Lab 11

YARA & SIGMA Rules

Lab Objectives

- Query the LLM to make a YARA Rule
- Query the LLM to make a SIGMA Rule
- See if using RAG changes how the rules are created

YARA vs SIGMA Rules

- Yet Another Redundant Acronym
- Sifts through an Executable Binary data to pattern match against known bad data corresponding to known malware or techniques
- Utilizes rules for each piece of malware
- Patterns are embedded within each rule

- SIGMA does not have a funny Acronym
- Utilizes rules to filter through specific event logs
- Utilizes rules based primarily by technique
  - Can have rules for specific tools

A Brief History of STUXNET

- Worm designed to target Siemens SCADA systems
  - Yknow, the ones used to control Nuclear Reactors
- Chained four 0-days, stolen certificates, and rootkit-like abilities to hide
- Spreads indescrimitaley, but will only activate on systems controlling the specific Siemens PLCs
- Used to target Iranian Nuclear Reactors
- May or may not have been made in part by the United States and Israel

Ask the LLM about Stuxnet

```
POST /_plugins/_ml/models/XHSdKZgBKJWqR4KGAMQ9/_predict
{
  "parameters": {
    "messages": [
      {
        "role": "assistant",
        "content": "How does Stuxnet Spread?"
      }
    ],
    "temperature": 0.5
  }
}
```

HTTP Method and Endpoint:

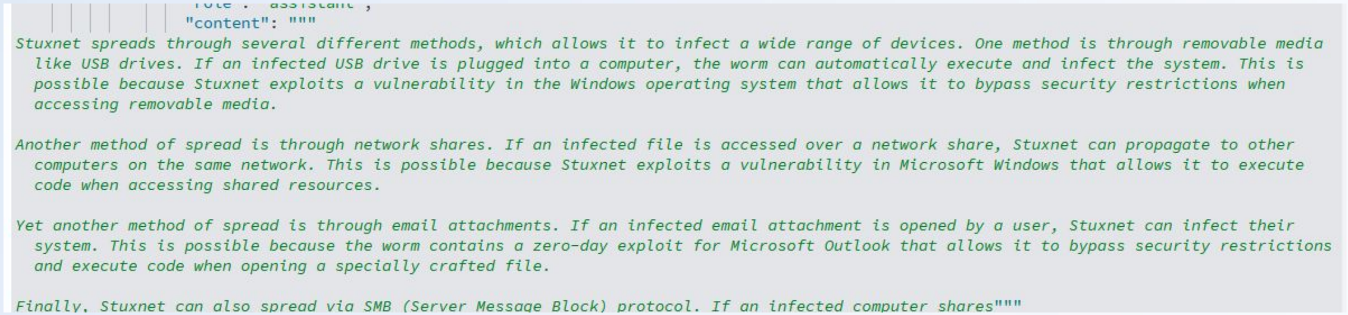
- `POST /_plugins/_ml/models/$MODELID/_predict`
- This is a POST request to the prediction endpoint of a specific machine learning model.

Request Body:

- **`parameters`**: Contains the parameters for the prediction request.
  - **`messages`**: An array with message objects.
    - **`role`**: "assistant" (indicating the role of the message sender).
    - **`content`**: "How does Stuxnet Spread?" (the actual content or question to be processed).
  - **`temperature`**: 0.5 (controls the randomness of the model's responses; a value of 0.5 indicates moderate creativity).

Variable Output

Due to how the LLM operates, your output may vary by a little, or even a lot- even to the point of “hallucinating”. Run the last query a few times- even try to change wording to see how the LLM reacts. Right below this is an example output.



```
                    "role": "assistant",
                    "content": """
Stuxnet spreads through several different methods, which allows it to infect a wide range of devices. One method is through removable media
  like USB drives. If an infected USB drive is plugged into a computer, the worm can automatically execute and infect the system. This is
  possible because Stuxnet exploits a vulnerability in the Windows operating system that allows it to bypass security restrictions when
  accessing removable media.

Another method of spread is through network shares. If an infected file is accessed over a network share, Stuxnet can propagate to other
  computers on the same network. This is possible because Stuxnet exploits a vulnerability in Microsoft Windows that allows it to execute
  code when accessing shared resources.

Yet another method of spread is through email attachments. If an infected email attachment is opened by a user, Stuxnet can infect their
  system. This is possible because the worm contains a zero-day exploit for Microsoft Outlook that allows it to bypass security restrictions
  and execute code when opening a specially crafted file.

Finally, Stuxnet can also spread via SMB (Server Message Block) protocol. If an infected computer shares"""
```

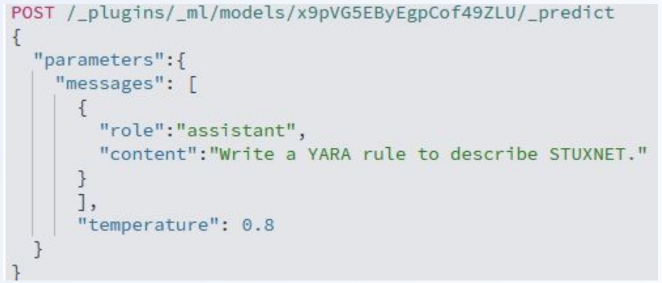
Now lets ask for a YARA Rule

HTTP Method and Endpoint:

- `POST /_plugins/_ml/models/$MODELID/_predict`
- This is a POST request to the prediction endpoint of a specific machine learning model.

Request Body:

- **`parameters`**: Contains the parameters for the prediction request.
  - **`messages`**: An array with message objects.
    - **`role`**: "assistant" (indicating the role of the message sender).
    - **`content`**: "Write a YARA rule to describe STUXNET" (the actual content or question to be processed).
  - **`temperature`**: 0.5 (controls the randomness of the model's responses; a value of 0.5 indicates moderate creativity).



```
POST /_plugins/_ml/models/x9pVG5EByEgpCof49ZLU/_predict
{
  "parameters":{
    "messages": [
      {
        "role":"assistant",
        "content":"Write a YARA rule to describe STUXNET."
      }
      ],
      "temperature": 0.8
  }
}
```

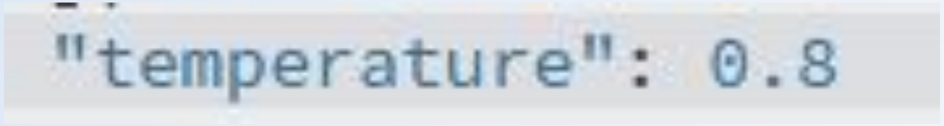
Other YARA Rules

Try to query the LLM for YARA Rules about different pieces of Malware. On the left are some examples of malware.

- Emotet
- TrickBot
- Zeus
- Ryuk
- Dridex
- Locky
- REvil
- Mirai
- Conflicker
- Wannacry
- NotPetya
- Agent Tesla

Temperature

Alter the temperature in your past queries. The lower the temperature, the more deterministic the model is. The Higher the Temperature is, the more… creative the model is. “1” is a default for most models



```
"temperature": 0.8
```

QUERY to make a SIGMA Rule

This will be pretty much the exact same query as your first basic one in this lab. Let's try to ask for a SIGMA rule to identify Cobalt Strike being used. Here is an example query:

“Write a SIGMA rule to identify Cobalt Strike Beacons being created”

Try to have the LLM make SIGMA queries for other known bad techniques- for example, service creations

See how close the LLM gets compared to published examples

RAG - A blessing.. Or a curse

We’ve discussed using RAG in order to reduce hallucinations. However, it can also cause issues if the RAG data is not accurate or helpful. To demonstrate the importance of RAG data, we have an index entitled “stuxnet_knn”. This contains some lines from a research paper by Paul Mueller and Babak Yadegari, entitled “The Stuxnet Worm”. We will build upon Lab 10’s RAG by using the knn index. In order to save time, the embedding model, knn index, and LLM model connector have already been set up.

Flow Agent

The prerequisites of this have been covered in Lab 7. A pipeline has been built that meshes a local(to OpenSearch) embedding model to search through the KNN index, and provides that output to the LLM running on LMStudio. The next page has a screenshot of the flow agent query. We will explore manipulating this in future labs. After the query shown in the next slide is ran, it will give us an AGENT_ID. We will use this AGENT_ID in lieu of a MODEL_ID in order to utilize the pipeline.

```
POST /_plugins/_ml/agents/_register
{
  "name": "Test Agent_For_RAG-stuxnet",
  "type": "flow",
  "description": "This is a test agent using the Development LMStudio host",
  "tools": [
    {
      "type": "VectorDBTool",
      "parameters": {
        "model id": "EBKWKZgB3rgc35q2GLG9",
        "index": "stuxnet knn",
        "embedding field": "description embedding",
        "source field": ["description text"],
        "input": "${parameters.question}"
      }
    },
    {
      "type": "MLModelTool",
      "description": "A general tool to answer any question",
      "parameters": {
        "model id": "XHSdKZgBKJWqR4KGAMQ9",
        "response filter": "$.choices[0].message.content",
        "messages": [
          {
            "role": "assistant",
            "content": "\n### Instruction:\n You are a professional data analyst. You will always answer questions
based on the given context first. If the answer is not directly shown in the context, you will analyze the data and
find the answer. If you don't know the answer, just say 'don't know'.  \n\n
Context:\n${parameters.VectorDBTool.output}\n\nHuman:${parameters.question}\n### Response:\n"
          }
        ],
        "temperature": 0.5
      }
    }
  ]
}
```

The Flow Agent Query Explained

HTTP Method and Endpoint:

- `POST /_plugins/_ml/agents/_register`
- Registers a new agent in the system.

Request Body:

- `name`: "Test_Agent_For_RAG-stuxnet"
    - The name of the agent.
- `type`: "flow"
    - Specifies the type of agent.
- `description`: "This is a test agent using the Development LMStudio host"
    - A description of the agent.
- `tools`: List of tools used by the agent.
    - `VectorDBTool`:
        - `type`: "VectorDBTool"
        - `parameters`:
            - `model_id`: "sGrdH5EBrbJh51EC43hd"
            - `index`: "stuxnet_knn"
            - `embedding_field`: "description_embedding"
            - `source_field`: ["description_text"]
            - `input`: "${parameters.question}"
    - `MLModelTool`:
        - `type`: "MLModelTool"
        - `description`: "A general tool to answer any question"
        - `parameters`:
            - `model_id`: "ntvbH5EBfzbu3jp_guZa"
            - `response_filter`: "$.choices[0].message.content"
            - `messages`:
                - `role`: "assistant"
                - `content`: Instruction and context for answering questions based on the output of `VectorDBTool`.
            - `temperature`: 0.5

Querying the Agent

HTTP Method and Endpoint:

- `POST /_plugins/_ml/agents/ytveH5EBfzbu3jp_muYs/_execute`
- Executes a specific agent identified by the ID `ytveH5EBfzbu3jp_muYs`.

Request Body:

- **`parameters`**:
    - **`question`**: "How does Stuxnet Spread?"
        - The question being submitted for processing by the agent.

```
POST
/ plugins/_ml/agents/kXSgKZgBKJWqR4KGgMS9/_exec
ute
{
  "parameters": {
    "question": "How does Stuxnet Spread?"
  }
}
```

VectorDB Tool Output

If you take a closer look at the query within the Flow Agent Creation, you will see a field representing the VectorDB output. This is the data our embedding model has pulled out of the “stuxnet_knn” index that it believes relevant to the question. Below is the context passed to the LLM to augment the answer it gives you

```
Stuxnet employs several methods to spread itself: Via USB flash drives The
ultimate destination of Stuxnet is the computers that control the centrifuges.
These are called PLCs (Programmable Logic Controllers), and are
special-purpose computers, used for controlling electronic devices or systems,
such as industrial systems. The PLCs are connected to computers that control
and monitor them, and typically, neither are connected to the Internet.
Therefore, Stuxnet needs some other vector to reach those computers, and so it
is capable of propagating via USB flash drives. In the case of Natanz, the
infected flash drives may have been introduced to the control computers via
outside contractors working at the plant. Different versions of Stuxnet use
different ways to do this: recent versions use an Windows LNK vulnerability
and older versions use an autorun.inf file vulnerability
```

Your new RAG Query should be more concise and specific compared to the non-RAG query.

Try rerunning both to see what output changes

Ruining SIGMA and YARA Rules

Since our RAG data is only about stuxnet, the model should be smart enough to not use the RAG data as it is not related. Try using the query from the previous slide to request a SIGMA rule for Cobalt Strike, as well as some other TTPs, to see how and when it uses data from the stuxnet_knn index.

Lab End