FortaKnight PRD Document

Team Lead: Nicholas Brown

Members: John Lin, Andy Wu, Khalid Mihlar, Alejandro Rojas Rodriguez

Introduction:

Web 3.0 and decentralized finance (DeFi) implements cutting edge blockchain technologies to provide user's with financial instruments that are not reliant on centralized financial institutions through the use of smart contracts on a blockchain network. This technology has the potential to revolutionize the financial experience of the user but also presents new, unprecedented challenges. The pseudonymity that blockchain technologies offer is a double edged sword as it allows hackers to steal large sums of money with minimal traceability. According to a report by Elliptic, DeFi users lost an estimated $10.5 billion to theft in 2021. These hacks are performed primarily by deploying an attacker smart contract that exhibits adversarial behavior on a blockchain network. Our project aims to reinforce the security that the Forta network provides by developing a bot that is capable of detecting attacker smart contracts through both static and dynamic analysis methods. The four attack stages consist of funding, preparation, exploitation, and money laundering. Our bot will seek to detect attacks before the exploitation stage because funds are virtually impossible to recover after they have been stolen due to the nature of the blockchain.

We are researching pre-existing malware detection strategies used in other areas of computing and are planning to apply these insights to develop a detection strategy for Web 3.0. We will look for high quality heuristics that can detect malicious activity. As

described previously, it is important to have a bot that can detect attackers as quickly as possible because there is no value in being able to detect attacks once it is too late to do anything about it. It is also important to avoid having a high false positive rate. Since most transactions on the blockchain are not malicious, a high false positive rate means that a large majority of our alerts will be for benign transactions which will cause users to have less trust in our alerts and will make the alerts for real issues more difficult to spot. Our goal is to strike a balance between speed and accuracy to ensure that our bot is as useful as possible.

We also aim to leverage machine learning techniques to more accurately detect suspicious activity. Since we have access to an in-depth dataset of past attacker smart contracts, we will be able to use this data to train a machine learning model that can detect malicious activity. We will study this data to find useful features that can indicate that a smart contract is malicious and use these features in a model. If this approach is more effective than using heuristics in terms of speed and accuracy we will use this instead of the heuristics. We can potentially use a combination of both approaches to get better accuracy for our alerts.

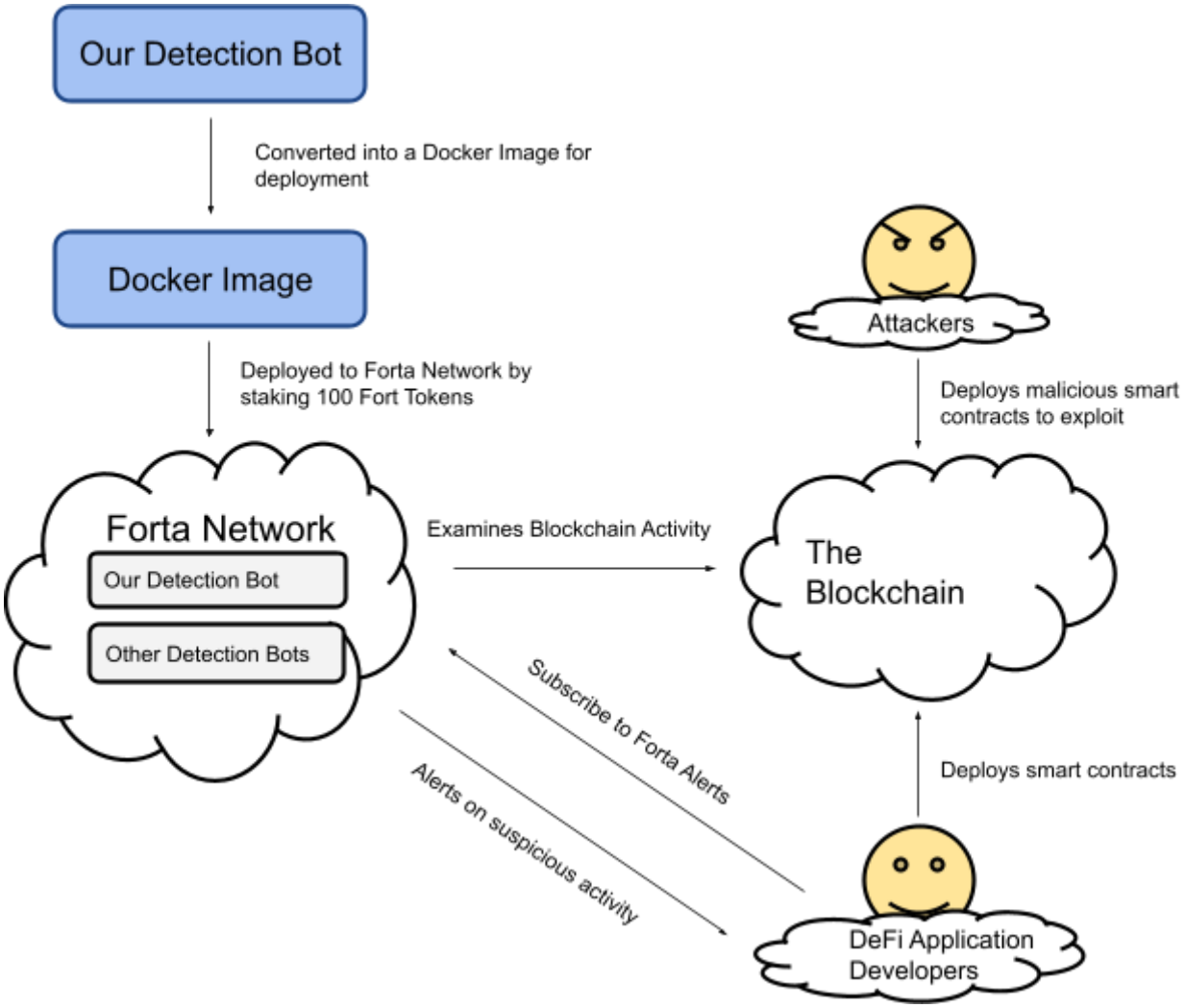
We will deploy our bot on the Forta Network which provides a simple way for users to subscribe to alerts and receive them in a convenient way. This will allow our bot to easily examine activity on the blockchain and log information for users. The Forta Network allows us to focus on our detection strategy without having to worry as much about how we will examine the blockchain and communicate with our users.

System Architecture Overview

User Interactions and Design:

The user will be notified when an attacker smart contract was identified by our bot. This would give the user an opportunity to mitigate the attack before any damage is done. Our detection bot will be able to send alerts on suspicious activity to users who have signed up to receive Forta Alerts.

High Level Diagram



Requirements:

1. As a user, I can subscribe to get an alert before an unauthorized smart contract exploits my contract protocols so that I can allow my system to deploy necessary resources to stop the exploitation before any money is lost.

   Github Issue:

   - Scenario 1: The bot detects an unauthorized smart contract before the exploitation phase and alerts the user system to deploy resources to stop exploitation by the malicious smart contract.
   - Scenario 2: The bot detects an unauthorized smart contract, but fails to alert the user system before the exploitation phase. The bot will deploy a warning message stating that an attack has, or may be currently occurring.

2. As a user, I can access the bot detection logs so that I can check on all attacks that have occurred on my system and to check bot activity if the system doesn't correctly detect an attack.

   Github Issue:

   - Scenario 1: The user requests an activity log, and the bot relays all detected attacks in chronological order.
   - Scenario 2: The user requests an activity log, and the bot relays all timely detected attacks, delayed detections, and money lost (if any)

3. As a user, I can search up a specific attack that has occurred on my system by clicking on the navbar so that I can instantly look up the attack instead of looking through the whole log.

4. As a user, I can filter detection bot logs by day, week, month, or year so that I can limit the amount of detection bot logs I see because I am interested in detection bot logs that happen on a specific date range.
5. As a user, I can download the bot detection logs so that I can have a copy of the bot detection logs.

Appendices:

GitHub to manage our Code Base, along with their Kanban Board

Python 3 to write the bot

Pytorch/numpy libraries for machine learning

Using Solidity to apply the smart contracts for detection