CANADIAN CENTRE FOR CYBER SECURITY | CENTRE CANADIEN POUR LA CYBERSÉCURITÉ

Geek Week 8 Team 5.2: CyberGPT

Overview and Takeaways - 14 July 2023

Team 5.2 - who are we?

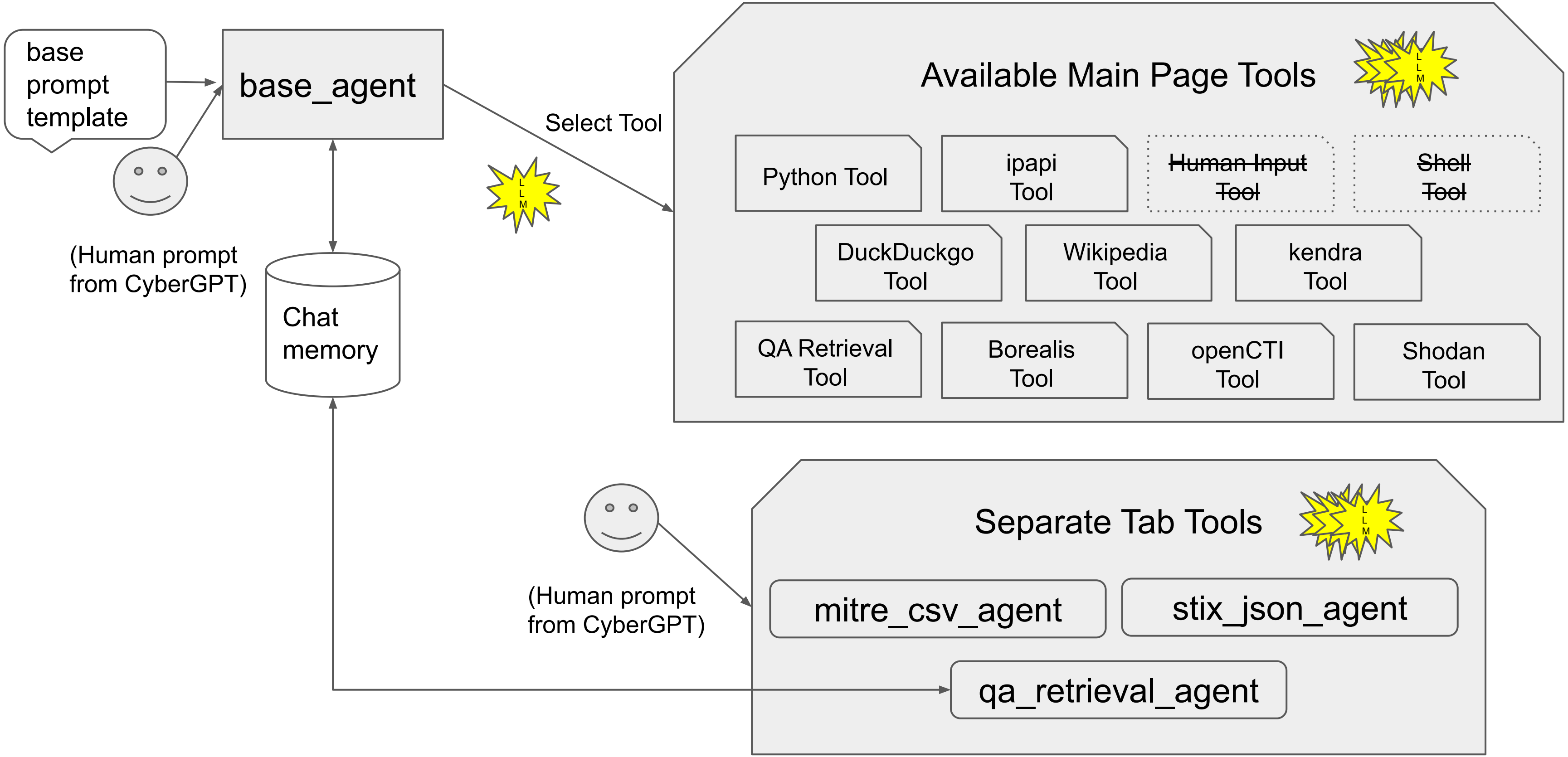
- Nina C - CCCS (Team lead)
- Casey C - Sandia
- Paul J - NCSC
- …

Team 5.2 - what were we aiming to do?

Create a minimal viable product of **CyberGPT**, demonstrating uses of generative AI in cybersecurity:

- Designing **specialized AI agents**
  - Selecting the appropriate foundational LLM to be used as the base of the agents and integrating the LLM into the agents.
  - Creating the code for the custom and non-custom agents, configuring initial prompt and type of agent.
  - Create agents that specialize in database analysis, API tool calls, Q&A over documents and web crawling
- Designing the **memory**
  - Create the vector DB setup for storing agent memories.
  - Research into ways of implementing long term entity memory for agents using vector DB.
- Designing the **knowledge bases**
  - Determine documents that could/should be part of agent knowledge base.
  - Store relevant documentation into vector DB and connect access to agents.
- Designing the **custom toolkits**
  - Create the custom tools using Langchain for cybersecurity APIs.
  - Connect tools to agents.
- Designing the **application user interface**
  - Create the application interface using Streamlit or React.
  - Create the configuration settings on the application interface (Agent selection).
  - Create a basic chat interface for interactions between user and AI.

CyberGPT - Main Page Tools



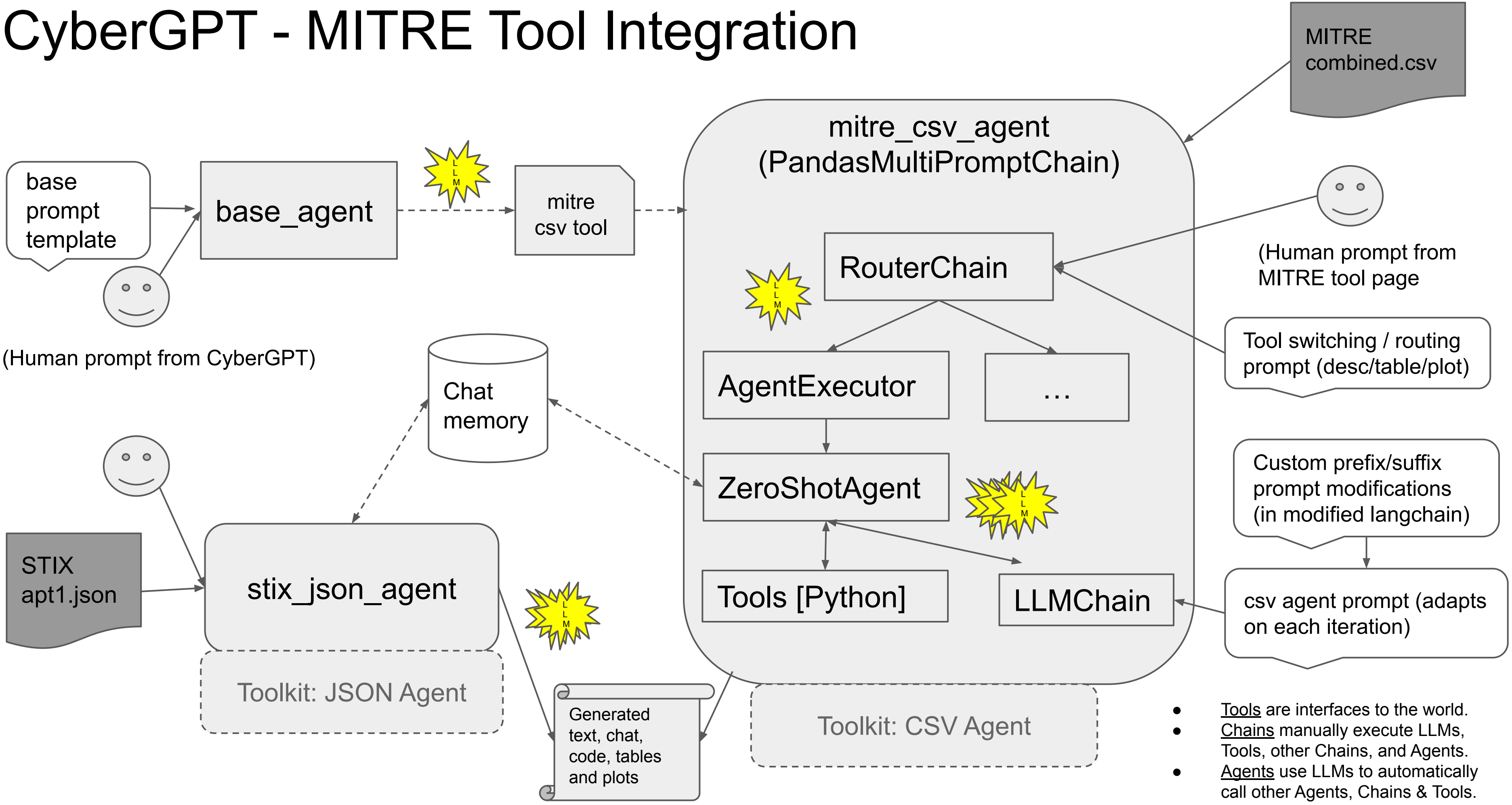
CyberGPT - Options for MITRE Tool Integration

- CSV Agent
  - Allowed us to get basic functionality very quickly, but limited our ability to change prompts or allow prompt switching.
  - Changing the prompts required modifying the underlying langchain code. This was somewhat burdensome, and currently we retain this modified copy as a subfolder in “agents”. Alternatives would be getting a langchain fork and installing or in the long term, cleaning it up and submitting a PR to langchain to add new functionality.
  - Another group’s look into this topic (using Pandas Dataframe Agent directly): https://levelup.gitconnected.com/talk-to-your-csv-how-to-visualize-your-data-with-langchain-and-streamlit-5cb8a0db87e0

- Router Chain
  - We created a subclass that could take AgentExecutors (Csv Agents) instead of LLMChains.
  - A base LLM is used to pick among different routes with descriptions (similar to an agent picking its tools).
  - Each destination chain has its own specialized prompts.

- JSON Agent
  - The out of the box option for JSON objects, but unlike the CSV agent, has no access to python tools.
  - Seemed to access top level or directly specified objects, but was unable to handle more complex requests.
  - Reconstructing the agent to get the desired behavior seems doable, though outside the scope of the time we had left in the workshop.

CyberGPT - MITRE Tool Integration


MITRE combined.csv
mitre_csv_agent (PandasMultiPromptChain)
base prompt template
base_agent
LLM
mitre csv tool
(Human prompt from CyberGPT)
RouterChain
LLM
(Human prompt from MITRE tool page
Tool switching / routing prompt (desc/table/plot)
AgentExecutor
…
Chat memory
ZeroShotAgent
LLM
Custom prefix/suffix prompt modifications (in modified langchain)
STIX apt1.json
stix_json_agent
LLM
Tools [Python]
LLMChain
csv agent prompt (adapts on each iteration)
Toolkit: JSON Agent
Generated text, chat, code, tables and plots
Toolkit: CSV Agent

- Tools are interfaces to the world.
- Chains manually execute LLMs, Tools, other Chains, and Agents.
- Agents use LLMs to automatically call other Agents, Chains & Tools.
- Toolkits are collections of Tools.

CyberGPT - MITRE Tool Integration

Example Prompts we got working:

- What techniques does FlawedAmmyy use? List them all alphabetically.
- Display a formatted table with the 10 techniques with the most unique mitigations.
- Select the set of rows with unique techniques and mitigations, and then select 10 techniques arbitrarily. Plot the number of unique mitigations for each of these 10 techniques.
- Use python to build a machine learning model to predict the Mitigation column using text from the Description fields, using only the first 10000 rows of the data frame. Then evaluate the model and output the accuracy results. Also plot a Confusion Matrix display using an sklearn ConfusionMatrixDisplay, with the generated matplotlib figure assigned to variable fig. Include the import statements in your python code. Don't set the display_labels parameter.
- Extract 100 Software Description entries and 100 Mitigation Description entries from the dataframe, embed them into 128-dimension vectors using TF/IDF representations, and use UMAP to reduce them to 2D and output as a plot. Don't show or save the plt at the end.

Unfortunately, there is still quite a bit of nondeterministic behavior, especially with the more complex queries, and especially when combining them with others. If one of these doesn’t work, try starting a new chat or restarting the app entirely. With the plots, you may need to run them several times before you get a successful plotting.

CyberGPT - MITRE Tool - Common Issues and Errors

- Not managing duplicate rows
  - Just adding ‘unique’ as a keyword seemed very effective here.
- Failing to import needed libraries
  - And getting stuck in loops trying to install and import things
- Trying to load new data files (rather than built-in dataframe)
  - Including hallucinating random data.csv files
- Inconsistent hanging
  - Adding a timeout allows it to (sometimes) gracefully recover from where it was in the chain.
- When plotting, defaulting to `plt.show()` even though that can’t be displayed to the page.
  - Specifying a specific variable `fig` and telling it to not use `plt.show() in the prompt templates (and sometimes again in the user prompt) helped.
- Inconsistent behavior/results on some queries, depending on what was run before or after, despite there being no explicit memory in the agent.
  - We never found a satisfying answer to this. We strongly recommend double checking generated code for correctness before accepting results.
- Interleaving of English description and code means it can often stumble on returning the final output.
  - This can be alleviated somewhat by telling it to return one or the other, or giving formatting tips (telling it to wrap code in ‘```’), but that doesn’t always work.

CyberGPT - MITRE Tool - Security Issues and Warnings

- Running the python tool involves arbitrary code execution.
  - Mitigations would need to be in place before tools using python should be run in any real environment (sandboxing, human checking before each run, which is slow, or otherwise limiting what kinds of code the LLM can produce/run).
  - Our implementation of plotting is also insecure; we try to run the plotting code the tool produces and load it in streamlit.
- Using OpenAI or other remotely hosted LLMs exposes organization data and queries and is not suitable for information that must be kept secure.
  - Locally hosted LLMs can be used to avoid this issue, but require significant GPU computational power and infrastructure to support.
- All the usual caveats of LLMs still apply.
  - No guarantee of correct information, verification of results is essential, especially if used for critical decision making. They should be used to enhance thinking, not be an excuse for replacing thinking. Humans are ultimately responsible for the use of LLM tool outputs.
- Don’t even think about enabling the Shell Tool! 🙈

CyberGPT - MITRE Tool - Sparks of Awesomeness

- Ability to following LLM’s sequence of Thoughts and Actions
  - This gets output to the terminal, and is invariably more interesting than the final results

- Generated Python Data Science Code always seems plausible
  - For data summarization, embedding/viz, training ML models, evaluating predictions, …
  - Plots were always nicely annotated.

- Automatic fixing of Python errors is great to watch
  - Library imports, fixing NaNs, column naming, …
  - Despite trying many times, pip installs never worked (a good thing for security!)

- …

CyberGPT - Key Takeaways (1/2)

- Langchain is immature and rapidly evolving…
  - many things feel incomplete and a little clunky
  - support for tools and toolkits is still quite minimal.
  - easy to use for ‘standard’ use cases, but wrappers make it complex to identify and change underlying behavior for more advanced use-cases.
- Tool and dataset descriptions really matter…
  - as they’re used directly by AI agents for deciding what to do (not just as documentation for humans!)
  - careful wording can make all the difference
- Testing non-deterministic agents is a challenge…
  - even with temperature=0, behaviour can be unpredictable
  - many different variables influence LLM behaviour
- Good prompt engineering is vital…
  - LLMs seem to often ignore certain prompt phrasing and instructions while other keywords were noted to have a large impact (e.g. ‘unique’).
  - Newer LLMs (e.g. GPT-4) claim to be better at following instructions (but we didn’t test this).

CyberGPT - Key Takeaways (2/2)

- Ability to automatically interpret Python errors (and sometimes fix them) is awesome to see, when it works!
  - ‘handle_parsing_errors’ option can also (sometimes) allow the agent to fix errors in its own prompt formatting in between iterations.
- The longer the chain of instructions, the greater the potential for error
  - In particular, selecting a specific tool and forwarding the input using the LLM often changes it - this makes merging a tool from a tab into the main agent challenging.

CyberGPT - Unresolved Questions…

- How to select the most appropriate foundation LLM?
- How to incorporate background knowledge into custom assistants?
- How to evaluate/benchmark the utility of AI-based cyber assistants?
- Will GPT-4 be better at understanding instructions, reasoning solutions, and fixing coding errors?
- What new security guidance should we issue for LLM developers?

CyberGPT - More Information & Resources…

- Our code, including example runs (private repo): https://github.com/NZ369/CyberGPT
- GeekWeek portal and apps: https://geek.collaboration.cyber.gc.ca/en/week/2023/portal/applications.html
- LangChain Documentation: https://python.langchain.com/docs/get_started/introduction.html
- Mitre Data Sources:
  - **(We mainly used this)** Python scripts for dumping MITRE ATT&CK to csv: https://github.com/mitre-attack/attack-scripts/
  - Python library for MITRE ATT&CK: https://github.com/mitre-attack/mitreattack-python
  - STIX Data for MITRE ATT&CK: https://github.com/mitre-attack/attack-stix-data
  - STIX Data for Atlas: https://github.com/mitre-atlas/atlas-navigator-data
- OpenAI GPT Best Practice Guidance: https://platform.openai.com/docs/guides/gpt-best-practices
- Academic Papers:
  - ReACT: Synergizing Reasoning and Acting in Language Models: https://arxiv.org/abs/2210.03629
  - Sparks of AGI - Early Experiments with GPT-4: https://arxiv.org/pdf/2303.12712.pdf