User Manual

Version 2

12/04/2025

Team Name: Cyber Recon

Sponsor: HighViz Security LLC

Team Mentor: Karthik Srivathsan Sekar

Team Members: Zachary Garza, Sean Weston, Jared Kagie, Christian Butler

Introduction

We are extremely grateful to have the opportunity to work on Cyber Recon for the betterment of Cybersecurity needs in the world today. Cyber Recon is an integrated system for vulnerability severity mapping that incorporates artificial intelligence and machine learning to provide real-time intelligence to severities that are able to be found through your current Nessus scanning. Some key highlights of this system include:

- Vulnerability enrichment through GitHub PoCs, KEV, NVD, and EPSS databases
- Model retraining capabilities after updated findings
- Anomaly detection for vulnerability outliers
- Documented reporting with formatting tailored to specific needs

The purpose of this manual is to provide the necessary documentation for installation, usability, and upgradability of Cyber Recon for HighViz and their needs. Our overall goal for this system is to remove some overhead that comes with research and vulnerability QA that takes multiple hours of research time while simultaneously providing the same intelligence enrichment practices that your team uses to determine severities.

Installation

To install this product, clone the repository to your local Apple or Mac system. This can be done with:

```
git clone <repository url>
```

Verify that package/hv-doctools contains the doctools provided by HighViz to the NAU team.

Should contain files like parse-nessus.py, findings_map.csv, nessus-sql-import.py, ect.

Next, install the requirements listed in requirements.txt. This can be done by running this command in the project directory:

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

After this, you should be good to go! See Daily Operation for use on how to run the program.

Configuration

This section explains the small amount of configuration you may want to change before using Cyber Recon in your environment.

1. Configuration Overview

After installation, Cyber Recon will run with sensible defaults. Most clients will not need to change any settings. This section covers the two items you may want to adjust: the main configuration file and the optional NVD API key.

2. System Configuration File

The main configuration file is: config/config.yaml

The key settings are:

a) API settings (optional)

api: nvd_api_key: ""

This lets you provide an NVD API key to speed up NVD lookups and improve rate limits.

b) Logging - level: "INFO"

INFO is recommended for normal use. DEBUG provides more detail if you are troubleshooting. WARNING and ERROR show fewer messages.

If you are happy with the default behavior, you can leave this file as is.

3. NVD API Key (Optional but Recommended)

Cyber Recon works without an API key, but adding one improves NVD rate limits and performance.

To configure an API key:

1. Request a free API key from the NVD website.
2. Open config/config.yaml in a text editor.

   Set: api: nvd_api_key: "your-api-key-here"
3. Save the file and restart any open Cyber Recon processes.

No other configuration changes are required for normal operation.

Daily Operation

This section explains what you will do in Cyber Recon during normal use.

1. Daily Operation Overview

On a day-to-day basis, you will mainly do three things:

1. Process Nessus scan files into reports.
2. Review and feedback your expert decisions on UNKNOWN findings.
3. Retrain the AI model whenever the knowledge base (findings_map.csv) is updated.

These are handled by three commands:

- Process a scan and generate reports:
  python scan.py
- Update the knowledge base with your expert decisions:
  python update_map.py
- Retrain the AI model from the updated knowledge base:
  python src/tools/train_from_findings_map.py

2. Processing Scans (scan.py)

This is the command you will use most often.

Purpose:

- Read a Nessus .nessus file.
- Enrich it with KEV, EPSS, and NVD intelligence.
- Apply Cyber Recon's AI model.
- Produce an Excel report and supporting files.

How to run:

1. Open a terminal or command prompt.
2. Navigate to the Cyber Recon project folder.
    - Run: python scan.py
3. A file picker window will open. Select your .nessus scan file.
4. Answer the prompts:
    - Verbose output? You can usually press Enter to accept the default (No).
    - Enable AI-powered risk assessment? Press Enter to accept the default (Yes).
    - Report format? Press Enter to accept the default (Excel).

If you just press Enter at each prompt, Cyber Recon will run with recommended defaults.

What you get:

For each scan, a new folder is created under output/, for example:

output/scan_20251205_143022/

Inside that folder you will see files such as:

- report.xlsx
  Your main, full vulnerability report with AI-adjusted severity, KEV and EPSS data, and helpful views for analysts and managers.
- unknown_findings_for_verification.xlsx
  Vulnerabilities where the AI is not fully confident and wants a human expert to review.
- high_risk_findings.csv (optional)
  A filtered list of the highest-risk items for quick action.

You can open report.xlsx in Excel and work from there. The sheets are organized for daily use (all findings, high risk only, by host, summary, etc.).

3. Updating the Knowledge Base (update_map.py)

As you review UNKNOWN items, Cyber Recon can learn from your decisions, so it does better next time.

When to use:

Run update_map.py after you have:

1. Processed a scan with python scan.py.
2. Opened unknown_findings_for_verification.xlsx and reviewed the items.
3. Record your final expert decisions (for example by exporting them to a CSV file such as verified_findings.csv).

How to run:

1. Open a terminal in the Cyber Recon project folder.
   - Run: python update_map.py
2. Use the file picker to select your verified findings CSV.
3. Answer the prompts:
   - Create backup before updating? We recommend Yes (press Enter).
   - Retrain ML models after updating? You may answer No here if you plan to run the training command yourself in the next step, or Yes if you want retraining to happen automatically.

What it does:

- Creates a timestamped backup of the current findings_map.csv.
- Merges in your new expert decisions.
- Expands the knowledge base with these new entries.

Next time you run python scan.py, Cyber Recon will recognize more patterns and suggest better severity adjustments automatically.

4. Retraining the AI Model (train_from_findings_map.py)

Every time findings_map.csv is updated, the AI model should be retrained so it reflects your latest expert knowledge. Updating the knowledge base without retraining means the new knowledge will not be used by the model during scans.

When to use:

- Run **python src/tools/train_from_findings_map.py** after each update to findings_map.csv.
- In practice, this means you should retrain immediately after running python update_map.py whenever new verified findings have been added.
- This process tends to take a couple minutes, so don't fret if it takes a while to go through a couple of steps.

How to run:

1. Open a terminal in the Cyber Recon project folder.
    - Run: **python src/tools/train_from_findings_map.py**
2. The script will:
    - Load the current findings_map.csv (your cumulative expert knowledge).
    - Extract features from text and structured fields.
    - Use SMOTE to balance classes.
    - Train and evaluate an ensemble model (targeting around 91% accuracy).
    - Save the new model under models/ (for example, models/severity_from_findings_map.joblib).

Training usually takes a few minutes on a modern machine. When it finishes successfully, future runs of python scan.py will use the updated model.

5. Recommended Workflow

For best results, treat update_map.py and train_from_findings_map.py as a pair:

1. Run python scan.py to process your scan.
2. Review UNKNOWN items in unknown_findings_for_verification.xlsx and export your verified decisions to a CSV file.
3. Run python update_map.py to add these decisions to findings_map.csv.
4. Immediately run python src/tools/train_from_findings_map.py so the AI model learns from the updated findings_map.csv.

Maintenance

This section explains what you should do on a regular basis to keep Cyber Recon healthy and useful over the long term.

1. Routine Tasks

Here is a simple schedule you can follow:

- Every vulnerability assessment
    - Run python scan.py to process your Nessus file(s).
- After you review UNKNOWN items from a scan
    - Run python update_map.py to add your expert decisions to findings_map.csv.
    - Immediately run python src/tools/train_from_findings_map.py so the AI model reflects the updated knowledge base.
- Periodically (for housekeeping)
    - Clean up old logs in the logs/ folder.
    - Clean up old output folders you no longer need in output/.
    - Copy findings_map.csv and the models/ folder to a safe backup location if you are about to make major changes or move servers.

Cyber Recon automatically creates backups of findings_map.csv whenever you update the knowledge base using update_map.py, but it is still a good idea to occasionally copy files to external storage (such as an internal file share or external drive).

2. Logs

a. All logs are stored under: logs/

These logs are useful for troubleshooting, but they can grow over time.

If the logs/ folder becomes large, you can safely delete older log files. We recommend keeping the last month of logs and removing anything older, especially in production use.

You can do this manually using your file browser by sorting by date and deleting older .log files. No special commands are required.

3. Output Reports

Scan results are stored under:

output/scan_YYYYMMDD_HHMMSS/

Each folder corresponds to one processed scan. Over months of usage, this can add up.

We recommend:

- Keeping recent assessments and any folders used for audits or ongoing work.
- Archiving older folders to a zip file or shared drive if you want to keep them.
- Deleting folders that are no longer needed.

This helps keep disk usage under control without affecting Cyber Recon's ability to run.

4. Backups

The most important files to protect are:

- package/hv-doctools/findings_map.csv
  Your accumulated expert knowledge.
- models/
  The currently trained AI models.

In addition to the automatic backups created by update_map.py, we recommend:

- Periodically copying findings_map.csv and the models/ folder to a separate backup location.
- Making a quick backup copy of the entire Cyber Recon project folder or Git repository before major changes or upgrades.

If something goes wrong, you can restore from one of these backups and avoid losing your training history.

5. Source Code and Repository Access

As part of delivery, you receive:

- Ownership of the Cyber Recon Git repository (transferred to your organization).
- A User Manual to assist in application use.

If you ever move Cyber Recon to a new server or environment, you can:

1. Clone or copy the repository.
2. Follow the installation steps.
3. Restore findings_map.csv and the models/ folder from your backups if needed.

By keeping up with this basic maintenance and retraining the model every time findings_map.csv is updated, Cyber Recon will continue to improve over time: each scan contributes more knowledge, and each retrain makes the AI a better reflection of how your own experts think about risk.

Troubleshooting

One of the early bugs we ran into was having duplicate data in our training datasets, which caused the model to overfit. While we have fixed this by removing the duplicate data, through retraining using new data, this problem could reappear. The solution to this problem is to make sure that whenever the model is retrained, that new data isn't already present in the training data.

Additionally, we tried to design the scan in such a way that it was simple to use. This means that when an error occurs, it will be easier to investigate because all the code and data are easy to access and understand. Overall, this reduces the time it takes scouring through code and logs trying to identify issues.

Finally, using the verbose and debug options when running a scan can give information about potential errors and why they occur. Additionally, the log files that are generated after a scan is completed contain similar information and can be accessed any time after a scan is run. Using these tools alongside the scan will help troubleshoot many of the errors that may occur when running the system.

Conclusion

This concludes with the user manual for Cyber Recon. Working on this project has been an incredible experience for all of us, and we are very thankful that you provided us with this opportunity. We hope that this product is everything that you want it to be and that it improves your team's workflow. From your Cyber Recon developers, we wish you and your team the best in your future endeavors. While we will be moving on to new prospects after this semester, we will be happy to stay in touch during these next months to answer any potential questions you may have. Again, thank you for all that you have done for us, and all the opportunities that we had because of this project.

Thank you for everything from the Cyber Recon team:

Christian Butler, Zachary Garza, Sean Weston, and Jared Kagie