Improving NDN Resilience: A Novel Mitigation Mechanism Against Cache Pollution Attack

Abdelhak Hidouri, Haifa Touati, Mohamed Hadded, Nasreddine Hajlaoui, Paul Muhlethaler, Samia Bouzefrane

Improving NDN Resilience: A Novel Mitigation Mechanism Against Cache Pollution Attack

Abdelhak Hidouri
National School of Computer Science (ENSI)
University of Manouba
Manouba, Tunisia
abdelhak.hidouri@lecnam.net

Haifa Touati
Hatem Bettahar IReSCoMath Lab
University of Gabes
Gabes, Tunisia
haifa.touati@univgb.tn

Mohamed Hadded
Abu Dhabi University
Abu Dhabi, United Arab Emirates
mohamed.elhadad@adu.ac.ae

Nasreddine Hajlaoui
Unit of Scientific Research, Applied College
Qassim University
Unayzah, Saudi Arabia
nasreddine.hajlaoui@fsg.rnu.tn

Paul Muhlethaler
INRIA Paris
Paris, France
paul.muhlethaler@inria.fr

Samia Bouzefrane
CEDRIC Lab
Conservatoire National des Arts et Métiers (CNAM)
Paris, France
samia.bouzefrane@lecnam.net

***Abstract*—Cache Pollution Attacks (CPA) are a growing concern in Named Data Networking (NDN) due to their potential to disrupt network services and compromise data integrity. While several defence mechanisms have been developed, they often struggle to keep up with the evolving nature of such attacks. This paper introduces a cutting-edge approach for detecting and mitigating CPA in NDN, utilizing Deep Reinforcement Learning (DRL). By employing a DRL framework, we leverage the power of deep neural networks to learn complex patterns within network traffic. Our DRL algorithm is designed to analyze the intricate dynamics of NDN environments and make informed decisions about cache management to protect against CPA. The agent's learning process involves continuous interaction with the network, allowing it to adapt to CPA attack vectors and evolving NDN network conditions. The DRL-based mitigation mechanism is evaluated using the official NDNSim simulation environment. The results show that the DRL agent effectively identifies and mitigates CPA with high accuracy, thereby improving the Cache Hit Ratio, while incurring an acceptable increase in memory usage.**

***Index Terms*—Cache Pollution Attack, NDN, Deep Reinforcement Learning, NDN security, IDS, Deep Q-Learning**

I. Introduction

More recently, the uses of today's Internet have evolved considerably over the years. Every day, users generate and consume massive amounts of data in an uncontrolled and scattered manner. In fact, the huge content distribution which mostly became based on the consumption of multimedia material has transformed data communication. If the infrastructure has improved with the deployment of optical fibres, Wi-Fi or mobile internet networks, the architecture of the Internet has not evolved and is still based on the TCP/IP protocol stack [1]. This architecture allows hosts to access the service of another host following the client-server model of the Internet based on IP addresses. However, The TCP/IP architecture was not built with the goal of providing wide-scale content delivery to numerous consumers. As a result, academics are reconsidering the design of the Internet infrastructure. Many solutions have been proposed in this context to cope with the massive volume of data flow. These architectures are called Information-Centric Networks (ICN). They focus on the data itself and not on the hosts in the network. Messages are thus no longer routed based on the host address, as was the case with the IP design, but rather on the name of the data. Several ICN designs, mostly in the United States and Europe, have been suggested in the literature [2]. The NDN architecture appears to be the most promising of all of these projects [3]. This new architecture is designed to enable large-scale content delivery. In NDN, data is the basis of the exchange. Each content is identified by a name or a prefix which has a hierarchical structure, like URIs. This content can be cached in intermediate NDN routers to deliver later requests for the same piece of content. Furthermore, NDN verifies the content's legitimacy, and each content is self-signed by its supplier, preserving by that the content's integrity and validity [4]. With these various new features, NDN is immune to the classical attacks of TCP/IP. However, despite its "secured-by-design" architecture, NDN is susceptible to attacks that primarily exploit the caching process, resulting in disruptions to data availability and the intentional prolonging of data retrieval such as the case of Cache Pollution Attack (CPA) [5]. Therefore, the development of effective mitigation mechanisms is crucial. This paper introduces a DRL-based approach to detect and mitigate CPA in NDN. It is carefully designed to adapt to the NDN environment dynamics and to update its decision in real-time based on the latest observations.

The rest of the paper is organized as follows: In Section II, we detail NDN security architecture design and its limits, in

which we introduce the in scope attack such as CPA attack. We give, in Section III, an overview of the related works in this field. We introduce our mitigation mechanism and its system model in Section IV. Then, we elaborate our simulation process and in depth analysis of our mitigation solution in Section V. And we conclude the paper in Section VI.

II. NDN Architecture and Security Limits

In NDN, when a user seeks to get content, he sends an Interest packet to the network, which is responded by a Data packet containing the requested data content [6]. The Interest packet includes the name of the data, i.e. the data's prefix, as well as a nonce and other potential optional fields, such as the request's validity or the public key of the owner who signed the message [7]. Data packets are composed of the data's name (the same prefix as the Interest packet), the data's content, and the content's signature. Each node in NDN implements three essential components such as the Content Store (CS), the Pending Interest Table (PIT) and the Forwarding Information Base (FIB). When the router receives an interest packet, it first looks in the CS for the matching data. The data packet is returned to the sender if it is found in the CS. If not, the router will conduct a PIT search. If a matching PIT entry is found, the incoming interface is added to the interface-list of that entry and the interest packet will be discarded. Otherwise, an entry is created in the PIT. Finally, the FIB sends the interest packet to the following hop in accordance with the longest matching prefix. After receiving the data packet, the router determines whether there is a matching entry in the PIT. If the entry is not found, the data packet is considered then as an unsolicited one, and it's going to be dropped. If it exists, then the data packet is forwarded to every interface present in the corresponding PIT entry. In the way to the requesting consumer, the CS caches that data packet [8]. Security and privacy are incorporated into the design of NDN. It includes new signature and cryptography technologies that protect data from being intercepted by malicious users. To ensure safe data transmission, its associated producer signs each data packet. Consumers and neighbour nodes can use this signature field to verify data integrity and provenance. Data caching is another important characteristic of NDN. The Content Store (CS), stores the data received for future requests by the same consumer or even other neighbour nodes, allowing the data to be retrieved more rapidly. Even with its "secure-by-design" architecture, NDN is vulnerable to attacks aiming primarily to abuse the caching process, disrupt data availability and maximize data retrieval delay, such as the case of the Cache Pollution Attack (CPA). According to [9], the aim of a CPA is to store unpopular content in the NDN router by requesting unpopular data packets, with the intention of rendering the cache inaccessible to legitimate consumers. Consequently, during the execution of the attack, there is a degradation in the cache hit ratio of the NDN routers CS, resulting in a decreased likelihood of a cache hit for requests coming from legitimate consumers [10].

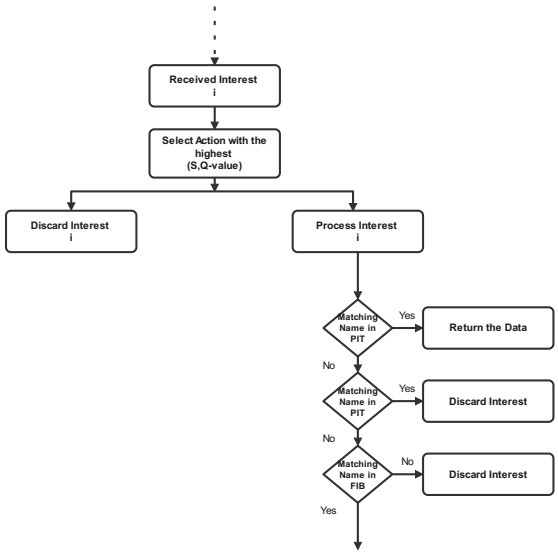

Fig. 1. Process of Action selection

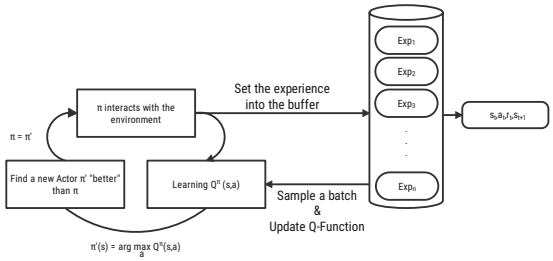

Fig. 2. Experience replay process

III. Related Work

Authors of [15] propose a secure framework called FuRL, which is based on Fuzzy Restricted Boltzmann Machine, to detect and mitigate network anomalies in NDN. The proposed framework includes a reward-based cache replacement (Re-Bac) algorithm to avoid CPAs. One of the metrics used in the methodology is the count of interest packets, which can be used to characterize the nature of the consumer (malicious/legitimate). The proposed framework may require significant computational resources, which may not be feasible in resource-constrained environments as the NDN environment.

Yang Cao et al. [16], propose an attack detection and defence schemes based on the consumers behaviour and network parameters such as: the number of interest packets in the network, the number of requesters, the distribution of the requesters, the core network traffic and the Cache Hit Ratio (CHR). The method uses SVM algorithm to distinguish between the legitimate and the malicious contents. The suggested mechanism might encounter numerous challenges, with the foremost concern being its potential to influence

the performance of the NDN network in relation to CPU utilization, memory overburden, and space storage depletion.

Authors of [17] propose a scheme based on Gini Impurity to detect network under cache pollution attack. they introduce monitor metrics for quantitative anomaly detection of requests. The paper also proposes an Interest throttling mechanism based on trust to reinforce NDN network under CPA. Moreover, the paper proposes a mechanism that generates attack tables that store malicious requests, contrary to the previous approach that simply drops malicious interests. However, there are some limitations to this paper, which are as follows: (1) The proposed defence mechanism may require additional computational resources, which may not be feasible in resource-constrained NDN environments. (2) The paper assumes that the attacker is not able to compromise the trust mechanism, which may not be true in all scenarios. (3) The proposed defence mechanism may not be effective against sophisticated attackers such as in LDA, which can escape the defence pattern.

Naveen Kumar et al. [18] introduce a new parameter for detecting CPA which is based on the number of distinct users requesting interest packets for a content over a period of time. This parameter is used to measure the popularity of contents and is not affected by the local popularity of the attacker's content. By analysing the number of distinct users requesting a content, the proposed approach can be identified if an attacker is repeatedly requesting unpopular contents to pollute the cache. This new parameter is a key component of the Interface-Based Popularity Caching (IBPC) approach, which aims to mitigate CPA in NDN. The IBPC approach caches content based on the number of interfaces that receive the content, and the popularity of contents is proportional to the ratio of the number of interfaces on which the content is received to the total number of interfaces. The mechanism doesn't consider the potential of changes in the pattern of the attack, such as the variation of the contents demanded by the malicious users, which can impact the overall the cache of the CS in NDN routers.

IV. DRL-BASED CPA DETECTION SOLUTION

The objective of our agent is to systematically examine all feasible states in order to identify the correct decision and decrease the frequency of CPA attack patterns. In this scenario, our agent carries out two distinct phases: (1) the exploration phase and (2) the exploitation phase. During the exploration phase, our agent utilizes the state space set as an input for our neural network to acquire knowledge about all potential actions in accordance with the optimal q-values obtained. Consequently, the agent strives to maximize the current reward and optimize all feasible pairs of (state, q-value) to attain higher future rewards. The subsequent step involves the exploitation phase, where our agent employs the resulting optimal pair of (state, q-value) to determine the associated action. In this context, the agent achieves a balance between the two aforementioned phases by employing the ϵ-greedy strategy, wherein the agent explores with a probability of epsilon and exploits with a probability of 1-ϵ. This approach enables the agent to gradually transition from exploration to exploitation as it gains more knowledge about the environment, ensuring that it explores new actions while also leveraging its existing knowledge to enhance performance.

In order to properly model our Deep Q-learning mechanism [11], several essential elements need to be specified such as:

- *Q-Value Prediction:* The Deep Q-Learning is a neural network that predicts the Q-values for given states and actions. This prediction is denoted by $Q(s, a; \theta)$.
- *Target Q-Values:* The target Q-values are the expected future rewards plus the discounted estimated Q-value of the next state. These are denoted by $Q_{\text{target}}(s', a'; \theta^-)$.
- *Experience Replay Buffer:* The replay buffer $\mathcal{R}$ stores past experiences, which consist of the current state, action taken, received reward, and the next state [12]. From this buffer, we sample a mini-batch of transitions as shown in Figure 2.
- *Loss Function:* The loss function is the mean-squared error between the predicted Q-values and the target Q-values. It is denoted by $L(\theta)$.
- *Parameter Update:* The parameters of the Q-Learning, θ, are updated to minimize the loss function using gradient descent.
- *Target Network Update:* The target network, θ^-, is periodically updated to match the parameters of the Deep Q-Learning. This helps stabilize the learning process.

The full update equation of the Deep Q-Learning algorithm can be written as follows:

$$L(\theta) = \mathbb{E}(s, a, r, s') \sim \mathcal{R}[(r + \gamma \max a' Q_{\text{target}}(s', a'; \theta^-) - Q(s, a; \theta))]^2$$

Where:

- $L(\theta)$ is the mean-squared error loss function.
- θ are the parameters of the main network.
- θ^- are the parameters of the target network.
- $\mathcal{R}$ denotes the replay buffer.
- $Q(s, a; \theta)$ is the Q-value predicted by the main network for state s and action a.
- $Q_{\text{target}}(s', a'; \theta^-)$ is the Q-value estimated by the target network for the next state s' and the best action a'.
- γ is the discount factor.
- r is the immediate reward.

The goal is to minimize this loss function to improve the network's ability to predict the Q-values accurately. For ease of description, as follows the steps that our mechanism is using to achieve its goal:

1) The Q-learning predictions are computed using the forward pass through the network with the current state and action as inputs.
2) The target Q-values are computed using the maximum Q-value over all possible actions for the next state, according to the target network.
3) The loss function is the mean-squared error between the predicted Q-values and the target Q-values.

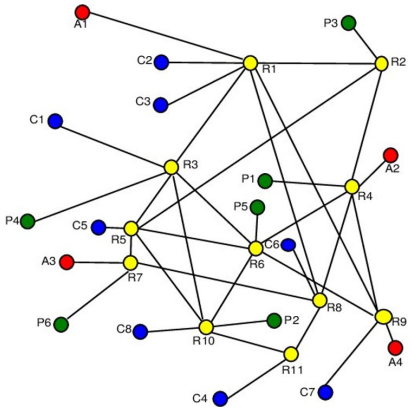

Fig. 3. DFN Topology.

4) The gradients of the loss function with respect to the network parameters are computed.
5) The network parameters are updated using stochastic gradient descent (SGD) with a learning rate (α) and the computed gradients.
6) Periodically, the target network parameters are copied from the Q-learning parameters to stabilize learning.

This process is repeated for a number of episodes until the Deep Q-Learning parameters converge, and the agent is able to make near-optimal decisions based on the learned Q-values. Furthermore, our agent is defined by three essential components, namely the *State* space, the *Action*, and the *Reward* function.

First, the state space of our agent presents the input of the neural network architecture. Based on our assessments on overall most impactful metrics that can be influenced by the presence by the CPA attack in our previous works [9] [13] [14], we defined the state space as combination of three major metrics: (1) The Average Cache Hit Ratio (AVG-CHR), (2) The Average Inter-Arrival Time (AVG-IAT) and (3) The Hop Count variation (HC):

$$S = \left[AVG-CHR; \quad AVG-IAT; \quad HC\right]$$

Second, the Action is modelled as follows:

$$A = \begin{cases} \text{Discard Interest } i \\ \text{Process Interest } i \end{cases}$$

Finally, the reward scaling function is given by:

$$R \ = \{AVG\_CHR_{t+1} - AVG\_CHR_t\}$$

V. Performance Evaluation

To assess the performance of our solution, we performed several simulations using the official network simulator (NS3) and the NDNSim2.8 module. We used the DFN topology as shown in Figure 3, in which we included four attacker nodes and eight legitimate consumers. The DRL hyper-parameters and the miscellaneous parameters are listed in Table I as well as the hardware and software configurations. Using these settings, we evaluated our solution in terms of CPA detection accuracy, AVG-CHR and memory usage.

TABLE I
Simulation Parameters

Environment Settings	**Description**
Operating System (OS)	Ubuntu 18.04.5 amd64
Memory (RAM)	20 GB
Processor (CPU)	Intel Core i3-1005G1
Graphic Card (GPU)	Nvidia 920 2 GB
NDN network Parameter	**Value**
Simulation time	106s
Number of Nodes	29
Number of Legitimate Consumers	8
Number of Attackers	4
Consumer type	ConsumerZipfMandelbrot
Attacker's Interest rate	120, 160, 200, 220, 260
Legitimate Consumers Interest rate	120
Time of Launching the Attack	Attack = 8s and No Attack = 6s
Router CS size	150
Cache policy	LRU
Our Agent Hyper-parameter	**Value**
Policy Hidden Sizes	[128,128,256,256]
Policy Hidden Activation	ReLU
Discount Factor	0.8
Learning Rate	0.1
Epsilon (ϵ)	0.99
Optimizer	ADAM
Miscellaneous parameters	**Value**
Evaluation Interval	1222
Evaluation Episodes	10

A. Accuracy and AVG-CHR while varying the DRL hyper-parameters

1) Variation of the epsilon (ϵ): The epsilon parameter in the epsilon-greedy technique is a crucial factor that determines the equilibrium point between exploration and exploitation in reinforcement learning algorithms. The epsilon-greedy strategy is a well-established approach utilized for determining the course of action that an agent should undertake. This particular strategy entails primarily selecting an action that maximizes the expected reward, while occasionally opting for a random action with a probability of epsilon. By embracing this methodology, the agent is able to effectively investigate the environment and potentially uncover superior actions, all while capitalizing on its existing knowledge in order to optimize the overall reward. To determine the optimal epsilon value, Figures 4 and 5 illustrate how accuracy and the average cache hit ratio (AVG-CHR) change with different epsilon values. By simulating a range of epsilon values from 0.1 to 0.99, we identified that 0.99 yields the highest performance. Specifically, it achieves 98.87% accuracy and an 80% AVG-CHR, surpassing all lower tested values. The reason behind this specific value, that in the case of detecting CPA attack in NDN architecture, this value encourages the agent to explore wider range of actions specially that attack pattern is too dynamic and can be unpredictable. To summarize, it's better to deeply explore in dynamic environment than exploiting and being more optimistic about the action taking [14].

2) Variation of the Learning Rate (α) and the Discount Factor (γ): The learning rate (α) and the discount factor (γ) are two important parameters in our DRL-based CPA detection mechanism. The learning rate α determines how much the new obtained information influences the update of the Q-Value for a state-action pair, while the discount factor (γ) determines the importance of future rewards in the Q-Value update. Figures 6 and 7 illustrate our exploration of various α and γ values, ranging from 0.1 to 0.9. Among these, the most favourable outcomes were observed when $\alpha = 0.1$ and $\gamma = 0.9$, achieving an accuracy of 98.87% and an AVG-CHR of 80%. These values outperformed the other combinations tested. This value of α is obtained because our agent takes longer time to update the Q-Value as per definition, a high learning rate means that the new information will significantly update the Q-value. This can be considered as an advantage for our mechanism, where a lower α value indicates a more cautious approach in action selection. On the other hand, the selection of γ stems from our agent's consideration of both immediate rewards and future rewards in the detection process of our DRL-based mechanism. As defined, a smaller discount factor emphasizes immediate rewards (near 0), while a larger discount factor places more weight on future rewards (near 0.99). Balancing between the Long-Term and short-term rewards, potentially making the agent more cautious and strategic in its actions of processing an interest or discarding it.

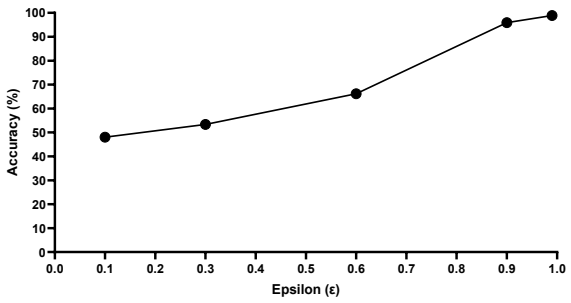

Fig. 4. The accuracy while varying Epsilon (ϵ).

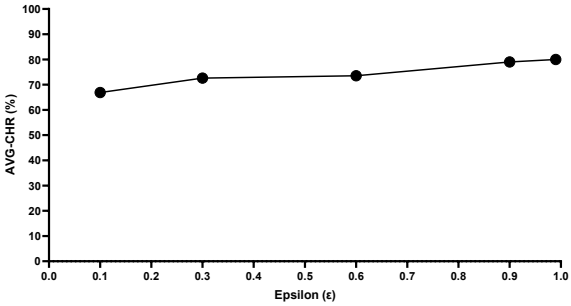

Fig. 5. The AVG-CHR while varying Epsilon (ϵ).

B. Memory usage evaluation

In this subsection, we assess the memory usage metric. Figure 8 illustrates the memory consumption of our DRL-based solution in comparison to our previous solutions, ICAN [13] and Q-ICAN [14].

ICAN [13] offers a robust solution for detecting CPA attacks, particularly in cases involving simple attack patterns. Despite utilizing multiple parameters—including Average Cache Hit Ratio, Average Inter-Arrival Time (IAT), Hop Count variation, and prefix variation—its memory consumption remains relatively low. This is attributed to its reliance on statistical measures for the detection process, resulting in an approximate memory usage of 135.849 MiB.

On the other hand, the Q-ICAN solution [14] consumes approximately 160.509 MiB which is low compared to other state-of-the-art solutions. In fact, this mechanism is based on the Q-Tables, where each cell corresponds to a state-action pair, and the value in each cell represents the expected return (or Q-value) for that pair. To properly calculate the memory usage of this mechanism, we need to calculate the memory consumption in each entry in the Q-Table. The memory consumption of each entry in the Q-table depends on the data type used to store the Q-values. Then the resulting values are multiplied by the number of actions and states. However, the DRL-based solution is proposed basically to outperform Q-ICAN and that's because Q-ICAN is limited in terms of states number which is solved by applying DRL. In Q-ICAN, increasing the number of entries in the Q-table directly correlates with increased memory consumption. This differs from the DRL-based solution, where memory usage for each neuron in a layer is primarily influenced by the weights it maintains for neurons in the preceding layer, along with a bias term. Therefore, to accurately gauge the memory usage of our DRL approach, we must calculate the consumption of each neuron's weight in addition to the bias value. Moreover, we need to consider the unpredictable limit of the pair state action in the running process of the simulation, and add to that the buffer replay and the action (interaction) between our mechanism and the NDN router. Consequently, the average memory usage is approximately 365,749 MiB. This level of memory usage is relatively low when compared to state-of-the-art solutions and is also not significantly higher than our previous solution.

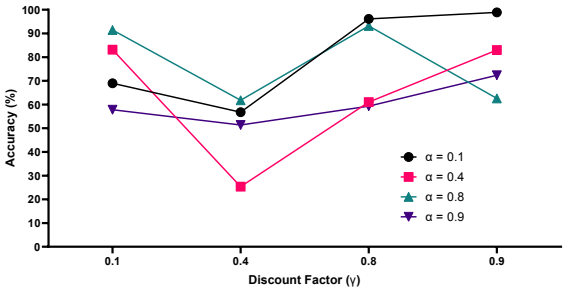

Fig. 6. The accuracy while varying the Learning Rate (α) and the Discount Factor (γ).

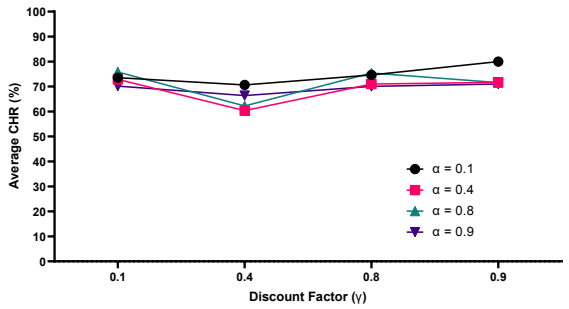

Fig. 7. The AVG-CHR while varying the Learning Rate (α) and Discount Factor (γ).

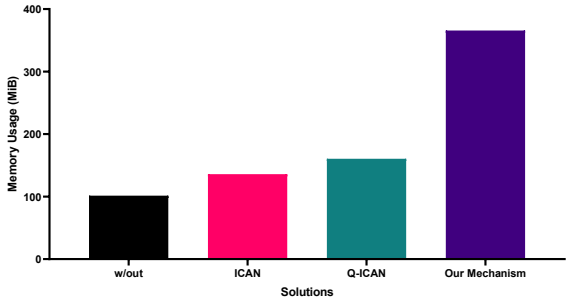

Fig. 8. The Memory Usage over different solutions

VI. Conclusion

This paper introduces a novel mitigation mechanism based on deep reinforcement learning (DRL).

The mechanism demonstrates a remarkable level of accuracy in real-time monitoring, detection, and mitigation of the cache of the Content Store (CS) in the presence of one of the most influential attacks in NDN, namely the Cache Pollution Attack (CPA). Additionally, our mechanism achieves high values of Average Cache Hit Ratio (AVG-CHR), making it a strong candidate for a new replacement policy that can significantly enhance the performance of this crucial component in the NDN architecture. Overall, this paper proposes a scaling approach for the parameters of our novel mechanism to effectively detect the CPA attack. As a perspective, we aim to assess our DRL mitigation method using a carefully designed testbed that replicates real-world conditions accurately.

References

[1] W. Shang, Y. Yu, R. Droms, and L. Zhang, “Challenges in IoT networking via TCP/IP architecture,” NDN, Washington, DC, USA, Rep. NDN-0038, 2016.

[2] Yuan, H., Song, T., & Crowley, P. . Scalable NDN Forwarding: Concepts, Issues and Principles. 2012 21st International Conference on Computer Communications and Networks (ICCCN). IEEE. doi: 10.1109/ICCCN.2012.6289305

[3] Zhang, L., Afanasyev, A., Burke, J., Jacobson, V., Claffy, K., Crowley, P., ...Zhang, B. (2014). Named data networking. SIGCOMM Comput. Commun. Rev., 44(3), 66–73. doi: 10.1145/2656877.2656887

[4] S. Mejri, H. Touati and F. Kamoun, ”Preventing unnecessary interests retransmission in named data networking,” 2016 International Symposium on Networks, Computers and Communications (ISNCC), Yasmine Hammamet, Tunisia, 2016, pp. 1-6, doi: 10.1109/ISNCC.2016.7746058

[5] Hidouri, A., Hadded, M., Touati, H., Hajlaoui, N., & Muhlethaler, P. (2022). Attacks, Detection Mechanisms and Their Limits in Named Data Networking (NDN). Computational Science and Its Applications – ICCSA 2022. Springer. doi: 10.1007/978-3-031-10522-7_22

[6] Touati, H., Aboud, A., & Hnich, B. (2022). Named Data Networking-based communication model for Internet of Things using energy aware forwarding strategy and smart sleep mode. Concurr. Comput. Pract. Exp.. 34. doi: 10.1002/CPE.6584

[7] Kumar, N., Singh, A. K., Aleem, A., & Srivastava, S. (2019). Security Attacks in Named Data Networking: A Review and Research Directions. J. Comput. Sci. Tech., 34(6), 1319–1350. doi: 10.1007/s11390-019-1978-9

[8] Abu, A. J., Bensaou, B., & Abdelmoniem, A. M. (2024). Dimensioning the pending interest table in content-centric networks. Future Gener. Comput. Syst., 152, 179–192. doi: 10.1016/j.future.2023.10.009

[9] Hidouri, A., Hadded, M., Hajlaoui, N., Touati, H., & Muhlethaler, P. . Cache Pollution Attacks in the NDN Architecture: Impact and Analysis. 2021 International Conference on Software, Telecommunications and Computer Networks (SoftCOM). IEEE. doi: 10.23919/SoftCOM52868.2021.9559049

[10] Hidouri, A., Hajlaoui, N., Touati, H., Hadded, M., & Muhlethaler, P. (2022). A Survey on Security Attacks and Intrusion Detection Mechanisms in Named Data Networking. Computers, 11(12), 186. doi: 10.3390/computers11120186

[11] Roderick, M., MacGlashan, J., & Tellex, S. (2017). Implementing the Deep Q-Network. arXiv, 1711.07478. Retrieved from https://arxiv.org/abs/1711.07478v1

[12] Liu, Y., Dong, M., Ota, K., Li, J., & Wu, J. . Deep Reinforcement Learning based Smart Mitigation of DDoS Flooding in Software-Defined Networks. 2018 IEEE 23rd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD). IEEE. doi: 10.1109/CAMAD.2018.8514971

[13] Hidouri, A., Touati, H., Hadded, M., Hajlaoui, N., & Muhlethaler, P. (2022). A Detection Mechanism for Cache Pollution Attack in Named Data Network Architecture. Advanced Information Networking and Applications. Springer. doi: 10.1007/978-3-030-99584-3_38

[14] Hidouri, A., Touati, H., Hadded, M., Hajlaoui, N., Muhlethaler, P., & Bouzefrane, S. (2023). Q-ICAN: A Q-learning based cache pollution attack mitigation approach for named data networking. Comput. Networks, 235, 109998. doi: 10.1016/j.comnet.2023.109998

[15] Rani, P. V., & Shalinie, S. M. (2020). FuRL: fuzzy RBM learning framework to detect and mitigate network anomalies in Information Centric Network. Sādhanā, 45(1), 1–13. doi: 10.1007/s12046-020-01331-3

[16] Cao, Y., Wu, D., Hu, M., & Chen, S. (2023). Detection and Defense Schemes for Cache Pollution Attack in Content-Centric Network. Emerging Networking Architecture and Technologies. Springer. doi: 10.1007/978-981-19-9697-9_49

[17] Liu, L., & Peng, S. (2022). Detection of A Novel Dual Attack in Named Data Networking. 2022 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom). IEEE. doi: 10.1109/ISPA-BDCloud-SocialCom-SustainCom57177.2022.00008

[18] Kumar, N., & Srivastava, S. (2023). IBPC: An Approach for Mitigation of Cache Pollution Attack in NDN using Interface-Based Popularity. Arab. J. Sci. Eng., 1–11. doi: 10.1007/s13369-023-07919-1

[19] Kumar, N., & Srivastava, S. (2023). IBPC: An Approach for Mitigation of Cache Pollution Attack in NDN using Interface-Based Popularity. Arabian Journal for Science and Engineering. https://doi.org/10.1007/s13369-023-07919-1

[20] Kar, P., Chen, L., Sheng, W., Kwong, C. F., & Chieng, D. (2023, August 22). Advancing Ndn Security: Efficient Identification of Cache Pollution Attacks Through Rank Comparison. doi: 10.2139/ssrn.4548729