Cache Performance in NDN Networks with ADMM Algorithm to Deal with Pollution Attacks and Traffic Load Changes

Atefeh Vaez-Shahrestani[1&2], Mohammad Reza Khayyambashi[3], Faramarz Safi-Esfahani[1&2]
atefehvaez@gmail.com, m.r.khayyambashi@eng.ui.ac.ir, fsafi@iaun.ac.ir
[1] Faculty of Computer Engineering - Najafabad Branch - Islamic Azad University - Najafabad - Iran
[2] Big Data Research Center, Najafabad Branch, Islamic Azad University,Najafabad, Iran
[3] Faculty of Computer Engineering - University of Isfahan - Isfahan – Iran
Receive Date: 03 July 2025 Revise Date: 13 July 2025 Accept Date: 27 July 2025

Abstract

As data-driven networks like Named Data Networks (NDN) continue to grow, securing cache systems has become an increasingly critical issue, especially in defending against cache pollution attacks (CPA). This paper introduces a novel algorithm based on the Alternating Direction Method of Multipliers (ADMM) method for adaptively tuning the β parameter in the PFP-β-DA cache policy. The algorithm works by decomposing the optimization problem into manageable subproblems and updating the parameters in a distributed fashion. This approach enhances the cache CHR and reduces cache pollution (CPR). The algorithm was tested in three network topologies—simple, hierarchical, and advanced—and evaluated under stable conditions, attack scenarios, and dynamic traffic. Simulation results demonstrate that ADMM outperforms traditional methods like LRU, LFU-DA, and PFP-β-DA, providing better stability, adaptability, and precision. Particularly, in high-attack scenarios, the proposed algorithm reduces cache pollution by as much as 5% and increases the cache hit rate by up to 4% over the baseline methods. These results confirm the effectiveness of ADMM as a scalable and intelligent solution for improving cache security and efficiency in NDN networks.

Keywords: Named data, ADMM algorithm, cache pollution, cache hit rate, dynamic networks

1-Introduction

In recent decades, the unprecedented growth of data volume, number of users, and diversity of Internet applications has presented computer networks with new challenges [1-3]. Traditional architectures based on the host-based model and IP addressing used in the classical Internet have faced limitations in responding to new user needs, such as fast and scalable content delivery [4]. In this regard, Information-Centric Networking (ICN), especially NDN, have been introduced as an alternative to traditional architectures [5]. In this model, the focus is shifted from "content location" to "content name", which not only simplifies data access, but also helps to increase network efficiency and scalability [6].

One of the prominent advantages of NDN is the use of in-network caching in intermediate nodes, which reduces data fetch latency, reduces bandwidth consumption, and improves user experience [7]. However, this feature can become a security vulnerability. CPAs are one of the major threats in this area that can affect network performance [8]. In pollution attacks, the attacker tries to fill the cache space and remove popular and frequently used data from the cache by sending repeated requests for low-value or fake data. This leads to a decrease in cache hit rate, increased response latency, and reduced overall network performance [9].

Traditional cache policies such as Least Recently Used (LRU) and Least Frequently Used (LFU) are unable to provide sufficient protection due to their lack of attention to the behavioral patterns of attackers or the origin of requests, especially in the face of complex attacks [7,10]. To counter these threats, newer algorithms such as the PFP algorithm (Popularity per Face) and its more advanced version, PFP-DA (Popularity per Face with Dynamic Aging), have been introduced, which are able to improve the accuracy of content popularity detection and reduce vulnerability to contamination attacks [11]. Also, the PFP-β version, using the parameter β, tries to model the popularity distribution according to the Zipf natural distribution and increase the detection accuracy [12]. However, the main challenges in dealing with cache contamination attacks in NDN include the lack of cache policies that are resistant to complex attacks, the lack of models that are adaptable to changes in demand behavior and traffic dispersion, and the lack of effective combination of cache policies with security threats [13,14]. These issues still require further research to improve cache efficiency and security in NDN networks [15].

In this research, the main objective is to present an optimal algorithm for cache management in NDN networks and to make it robust against contamination attacks. The proposed algorithm uses the PFP-DA model and, by using the optimization method of the Alternating Direction Method of MultipliersADMM [16] adds the tuning parameter β to more accurately model the distribution of content popularity. This method is specifically designed to improve the efficiency of the algorithm in real and complex scenarios and increase the accuracy in distinguishing legitimate content from malicious attacks. In addition, the use of ADMM allows the algorithm to effectively update the cache parameters in a distributed and adaptive manner and improve its performance against contamination attacks. In addition to providing a robust solution against contamination attacks, the proposed method also examines the adaptability of the algorithm to changes in traffic and network needs in different scenarios. The results of this research can pave the way for improving cache management policies in future networks and effectively solve security and performance problems in NDN networks.

The innovation of the proposed method is that by using ADMM, the algorithm is able to update cache parameters effectively in a distributed manner and without the need for central control. In addition to improving the accuracy in detecting legitimate content and malicious attacks, this approach also enables rapid convergence and adaptability of the algorithm to changes in network traffic dynamics. Thus, ADMM allows the algorithm to perform more optimally in different network conditions by reducing computational complexity.

The rest of this article is divided into the following sections. In the second section, the research background is reviewed. In the third section, the proposed method is presented. In the fourth section, the results and analysis are presented. Finally, in the final section, the conclusions are presented.

2-Research Background

In [17], it has been shown that traditional cache policies such as LRU lack the ability to distinguish between valid content and malicious requests due to their focus on the frequency of use. In [18], a method for detecting CPA is presented, which detects abnormal changes by analyzing the latency and state transition matrix. The disadvantages of this method include the analytical complexity and the need to calculate large transition matrices, which can limit operational scalability. In [12], the PFP algorithm is introduced to CPAs and increase the hit rate in ICN networks, which prevents artificial concentration of popularity by analyzing the normalized popularity of content and separating it based on input/output ports. The disadvantages of this method include the need to maintain separate data structures for each data port and real-time processing of changes in popularity distribution, which may cause complexity and resource consumption in resource-constrained nodes. In [19], the BO-CatBoost algorithm was introduced for the simultaneous detection of combined cache contamination (CPA) and Improved Collusive Interest Flooding Attacks (I-CIFA), which uses Bayesian optimization and the CatBoost learning algorithm to achieve accurate detection of multi-attack threats. The disadvantages of this method include the need for high processing power, more storage space, and fine-tuning of parameters, which can limit its widespread applicability.

In [20], a method based on popularity prediction using a gray model was introduced, which is able to identify abnormal deviations in content popularity by analyzing historical trends in request behavior. The disadvantages of this method include the dependence on the accuracy of the prediction model and the need for valid past data, which can create challenges in its implementation. In the study conducted in [21], three resilient cache algorithms, including PFP, PFP-DA with dynamic aging, and PFP-β were introduced and evaluated, which specifically focus on the inherent resilience of cache algorithms against smart and single-input attacks. Simulation results showed that the PFP-DA algorithm performs better than common algorithms such as LRU and LFU-DA under normal conditions and when faced with contamination attacks. In [22], it was shown that attackers can exploit cache behavior to perform indirect attacks such as targeted denial of service (DoS) attacks on data-centric networks, which reduces cache hit rates, increases data fetch latency, and reduces energy efficiency. This research emphasizes the need to design DoS-resistant cache algorithms that, in addition to analyzing content popularity, also pay attention to temporal and behavioral access patterns.

Various studies on CPAs and improving the performance of cache algorithms in data-driven networks have shown that each existing approach has its own advantages and limitations. For example, traditional algorithms such as LRU are unable to distinguish between valid content and malicious requests due to their focus on frequency of use. Also, methods such as latency analysis and state transition matrix for detecting CPA attacks, although accurate, have high analytical complexity

and limit scalability. The PFP algorithm and its improved versions, such as PFP-DA and PFP-β have better performance in detecting threats and increasing the hit rate, but they require maintaining complex data structures and consuming large resources in nodes with limited resources. In this regard, the BO-CatBoost algorithm using machine learning algorithms has been able to provide accurate performance in detecting hybrid threats, but it requires high processing power and more storage space. Also, methods such as popularity prediction with the gray model, despite high detection accuracy, are dependent on valid past data, which can pose challenges. Given the limitations of current algorithms, the use of optimization methods such as the ADMM method can be considered as an innovative solution to improve cache resilience against various attacks. With the ability to update cache parameters in a distributed and adaptive manner, ADMM can reduce the complexities and challenges of previous algorithms and, in addition to improving efficiency, provide better scalability and flexibility in the face of various attacks.

3-Proposed Method

In this section, a novel approach for adaptive tuning of the β parameter in the PFP-β-DA algorithm will be presented. The effect of the β parameter on the cache performance in the NDN is directly applied. Smaller β reduces the pollution percentage and increases the hit rate in the cache, while larger β values, due to the reduced sensitivity of the algorithm, may cause unpopular content to be stored in memory.

The PFP-β-DA cache replacement algorithm is used to investigate its effectiveness in reducing cache pollution. The PFP-β-DA algorithm is a modified version of the PFP-DA algorithm, which reduces the difference in PFP contribution between higher and lower ranked content faces by modifying the overall popularity calculation formula. This modification ensures that even a request for low-ranked content can be cached if the request is received from multiple content faces. Conversely, a request with a high attacker rank may not be cached if that content is requested from only one or a few content faces.

Formula (1) is used to rank the requests for each content face and determine the overall popularity of those requests in this way:

$$\boldsymbol{P(C,\beta)} = \sum_{i=1}^{n} \frac{1}{r_i^{\beta}} \qquad -\infty \leq \boldsymbol{\beta} \leq +\infty \tag{1}$$

This formula calculates the overall popularity of a content request C by considering the requests for each category of content C and using the rank (r_i) of the content C in each Face (i). The parameter (r_i) indicates the rank of the requested content C in the popularity ranking list of Face (i). In formula (1), the smaller the value of β, the smaller the difference between the shares of each Face for each content request. However, using a fixed value for β will be inefficient due to the dynamic and changing network conditions as well as the changing attack patterns. Since NDNs naturally face continuous changes in the number of users, attackers, and traffic patterns, it is necessary to develop an approach that can automatically

and adaptively adjust the value of β based on the current network conditions.

The optimization problem of setting the value of β will be modeled as a parametric optimization problem. In this model, the two main criteria of NDNs, namely hit rate and cache pollution percentage, are considered as functions of β. The value of β will be set in such a way that the hit rate is maximized and the pollution percentage is minimized. The problem model is considered as follows:

$$\min_{\beta}(-HitRate(\beta) + \lambda.\,\text{PollutionRate}(\beta) + \frac{\rho}{2}(\beta - z + u)^2) \quad (2)$$

$$\text{Subjet to } 0< \beta<1$$

In the above equation, β is known as the decision variable and is defined in the interval [0, 1]. Hit Rate(β) will be defined as a function of the cache hit rate. Pollution Rate(β) will be considered as a function of the percentage of cache pollution. λ will be defined as a weighting parameter to control the relative importance of pollution versus hit rate. The weighting parameter determines the importance of reducing cache pollution relative to increasing the cache hit rate. ρ>0 is a penalty parameter in ADMM that controls the speed and stability of the algorithm's convergence. Z is an auxiliary variable in ADMM that divides the problem solution into simpler subproblems.where u is the dual variable (Lagrange coefficient) that ensures coordination between the updates β and z.

The ADMM algorithm uses methods based on iterative optimization to divide a complex optimization problem into simpler subproblems to achieve faster and more accurate convergence. In this method, β is used as the decision variable, and the auxiliary variables z and u are used to maintain coherence and improve the stability of the optimization process. The algorithm includes three main steps in each iteration: updating β, updating z using the projection operator to restrict it to the interval [0, 1], and updating the dual variable u to ensure convergence and coherence between β and z. Finally, this process is repeated continuously until the convergence conditions are met (using the thresholds ε_2 and ε_1 for β and z). In the following, the pseudocode for this method will be explained in detail in Figure 4-1. In the ADMM algorithm, to optimize the parameter β, initial values for β, z, u, ρ, and λ are first determined. Then, the convergence thresholds ε_2 and ε_1 are also considered to check the stopping conditions of the optimization process. In addition, the number of iterations, known as MaxIter, is adjusted so that the optimization process can continue until convergence is achieved.

In the first step: β is updated. This is done in such a way that the optimal value of β is found,which leads to an improvement in the hit rate and a reduction in cache pollution. The goal of this update is to reduce cache pollution while maintaining a high hit rate. To do this, the algorithm uses an objective function that uses a combination of hit rate and cache pollution as optimization criteria and updates β in such a way that both criteria are optimized simultaneously.

In the second step: The auxiliary variable z is updated. This step is done to ensure that the value of β always remains

in the allowed range [0, 1]. In other words, after updating β, any change in the value of β must be such that its value does not go out of the desired range. To do this, a projection operator is used that adjusts the changes in β in such a way that it always remains in the range [0, 1].

In the third step: the dual variable u is updated. This step is performed to maintain the coordination between the updates of β and z. In fact, the dual variable u plays the role of a coordinating factor that ensures that the changes of β and z effectively converge and remain coordinated.

In the fourth step: the convergence conditions are checked. The algorithm checks whether the changes in β and z have become small enough. If the changes in β and z are less than the specified thresholds, the optimization process stops and the algorithm converges. Otherwise, the process continues until the algorithm reaches the convergence point or reaches the maximum number of iterations. Finally, the algorithm automatically adjusts the value of β to maintain the best performance under the changing conditions of the networks. Figure 1 shows the pseudocode of the proposed method. In summary:

- At each iteration, the algorithm divides the problem into simpler subproblems.
- β-update tries to find the best β to improve HitRate and PollutionRate.
- z-update ensures that β remains in the range [0,1].
- u-update increases the coherence between β and z.
- Finally, β is adaptively adjusted to maintain the best performance under changing network conditions.

Initialize:

- Set initial values for β, z, u, ρ, λ
- Set convergence thresholds $\boldsymbol{\varepsilon_1}$, $\boldsymbol{\varepsilon_2}$
- Set maximum number of iterations MaxIter
- Set initial values for HitRate(β) and PollutionRate(β)

For k = 0 to MaxIter:

1. Update β using the minimization formula:

$$\boldsymbol{\beta}^{(k+1)} \leftarrow \arg\min_{\boldsymbol{\beta}} \left(-\boldsymbol{HitRate}(\boldsymbol{\beta}) + \boldsymbol{\lambda}.\mathbf{PollutionRate}(\boldsymbol{\beta}) + \frac{\boldsymbol{\rho}}{\mathbf{2}} \left(\boldsymbol{\beta} - \boldsymbol{z}^{(k)} + \boldsymbol{u}^{(k)} \right)^{\mathbf{2}} \right)$$

$$\textbf{Subjet to } 0 < \boldsymbol{\beta} < 1$$

2. Update z using the projection operator:

$$\mathbf{z}^{(k+1)} \leftarrow \mathbf{Projection}_{[0\ 1]}(\boldsymbol{\beta}^{(k+1)} + \boldsymbol{u}^{(k)})$$

3. Update dual variable u:

$$\mathbf{u}^{(k+1)} \leftarrow \mathbf{u}^{(k+1)} + (\boldsymbol{\beta}^{(k+1)} + \boldsymbol{z}^{(k+1)})$$

4. Check for convergence:

If

$$\left|\boldsymbol{\beta}^{(k+1)} - \boldsymbol{\beta}^{(k)}\right| < \boldsymbol{\varepsilon_1}\ \boldsymbol{and}\ \left|\mathbf{z}^{(k+1)} - \boldsymbol{z}^{(k)}\right| < \boldsymbol{\varepsilon_2}$$

Break (converged)

End for

Return $\boldsymbol{\beta}^{(k+1)}$, $\mathbf{z}^{(k+1)}$,$\boldsymbol{u}^{(k+1)}$

Fig.1. ADMM algorithm in cache improvement

4- Results

In this section, the performance of the proposed algorithm is investigated in different conditions of NDN and in the face of security threats such as CPAs. The main goal of this evaluation is to measure the efficiency of the proposed method in improving the cache hit rate and reducing cache pollution compared to existing and standard methods. For the simulation and analysis of this research, a computer with an Intel Core i7 processor and 8 GB of RAM in a Windows 10 environment was used. The proposed algorithm was first implemented in MATLAB 2020Rb software for numerical analysis and optimization of the β parameter, and then the NS-3-based ndnSIM simulator was used to practically evaluate its performance in a network environment. The combination of these two tools has enabled a detailed mathematical investigation and realistic evaluation of the algorithm in the NDN architecture.

In this research, two main criteria have been used to evaluate the performance of the proposed method, which play a key role in the analysis and comparison of algorithms. The first metric, Cache CHR, represents the percentage of requests answered directly from the cache and is a fundamental metric for measuring the efficiency of a cache system, as increasing this rate leads to reduced latency in content delivery and improved user experience.

$$\mathrm{CHR}\% = \frac{\text{Number of cache hits}}{\text{Total number of content requests}} \times 100 \quad (3)$$

In this regard, the Number of cache hits is the number of requests that have been answered from the cache (the number of cache hits), and the Total number of content requests is the total number of user content requests (regardless of whether they have been answered from the cache or not). The second metric, Cache Pollution Percentage (CPR), is an important indicator in identifying and mitigating useless content and CPAs and indicates the algorithm's ability to maintain the quality of stored data.

$$\mathrm{CPR}\% = \frac{\text{Number of polluted contents in the cache}}{\text{TTotal number of cached contents}} \times 100 \quad (4)$$

In this regard, the Number of polluted contents in the cache is the number of polluted or unpopular data in the cache and the Total number of cached contents is the total number of data stored in the cache.

4-1-Topology

In this study, in order to evaluate the performance of the proposed algorithm based on ADMM, three different network topologies have been designed and implemented, each of which includes three key scenarios. The simple topology (Figure 2a) is a three-layer structure with producer nodes, consumers, and intermediate routers with cache, where the proposed algorithm is executed independently in each node. This topology is designed for basic analysis of the algorithm under different conditions, including a stable network, CPA, and traffic fluctuations. The hierarchical topology (Figure 2b) also has a similar structure but focuses more on the interaction between layers, simulating more realistic conditions of the algorithm's

performance and allowing for a more accurate analysis of the cache behavior in the network. Finally, the advanced topology (Figure 3c) as the most complete structure, with precise layering and simulation of complex environments, uses adaptive and dynamic optimization for the key parameters β, ρ, λ. In all three topologies, three fixed scenarios are defined: (1) a stable network without attacks, to evaluate the cache CHR and cache pollution percentage (CPR) under normal conditions; (2) a CPA attack scenario, to measure the algorithm's resilience against malicious requests; and (3) a traffic load change scenario, to evaluate the algorithm's adaptability in dynamic and changing environments

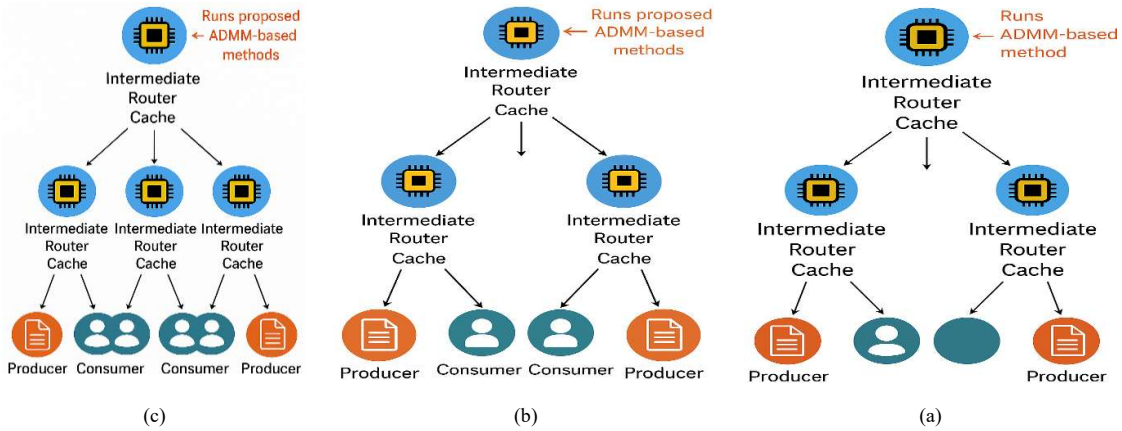

Fig.2. Scenarios considered: a) Simple, b) Hierarchical, c) Advanced

Table 1. Comparison of network topologies and scenarios for evaluating the proposed algorithm

Scenario Objective	Evaluation Features	Scenario	Topology
Investigate the algorithm's performance under normal conditions	Evaluate CHR and CPR without the presence of a threat	Stable Network	Simple Hierarchical
Evaluate the algorithm's resistance to malicious requests	Maintain cache hit rate and reduce cache contamination percentage	CPA Attack	
Evaluate the algorithm's adaptability under dynamic conditions	Adaptive tuning of β, ρ, and λ parameters	Traffic Load Shifting	
Realistic analysis of the algorithm's performance	Interaction between layers, CHR and CPR review	Stable Network	Topology Simple Hierarchical
Evaluate the algorithm's performance against attacks in the middle layer	Intelligent local defense, reducing the impact of fake content	CPA Attack	
Evaluate the algorithm's response to network load fluctuations	Maintain cache stability in changing conditions	Traffic Load Shifting	
Detailed investigation of the algorithm in a complex environment	Performance with adaptive tuning in multiple layers, minimizing CPR	Stable Network	Advanced
Evaluate the algorithm's security and stability against attacks	Dealing with fake data in multi-port and aggressive conditions	CPA Attack	
Evaluate flexibility in real time	Rapid algorithm adaptation to instantaneous changes in load and network behavior	Traffic Load Shifting	

This comprehensive framework provides a suitable platform for analyzing the algorithm's performance and stability under various conditions of data-driven networks. Table 2 reviews the characteristics of these topologies and scenarios.

4-2-Hierarchical Topology Evaluation

To evaluate the performance of the proposed ADMM-based algorithm in the hierarchical topology of NDN, three key scenarios are designed. These scenarios are designed to accurately simulate the behavior of the algorithm in the face of different network conditions, including changing cache capacity, changing attack intensity, and dynamic changes in network traffic load. In the first scenario, the cache memory capacity of the intermediate nodes is variably adjusted to three values of 500, 1000, and 2000 contents to investigate the effect of cache size on cache CHR and cache contamination percentage (CPR). The second scenario examines the resilience of the algorithm against different intensities of cache contamination attacks. In this scenario, the rate of sending malicious requests is increased from 5 to 20 requests per second, and the adaptive performance of the algorithm in controlling contamination and maintaining quality of service is analyzed. Finally, the third scenario is designed to evaluate the adaptability of the algorithm in the conditions of varying network traffic load. In this scenario, the request rate of legitimate users gradually changes from 5 to 20 requests per second to measure the algorithm’s ability to maintain optimal performance in the face of dynamic changes. Table 2 shows the specifications of these scenarios.

Table 2. Simulated scenarios in hierarchical topology

Method	Fixed (Key) Parameters	Objective	Scenario
Cache capacity = 500, 1000, 2000 contents	User Request Rate: 10	Investigate the sensitivity of the algorithm to cache memory capacity	Cache capacity change
Malicious request rate: 5, 10, 20 requests/sec	Attack Scenario: 20%	Evaluate resilience against different attack intensities	Attack severity change
Legitimate request rate: 5 to 20 gradually	Cache Capacity: 1000	Measure adaptability in dynamic conditions	Traffic load change

In a comparative analysis of the performance of the algorithms in three main scenarios, the results show that the ADMM-based algorithms, especially their improved version, have outperformed conventional methods such as LRU and LFU-DA in all conditions. In the first scenario (stable network) shown in Table (3), the ADMM algorithm has succeeded in increasing the cache CHR and cache rate for the Top 50 and Top 100 popular data, while the cache contamination percentage (CPR) has remained at a lower level than other methods. In the second scenario (cache contamination attack - Table 4), despite the frequent sending of

fake data by malicious nodes, the proposed algorithms have been able to significantly reduce cache contamination and maintain the CHR at a desirable level, unlike LRU and LFU-DA, which were clearly vulnerable to the attack. Finally, in the third scenario (traffic load changes – Table 5), the ADMM algorithm and its advanced version showed high adaptability and, by automatically adjusting the parameters, were able to maintain cache performance and control pollution even in the face of load fluctuations. These results indicate that the use of ADMM as an adaptive optimization method provides an effective, stable, and secure solution for cache management in NDN architecture.

Table 3. Scenario 1: Stable network (no attack)

Algorithm	CHR	CPR	Hit rateTop 100	Hit rate Top 50
PFP-DA	21.00%	10.00%	68.00%	78.00%
LFU-DA	15.00%	20.00%	55.00%	65.00%
LRU	12.00%	25.00%	48.00%	58.00%
ADMM	23.00%	8.00%	70.00%	80.00%

Table 4. Scenario 2: Network under Contaminant Control Attack (CPA)

Algorithm	CHR	CPR	Hit rateTop 100	Hit rateTop 50
PFP-DA	19.02%	15.00%	62.80%	75.53%
LFU-DA	12.67%	36.00%	41.17%	50.00%
LRU	8.82%	36.00%	28.72%	34.64%
ADMM	21.00%	12.00%	65.00%	77.00%

Table 5. Scenario 3: Traffic load changes (dynamic)

Algorithm	CHR	CPR	Hit rateTop 100	Hit rateTop 50
PFP-DA	18.00%	14.00%	60.00%	70.00%
LFU-DA	11.00%	28.00%	45.00%	52.00%
LRU	9.00%	30.00%	40.00%	48.00%
ADMM	20.00%	11.00%	63.00%	75.00%

4-3-Evaluation Results in Simple Topology Scenarios

In simple topology, the performance of the algorithms has been evaluated in three key scenarios, the results of which are reflected in Tables (6) to (8). In the first scenario (stable network), the proposed ADMM-based algorithm has recorded a higher cache CHR and a lower cache pollution percentage (CPR) than classical methods such as LRU, LFU-DA, and even PFP-DA. This superiority is due to the adaptive parameter tuning and intelligent cache management. In the second scenario CPA, its ADMM algorithm has shown very high resistance to malicious data and has succeeded in significantly reducing cache pollution compared to other methods, while classical methods have experienced a sharp drop in the hit rate and an increase in pollution. In the third

scenario (dynamic traffic), the proposed algorithms were able to maintain their adaptability and provide stable and acceptable performance in changing network conditions by automatically adjusting the parameters. While traditional algorithms suffer from performance degradation when faced with load fluctuations, this analytical trio well confirms the high performance of the ADMM algorithm in various network conditions and proposes it as a powerful and stable solution for cache management in NDN networks.

Table 6.Scenario 1 – Stable network (no attack)

Algorithm	CHR	CPR	Hit rateTop 100	Hit rateTop 50
PFP-DA	20.00%	11.00%	67.00%	77.00%
LFU-DA	13.00%	21.00%	53.00%	63.00%
LRU	10.00%	27.00%	47.00%	57.00%
ADMM	22.00%	9.00%	69.00%	79.00%
	24.00%	8.00%	71.00%	81.00%

Table 7. Scenario 2 – Contaminant Control Attack (CPA)

Algorithm	CHR	CPR	Hit rateTop 100	Hit rateTop 50
PFP-DA	18.50%	16.00%	61.00%	73.00%
LFU-DA	11.50%	35.00%	40.00%	49.00%
LRU	7.50%	37.00%	27.00%	33.00%
ADMM	20.00%	13.00%	64.00%	75.00%
	22.00%	11.00%	67.00%	78.00%

Table 8. Scenario 3 – Traffic Load Change (Dynamic)

Algorithm	CHR	CPR	Hit rateTop 100	Hit rateTop 50
PFP-DA	17.00%	15.00%	58.00%	68.00%
LFU-DA	10.00%	29.00%	44.00%	50.00%
LRU	8.00%	31.00%	39.00%	46.00%
ADMM	19.00%	12.00%	62.00%	73.00%
	21.00%	10.00%	65.00%	76.00%

4-4-Evaluation Results for Advanced Topology

The results of the three-scenario analysis in the advanced topology (Tables 9 to 11) indicate the superiority of the proposed algorithm based on ADMM in managing the cache memory of NDN networks. In stable conditions, these algorithms showed better performance than LRU, LFU-DA, and PFP-DA by maintaining high CHR and low CPR, especially for popular data. In the face of CPA attacks, the ability of ADMM to adaptively adjust parameters significantly reduced cache pollution, while conventional algorithms had a severe performance drop. Also, in dynamic scenarios with fluctuating traffic load, the proposed algorithm maintained stable performance with automatic adaptation and

did not suffer from a decrease in the hit rate. These findings prove that the use of ADMM with distributed optimization is an effective solution for improving cache security and efficiency in data-centric networks.

Table 9. Advanced Topology Scenario 1: Stable Network (No Attack)

Algorithm	CHR	CPR	Hit rateTop 100	Hit rate Top 50
PFP-DA	25.00%	8.00%	70.00%	80.00%
LFU-DA	18.00%	18.00%	60.00%	70.00%
LRU	15.00%	22.00%	52.00%	62.00%
ADMM	27.00%	6.00%	72.00%	82.00%
	29.00%	5.00%	75.00%	85.00%

Table 10. Advanced Topology Scenario 2: Network under CPA

Algorithm	CHR	CPR	Hit rateTop 100	Hit rateTop 50
PFP-DA	23.00%	12.00%	65.00%	77.00%
LFU-DA	16.00%	30.00%	50.00%	60.00%
LRU	11.00%	32.00%	40.00%	48.00%
ADMM	25.00%	10.00%	68.00%	80.00%
	27.00%	8.00%	70.00%	82.00%

Table 11. Advanced Topology Scenario 3: Traffic Load Changes (Dynamic)

Algorithm	CHR	CPR	Hit rate Top 100	Hit rate Top 50
PFP-DA	21.00%	10.00%	63.00%	74.00%
LFU-DA	14.00%	24.00%	47.00%	56.00%
LRU	12.00%	26.00%	42.00%	52.00%
ADMM	24.00%	8.00%	66.00%	78.00%
	26.00%	7.00%	68.00%	80.00%

4-5-Comparison with the base paper

In two scenarios reviewed in [12], the performance of the PFP-β-DA algorithm under steady-state and under attack conditions is evaluated. In the first scenario (no attack), it was found that decreasing β (up to 0.05) increased the cache CHR and reduced the cache complete contamination rate (CPR). In the second scenario (attack with a load of 5% to 30%), PFP-β-DA was robust against a small number of attackers, but its performance decreased as the attack intensity increased. In contrast, the proposed ADMM-based algorithm, with adaptive tuning of the parameters β, ρ, and λ, not only provides a higher CHR under steady-state conditions, but also has higher resilience against severe attacks. This adaptability is the key advantage of the proposed method over PFP-β-DA.

In the comparison between the proposed algorithm based on ADMM and the PFP-β-DA methods, the results of the three scenarios can be summarized as follows in Tables 12 to 14. Table 12 examines (stable network). In this scenario, PFP-β-DA with β=0.05 reduced the contamination to zero, but ADMM

with adaptive tuning and its advanced version with optimization of ρ and λ provided higher hit rates (CHR) and closer to ideal results for Top 50 and Top 100 content. Table 13 examines (attack with 5% load). In light attacks, ADMM and its advanced version not only controlled the contamination like PFP-β-DA, but also recorded higher hit rates thanks to the adaptive learning mechanism. Table 14 (30% load attack): In heavy attacks, the performance of PFP-β-DA dropped, but the ADMM with adaptive architecture was able to provide 5% less contamination and up to 4% higher hit rate. The advanced version of ADMM recorded the best performance with high CHR rate (for Top 50 up to 73%) and low contamination (18%). These three tables clearly show that the adaptive architecture of ADMM provides better stability, flexibility, and accuracy in NDN cache management than static methods, both in normal conditions and in threat scenarios.

Table 12. Comparison of algorithm performance in scenario 1 (without attack)

Algorithm	CHR	CPR	Hit rate Top 100	Hit rateTop 50
PFP-DA	21.3%	10.3%	68.2%	75.5%
PFP-β-DA (β=0.05)	25.0%	0.0%	93.1%	98.8%
ADMM	28.0%	0.0%	95.0%	99.0%
	30.0%	0.0%	96.5%	99.5%

Table 13. Comparison of algorithm performance in scenario 2 (attack with a diverse number of attackers) with 5% attacker load

Algorithm	CHR	CPR	Hit rateTop 100	Hit rateTop 50
PFP-DA	19.0%	10.0%	65.0%	75.0%
PFP-β-DA (β=0.05)	23.0%	0.0%	70.0%	80.0%
ADMM	26.0%	0.0%	74.0%	84.0%
	28.0%	0.0%	77.0%	87.0%

Table 14. Comparison of algorithm performance in scenario 2 (attack with a diverse number of attackers) with 30% attacker load

Algorithm	CHR	CPR	Hit rate Top 100	Hit rate Top 50
PFP-DA	10.0%	30.0%	50.0%	60.0%
PFP-β-DA (β=0.05)	12.0%	25.0%	55.0%	65.0%
ADMM	16.0%	20.0%	60.0%	70.0%
	18.0%	18.0%	63.0%	73.0%

5-Conclusion

In this study, in order to effectively deal with CPAs in NDN, an adaptive algorithm based on ADMM was presented to optimally adjust the β parameter in the PFP-β-DA cache policy. Simulation results in different topologies and scenarios (stable network, 5% and 30% load attack, and dynamic traffic) showed that the proposed algorithm is able to significantly

increase the cache CHR and at the same time minimize the CPR. Compared with basic methods such as LRU, LFU-DA, and even static versions of PFP-DA and PFP-β-DA, the ADMM-based algorithm was able to demonstrate more stable performance against heavy attacks and dynamic network changes. It also indicates the high ability of this method to adapt to real network conditions. Overall, this research proved that using the ADMM distributed and adaptive optimization structure not only improves cache performance under normal conditions, but also provides a robust, scalable, and practical solution to deal with complex security threats in the NDN architecture.

References

[1] R. Alubady, M. Salman, and A. S. Mohamed, "A review of modern caching strategies in named data network: Overview, classification, and research directions," Telecommunication Systems, vol. 84, no. 4, pp. 581-626, 2023.

[2] H. Khelifi, S. Luo, B. Nour ,H. Moungla, Y. Faheem, R. Hussain, and A. Ksentini, "Named data networking in vehicular ad hoc networks: State-of-the-art and challenges," IEEE Communications Surveys & Tutorials, vol. 22, no. 1, pp. 320-351, 2019.

[3] N. Sadat, and R. Dai, "A Survey of Quality-of-Service and Quality-of-Experience Provisioning in Information-Centric Networks," Network, vol. 5, no. 2, pp. 10, 2025.

[4] S. Yassine, N. Najib, and J. Abdellah, "Routing approaches in named data network: a survey and emerging research challenges," International Journal of Computers and Applications, vol. 46, no. 1, pp. 32-45, 2024.

[5] X. Zhang, Y. Zhou, D. Wu, Q. Z. Sheng, S. Riaz, M. Hu, and L. Xiao, "A Survey on Privacy-Preserving Caching at Network Edge: Classification, Solutions, and Challenges," ACM Computing Surveys, vol. 57, no. 5, pp. 1-38, 2025.

[6] L. V. Yovita, and N. R. Syambas, "Caching on Named Data Network: a Survey and Future Research," International Journal of Electrical & Computer Engineering (2088-8708), vol. 8, no. 6, 2018.

[7] N. U. Saqib, and S. Isnain, "A Survey on Mitigation of Cache Pollution Attacks in NDN," Acta Technica Jaurinensis, 2025.

[8] P. Kar, L. Chen, W. Sheng, C. F. Kwong, and D. Chieng, "Advancing NDN security: Efficient identification of cache pollution attacks through rank comparison," Internet of Things, vol. 26, pp. 101142, 2024.

[9] P. Chaudhary, and N. Hubballi, "PeNCache: Popularity based cooperative caching in Named Data Networks," Computer Networks, vol. 257, pp. 110995, 2025.

[10] A. Karami, andM. Guerrero-Zapata, "An anfis-based cache replacement method for mitigating cache pollution attacks in named data networking," Computer Networks, vol. 80, pp. 51-65, 2015.

[11] A. Hidouri, N. Hajlaoui, H. Touati, M. Hadded, and P. Muhlethaler, "A survey on security attacks and intrusion detection mechanisms in named data networking," Computers, vol. 11, no. 12, pp. 186, 2022.

[12] J. Baugh, and J. Guo, "Enhancing Cache Robustness in Information-Centric Networks: Per-Face Popularity Approaches," Network, vol. 3, no. 4, pp. 502-521, 2023.

[13] N. Kumar, and S. Srivastava, "IBPC: An Approach for Mitigation of Cache Pollution Attack in NDN using Interface-Based Popularity," Arabian Journal for Science and Engineering, vol. 49, no. 3, pp. 3241-3251, 2024.

[14] T. Lauinger, N. Laoutaris, P. Rodriguez, T. Strufe, E. Biersack, and E. Kirda, "Privacy risks in named data networking: What is the cost of performance?," ACM SIGCOMM Computer Communication Review, vol. 42, no. 5, pp. 54-57, 2012.

[15] Z. N. S. ALMUDAYNI" ,Improving Energy Efficiency and Network Lifetime in IoT Systems: A Novel Theoretical Framework and Experimental Validation," La Trobe, 2025.

[16] D. Han, and X. Yuan, "A note on the alternating direction method of multipliers," Journal of

OptimizationTheory and Applications, vol. 155, pp. 227-238, 2012.

[17] S. G. Mandapati, C. Ranaweera, and R. Doss, "Early Detection and Mitigation of Cache-Based Attacks in IoT-NDN," Deakin University, 2025.

[18] H. Wang, D. Man, S. Han, H. Wang, and W. Yang, "Detection and Defense of Cache Pollution Attack Using State Transfer Matrix in Named Data Networks." pp. 545-556.

[19] L. Liu, and S. Peng, "Detection of A Novel Dual Attack in Named Data Networking." pp. 1-8.

[20] L. Yao, Y. Zeng, X. Wang, A. Chen, and G. Wu“ ,Detection and defense of cache pollution based on popularity prediction in named data networking,” IEEE Transactions on Dependable and Secure Computing, vol. 18, no. 6, pp. 2848-2860, 2020.

[21] J. P. Baugh, “Enhancing Cache Robustness in Named Data Networks,” 2018.

[22] J. B. Gouge, “A targeted denial of service attack on data caching networks,” 2015.