* **Visualization Scripts for Word Clouds and Bar Charts Script**

Generates visual summaries of frequent entities and MITRE ATT&CK techniques detected, enabling quick threat landscape understanding.

import json

import matplotlib.pyplot as plt

from wordcloud import WordCloud

import pandas as pd

# --- Load and process extracted\_entities.json ---

with open('extracted\_entities.json', 'r') as f:

entities = json.load(f)

entity\_counts = {}

for e in entities:

# Handle if entity is a list

if isinstance(e, list):

# Try first element as string key

key = e[0] if e else 'unknown'

# Handle if entity is dict

elif isinstance(e, dict):

# Try common keys, adjust if your structure is different

key = e.get('entity') or e.get('name') or str(e)

else:

# Assume e is string

key = str(e)

entity\_counts[key] = entity\_counts.get(key, 0) + 1

# Sort top 30 entities by frequency

top\_entities = dict(sorted(entity\_counts.items(), key=lambda item: item[1], reverse=True)[:30])

# Word Cloud

wordcloud = WordCloud(width=800, height=400, background\_color='white').generate\_from\_frequencies(top\_entities)

plt.figure(figsize=(12, 6))

plt.imshow(wordcloud, interpolation='bilinear')

plt.axis('off')

plt.title('Top Extracted Entities Word Cloud')

plt.show()

# Bar chart of top entities

plt.figure(figsize=(14, 7))

plt.bar(top\_entities.keys(), top\_entities.values(), color='skyblue')

plt.xticks(rotation=45, ha='right')

plt.title('Top 30 Extracted Entities Frequency')

plt.xlabel('Entity')

plt.ylabel('Frequency')

plt.tight\_layout()

plt.show()

# --- Load and process mitre\_mappings.json ---

with open('mitre\_mappings.json', 'r') as f:

mitre\_data = json.load(f)

# Prepare data for DataFrame

rows = []

for item in mitre\_data:

# If item is dict and contains needed fields

if isinstance(item, dict):

technique = item.get('technique') or item.get('technique\_name') or 'Unknown'

score = item.get('score')

if score is not None:

try:

score = float(score)

except:

score = None

if score is not None:

rows.append({'technique': technique, 'score': score})

df\_mitre = pd.DataFrame(rows)

if not df\_mitre.empty:

avg\_scores = df\_mitre.groupby('technique')['score'].mean().sort\_values(ascending=False).head(20)

plt.figure(figsize=(12, 6))

avg\_scores.plot(kind='bar', color='coral')

plt.title('Top 20 MITRE Techniques by Average Score')

plt.xlabel('MITRE Technique')

plt.ylabel('Average Score')

plt.xticks(rotation=45, ha='right')

plt.tight\_layout()

plt.show()

else:

print("No valid MITRE data found to plot.")

# --- Print summary\_report.txt ---

print("\n===== Summary Report =====\n")

try:

with open('summary\_report.txt', 'r') as f:

print(f.read())

except FileNotFoundError:

print("summary\_report.txt not found.")