Forta: a decentralized runtime security solution for automated threat detection and prevention on smart contracts

Last updated December 2021. Published July 2022 with the permission of OpenZeppelin.This whitepaper was used to explain and provide detail of the technical architecture of the Forta Network and the potential use cases of the planned Forta Network beyond the public code underlying the protocol that had been shared with the community. It describes the mission of Forta, as originally articulated by OpenZeppelin. It should not be considered a source of truth for the ultimate Forta Network design, and it should not be relied upon as a roadmap for any future development of Forta by OpenZeppelin, the Forta Foundation or any other members of the Forta community. At the date of publishing, governance over Forta's future evolution is dictated by its community. Neither OpenZeppelin or Forta Foundation has updated this document, nor do they undertake any obligation to update it in the future. For a more fulsome description of the current structure and functionality of Forta, please see https://docs.forta.network/en/latest/.

Web3 has seen amazing growth in the past 2 years, including huge growth in transaction volume and Total Value Locked. However, there have also been many highly-publicized hacks and exploits. Such events are cause for concern among new and experienced users alike. These security concerns also extend to the exploding NFT market and other blockchain-based solution areas such as supply chain, e-commerce, public records, utility management or elections.

Measures prior to release, such as security testing and audits, are critically important but have proven insufficient to identify all risks and potential exploits. Also, slow or delayed response to attacks and zero-day vulnerabilities has led to larger losses. Therefore, it has become clear that thorough system scanning for threat detection and prevention -- often referred to collectively as runtime security -- must also be implemented on smart contracts to mitigate risks and losses.

Forta exists to address the critical need for public runtime threat detection and prevention on smart contracts, through a multi-chain smart contract protocol and associated node software. Forta will support a decentralized network of security detection bots and nodes for continuous scanning of transactions and block-by-block state changes on smart contracts, with the ability to monitor for and alert on threats discovered on L1s, L2s, simulation networks, or mempools. Forta will be secured and governed by smart contracts and the FORT utility and governance token.

The Need for Smart Contract Threat Detection and Prevention

In 2021, over $2 billion was lost in a variety of Web3 hacks and exploits. In most cases, the losses could have been avoided or dramatically reduced through an early threat detection and rapid response system. Such a system would afford users greater protection by providing them with a warning of the potential threat and give them the time to act accordingly.

While large losses from hacks and exploits have been much publicized and discussed, there are many other stories, some told and some untold, of protocols that discovered critical vulnerabilities and then worked nervously to address the issues before they were attacked.

These protocols would have greatly benefited from a way to protect their systems against zero-day attacks.

Due to the complexity of smart contracts, composability, and the rapidly changing landscape of smart contract programming languages and L1s and L2s, it is impossible to catch all risks and potential exploits before new protocols, apps, and versions are released. The risks also extend to the smart contract services that protocols and users rely on such as stablecoins, exchanges, oracles and smart wallets which are all subject to attacks too.

Technical Background

Centralized financial service providers rely on many commercial solutions for threat detection and prevention, including antivirus scanners (for example BitDefender, Trend Micro, Kaspersky), network traffic and log scanners (for example AWS, Azure, IBM, Secureworks, Imperva, Splunk, Rapid7, Palo Alto Networks and more), and homegrown security event scanners (for example those built using Amazon Kinesis or Apache Kafka). These solutions are fundamental to security and risk mitigation in centralized financial services, e-commerce, and many other industries. Typically, these solutions involve the use of detection bots and scanners and (more recently) machine learning.

Many security solutions exist to find issues in source code, libraries, and binaries, but due to the complexity of software and networks they are not able to discover all vulnerabilities or protect against all variety of hacks. Runtime security solutions are designed to detect and thwart attacks and they often help to discover vulnerabilities after the fact. Runtime scanning and anomaly detection has therefore proven critical to mitigate the risks associated with software and computer networks, and is considered a key part of "layered security" best practices.

Smart contracts, especially those running on public and permissionless platforms, present new and different types of attack and exploit vectors. As a result of composability -- the ability to integrate different smart contract protocols together -- these attack vectors can also change rapidly. Furthermore, with the increase in multi-chain smart contract deployments comes an increase in vulnerabilities across multiple chains.

Most of the effort on smart contract security until now has focused on vulnerability detection. Since smart contracts are immutable after deployment, there have rightly been great efforts to ensure that the contracts do not have serious vulnerabilities. However, as the accumulated losses have shown, vulnerability detection efforts alone cannot protect smart contracts from attacks and exploits. Despite continued best efforts in vulnerability avoidance, serious risks for users and the protocols themselves do and will remain. By definition, most of these runtime risks are unknown or unanticipated until they are discovered or exploited.

Bug bounties have proven useful to finding issues after deployment, however bounties alone cannot solve issues with zero-day exploits (e.g. protocols still have a problem determining how to block reported vulnerabilities until they are fixed). And in many cases, as we have seen in

DeFi, the value of exploitation is so high (with tens of millions $ at risk) that black hat hackers have much higher incentives than the bounties offered to white hats.

As in traditional industries, in order to reduce risks and losses, it has become clear that we must complement smart contract vulnerability reduction efforts with runtime security focused on detecting and thwarting attacks and exploits. While some suitable solutions exist for scanning and monitoring for individual protocols and teams, there is no decentralized exploit detection solution capable of covering all smart contracts on all L1s and L2s. Also, in a world of composable and multi-chain smart contracts, individual teams are not capable of implementing all the threat detection they require to protect the value within their contracts. And finally, in order to protect public networks of permissionless, unstoppable, censorship-resistant smart contracts, a suitable runtime security network should itself be public, permissionless, unstoppable, and censorship-free.

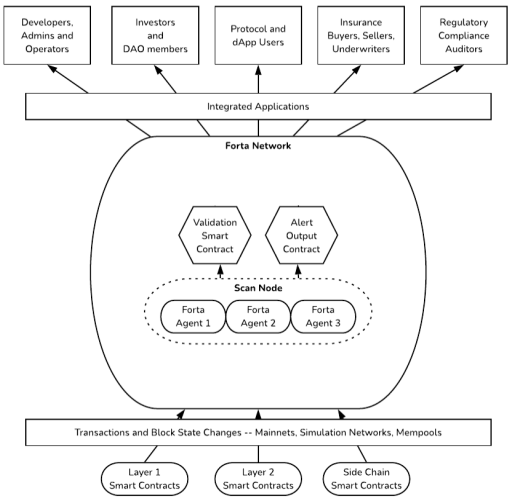
Forta Technical Overview

Forta is a permissionless network of threat detection bots and scan nodes working with multi-chain smart contracts launched in 2021, capable of supporting any EVM compatible L1 or L2.

Forta node software, run by independent node operators, acts as a decentralized network of continuous scanners and threat information providers. Scan nodes scan all transactions and block-by-block state changes and execute Forta bots which contain the logic for threat detection (see next section for more information on Forta bots). When a Forta bot discovers a potential threat, the scan node broadcasts an alert to the Forta analysis layer which provides a public API of all alert data along with publicly verifiable receipts of all scan node results on IPFS. Forta bots are assigned to multiple scan nodes according to parameters defined in the Forta smart contracts, and alert subscribers can choose the amount of consensus they require when consuming alerts.

Use of the Forta Network will create a public solution for runtime security. Threat detection alerts emitted by Forta may be integrated into a variety of applications for a wide variety of interested parties which may include:

- Developers and administrators who want to avoid losses on their protocols
- Investors who want to evaluate protocol risks and protect their holdings
- DAO members who want to protect their protocol holdings and investments
- Users of protocols and dApps who want to ensure security of the applications
- Insurance underwriters who want to analyze and monitor the risk of smart contracts
- Insurance buyers and providers who want to automate coverage claims
- Government regulators and industry auditors who need to ensure that there are sufficient security protections for relevant smart contracts



Forta node software is written in Golang and will be able to run on any standard server. Forta node runners are required to provide the server and need to have an internet provider along with connectivity to the Forta smart contracts and any other L1 or L2 they support. Each Forta node automatically generates a private key which is used for registration and security. The Forta node software is initially distributed under a license that restricts production use to unmodified versions connected to the network itself (in order to protect against free-riders and forked competitors) and will become fully open source after a fixed period of time.

After an initial development phase, anyone will be permitted to become a Forta node runner. Node runners will be required to stake a security bond using the native FORT network token. Each node's staked bond will be subject to slashing if the node runner fails to carry out the required functions or meet the quality of service standards defined by the Forta community.

Forta Bots

Forta bots perform the individual checks on transaction and smart contract state changes to detect security issues, attacks or exploits, and emit alerts when issues are discovered. Forta bots may have specific logic tied to specific protocols and smart contracts, and may utilize time-series or machine learning and other techniques for anomaly and issue detection. It is expected that Forta bots will continuously evolve and adapt as more smart contracts are deployed and new attack and exploit vectors are identified by the Forta community.

Forta bots are created by independent developers and security researchers, who register their bots with the network and who will be required to stake tokens for security.

Forta bots are deployed via Docker containers which are registered by the creators in Forta smart contracts. Containers are used to ensure that the scan nodes can run the Forta bots securely without the possibility that one set of Forta bots maliciously or inadvertently affect users, nodes, or other bots. Utilizing containers also allows the bot developers to include

whatever capabilities they need to support their bots, such as in-memory state storage or machine-learning modules.

The bot logic may be defined in any Docker-compatible programming language such as JavaScript, Rust, Go or Python. Libraries are provided for interacting with the scan node services (for example to obtain external data or emit alerts). Bot and container metadata specify which L1s, L2s, and protocols are supported. Some bots may support multiple L1s, L2s and protocols.

Bots are executed in a secure runtime environment within each scan node that also provides them access to L1s, L2s, and web services. The scan node runtime environment continuously scans L1s and L2s for new transactions and block changes, automatically filters transactions according to the requirements identified for each agent container, and then invokes each bot for execution.

When a scan node executes a bot, it passes the relevant log data for each transaction and block to be analyzed. The bot then executes within the containerized environment, and each bot may emit alerts and details according to its logic.

An unlimited number of specialized detection algorithms and alerts may be implemented using Forta bots. These may also be specific to protocols and solution domains. Examples of issues that might be detected by Forta agents include:

- Anomalous transactions (e.g. very large amounts or flash loans and large gas payments)
- Rapid vault liquidations, rapid changes in collateralization
- Anomalous administration transactions
- Anomalous withdrawals, or unusual volume of anomalous transactions
- Transactions that target identified vulnerabilities that have not been remediated

Governance and the FORT Token

The development and operations of Forta will rely on a wide range of contributors. In order to coordinate activities and ensure security and reliability of the network, Forta will use the native FORT token for staking and slashing. Governance will ultimately be conducted through voting by token holders or their delegates. The token holders and their delegates will govern things like:

- Change proposals on Forta node software and smart contracts
- Policies for slashing tokens staked by participants
- Use of token holdings and any treasury funds