Will Young
Student ID: 924230057
Week 4 Report
2/20/2026

Hello Professor,

Below is an update for my week 4 progress In addition, I have pushed my updates to my github repo: https://github.com/Woyoung21/LastMile-Sec. For week 4, the main tasks were to integrate the Reporter agent and ensure the noise was reduced in the original documents we are parsing. I was also going to test the Lang Extract method from google, utilizing Gemini to parse PDF files.

I will start by discussing my experience with Lang Extract, which did not go smoothly. I created a paid Google Studio's API that gave me up to $300 in api credits to avoid hitting any rate limits the free tier API usually have. While I was developing and testing the Lang Extract method, it kept exhausting my defined API rate limits, which I will add are very high. I wasn't afraid to spend some of the credits to test and fine tune this method. However I was unable to get past this hurdle, LangExtract kept hitting parallel inference limits to chunk the original PDF report. Mind you these reports can be hundreds of pages long. After troubleshooting for a while, I decided to pivot and bolster the original parser. I originally was trying to create a generic parser for any PDF, and I think that is why it didn't work so well at first. I tailored the parser to the structure of the vulnerability reports my tools produce, and it worked great. This finished the last portion of section 1.

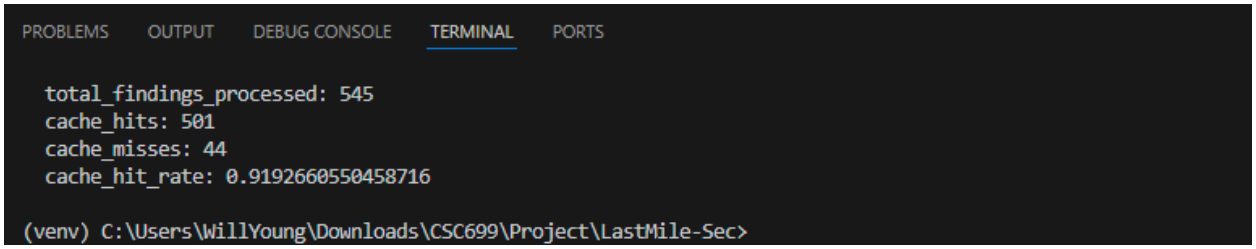
Now that the parsers were all dialed in, I was able to begin section 2 of the project. In this section, we take the JSON packet and first send it to the reporter agent. The reporter agent then takes in the information from the JSON packet and creates a succinct, technical sentence describing the event. I pivoted from last week's plan and decided to use Gemini 2.0 for this step because I have the credits to use a highly accurate frontier model. I will now walk you through how this is setup.

In our data folder, we have a processed folder where the JSON files go after traversing our parser. That JSON file is fed into our reporter.py file, where we look at the title and the first 100 characters of the description and create an MD5 hash. We then look in the cache folder for any matching hashes, and if it is found, we serve that summarization. If one isn't found, we package that data and sent it Gemini's API and wait for it to summarize the event. We then save this response in a new file for future reference. We chose this path because many devices have the exact same vulnerability, for instance 10 computers could have an deprecated Open SSH version, but instead of sending each packet to Gemini, we only needed to do it once, saving on tokens. It was also a great choice because it incorporated a few backoff timeouts if we hit the 429 RESOURCE_EXHAUSTED message.

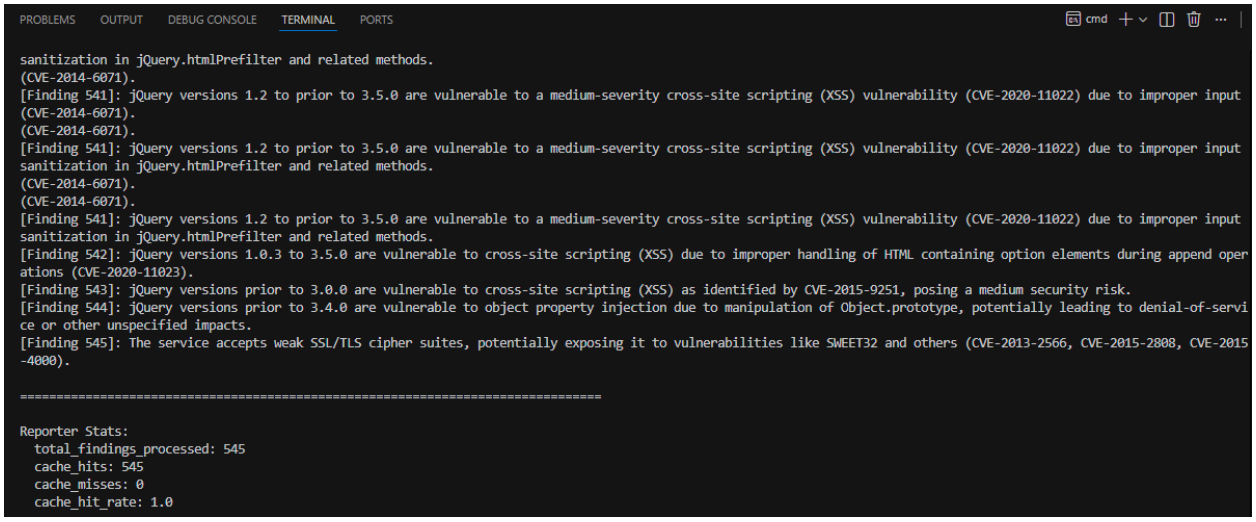
After all of the events are processed, the summarization sentence is added back into the original JSON under the new metadata category of technical_summary and we added a timestamp for when we ran the summary in the summary_generated_at field. Editing data packets is a core component of security engineering, over my internship last summer, I used a product called Cribl. Cribl acts as an intermediary data enricher, you basically feed it all of your

log sources and you can cut down on the fields you want to then send to your SIEM, or your can enrich the packets by adding fields that will help correlation once it is received by the SIEM.

The image below is an example of the debugging text after we process the JSON files. Here you will see that we identified 545 total security events. Of those 545 events, 501 were not unique, so we were able to immediately serve the summation of those events. The other 44 events were unique, so we reached out to Gemini to create the one sentence summarization for us.



The image below I reran to show what the single sentence summarizations look like. Here we are focusing on the severity of the vulnerability and any associated CVE numbers that may help us with the MITRE mapping.



For our system prompt, I used few-shot prompting, where I provided the LLM with a few examples of what I would like the output to look like. I also started out assigning the agent a persona, "*You are a cybersecurity technical analyst specializing in vulnerability assessment and threat analysis. Your task is to generate a single, concise technical summary sentence that captures the essence of a security finding*." I also added requirements for what needs to be in the summary as well as what not to include in the summary sentence. I may continue to develop this as there are some core pieces of information I would like to be considered.

Will Young
Student ID: 924230057
Week 4 Report
2/20/2026

Next week we will be working on fine tuning our MITRE Mapping agent. I will also be reviewing the research paper more in depth to better understand how the reporter and mapper work together. Based on my early readings I may have to utilize a RAG database to match the MITRE ID's to their actual textual meanings. If this is the case, I may try to use Actian's VectorAI database I used in SFHACKs. The semantic cache feature was incredibly fast and efficient, plus it's on-machine.