Team 7- Fortafy

Members: Amy Wang, Anuragh Sundar, Joe Zhang, Lawrence Chen, Max Fletcher

Introduction

Problem: Web 3, the next generation of the internet characterized by decentralized, user-owned data and applications, built on blockchain technology, is being hacked. It has been susceptible to various hacking attempts due to vulnerabilities in decentralized applications, smart contracts, and blockchain protocols. Some common methods of hacking in the Web 3 ecosystem include smart contract vulnerabilities and phishing attacks. Phishing attacks are a major concern for individuals and organizations alike. In the context of blockchain, phishing attacks can take the form of "ice phishing," where attackers modify the contract spender's address to their own address, thereby gaining access to the victim's funds. This differs from the common scam of trying to gain access to a user's private key. In an ice phishing scam, the attacker only needs to secure a transaction in the form of a smart contract to exploit a user. Ice phishing scams are almost entirely recorded on the blockchain, and are thus more straightforward than other scams because of the predominance of on-chain evidence that we can work with. It is imperative to work on ice phishing scams because they are prevalent and have the potential to result in substantial financial losses. Currently, automated phishing alerts do not have a high accuracy and have to be manually confirmed, which takes time and labor. Additionally, there is a lot of room for bias and human error. Given alerts that contain a user address on the blockchain we should be able to return an automated verdict of a true or false positive alert.

Innovation: To address this problem, our project proposes using an LLM, ChatGPT, to augment human verification. ChatGPT is an AI Large Language Model chatbot that can handle inquiries related to a variety of topics, including phishing emails quickly. By parsing transaction

data from the blockchain and identifying anomalies that may indicate a scam, ChatGPT can evaluate if a scam alert is a true positive or false positive. We focus on training ChatGPT with information such as tools and examples of ice phishing attacks, to make an informed decision on identifying a potential scam. It will be necessary to enhance the ability of ChatGPT with the usage of the Chat Completions API, which will allow the chatbot to access the internet for resources such as Etherscan. This will allow for a far more accurate and efficient method of evaluating scams with less bias, since scam alerts are currently being monitored manually by humans.

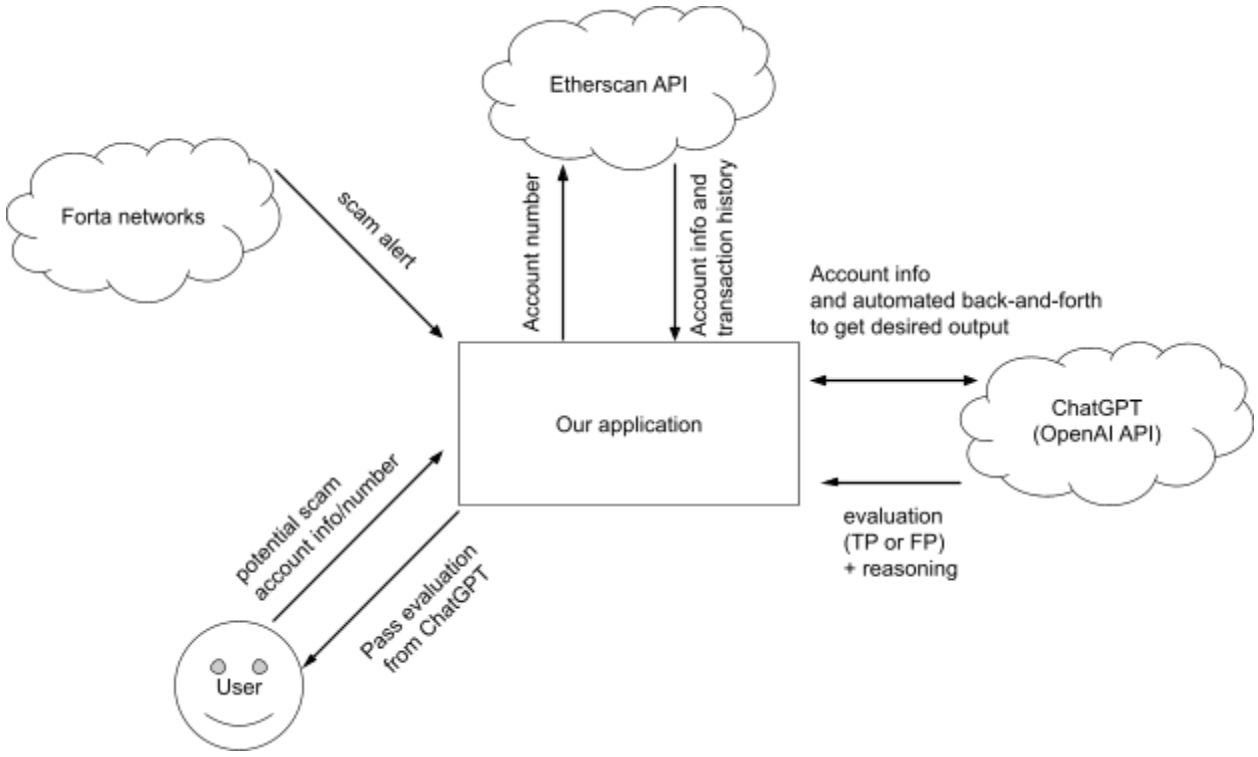
Science: The project leverages natural language processing (NLP) and machine learning (ML) techniques to identify ice-phishing on the blockchain. ChatGPT is based on GPT-4, a deep learning model that has been shown to be effective in making social engineering attacks such as phishing or business email compromise scams harder to detect and easier to pull off. By using NLP, ChatGPT can analyze transaction data from the blockchain and identify patterns that may indicate a scam. As ChatGPT continues to receive data, ChatGPT can continuously refine its ability to detect unusual behavior while improving its pattern recognition. As for demonstrating the project, we will implement either a frontend web application, utilizing React and Javascript or an UI, utilizing Python. React and Javascript have a long history of libraries and community support that allows for easier utilization. For the UI, Python contains a wide variety of libraries and advanced frameworks in Python allows for complex and scalable applications.

Core technical advancements: The core technical advancements of this project include developing an automated system that can properly extract the appropriate data from etherscan, and then have ChatGPT parse the transaction data from the blockchain and identify anomalies that may indicate a scam. This system will leverage ChatGPT's NLP and ML capabilities to

evaluate if a scam alert is a true positive or false positive. By doing so, we aim to improve incident response times and reduce the impact of phishing attacks on individuals and organizations.

System architecture overview

High level diagram



Requirements (functional and non-functional)

As Forta, I should be able to integrate this program with our already existing alerts program to forward alerts and receive a verdict on all alerts.

- Acceptance Criteria: Our application should be able to take alerts directly from Forta's bot and output a verdict that can be forwarded to their reporting program.

As a new user, I want to see a clean and easy page so that I can easily drop my CSV's and expect a modified CSV including a true or false positive verdict in return.

- Acceptance Criteria: The website should be simple and have a drop box and have a button for the user to press to download the file or it automatically downloads for them

As a user, I can download the output of the modified CSV to my own machine so that I can have my own local copy.

- Acceptance Criteria: There is a button that will download the CSV to the user's local device when clicked.

As a user, I want to be able to easily see if a scam alert is a true positive or false positive.

- Acceptance Criteria: When ChatGPT's output is ready, the UI should indicate that the alert is a true positive or false positive.

As a user, I want to be able to ask for a report so that I can view the reasoning behind why a certain scam alert was marked as a true positive or false positive by ChatGPT.

- Acceptance Criteria: When the final result is delivered to the user, it should include ChatGPT's reasoning.

As a user, I want to be able to download the report as a TXT file so that I can have my own local copy.

- Acceptance Criteria: There is a button that saves the report to the user's device when clicked.

As a user, I want to be able to enter my OpenAI API key in case it is missing/not working

- Acceptance Criteria: When the key is entered and saved in the UI, it is used for all OpenAI API requests.

As ChatGPT, I want to be able to get more data if needed.

- Acceptance Criteria: Our program includes the function calling feature of the Chat Completions API, which will allow ChatGPT to get more information if needed.

As the application, I want to resend a request to ChatGPT if it refuses to answer, so that the user does not have to.

- Acceptance Criteria: The user does not get results that indicate ChatGPT refused to answer.

As the user, I want to know the status of ChatGPT so that I know the program isn't hanging.

- Acceptance Criteria: The stream parameter in the Chat Completions API is used to return chunks of text at a time such that there is no long delay with nothing happening.

Appendices

- ChatGPT (LLM, NLP)
- Python (Main Script)
- Javascript / React (Web page)
- Python (UI)