□ D3FEND ENGINEERING LABS: DEFENSIVE COUNTERMEASURE DESIGN

Version 1.0 - Tactical Engineering Series

□ LAB 1: THE D3FEND ONTOLOGY HUNT (DAY 141)

Objective: Programmatically query the D3FEND knowledge graph to identify defensive strategies.

Tasks:

1. **Setup:** Install `rdflib` via pip.

```
pip install rdflib
```

2. **Analysis:** Load `d3fend-protege.ttl` and find the `definition` of `ProcessSpawnAnalysis`.
3. **SPARQL Drill:** Use the following query to list all countermeasures related to **Network**:

```
PREFIX d3f: <http://d3fend.mitre.org/ontologies/d3fend.owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT ?did ?label WHERE {
  ?x rdfs:subClassOf+ d3f:NetworkSurveillance .
  ?x d3f:d3fend-id ?did .
  ?x rdfs:label ?label .
}
```

4. **Deliverable:** A JSON file mapping 5 DIDs to their human-readable definitions.

□ LAB 2: DECEPTIVE ARTIFACT DEPLOYMENT (DAY 148)

Objective: Deploy a "Honey-Registry-Key" and monitor for unauthorized access.

Tasks:

1. **Engineering:** Create a decoy registry key that mimics a sensitive application configuration.

```
reg add "HKLM\SOFTWARE\AlphaProject\Security\Tokens" /v "AdminToken" /t REG_SZ /d "XYZa123_DECOY" /f
```

2. **Detection:** Configure Sysmon to monitor `RegistryEvent` (ID 13) for this specific path.

3. **Correlation:** Map the detection event to D3FEND DID **D3-DA** (Deceptive Artifact).
4. **Deliverable:** A screenshot of the SIEM alert showing the user and process that accessed the honey-key.

□ LAB 3: APP-PATH HARDENING (DAY 149)

Objective: Operationalize **D3-APH** to prevent non-standard execution.

Tasks:

1. **Engineering:** Identify common "Writable" directories (e.g., `C:\Users\Public\Downloads`).
2. **Hardening:** Configure a WDAC (Windows Defender Application Control) or AppLocker policy to BLOCK execution from these directories.
3. **Test:** Attempt to run `calc.exe` from the Public folder.
4. **Deliverable:** The exported AppLocker XML policy and a log of the blocked execution (Event ID 8004).
