Cybersecurity Psychology Framework (CPF) - SOC Integration

Contents

Overview **1**

Repository Structure **2**

Getting Started **2**

Prerequisites . . . 2

Installation . . . 2

Configuration . . . 3

Usage **3**

Running a Single Metric . . . 3

Integrating with an LLM (Advanced) . . . 4

Contributing **4**

License **4**

Disclaimer & Ethical Use **4**

Cite This Work **4**

Contact **5**

This repository contains the official implementation of the **Cybersecurity Psychology Framework (CPF)**, a novel method for quantifying human-centric risk in Security Operations Centers (SOCs). It provides a suite of algorithms to measure psychological vulnerabilities defined by the CPF taxonomy using data from standard SOC tools.

> **Academic Reference**: This work is the practical implementation of the methodology described in the preprint: **"The Cybersecurity Psychology Framework (CPF): A Method for Quantifying Human Risk and a Blueprint for LLM Integration"** *by Giuseppe Canale, CISSP* Link to Paper

Overview

The human factor is the most critical vulnerability in cybersecurity. Traditional tools focus on technical indicators, leaving psychological states unmeasured. This project operationalizes the

CPF, translating its theoretical categories into specific, measurable algorithms that analyze data from tools like Splunk, Elasticsearch, Qualys, Jira, and Slack.

The framework is organized into the ten primary CPF categories. Each category directory contains implementations for its specific subcategories (e.g., `1.1-unquestioning-compliance.py`).

Repository Structure

```
cpf-soc-integration/

   implementation/
      1.x-authority/              # Authority-Based Vulnerabilities
          1.1-unquestioning-compliance.py
          1.2-diffusion-responsibility.py
          ... (other subcategories)

      2.x-temporal/               # Temporal Vulnerabilities
      3.x-social/                 # Social Influence Vulnerabilities
      4.x-affective/              # Affective Vulnerabilities
      5.x-cognitive/              # Cognitive Overload Vulnerabilities
      6.x-group/                  # Group Dynamic Vulnerabilities
      7.x-stress/                 # Stress Response Vulnerabilities
      8.x-unconscious/            # Unconscious Process Vulnerabilities
      9.x-ai/                     # AI-Specific Bias Vulnerabilities
      10.x-convergent/            # Critical Convergent States

   docs/
      CPF-Taxonomy-Complete.pdf   # The full CPF taxonomy
      operational-sheets/         # Detailed sheets for each subcategory

   config/
      example.config.yaml         # Example configuration for API keys and endpoints

   requirements.txt
   README.md
```

Getting Started

Prerequisites

- **Python 3.8+**
- Access to SOC data sources (e.g., Splunk, Elasticsearch, Jira, Slack, Qualys APIs)
- API keys/tokens for the aforementioned services

Installation

1. **Clone the repository:**

   ```
   git clone https://github.com/your-username/cpf-soc-integration.git
   ```

```
cd cpf-soc-integration
```

2. **Create a virtual environment (recommended):**

   ```
   python -m venv venv
   source venv/bin/activate  # On Windows: .\venv\Scripts\activate
   ```

3. **Install the required dependencies:**

   ```
   pip install -r requirements.txt
   ```

Configuration

1. Copy the example configuration file and adapt it to your environment:

   ```
   cp config/example.config.yaml config/config.yaml
   ```

2. Edit `config/config.yaml` with your specific details:

   ```
   splunk:
     host: "your-splunk-host.com"
     port: 8089
     username: "your_username"
     password: "your_password"

   jira:
     server: "https://your-company.atlassian.net"
     email: "your.email@company.com"
     api_token: "your_jira_api_token"

   # ... configure other data sources
   ```

Usage

The primary use of this repository is to calculate metrics for specific CPF subcategories. Each script is designed to be run independently or integrated into a larger analytics pipeline.

Running a Single Metric

To calculate a specific metric, run the corresponding Python script. Most scripts will accept parameters or pull from the central configuration.

Example: Calculating Compliance Fatigue (5.1 Alert Fatigue Desensitization)

```
# Navigate to the cognitive overload directory
cd implementation/5.x-cognitive

# Run the algorithm for a specific analyst or team
python 5.1-alert-fatigue-desensitization.py --analyst-id "analyst_john.doe" --start-date "2023-
```

Expected Output:

```
Calculating Compliance Fatigue for analyst_john.doe (2023-11-01 to 2023-11-30)...
```

```
MTTA: 18.7 hours
Ignore Rate: 22.5%
[STATUS] YELLOW: Moderate fatigue detected. Recommend task rotation.
```

Integrating with an LLM (Advanced)

As outlined in the accompanying paper, these algorithms are designed to feed a Retrieval-Augmented Generation (RAG) pipeline for a lightweight LLM. The output of these scripts (the metrics and relevant data snippets) can be indexed into a vector database (e.g., ChromaDB, FAISS) to provide context to an LLM for sophisticated psychological risk analysis.

Contributing

Contributions are welcome! This is a large-scale research-to-practice project.

1. **Fork the repository.**
2. **Create a feature branch:** `git checkout -b feature/amazing-algorithm`
3. **Implement your algorithm** for a CPF subcategory within the appropriate directory.
4. **Commit your changes:** `git commit -m 'Add amazing algorithm for subcategory X.Y'`
5. **Push to the branch:** `git push origin feature/amazing-algorithm`
6. **Open a Pull Request.**

Please ensure your code is well-commented and includes a docstring explaining the metric, formula, and data sources.

License

This project is licensed under the MIT License - see the LICENSE.md file for details. This license allows for academic and commercial use with attribution.

Disclaimer & Ethical Use

This tool is designed for measuring organizational health and team-level psychological risk, NOT for monitoring individuals.

- **Privacy:** Always anonymize data where possible. Follow the principle of data minimization.
- **Ethics:** The implementation of this framework must be transparent to employees and comply with all local data protection regulations (e.g., GDPR, CCPA). It should be used to support and augment security teams, not to punish them.
- **Accuracy:** These metrics are proxies for psychological constructs. They should be used as leading indicators and supplemented with human-led audits and interviews.

Cite This Work

If you use this framework in your research or work, please cite the accompanying paper:

```
@article{canale2025cpf,
  title={The Cybersecurity Psychology Framework (CPF): A Method for Quantifying Human Risk and
  author={Canale, Giuseppe},
  journal={Preprint},
  year={2025},
  url={https://github.com/xbeat/CPF}
}
```

Contact

Giuseppe Canale, CISSP - g.canale@cpf3.org | ORCID

Project Link: https://github.com/xbeat/CPF