for Transaction Order Dependence
- [x] Test for DoS with Failed Call

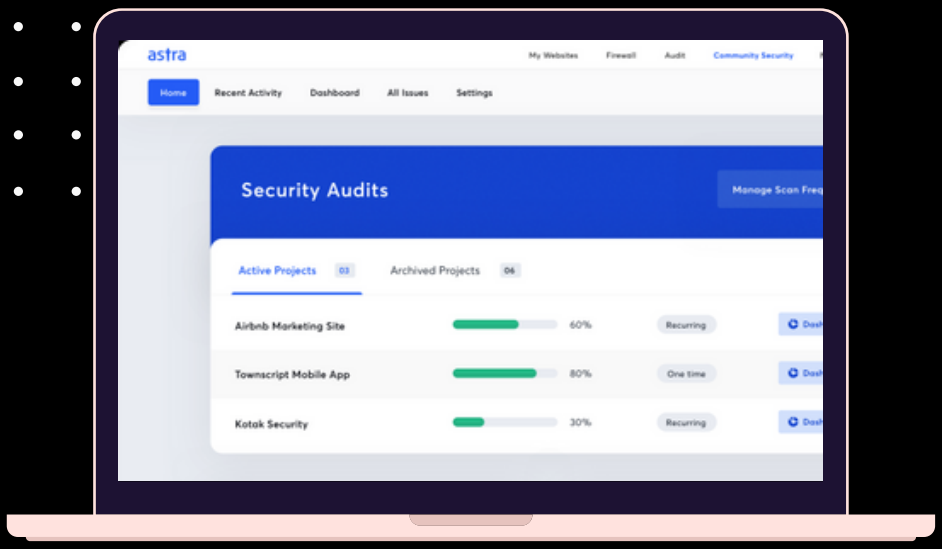
- [x] Test for Delegatecall to Untrusted Callee
- [x] Test for Use of Deprecated Solidity Functions
- [x] Test for Assert Violation
- [x] Test for Uninitialized Storage Pointer
- [x] Test for State Variable Default Visibility
- [x] Test for Reentrancy
- [x] Test for Unprotected SELFDESTRUCT Instruction
- [x] Test for Unprotected Ether Withdrawal
- [x] Test for Unchecked Call Return Value
- [x] Test for Floating Pragma
- [x] Test for Outdated Compiler Version
- [x] Test for Integer Overflow and Underflow
- [x] Test for Function Default Visibility

Blockchain Security Testing Tools

- **Mythril** - Open-source EVM bytecode security analysis tool.
- **SWC-registry** - The Smart Contract Weakness Classification Registry (SWC Registry) is an implementation of the weakness classification.
- **MythX** - Security verification platform and tools ecosystem for Ethereum developers.
- **Echidna** - It's a Haskell program designed for fuzzing/property-based testing of Ethereum smarts contracts.
- **Manticore** - Symbolic execution tool on Smart Contracts and Binaries.
- **Oyente** - Alternative static smart contract security analysis.
- **Securify** - Security scanner for Ethereum smart contracts.
- **SmartCheck** - Static smart contract security analyzer.
- **Octopus** - It's a security analysis framework for WebAssembly module and Blockchain Smart Contract.
- **Surya** - Surya is a utility tool for smart contract systems.
- **Solgraph** - Visualise Solidity control flow for smart contract security analysis.
- **Solidity security blog** - Comprehensive list of known attack vectors and common anti-patterns.
- **Awesome Buggy ERC20 Tokens** - A Collection of vulnerabilities in ERC20 Smart Contracts with tokens affected.

***Hyperlinks are attached to the respective tool names.**

Looking for a professional Security Audit & VAPT for your Blockchain? Astra Security can help.



astra

Security audits based on industry leading practices such as OWASP, OSSTMM, WASC, CREST, NIST etc.

Astra Security's vulnerability management dashboard comes with a birds eye view for management keeping you always on the top of security assessment status.

Video PoCs, selenium scripts & collaboration with security team enables your developers to fix the vulnerabilities in record time. With Astra Security, VAPT takes 40% less time than other solutions.

Contact us to get a free demo

hello@getastra.com

fb.com/getAstra

Schedule a Call

@getastra

www.getastra.com

linkedin.com/company/getastra

References: https://github.com/ConsenSys/ethereum-developer-tools-list#security-tools
https://developer.ibm.com/technologies/blockchain/articles/how-to-secure-blockchain-solutions/