EE / CprE / SE 492 Bi-Weekly Report 4

Project title: Knowledge Graphs for Cybersecurity Reasoning

03/25/23- 04/08/23

Group number: sdmay23-01

Client & Advisor: Benjamin Blakely

Team Members:

Brandon Richards - *Frontend Development Lead*

Micah Gwin - *Python/ML Development*

Alice Cheatum - *Programmer*

Nicklas Cahill - *Tester/Programmer*

Michael Watkins - *Python/ML Development*

Carter Kitelinger - *Client Interaction*

Summary

Deployed and setup the neo4j EC2 instance in AWS and setup networking resources to allow access on neo4j ports so the database can be queried by the parser and the frontend. Finished implementing the CVE background job and added it to the parser. Continued improvements for the scraper lambda and parser. Made the pipeline much faster by skipping terraform steps if nothing needing to deploy has changed and caching the build dependencies step for the parser, scraper and frontend

Past Two Week Accomplishments

- Download latest models in parser – Brandon
  - Add step to upload hash of annotation file along with models to S3 from trainer
  - Parser checks if models are missing locally, and if so downloads them from S3
  - If models aren't missing, checks local hash against remote and downloads if it differs
- Lambda Scraper Enhancements – Brandon
  - Enable running scraper locally by rearranging package structure and imports
  - Don't scrape already-scraped articles by checking DynamoDB for if they already exist
- Trainer 1.2 - Brandon
  - Save outputs of evaluation scores to file
  - Increase NER and RE scores by reviewing all gold-standard annotations
- Frontend

- Migrate from using Next.js to vanilla React
    - Query Neo4j using raw string and display returned response
- Neo4J EC2 – Micah
    - Deployed a t2.micro ec2 with an image from the AWS marketplace that contained a neo4J server. Then configured security groups to allow SSH access from our IPs and then full access for any IPv4 from Neo4j ports. Now the graph database can be queried from a browser.
- Pipeline Speed Increase – Micah
    - Used a GitHub diff action in the pipeline to determine if certain folders or files had changed in our infrastructure configuration and skip the terraform if none had. In addition, a cache action was added for the dependencies steps to save time.
- Article Annotation – Carter
    - Verified previously-annotation articles follow the latest entity and relation specification
    - Performed more annotations for gold-standard dataset
- Parser CVE data background job - Alice
    - Finish integrating CVE fetch script into the parser, currently in PR
- Pipelined parser – Michael
- Annotated articles - Nicklas
- Reevaluate 'used' relations – Everyone
    - Reduce usage of 'used' relation by creating new more-refined relations between specific entities

Pending Issues

- Even though the added cache fixed some dependency problems, the parser dependencies are 2.2GB which far exceeds the maximum unzipped size for a lambda function and its dependencies. However, the team found a solution which is to make a docker container with our own custom lambda runtime that has all the dependencies built in. This will be implemented in the next sprint.

Individual Contributions

Name	Hours past two weeks	Hours cumulative
Brandon Richards	13	91.5
Micah Gwin	8	63
Alice Cheatum	6.42	54.25
Nicklas Cahill	4	37.33
Michael Watkins	4	35
Carter Kitelinger	5	45.2

Summary of weekly advisor meeting

Performed a retrospective and discussed our progress and trajectory. Concluded we are perfectly on schedule to wrap up version 1 of the system by the next sprint. Selected tasks to complete this week to remain on schedule. Decided on started the frontend now to allow the client to visually see the knowledge graph in version 1. Also decided to writing unit and integration test for components and document our progress in terms of test percentages.

Plans for Coming Two Weeks

- Implement custom lambda runtime – Micah
    - As described in the pending issues this will involve creating a docker container with the needed dependencies and uploading it to ECR, which is the AWS container repository. Then the parser lambda can pull the container with all dependencies.
- Create the EC2 Hibernator Lambda – Micah
    - In order to save money on EC2 compute hours, I will create a simple lambda triggered by a cron time expression during the morning and evening to turn the EC2 on and off using boto3.
- Frontend – Brandon
    - Create query builder
    - Improve UX
- Alice
    - CVE background job
        - Fix further issues if any are found
        - Merge into develop
    - Annotate articles for training
    - Other tasks as necessary