CHAPTER 9: ANCHORING DOWN

Mission 9: Domain History Analyzer

Every prompt from the chapter, ready to copy, plus your results sheet and pre-deployment checklist.

OPERATION NAME: NEURAL NEXUS
CLASSIFICATION: TLP RED
MISSION FILE: CHAPTER 9 // ANCHORING DOWN

Analysts:
Phantom Group has been observed rotating infrastructure to avoid detection. Your team has collected a list of suspicious domains through open source reporting, but manually checking each domain's historical activity is too slow to support timely analysis.
Build an AI-assisted workflow that uses the Internet Archive's Wayback Machine to review domain history at scale, identify when sites were active or inactive, and surface timeline anomalies that may indicate infrastructure changes, content shifts, or operational staging.

```
Director of AI Intelligence Agency
Office of Intelligence
```

SAFETY

These are potentially live threat actor domains. Do not browse to them directly. Remove the [] brackets only inside Colab.

DOMAINS Potential Sandworm domains used in the chapter

Domain (defanged)
Kmsupdate2023[.]com
Windowsupdatesystem[.]org
kms-win11-update[.]net
Kalambur[.]net
ratiborus2023[.]com

Source: EclecticIQ Threat Intelligence, "Sandworm APT Targets Ukrainian Users with Trojanized Microsoft KMS Activation Tools," February 11, 2025.

PROMPTS Copy these into your AI tool, one step at a time

Open Google Colab, create a new notebook, and name it Domain History Analyzer. Remember the loop: prompt, read, explain, verify, run. You are generating your own code, not copying mine.

STEP 1: INSTALL YOUR LIBRARIES

```
Write me a single line of Python that installs the requests and pandas libraries in Google Colab.
```

Good output: one line starting with ! then pip install and the two library names. Run it and wait.

STEP 2: IMPORT YOUR LIBRARIES AND CREATE YOUR DOMAIN LIST

```
Write me a Python cell for Colab that imports the requests, pandas, and time libraries and creates a list variable called domains containing five placeholder domain names. Remove the [] brackets from the domains: Kmsupdate2023[.]com, Windowsupdatesystem[.]org, kms-win11-update[.]net, Kalambur[.]net, ratiborus2023[.]com
```

Good output: no more than 8 lines, the word import 3 times, and the five domains below it.

STEP 3: BUILD THE LOOKUP FUNCTION

```
Write me a Python function called check_domain_history that takes a single domain name as input, queries the Internet Archive Wayback Machine CDX API to find the first time that domain was ever archived and the most recent time it was archived, and returns a dictionary containing the domain name, first seen date formatted as YYYY-MM-DD, last seen date formatted as YYYY-MM-DD, the HTTP status code from the first capture, and a flag field that says NO HISTORY if the domain has never been archived, RECENTLY REGISTERED if the first capture was after 2022, and No anomalies detected otherwise. Include error handling so the function returns an error message instead of crashing if something goes wrong.
```

Good output: the Wayback Machine URL in the code, four fields in the return statement, and a try and except block.

STEP 4: RUN IT AGAINST YOUR LIST

```
Write me a Python cell that loops through a list called domains, calls check_domain_history for each one, waits one second between each call, collects all the results in a list called results, converts that list to a pandas DataFrame called df, and prints the DataFrame.
```

Good output: includes time.sleep so you don't hammer the API and get your IP blocked.

STEP 5: VISUALIZE AND EXPORT YOUR RESULTS

```
Write me a Python cell that takes a pandas DataFrame called df and creates a horizontal bar chart using matplotlib showing each domain on the y-axis and the number of total captures on the x-axis. Color the bars red if the flag field contains NO HISTORY or RECENTLY REGISTERED and green otherwise. Add a title that says Sandworm Group Domain History Analysis, label the axes, and save the chart as domain_history_chart.png.
```

Heads up: this one may throw an error. Use Colab's Explain Error button. Fixing it is part of the lesson.

Analyst (that's you) | **Date** | **Tools used**

RESULTS Timeline per domain

Domain	First seen	Last seen	Flag	Anomalies or gaps
Kmsupdate2023[.]com				
Windowsupdatesystem[.]org				
kms-win11-update[.]net				
Kalambur[.]net				
ratiborus2023[.]com				
My own domain:				

DELIVERABLE What the archive says, and doesn't

One sentence: what the historical data suggests

One sentence: what cannot be concluded from the archive alone

What the tool shows clearly vs. what still needs human interpretation

CHECKLIST Pre-deployment: try to break your tool

Check	Ask	Pass / Fail
Data trail	Can you trace where all data comes and goes?	
Access	What does the tool have access to, and does it touch anything beyond what it should?	
Error handling	Did you build in instructions for handling errors, or does it just crash?	
Quality	Did you test it with bad data to see how it handles it?	
Output test	Did you verify the output against a target with known results?	
AI attack	Have you asked AI to look for failure points?	
Documentation	Have you documented edge cases, limitations, how it works, data sources, confidence, accuracy?	
Terms of service	Does the tool comply with the ToS for every API it uses?	