Team Binary Bros: PRD Version 1

Project: TackleBox

Developers

Christopher Garsia (Team Lead), Sam Pettus (Scribe),

Brice Redmond, Eric Guerrero, Ori Mizrahi

Company

Novacoast

Introduction

Problem and Importance:

2020 has seen a sharp rise in the frequency of cybersecurity attacks and with it an increase in their sophistication. These improvements make the attacks harder to spot, thus allowing them to compromise higher-value targets. Often these attacks will target people as the weakest link in the security chain and as a result phishing scams—a tactic which uses fraudulent emails to trick receivers into revealing sensitive information—are becoming ever more apparent. Once sensitive information—like a sysadmin password— has been stolen, the hacker may be able to steal integral data from the company, encrypt it, and then demand money for the encryption key. Since currently there is no way to decrypt the data without the key, the organization is forced to pay the ransom. As a result, prevention is the only method to combat these attacks. Preventing employees from accessing these fake domains can be difficult though even with proper safety training. The process of registering a fake domain, pushing malicious code and emailing victims can take less than an hour total. This speed paired with high volumes of attacks allows some fake domains to exist for weeks or months without being spotted and thus it is only a matter of time before someone makes a mistake and falls for the trap. Furthermore, these attacks are not only tricky to combat, but are also extremely costly—The FBI issued a PSA that estimates that over $12 billion were lost due to email phishing between 2013 and 2018.

Current Implementations:

To combat phishing, spam filter engines are typically the first line of defense. However, this solution cannot match the rampant speed at which attacks are being launched. Spam filter engines usually

pull suspect domains from older data sources and therefore do not provide a real time solution. On the other hand, educational programs can provide real time solutions by bolstering security through teaching a company's workforce to detect and avoid phishing scams. Although effective, companies are still vulnerable to phishing due to human error. Finally, there are products which allow users to locate potential spoofing domains, but these are not automated and thus require a user to search for their domain in order to get real time results.

Core Technical Advancements:

Our product seeks to leverage open source intelligence software—like dns twist—to handle the discovery of malicious domains for phishing prevention. We will then improve upon these by making recurrent calls to their APIs at a cadence specified by our users through our React Frontend. Thus, a large advancement in our product is the automation of subsequent checks for spoofing domains which—once configured for a user—will provide real time updates. Additionally, our product will contain a domain analysis tool on top of the automation, which assesses the risk that a domain is malicious using regex matching. It will then send automated reports to the corresponding company's legal team or other specified points of contact with a breakdown of recent spoofing domains. Lastly, our software will expand its ability to prevent phishing attacks by integrating with Novacoast's firewall to block users who accidentally travelled to malicious sites.

Team Goals and Objectives:

Our goal is to create an automated anti domain-spoofing service, which will assist our users in identifying, tracking and dismantling malicious websites by:

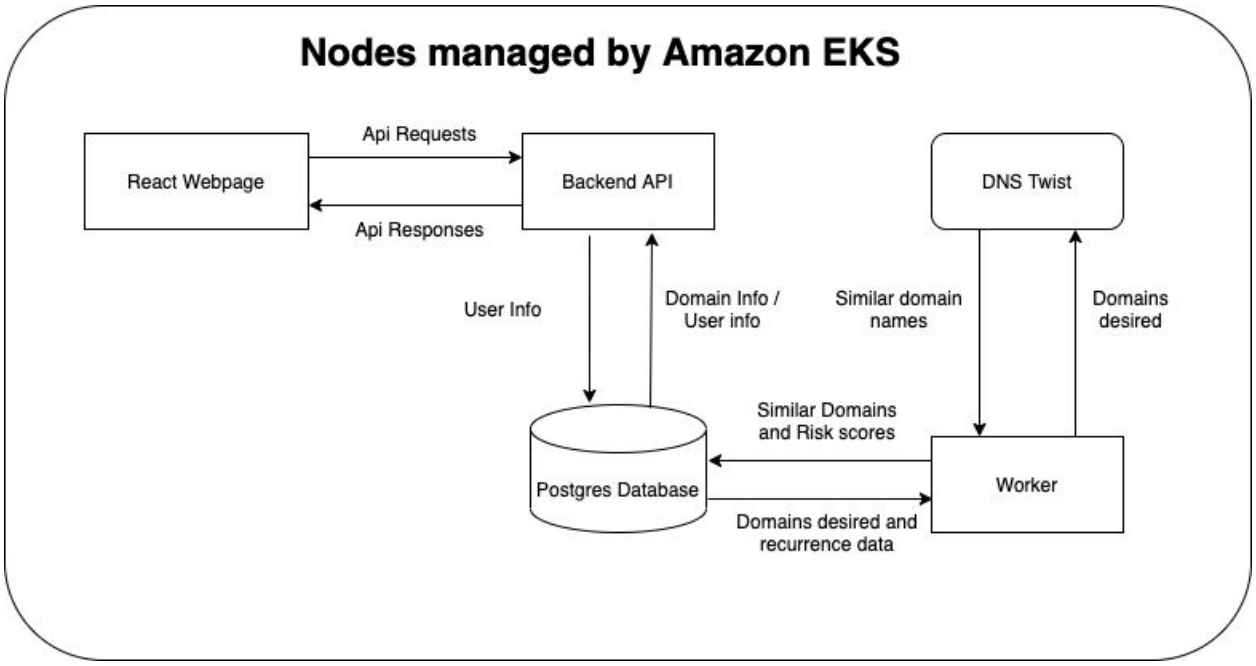
- Blocking navigation to these websites for Novacoast platform users
- Notifying the company's legal team if the phishing website uses stolen intellectual property so they can take action against the creator of the website
- Sending reports of sites that have been made to steal data from a company
- Posting the data we have collected to open source projects so that others can benefit from what we have discovered.

Assumptions:

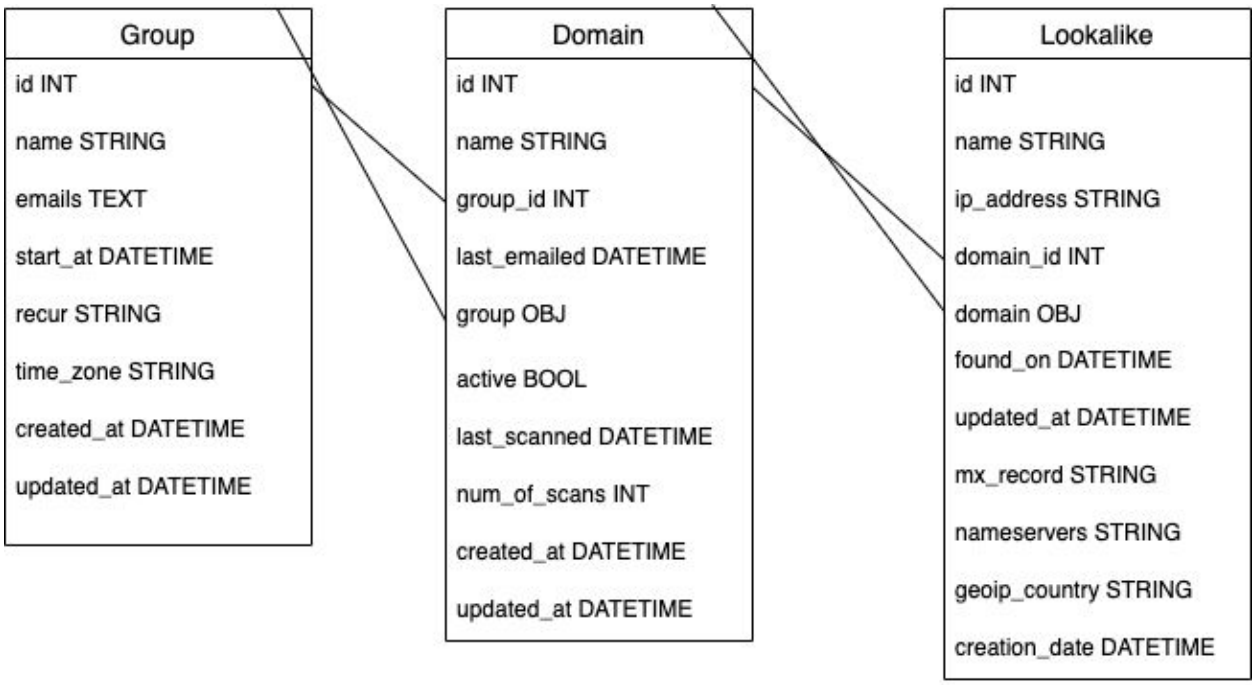
Users have their own website and are subscribers of Novacoast's services.

System Architecture Overview

High Level Diagram:



Database Schema:



User Stories & Use Cases

#1: Receive Scheduled Reports

User Story: As a user I will receive reports on potential phishing domains with addresses similar to those I am registered for, so I can seek action to protect my company from phishing attacks.

Acceptance Criteria:

- Report emailed to addresses specified at correct frequency
- Report shows a table of each lookalike domain
- Email returns a table of similar domains for each domain managed by our group

Card: https://trello.com/c/MBV8VyY8/36-user-story-receive-reports-of-similar-domains

#2: Create a Domain

User Story: As a user I can create a domain in the frontend so that all of its information is persisted in the database as well as the frontend.

Acceptance Criteria:

- Ability to create domains in the frontend, with emails, groups and recurrence information
- Persistence of data despite clearing of cache, refreshing, etc...

Card: https://trello.com/c/w0OPGUHb/35-user-story-create-a-domain

#3: View Group Details

User Story: As a user I can navigate to a group and see its domains, with relevant lookalike counts from previous scan results.

Acceptance Criteria:

- Frontend renders count of lookalike domains within group
- Frontend renders most recent lookalike found
- Frontend renders the date of the most recent lookalike found

Card: https://trello.com/c/NFtddJxy/37-user-story-view-group-details

#4: Delete a Domain

User Story: As a user I can remove a domain from its associated group so that I no longer track unnecessary domains.

Acceptance Criteria:

- Delete persists in the frontend despite clearing of cache, refreshing, etc...
- Deletion also removes entry from all rows in database where it occurs

#5: Delete a Group

User Story: As a user I can remove a group and its associated domains so that I no longer track unnecessary groups.

Acceptance Criteria:

- Delete persists in the frontend despite clearing of cache, refreshing, etc...
- Deletion also removes entry from all rows in database where it occurs for both groups and domains connected to that group

Card: https://trello.com/c/DaYauG4t/49-user-story-delete-group

#6: View Lookalike details

User Story: As a user I can view details including date registered, IP Address, and similarity for each domain so that I can track malicious activity

Acceptance Criteria:

- Renders date registered in frontend
- Renders IP Address
- Renders similarity analysis to associated domain

#7: Change recurrence information on a Group level

User Story: As a user I can update the recurrence information at the group level so that I can change the lookalike scan frequency by day, week, and month.

Acceptance Criteria:

- Ability to customize recurrence date, time zone and frequency
- Updating Recurrence information is reflected in both frontend and backend
- Lookalike scan updates frequency based off the new recurrence

#8: Continuous Front-end Deployment

Use Case: Whenever developers push code to the git code base, the updates are automatically deployed to the team's AWS hosted domain.

Card: https://trello.com/c/87oYRM8Y/40-use-case-continuous-front-end-deployment

#9: Scheduled Database Update

Use Case: After a scheduled dns twist scan completes, it should update the database with the new lookalikes found.

Acceptance Criteria:

- New lookalikes are present in the database after each scan
- The new lookalikes can be pulled from the database

#10: Ping Legal Team

User Story: As a user I want my legal team to be automatically contacted if there is a high enough similarity between my domain and a look alike so that they can reach out to the phishing domain directly and pursue legal action against them.

Acceptance Criteria:

- Report is sent to legal team if a certain similarity analysis is reached
- Report contains all necessary information for team to seek legal action

Appendix

Technologies:

Frontend - React

Backend - Flask (Python)

Database - PostgreSQL

Hosting - AWS EKS for kubernetes, AWS ECR for our registry

Separate APIs - RabbitMQ for messaging, DNSTwist for identifying domains

Containerizations - Kubernetes, Docker