Reinforcement Learning for Cyber Security with a Gymnasium Simulation Environment: the HTTPS Intrusion Detection System

George Mason University | Cyber Security Engineering 689 | Final Project 2.1
Chris Limson | *climson@gmu.edu* | May 7, 2024

Introduction

The HTTPS Intrusion Detection System (HIDS) determines ideal decisions of whether to signal an alert due to an attempted malicious attack, or allow the web server to proceed with nominal operations of accepting remote requests. Reinforcement Learning (RL) will simulate scenarios for the system and provide feedback with reward values that teach HIDS how to respond.

Background

HIDS is motivated by the researcher's own personal website, run with NGINX which would record access and error logs that monitor requests to the server. Inspection of the logs shows more attempts than intuition suggests, and this is for just a low-profile non-commercial static website. High-profile corporate organization websites must be more constantly requested.

Many cases are harmless web crawlers, or bots whose only purpose is to index content all over the Internet. The access and error logs, however, exhibit some request headers that search for vulnerabilities by feeding SQL injection code, direct calls to PHP scripts, hexadecimal-encoded strings, etc., and those should be flagged as harmful. Identifying each exploit's strategy, RL can defend against similar intrusion attacks in the future.

Solution

A related course-assignment implements a Monte Carlo RL simulation with plain Python code. This solution's infrastructure of the Farama Foundation's Gymnasium API is a library of enhanced RL features. Due to time-constraints, however, HIDS does not yet entirely conform to the API and still uses basic constructs. For example, while modeling the sequential attack scenario does make observation and action spaces out of the Discrete class, the values still map to a table of Python strings, instead of using space Text classes.

Several types of attacks are found in the logs. Those types will be represented by the RL observation. The benign site visit is also represented and will be the initial observation.

An RL action either allows the visit to continue with no alert, or to signal the alert of a possible threat.

Implementation

In `hids.py`, the variable `info` is a supplement dictionary that will not be used now for simplicity, but is included as a requirement of the Env class.

The `reset` function is implemented by bringing the webserver to a ready *state*, or *environment* in the context of the API, *observation* in the fundamental context of RL.

The `step` function logic decides whether to signal an alert or to continue. Here a value is rewarded, based on whether the Agent correctly assesses a valid visit, a threat or makes a mistake. The concept of this response is the RL *action*.

The `make` function is already implemented by the registration of the API's custom environment. Upon installation, the `cymnasium` package is added to the user's sandbox. With `make`, a simulation or Agent only needs to call the external environment ID, `HIDS-v0`.

The included environment simulation `hids_env_sim.py` would be replaced with the scope of the Agent, which is to determine the solution for the best decisions.

Conclusion

This project was conducive to understanding the basics of Gymnasium API and Reinforcement Learning. Learning how to use the API, however, took time and effort to the point that a simple application needed to be done for clarity. The project can then be built on for more elaborate research and development.

Notes

The directory structure, code, output, text listing of logs, and two sample logs themselves can be found in the GitHub repository created for this project,

```
https://github.com/chrimson/Cymnasium
```

This project has been tested successfully on Centos Stream 9, Ubuntu 24 and AWS Linux 2023. On bare installations of these OS's, the following command (or similar) may need to be done,

```
yum install -y gcc gcc-c++ git python3-devel python3-pip
```

The warning of the pip install versions may probably be ignored,

```
WARNING: You are using pip version 21.3.1; however, version 24.0 is available
You should consider upgrading via the 'python3 -m pip install --upgrade pip' command
```

To suppress that warning

```
export PIP_DISABLE_PIP_VERSION_CHECK=1
```

For this reason, using a virtual Python environment is recommended

```
python -m venv cyber
. cyber/bin/activate
```

References

https://gymnasium.farama.org

https://github.com/Farama-Foundation/Gymnasium

https://github.com/Farama-Foundation/gym-examples