GCPiN: Group Caching for Privacy in Named Data Networking

Amita Ajith Kamath, Chirag Jamadagni, Abhijith Anilkumar, Kevin T. Mathew, Mohit P. Tahiliani
Wireless Information Networking Group
National Institute of Technology Karnataka, Surathkal
Mangalore, India
Email: amita.a.kamath@ieee.org, chirag.jamadagni@gmail.com, abhijithanilkumar@live.com,
kevin.thomasmathew@gmail.com, tahiliani@nitk.edu.in

***Abstract*—Router architecture in Named Data Networks (NDN) is intricate, and entails in-network caching of Data packets. Protecting the privacy of this cached content, while maintaining the performance benefits obtained due to caching, is a major concern. Although this problem has been addressed, most existing solutions compromise the performance gain that NDN provides in order to ensure data privacy. In this paper, we formulate a new approach to enhance the privacy of cached content at each NDN router, while ensuring minimal performance loss. This is achieved by segregating NDN routers into groups, and maintaining a Distributed Content Store across each group. The proposed approach has been named Group Caching for Privacy in NDN (GCPiN). We provide a mathematical analysis to demonstrate that GCPiN is a promising approach, and successfully balances the tradeoff between privacy and performance.**

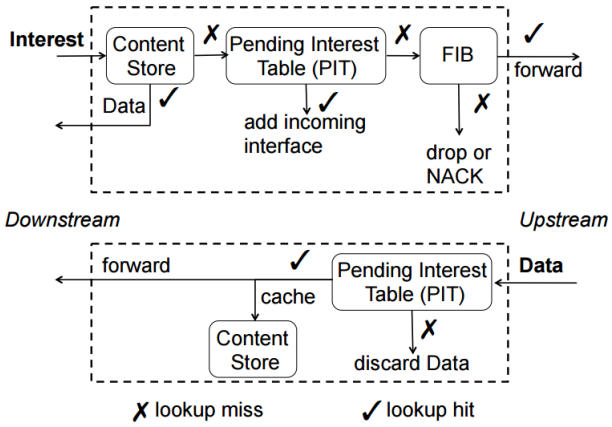
I. Introduction

Named Data Networking (NDN) [1] is a new internet architecture designed to overcome the limitations of the current IP architecture. While the IP architecture is communication-oriented, the Internet has evolved into a significantly more content distribution-oriented system. This has led to excessive data redundancy, which limits the scalability and hence suitability of the current IP architecture for future applications. The NDN architecture removes the concept of IP addresses, and instead gives each packet a *name*. This results in a much more generalized structure, keeping the emphasis on the *data* itself rather than the *source* and *destination* of that data in the network.

NDN has two types of packets: an Interest packet and a Data packet. The consumer requests a Data packet by sending an Interest packet to the producer, and the producer replies by sending a Data packet when it receives an Interest packet from the consumer. However, connections in NDN may not always be end-to-end because NDN provides in-network caching of Data packets. Thus, all Interest packets may not be forwarded to the producers if the requested content is already cached in the network. This in-network caching has several benefits, including improved latency, reduced network traffic, and lightens the load on producers to respond to a large number of requests from consumers.

However, these advantages come at the cost of an increase in complexity of NDN router architecture. Fig. 1 presents the general architecture of a NDN router.

Fig. 1. Routing in NDN [1]



Whenever an Interest arrives, the router checks whether the requested Data has been cached in its local memory, called its Content Store. If so, the Interest is not propagated further, and the Data packet with requested content is sent back to the consumer. If not, the router checks whether a matching Interest was received earlier, by consulting its Pending Interest Table (PIT). PIT is a list of Interest packets that have been forwarded by router to the respective producer but have not yet been satisfied. If an incoming Interest matches an existing entry in PIT, it is added to PIT. Otherwise, the router consults its Forwarding Information Base (FIB) and forwards it towards the producer. FIB is similar to a routing table in current IP routers. Although this approach significantly reduces the response time, ensuring privacy of Data cached in the Content Store is a challenging task.

There are two aspects involved in ensuring privacy of cached content: hiding the identity of consumers who request content, and hiding the actual content itself. While NDN inherently takes care of the former by excluding consumer identity from Data packets, the fact that Data packets are cached opens the NDN architecture to vulnerabilities in the form of *Cache Snooping* [2]. Cache snooping is a method by which a malicious entity can determine the contents of a cache (here, a router's Content Store) by requesting a sequence of data packets. By measuring the response time taken for each

request, the malicious entity can establish whether each request resulted in a cache hit or a cache miss, thus cumulating in an understanding of the contents of the router's Content Store [3]. Furthermore, once this understanding is gained, mapping specific content to the corresponding consumer is possible, particularly when the router services only a small number of consumers [4]. Existing solutions to address this problem include: disable caching of data at the router, artificially increasing the response time to prevent cache snooping [3][4], etc. Although these solutions enhance privacy, the associated performance loss does not allow them to be of practical use.

This paper makes two contributions: firstly, we propose Group Caching for Privacy in NDN (GCPiN), an effective solution to the tradeoff between privacy and performance. Secondly, we provide a mathematical proof to demonstrate the effectiveness of GCPiN. GCPiN involves the geographical segregation of routers into small groups. Using distributed cache policies, a *Distributed Content Store* is maintained across every group of routers. When an Interest packet arrives and the requested data is cached in the Content Store, the router artificially manipulates the response time so that a malicious entity is unable to determine which of the routers belonging to that group had the data cached in it. Although this approach does affect performance, it does so significantly lesser than existing solutions. Additionally, the increase in privacy is considerable: a malicious entity would find more difficult to map a particular content to its corresponding consumer, as the size of cached content is multiplied by the number of routers in the group.

The remainder of the paper is organized as follows: Section 2 discusses existing work in the field of privacy in NDN. Section 3 presents GCPiN and details its functionality. Section 4 provides a mathematical proof to demonstrate the effectiveness of GCPiN, and Section 5 concludes the paper with directions for future work.

II. Related Work

This section discusses related work in the fields of Privacy and Distributed Caching in NDN.

A. Privacy

All protocols that involve a form of caching, such as DNS, have problems with respect to privacy [2]. Privacy has also been a topic of study in all types of Information-Centric Networks (ICN), including NDN. Lutz [5] discusses the advantages and disadvantages of ICN in terms of its effect on privacy and security. New issues such as cache, content, name and signature privacy were brought up. Acs et al [3] discuss violation of cache privacy in NDN specifically, proposing several solutions. However, the countermeasures proposed in these works all reduce performance to a great extent. Lauinger et al [4] study privacy risks in NDN, discussing several *elementary* and *selective* countermeasures, and their drawbacks.

Elementary countermeasures artificially increase the response time such that Interests causing cache hits and those causing cache misses take the same amount of time to be responded to [4]. However, this causes a significant drop in performance, and defeats the purpose of using a cache, except for improved efficiency of the network due to fewer requests being forwarded.

Selective countermeasures depend on users marking their data as "privacy-sensitive". Protocols for selective caching (ensuring privacy-sensitive data is never cached) or selective tunneling (ensuring privacy-sensitive data is always tunneled) could be used, assuming the consumer is willing to experience a lag in performance to ensure privacy is maintained. However, not all consumers are aware of the sensitivity of their requests, and marking all data privacy-sensitive would lead to elimination of the performance benefits of NDN. Hence, a generic solution is preferred, if it can lead to a better balance in the tradeoff between privacy and performance.

B. Distributed Caching

Li et al [6] discuss coordinated caching in NDN based on popularity of the content. Hu et al [7] find a solution to distributed in-network cooperative caching by considering it a constrained optimization problem. Sourlas et al [8] propose a method of distributed caching involving "distributed managers" in each router that collectively make decisions regarding placement and removal of data. However, these caching mechanisms do not take privacy issues into consideration. In this paper, we use distributed caching as a solution to the issue of privacy in NDN.

III. Group Caching for Privacy in NDN (GCPiN)

A. Overview

The concept of distributed caching in ICN has been proposed in [8], in which required algorithms have been derived and discussed. GCPiN builds upon this work to customize it to NDN, specifically to provide privacy. GCPiN has three major functions: segregating routers into groups, distributing cache content across the routers in a group, and manipulating response time to provide privacy to consumers.

Each router has a *GCPiN manager*, which handles the aforementioned functions of GCPiN. Below are the responsibilities of the GCPiN manager:

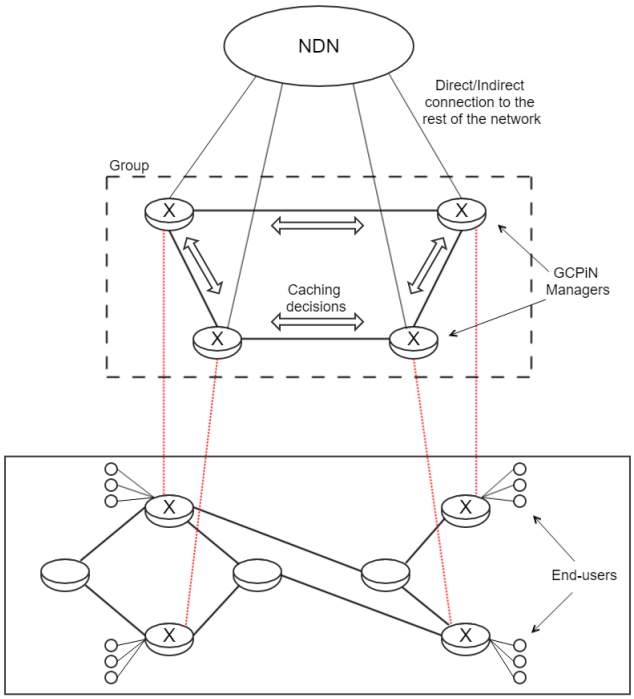
- To determine which data should or should not be cached in its router. This decision is made considering popularity of the content in the network, local to that router.
- To respond to "join" requests from other routers based on a standard quantitative scale of its group's need for an additional router. It would do this by monitoring the network for number of Interest packets, cache hits and misses, etc.
- To handle manipulation of response time for requests to that router, such that malicious entities are unable to determine which router belonging to the group had the data cached in it.

The three functions of GCPiN are discussed in the following subsections.

B. Grouping Routers

GCPiN geographically groups the routers that are closest to the end-users; if there is any hierarchy in the network, these would be the edge routers.Collectively, the caches of the routers of the group form the *Distributed Content Store*. Fig. 2 shows an example network topology, with the routers closest to the end-users marked with an X. These routers are grouped, and each router has a GCPiN manager, as shown in an overlay. GCPiN managers communicate with each other to collectively make caching decisions.

Fig. 2. Structure of GCPiN



When a router joins the network, it contacts all GCPiN managers nearest to it geographically. These GCPiN managers respond with a quantitative answer regarding the degree to which they need another router for their respective groups. The router joins the group that needs it the most, i.e., is facing the most amount of traffic with the fewest amount of resources in terms of the size of Distributed Content Store. If no GCPiN managers are geographically near a router, it forms its own group. Other routers may then join it at a later time.

When a router desires to leave a group, it may simply drop out. This may also be unintentional, for example when a router goes down. The content it had stored in its cache will then not be available to the other routers of the group, and if requested a cache miss will occur to the Distributed Content Store. This content will then be retrieved from the network in the usual way, by forwarding the Interest towards the producer until the information is found and returned.

C. Distributing Cache Content

In GCPiN, the concept of a Distributed Content Store is equivalent to a Content Store distributed across a group of routers, in a manner similar to a distributed caching management strategy proposed by [8] for ICN. This strategy introduces a "distributed manager" in each router, which communicates with other distributed managers such that the routers are collectively and dynamically able to assign data items to the caches. Decisions regarding whether to keep or remove a Data packet in a cache are made by comparing the overall network performance loss caused by removing the packet with the overall network performance gain caused by keeping the packet.

In our approach, we extend the concept of distributed managers into GCPiN managers. The additional aspects we have added in GCPiN are: the consideration of the popularity of the content at that router, and the addition of privacy measures. The strategy proposed by [8] showed a remarkable improvement in performance compared to non-distributed caching mechanisms, at the cost of an increase in the number of messages in the network and in the computation required. GCPiN would have the same improvements and costs associated.

The caching strategy followed in GCPiN ensures that the optimal placement of data takes place across the routers of the group, with popular data in a subnetwork cached close to that subnetwork. Although this would not improve performance from the consumer's perspective due to the artificial increase of response time, the network traffic would be kept to a minimum if popularity-based division is followed.

The GCPiN managers also attempt to maintain distinctness while distributing cache content, in order to maximize the collective content of the Distributed Content Store. Due to this distinctness, the size of the cache in GCPiN is effectively multiplied by the number of routers in the group. Hence, the hit ratio at the Distributed Content Store increases with an increase in cache size in accordance with an exponential function of form:

$$y = A - Be^{-Cx}$$

where y is the hit ratio, x is the cache size, and a, b and c are constants. This relation, presented in [9] and [10], concurs with our findings shown in Fig. 3.

However, the increased hit ratio comes with a performance cost. The time taken to search for data from a larger Content Store increases as *O(log N)*. Additionally, the time taken to retrieve the data from the group is longer than the time taken to return the data from the router itself, due to network latency within the group. These performance drops have been considered when comparing the performance of our proposal with the elementary approach of artificially making cache hit times equal cache miss times.

D. Providing Privacy

GCPiN increases privacy by having the GCPiN manager artificially increase response times. The concept of manipulat-

ing response time has been introduced in [3] wherein cache hit times are increased to equal cache miss times to yield a common response time, making it difficult to determine the cache contents by snooping. We term this artificial increase as "padding" the cache hit time "to" the cache miss time.

The GCPiN manager, however, does not pad a cache hit at the router to a cache miss; rather, it pads it to a cache hit at the groups Distributed Content Store. In the rare event that a router is alone in a group, which would occur if it is the first router in the group, or if all other routers of the group dropped out, the cache hit time is padded to a cache miss time.

In Section 4, we show that GCPiN performs better than the approach suggested in [3]. Furthermore, GCPiN integrates well with current approaches to improve cache performance in the Content Store [11], as well as approaches to increase privacy in cache-based network protocols [12]. For example, measures to prevent cache snooping and poisoning could be run in parallel across all the routers belonging to that group.

IV. Proof of Concept

This section provides a mathematical proof of concept for GCPiN. The variables considered in this proof and their meanings are depicted in Table 1. In the following subsections we derive relationships between H_r and H_g, and T_r and T_g, then use these relations to prove the validity of GCPiN.

TABLE I
VARIABLES CONSIDERED AND THEIR MEANING

VARIABLE	MEANING
n	Number of routers in the group
H_r	Hit ratio at a router
H_g	Hit ratio at the group
T_r	Average time taken to search a router's Content Store
T_g	Average time taken to search the group's Distributed Content Store
T_d	Average RTT between two neighboring routers
R_e	Average response time of existing solution
R_p	Average response time of proposed solution

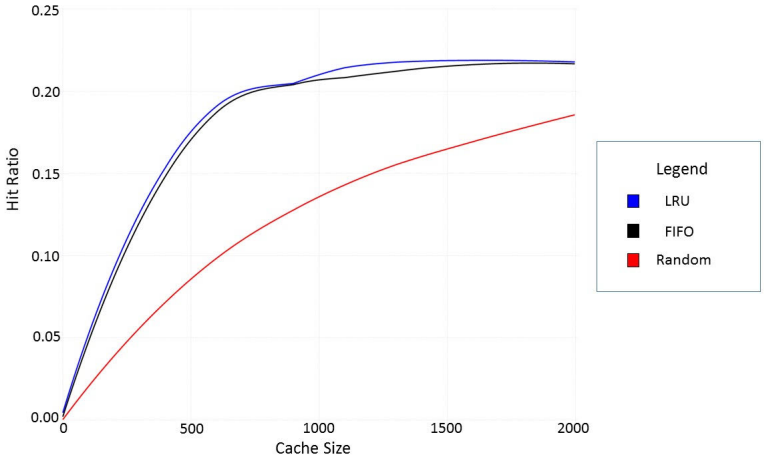
A. Relation between H_r and H_g

This section determines the relation between H_r and H_g, which demonstrates the increase in cache hit ratio in GCPiN, obtained due to grouping of the routers. To find this relation, we first study how cache hit ratio changes with cache size, because in GCPiN, cache size is effectively multiplied by *n* when a group with a Distributed Content Store is formed. We use ndnSIM [13] to study this relation, mapping cache hit ratio to absolute cache size for a variety of caching algorithms: FIFO, LRU and RANDOM. Steps to reproduce our results are available at [14].

As mentioned in Section 3.3, Fig. 3 confirms that our results are in accordance with those shown in [9] and [10], and the plot lines all follow the form

$$y = A - Be^{-Cx} \tag{1}$$

Fig. 3. Relation between Cache Size and Hit Ratio



where y is the hit ratio H, x is the cache size, and A, B and C are constants. We further verified our results using Icarus [15], a caching simulator for ICNs. Various caching strategies have been compared, and the results obtained [1] match our ndnSIM results, also yielding exponential graphs of the same form. Thus, we use equation (1) to derive the relation between H_r and H_g, as shown below.

If x_1 is the size of the router's Content Store, we get

$$H_r = A - Be^{-Cx_1} \tag{2}$$

The size of the group's Distributed Content store would then be nx1, similarly yielding

$$H_g = A - Be^{-Cnx_1} \tag{3}$$

Using equations (2) and (3), we arrive at the relation between H_r and H_g as shown below:

$$H_g = \frac{A - Be^{-Cnx_1}}{A - Be^{-Cx_1}} H_r \tag{4}$$

B. Relation between T_r and T_g

This section determines a relation between T_r and T_g, which indicates the increase in cache search time due to grouping in GCPiN.

The time taken to search the Content Store (T_r) varies as *O(log N)*, where N is the number of entries, if it is implemented as a skip list [2]. Thus, we have

$$T_r = C + D \log N \tag{5}$$

where C and D are implementation-specific constants.

Assuming there are *n* routers, each having N entries in its Content Store, the time taken to search a Distributed Content Store in GCPiN (T_g) would be

$$T_g = C + D \log nN$$

$$T_g = C + D(\log n + \log N)$$

[1] available at [14]
[2] ndnSIM implements Content Store as a skip list

$$T_g = C + D\log n + D\log N$$

$$T_g = T_r + D\log n \tag{6}$$

Thus, T_g varies as *O(log n)* if the size N of each Content Store remains constant. Equation (6) presents the relation between T_r and T_g.

C. Evaluation of GCPiN

In this section, we use equations (4) and (6) to derive a mathematical proof that evaluates the performance of GCPiN in comparison with the existing solution proposed in [4], which recommends that the routers pad the response time of a cache hit to that of a cache miss. Thus, the average response time (R_e) would be

$$R_e = 22(T_r + T_d) \tag{7}$$

The value 22 in equation (7) represents the average number of hops required to fetch data from the Internet. This was found to be 13-23 for different domains [16]. This number would reduce for NDN, as the data is cached closer to the end users, but we consider the number of hops from the first router (at which the difference in solutions begins) to be 23 – 1 = 22, i.e., the worst-case scenario.

Similarly, the average response time of GCPiN (R_p) would be:

$$R_p = H(T_g + T_d) + (1 - H) \times 22 \times (T_r + T_d) \tag{8}$$

where, H is the probability of a cache hit in the Distributed Content Store, resulting in a response time of T_g + T_d, and (1 - H) is the probability of a cache miss in the Distributed Content Store, which results in fetching information from the nodes that may be several hops away.

Subtracting equation (8) from equation (7), we get

$$\begin{aligned} R_e - R_p &= 22(T_r+T_d) - H(T_g+T_d) - (1-H)\times 22\times(T_r+T_d) \\ &= -H(T_g + T_d) + H \times 22 \times (T_r + T_d) \\ &= H(22(T_r + T_d) - (T_g + T_d)) \\ &= H(22T_r + 22T_d - T_r - D\log n - T_d) \\ &= H(21T_r + 21T_d - D\log n) \end{aligned}$$

Which is greater than 0 (i.e. $R_p < R_e$) for

$$D\log n < 21(T_r + T_d)$$

$$\log n < \frac{21(T_r + T_d)}{D}$$

$$n < e^{\frac{21(T_r+T_d)}{D}}$$

The right hand side of the inequality would be a very high number, rendering the inequality true for all practical cases of n. Thus, GCPiN enhances the privacy in NDN while incurring minimal performance loss in terms of the response time.

V. Conclusion and Future Work

In this paper, we have proposed GCPiN, an approach to increase privacy in NDN at a reduced performance cost. GCPiN involves geographically segregating routers closest to the end users into groups, then maintaining a Distributed Content Store across them. The latter is done using a distributed cache management strategy, modified to consider popularity of content in order to minimize network traffic. The working of GCPiN has been discussed in this paper, and we prove mathematically that GCPiN outperforms the existing solution to increase privacy.

Future work would be to determine the optimal number of routers in a group depending on network traffic, and to determine other aspects such as whether it would be advantageous to replicate popular content within a group. Also, added factors could be taken into consideration about whether the data requires high levels of protection or not: interactive data would normally have low-latency requirements over privacy requirements.

References

[1] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, k. clay, P. Crowley, C. Papadopoulos, L. Wang, and B. Zhang. Named data networking. SIGCOMM Comput. Commun. Rev., 44(3):6673, July 2014.

[2] S. Krishnan and F. Monrose. Dns prefetching and its privacy implications: When good things go bad. In Proceedings of the 3rd USENIX Conference on Large-scale Exploits and Emergent Threats: Botnets, Spyware, Worms, and More, LEET10, pages 1010, Berkeley, CA, USA, 2010. USENIX Association.

[3] G. Acs, M. Conti, P. Gasti, C. Ghali, and G. Tsudik. Cache privacy in named-data networking. In Distributed Computing Systems (ICDCS), 2013 IEEE 33rd International Conference on, pages 4151, July 2013.

[4] 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? SIGCOMM Comput. Commun. Rev., 42(5):5457, Sept. 2012.

[5] R. Lutz. Security and privacy in future internet architectures - benets and challenges of content centric networks. CoRR, abs/1601.01278, 2016.

[6] J. Li, H. Wu, B. Liu, J. Lu, Y. Wang, X. Wang, Y. Zhang, and L. Dong. Popularity-driven coordinated caching in named data networking. In Proceedings of the Eighth ACM/IEEE Symposium on Architectures for Networking and Communications Systems, ANCS 12, pages 1526, New York, NY, USA, 2012. ACM.

[7] X. Hu and J. Gong. Distributed in-network cooperative caching. In 2012 IEEE 2nd International Conference on Cloud Computing and Intelligence Systems, volume 02, pages 735740, Oct 2012.

[8] V. Sourlas, L. Gkatzikis, P. Flegkas, and L. Tassiulas. Distributed cache management in information-centric networks. IEEE Transactions on Network and Service Management, 10(3):286299, September 2013.

[9] D. Z. Y. y. M. Z. y. X. Jian hua Ran, Na Lv. Advances in intelligent systems research. ICACSEI-13, July 2013.

[10] A. Smith. Disk cachemiss ratio analysis and design considerations. ACM Trans. Comput. Syst. 3, 3, pages 161-203, August 1985.

[11] I. Psaras, W. K. Chai, and G. Pavlou. Probabilistic in-network caching for information-centric networks. In Proceedings of the Second Edition of the ICN Workshop on Information-centric Networking, ICN 12, pages 5560, New York, NY, USA, 2012. ACM.

[12] N. Ntuli and S. Han. Detecting router cache snooping in named data networking. In 2012 International Conference on ICT Convergence (ICTC), pages 714718, Oct 2012.

[13] A. Afanasyev, I. Moiseenko, and L. Zhang. ndnSIM: NDN simulator for NS-3. Technical Report NDN-0005, NDN, October 2012.

[14] ndnSIMawk: Awk Script to compute the cache hit ratio in NDN Content Store. https://github.com/abhijithanilkumar/ndnSIMawk

[15] L. Saino, I. Psaras, and G. Pavlou. Icarus: A caching simulator for information centric networking (icn). In Proceedings of the 7th International ICST Conference on Simulation Tools and Techniques, SIMUTools 14, pages 6675, ICST, Brussels, Belgium, Belgium, 2014. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[16] A. Fei, G. Pei, R. Liu, and L. Zhang. Measurements on delay and hop-count of the internet. In in IEEE GLOBECOM98 - Internet Mini-Conference, 1998.