Weems - Adaptive IP Traffic Filter using DQRN

Lucas Miller
University of Colorado Boulder
ASEN 5264 - DMU
Boulder, CO
Lucas.Miller-6@colorado.edu

Jacqulin Justin
University of Colorado Boulder
ASEN 5264 - DMU
Boulder, CO
jaju8756@colorado.edu

The authors grant permission for this report to be posted publicly. Code is available at `https://github.com/luc-spec/weems`.

Abstract— **Adaptive network filtering is a fickle task. We demonstrate some moderate success in implementing Deep Recurrent Q-Networks (DQRN) to make sequential traffic filtering decisions via OpenSnitch (a popular firewall application which presents adaptive firewall decisions to a user). We bolster our adaptive network policy with large, well-refined, block list sets from popular lookup table DNS filters. The core of this approach is a dueling structure separating value and advantage streams. The largest gap in this work is a robust simulation that fully represents the depth of uncertainty that a system like this would face in practice. The variability in network traffic from user to user or environment to environment makes a truly representative simulation difficult to generate dynamically without access to massive amounts of data. Regardless, we achieve block rates that are slightly more aggressive than blocklist filters while allowing enough traffic through in the proof of concept that a production implementation with proper OpenSnitch hooks could conceivably yield an acceptable user experience.**

I. Introduction

More and more, applications are generating network traffic that is not strictly vital to their function. Telemetry, usage data, and more are all transmitted from an application with and without clear consent from a user. Static DNS filtering approaches, such as PiHole, are limited in their application by nature of being static. Other tools like OpenSnitch are more dynamic but implement simple policies (a timeout to default action) in lieu of user interaction for every connection. This paper proposes an intelligent policy implementation to perform adaptive IP network traffic filtering, akin to "DELM" by Ahmed et al and "Adaptive blacklist-based packet filter" from Meng et al. Existing technologies provide hooks to simplify implementation and focus on policy formulation. Our proposal minimizes scope by creating a basic learning policy and integrating with existing tools. Future iterations can extend this approach to develop per-application adaptive policies for greater tailoring to a given environment.

II. Background

The increasing sophistication and volume of cyber threats have driven a shift from traditional static filtering methods toward adaptive, intelligent traffic control systems. Static blacklists and signature-based Network Intrusion Detection Systems (NIDS) have historically been effective due to their low false-positive rates; however, they struggle to maintain performance under high-throughput or zero-day attack scenarios where attackers use polymorphic or evolving behaviors to bypass detection.

To address this challenge, Meng and Kwok (2014) proposed an adaptive blacklist-based packet filtering system that augments NIDS with a lightweight filtering layer. Their approach relies on dynamically updated IP confidence scores based on observed signature matches, allowing the system to proactively block likely malicious packets before deeper inspection. This method reduces processing overhead while maintaining detection efficacy, particularly in Distributed Denial of Service (DDoS) mitigation scenarios.

Building upon the need for adaptive and intelligent models, Ahmed et al. (2024) introduced DELM (Deep Ensemble Learning Model), which combines Feedforward Deep Neural Networks (FDNNs) and Deep Belief Networks (DBNs), supported by Adaptive Feature Aggregation (AFA). DELM leverages Conditional Generative Adversarial Networks (CGANs) to synthetically balance training datasets, thereby enhancing its robustness in detecting malicious traffic under class imbalance conditions. Their model demonstrates strong adaptability to shifting network traffic patterns through dynamic feature selection and deep learning-based generalization.

At the infrastructure level, Sonchack et al. (2023) proposed Dapper, a programmable packet filtering framework that allows administrators to define application-specific filtering rules using packet metadata. Dapper brings deep visibility into traffic flows, supporting connection-aware filtering across diverse protocols. However, its performance can degrade when rules become overly complex or when application behaviors change rapidly.

Zhang et al. (2022) introduced TeeFilter, a scalable DDoS detection system that "tees" packets for sampling and analysis in parallel with forwarding. This architecture enables detection of large-scale volumetric attacks with low latency impact. However, TeeFilter's dependence on sampling intervals limits its ability to capture bursty or short-lived attack patterns, requiring a trade-off between visibility and processing efficiency.

In contrast, Meng et al. (2014) highlighted the importance of fairness in traffic filtering by designing a system that reduces collateral damage when dropping packets. Their adaptive strategy ensures that legitimate traffic retains access to bandwidth, a consideration especially important in enterprise or shared environments where over blocking can harm service availability.

III. Problem Formulation

Our approach is to form the task of network filtering as a POMDP, with a Deep Recurrent Q-Network architecture for sequential decision making. At every evaluation step we have a destination (FQDN or IP address) and a simple set of actions, *Allow* or *Block*, each with a time association of “Once”, “Until Restart”, and “Forever” for a total of six possible actions. *Allow* or *Block* actions are required to progress on to another evaluation point.

Our state space is similarly simplistic, spanning: " Legitimate”, "Suspicious”, and “Malicious" depending on their reputation. At a given state (belief regarding a connection request) we make additional observations via research available to model such that it can gain more information about a particular traffic destination to refine belief. We derive a reputation value-based combination of biases, action history, and additional research actions. Additional observations such as blocklist lookups, or URL classification come at the cost of temporal delays which increases risk of impacting the utility of time-sensitive traffic. In this effort we make the conscious decision not to penalize these delays and possible disruptions. We would rather optimize for making the correct decision, especially during training, as opposed to simply rewarding fast decision making.

In this work, we enhanced a domain-based request classifier by incorporating key preprocessing steps to improve accuracy and consistency. Specifically, we cleaned up blocklist files to eliminate noise and normalize domain formats. These adjustments significantly improved the classifier's performance, resulting in sharper probability distributions and more reliable predictions. Additionally, by including both the destination IP and the process arguments in the feature set, we ensured a more comprehensive understanding of request context, reducing potential loopholes that malware could exploit to evade detection.

$$S = \{Legitimate,\ Suspicious,\ Malicious\}$$

Equation 1 - State Space

$$A = \left\{ \begin{matrix} Allow\ Once, \\ Allow\ Until\ Restart, \\ Allow\ Always, \\ Block\ Once, \\ Block\ Until\ Restart, \\ Block\ Always, \\ Ask\ User \end{matrix} \right\}$$

Equation 2 - State Space

Our implementation uses Deep Recurrent Q-Networks (DRQN) as the core reinforcement learning architecture for making intelligent firewall decisions. The DRQN combines deep learning with recurrent neural networks, specifically using LSTM layers to maintain temporal awareness of network connection patterns. This is particularly valuable for a firewall application, as it allows the system to recognize sequential patterns in network traffic that might indicate either legitimate application behavior or potential threats.

The DRQN architecture in our code employs a dueling network structure, separating value and advantage streams. This design choice enables more stable learning by allowing the model to independently evaluate the value of being in a particular state versus the advantage of taking specific actions from that state. For network security decisions, this separation helps prevent overestimation of action values and leads to more conservative, security-focused decisions when appropriate.

Thompson Sampling serves as our exploration strategy, providing a Bayesian approach to the exploration-exploitation dilemma. Rather than using simpler methods like epsilon-greedy, our implementation maintains probabilistic models of action values using Beta distributions. This gives the agent a structured way to handle uncertainty about optimal actions. As the agent gathers more experience with certain applications and destinations, the exploration strategy naturally shifts toward exploitation for familiar patterns while maintaining healthy exploration for novel situations. The implementation tracks success and failure counts separately for each state-action pair, allowing the model to develop increasingly refined probability distributions for action selection.

What's particularly elegant about our implementation is how Thompson Sampling interfaces with the prioritized experience replay mechanism. The exploration strategy helps gather diverse, informative experiences, while the replay buffer prioritizes learning from the most surprising outcomes (those with high TD error). This combination accelerates learning in the large state space typical of network security applications, where certain critical edge cases might occur rarely but require special handling.

Our approach may be unusual in the world of packet filtering, where strategies typically arise from packet structures or threat models. Instead, this approach focuses on maximizing the utility that the network provides to the user. We assert that a filtering learned policy makes it possible to perform tailored packet filtering without requiring constant decision making from the user.

Our solution emulates egress hooks from OpenSnitch to gain access to network traffic. Then we form a Deep Recurrent Q-Network (DRQN) with Thompson Sampling to perform sequential decision making under partial observability. As the tailored model grows, it represents the user’s preferences more accurately and forms a more accurate belief about the reputation of consistent traffic. Thus, the user will only be prompted for input as network patterns change rather than constantly receiving prompts for similar input.

Over the course of simulation, we feed the model a series of applications and destinations from fixed lists. The simulation chooses samples at random from a short list of applications with a small number of malicious entries made clear by their names. Destinations are handled slightly differently. Samples meant to represent known or common applications that this model would be likely to see in practice are in a fictious destinations list. “Bad” or malicious destinations, on the other hand, are more realistic. These come directly from massive blocklists curated for AdGuard Home and PiHole. These are projects that rely on gigabytes of block lists and rules instructing which lists the filter should check against. By using this realistic data with sparse repetition, we make a somewhat reasonable approximation of the breadth that a filter of this nature would see in practice.

Importantly, our approach presents significant improvements over a user setting “Allow Forever” rules manually. In practice it is impossible for a user to track and understand dozens or hundreds of long-term rules in existing adaptive firewall paradigms. In our approach, the model maintains history and improves upon belief regarding destination and application reputation. This reinforcement overcomes the potential staleness of past rules. The model can request user feedback to ensure that massive numbers of network rules actually track the user’s intent.

REWARD	**STATE**	**ACTION**
3	Legitimate	Allow
-5	Legitimate	Block
-10	Malicious	Allow
1	Suspicious	Block
-1	-	Ask User

TABLE 1 - BASE REWARD FUNCTION

Our evaluation of this approach is comprised of 1000 decisions with occasional exploration of asking the user at no cost to the model, on top of the available action (at a cost) to ask the user for input. Then, we run 10,000 evaluation steps and track decision making. A generalized plot of decision making over time is found in Figure 1. One common trend during the course of this testing was saturation of decision making over a long period of time. While some sections of the process appear relatively stable, it is clear that our model resists holding a consistent balance. It is possible that this is due to the very short training interval, predictability of our “good” destinations relative to our “bad” destinations, or an unseen flaw in our architecture or implementation.

The greatest challenge in simulation is beginning to accurately represent the variability in realistic network traffic, especially between servers and one user to another. Properly representative simulation may be a project of this scale in and of itself. We began this process by attempting to capture network traffic in a virtual machine with some typical processes running. However, traffic rates proved to be With the relatively small input variation from text file sampling, we produced significant variability in the learning process. The plots below begin to capture statistics regarding the rates of allowed or blocked traffic size is far too small to truly reflect utility and effectiveness. For more objective results, we would need a more robust unsupervised simulation than the proof of concept written for this work.

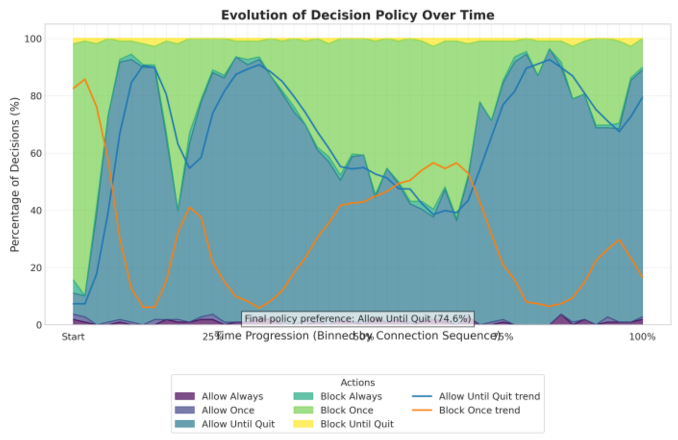

FIGURE 1 – DECISIONS OVER TIME

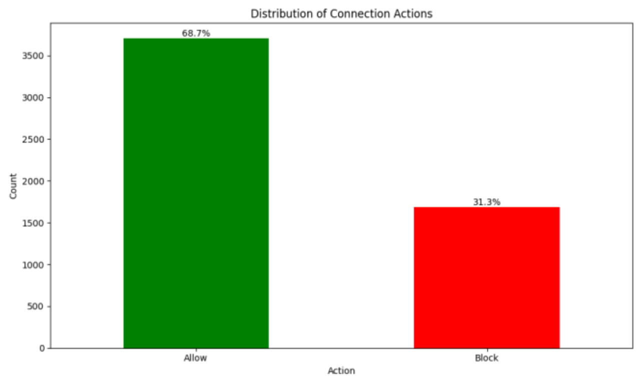

FIGURE 2 – OVERALL BLOCK RATE

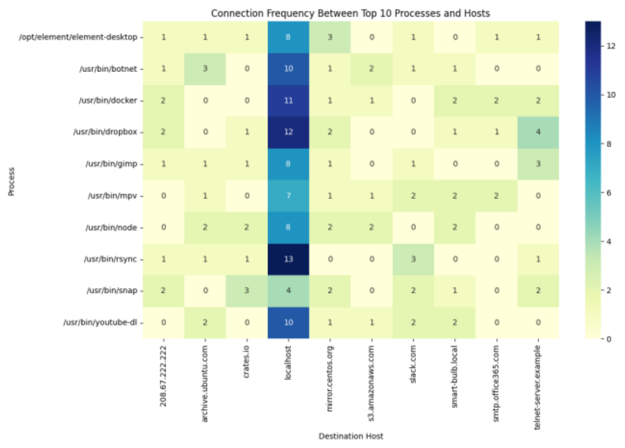

FIGURE 3 – SIMULATED CONNECTION HEAT MAP

IV. FUTURE WORK

Future efforts could expand our approach in a few ways. One interesting starting point might be to form belief regarding the relationship between applications *and* destinations, rather than focusing on destinations like we do here. From training it is clear that people make decisions to allow or block traffic based on who is asking, not just where the traffic is headed. Another reasonable modification could be to experiment with different algorithms to replace our DRQN implementation, or a survey of algorithms to form a better understanding of performance in this use case.

Another method would be to form this problem in a game structure. In our work, applications are simply part of the environment and the user's decisions form a sort-of truth that we can form our belief around. Future work could change this paradigm such that applications (including the smart firewall) and the user could all be considered players. It may be interesting to investigate how symbiotic or antagonistic differing reward functions could be, in addition to possible equilibria that are exposed by a game framework.

V. CONTRIBUTIONS AND RELEASE

Lucas was the primary developer in this work. While the codebase makes heavy use of PyTorch patterns and infrastructure, the DQRN agent, policy, and hooks were effectively implemented from scratch for this work. Full list of contributions below.

- System design
- Blocklist data selection
- DRQN selection
- Agent Implementation
- UI Implementation
- Simulation
- Visualizations
- Opensnitch stub (gRPC interface)

Jacqulin implemented observation actions like blocklist lookups and off-the-shelf text classification.

- URL classification
- Blocklist Lookups

VI. REFERENCES

[1] L.-F. K. Yuxin Mengn, "Adaptive blacklist-based packet filter with a statistic-based approach," *Network and Computer Applications,* p. 92, 2013.

[2] J. C. E. A. R. N. A. S. A. L. MUKHTAR AHMED, "DELM: Deep Ensemble Learning Model for Anomaly Detection in Malicious Network Traffic-based Adaptive Feature Aggregation and Network Optimization," *ACM Trans. Priv. Sec.,* vol. 27, p. 36, 2024.

[3] J. Z. X. Z. X. C. a. X. Y. Wenlong Shi, "DTS-AdapSTNet:anadaptivespatiotemporal neuralnetworksfortrafficprediction withmulti-graphfusion," *PeerJ Comput. Sci.,* p. 37, 2024.

[4] N. B. T. M. Jonas Röckl, "TeeFilter: High-Assurance Network Filtering Engine for High-End IoT and Edge Devices based on TEEs," in *In ACMAsia Conference on Computer and Communications Security (ASIACCS '24)*, Singapore, 2024.

[5] D. S. Y. F. L. S. M. Z. C. E. P. R. Jun Li, "Adaptive Distributed Filtering of DDoS Traffic on the Internet," *2023 IEEE Conference on Communications and Network Security (CNS),* p. 12, 2023.