On the Tradeoff between Performance and User Privacy in Information Centric Networking

Giulia Mauri and Giacomo Verticale
Department of Electronics, Information, and Bioengineering, Politecnico di Milano, Italy
{name.surname}@polimi.it

***Abstract*—Widespread use of caching provides advantages for users and providers, such as reduced network latency, higher content availability, bandwidth reduction and server load balancing. In Information Centric Networking, the attention is shifted from users to content, which is addressed by its name and not by its location. Moreover, the content objects are stored as close as possible to the customers. Therefore, the cache has a central role for the improvement of the network performance but this is strictly related to the caching policy used. However, this comes at the price of increased tracing of users communication and users behavior to define an optimal caching policy. A malicious node could exploit such information to compromise the privacy of users. In this work, we compare different caching policies and we take the first steps for defining the tradeoff between caching performance and user privacy guarantee. In particular, we provide a way to implement prefetching and we define some bounds for the users' privacy in this context.**

***Index Terms*—Information Centric Networking; Content Centric Networking; Named-Data Networking; Caching Policy; Prefetching; User's Ranking; Privacy; Data Perturbation;**

I. Introduction

The number of contents in the Internet quickly grows. Users continuously request content objects and want them as soon as possible. The need for a network of caches is obvious. A lot of research projects are focusing on this idea: CCN [1], NDN [2], DONA [3].The core paradigm is that users send to the network an interest message indicating the name of a content and the network delivers it from the nearest node cache.

Nodes can either implement a reactive caching policy or use prefetching, which if the user behavior can be well predicted, provides better performance. The information for determining which objects to prefetch can be either generated using a server-hint method or a local method. In the former, a server provides hints, which are based on previous requested objects, to routers closer to the client. The routers prefetch the contents according to the hints received. In the latter, the local prefetcher uses only local information to determine what to prefetch. In this paper, we consider prefetching policies based on information deduced from user's ranking.

Users have different preferences over the objects. Consequently, the probability of requesting a content differs from one user to another. This paper captures this aspect by defining a dissimilarity metric between users. Our work stems from the naïve consideration that a prefetching policy that exploits optimal per-user knowledge has optimal performance but is not privacy-friendly. This paper confirms both intuitions. We compare prefetching and reactive algorithms in scenarios with growing dissimilarity. Then, we try to enhance the privacy of the prefetching per-user policy and evaluate its cost in terms of performance degradation. In order to achieve this goal, we propose a mathematical definition of user privacy suitable for Information Centric Networks. Finally, we evaluate the tradeoff between privacy and latency.

The remainder of the paper is structured as follows: Section II provides an overall view on related work about the possible improvements of caching policies and the problems related to user privacy. Section III presents the model: we define the user dissimilarity, i.e. how user differs from each other, and we consider different caching policies. Section IV proposes a possible way to implement prefetching in ICN scenario. The adversary model is given in Section V, together with the privacy definitions and a possible countermeasure against privacy leakage. Section VI shows the results for the latency perceived by user depending on popularity classes, the mean delay for the most popular content classes, using data perturbation or not, and the tradeoff between privacy and latency. Conclusion and future research directions are left for the final Section VII.

II. Related Work

There is a wide literature centered on the improvement of caching and prefetching and on the privacy challenges in the content centric scenario. However, only a few considers the tradeoff between performance and privacy.

The authors of [4] analyze the caching potential to improve network performance and take into account the need for preserving users privacy-sensitive information. Moreover, they propose some countermeasures to detect and prevent information leaks. However, paper [4] considers caching policies based only on past communication and does not point out the improvement in performance. In this paper, we compare three proactive algorithms and two reactive algorithms and we give results for the latency achieved by each one.

The proposal for an high performance caching scheme that also prevents cache pollution attacks is given in [5]. CacheShield is an add-on that can be used with any cache replacement algorithm. Based on past requests, a shielding function determines whether to cache a new content. Using this function, it is possible to prevent cache pollution attacks maintaing the cache robustness.

Another solution for efficient caching is given in [6]. The authors propose WAVE, a collaborative in-network algorithm, where upstream nodes suggest the number of chunks to be

978-1-4799-3223-8/14/$31.00 ©2014 IEEE

stored to their downstream nodes. The number of chunks to be cached exponentially increases according to the content popularity. Results show that the hit ratio is improved and less frequent replacements are needed.

Both previous papers consider the same Zipf-like popularity distribution for all the contents and they do not take into account problems related to users privacy.

Cache privacy is also in [7]. The authors analyze timing and probing attacks and then they propose some countermeasures. First, they propose to mark content as private; however this method lowers the cache performance. Secondly, they describe some methods that guarantee a tradeoff between performance and privacy based on random caching. Our work considers the tradeoff between privacy and latency. However, it proposes also prefetching policies, a different adversary model and a different countermeasure.

III. Definitions

A. User Dissimilarity

We assume a set of I different contents uniformly distributed into C classes of popularity. Each item of class c has a probability of being requested equal to p_c, with $c = 1, ..., C$. The probability distribution follows the Zipf law, hence $p_c = K/c^{\alpha}$, where $K = 1/\sum_{i=1}^{C} \frac{1}{c^{\alpha}}$ and α is the slope of the distribution. In a more general scenario, the probability associated to the same content could be different for distinct users; therefore, we introduce a variable $p_{c,u}$ that defines the probability for a content of class c of being requested by the user u. This probability depends on the ranks that each user gives for each content class $r_{c,u}$, hence $p_{c,u} = K/r_{c,u}^{\alpha}$. Notice that the smaller the rank $r_{c,u}$, the more important is the content class c for the user u.

To capture how users differ from one another, we define the *dissimilarity* between two users as the number of swaps of two adjacent elements in order to obtain a permutation that maps the first user's ranks into the second one. In addition, we define dissimilarity of a set the minimum dissimilarity between all the users in the set and the natural ranking that gives the highest probability to class 1, the second highest to class 2, and so on.

B. Caching policy

There are various methods for choosing whether a content should be cached or not. Here, we consider five caching policies that can be distinguished into two classes: coordinated and uncoordinated. In the uncoordinated case, each router decides whether to cache a content using a reactive replacement strategy, e.g. LRU (Least Recently Used) and LFU (Least Frequently Used). While in the coordinate case, the routers take decisions based on a proactive computation on users ranking and on what other routers do.

We present the following prefetching and caching policies:

- **Prefetch by Popularity (PxP)**: the contents are stored into caches based on a reference popularity distribution. Assuming that the contents are organized into C classes of popularity, the ordered vector of content classes c = 1,..,C is used as reference for prefetching. The contents with a smaller class number are stored first into the nearest caches to the users. The caches that have the same distance from the users store the same contents.
- **Prefetch by Ranking (PxR)**: the contents are cached according to the popularity ranking resulting from the actual set of user in the network. The first content class to be stored is the class with the highest request probability obtained by averaging the request probability of each content class over all the users, i.e. $\sum_{u=1}^{N} r_{c,u}/N \quad \forall c \in C$, where N is number of users in the network. Also in this case, the caches that have the same distance store the same contents.
- **Prefetch by User (PxU)**: the contents are put into caches according to the popularity ranking of the downstream users. The ranking of each user is used as reference, i.e. $r_{c,u}$. The most popular contents for a user are stored first in the cache nearest to this user. Each cache stores different content classes because users are different.
- **Least Recently Used (LRU)**: the least recently used item is discarded first from the cache. This algorithm requires keeping track of all contents when they are used.
- **Least Frequently Used (LFU)**: the contents that are used least often are discarded first. This algorithm requires counting how many requests for each content are received.

As can be noted from the previous descriptions, we need different amounts of information to choose what contents should be stored into the caches. A cache that uses a reactive policy should keep track of past requests and then, it decides which content should be discarded or stored for each interest received. Whereas, the family of prefetching methods is based on users' a priori preferences. In particular, these caching policies exploit the values of $r_{c,u}$ to store a content before it is requested. Unfortunately, it is not easy to know the users' rankings because they are privacy sensitive information. Moreover, obtaining more details on ranks produces an improvement in caching performance. In the next Section, we propose how to implement prefetching by user in a content centric scenario.

IV. Prefetching by User in ICN

This section discusses a way to implement prefetching by user for the ICN nodes, according to the definition in section III. An estimator installed over the router nodes computes the likelihood that a content will be requested in the next few times and gives this information to the neighbor nodes. Then, the node can choose whether or not to prefetch the content.

The estimator uses a prediction algorithm that could be based on that described in [8]. It is out of the scope of this paper to define the detailed algorithm. However, we provide a brief description of it. However, the node can choose to send a *Prefetching Interest* for the content suggests by the algorithm. To distinguish between Interest and Prefetching Interest, we propose to modify the Interest packet adding the option `Prefetching` into the *Selector* field. In particular, the `Prefetching` option can assume the following values: (i)

0: it means that the Interest is an ordinary interest packet; (ii) **1**: it indicates that the Interest is sent for prefetching contents. Clearly, we also need to add an option to the PIT (Pending Interest Table), where it would be possible to keep track of the `Prefetching` bit.

V. Adversary Model and Countermeasure

Indeed, the ICN scenario solves the problem of privacy related to personally identifiable information, e.g. name, address, etc., but poses new challenges into the privacy of users behavior. In this Section, we present our attacker model, then we define a possible countermeasure to guarantee user privacy.

A. Adversary Model

First, we consider an attacker $\mathcal{A}$ that interacts with a challenger CH and we denote the interaction as $\mathcal{A}^{\text{CH}}$, where both the attacker and the challenger are Turing Machines (TM) computationally bounded. Particularly, we consider probabilistic polynomial time (PPT) TMs. The adversary goal is to identify a user within a set of subjects observing the contents stored in a chosen cache. Thus, the interaction continues until $\mathcal{A}$ returns an output.

The interaction between the challenger CH and the adversary $\mathcal{A}$ is described in the following Algorithm 1:

Algorithm 1 The $\mathcal{A}^{\text{CH}}$ game

1. CH chooses a user $u \xleftarrow{R} \mathcal{U}$ within the user space $\mathcal{U}$;
2. CH chooses a cache $k \xleftarrow{R} \mathcal{K}$ within the cache space $\mathcal{K}$;
3. CH gives l, the level of the cache k in a tree topology, and $k_{c_{(i)}}$, the i contents' classes stored in k to $\mathcal{A}$.
4. $\mathcal{A}$ outputs T, True, or F, False.

The Adversary wins the $\mathcal{A}^{\text{CH}}$ game if its output is:

$$\mathcal{A}_{win} = (T \wedge u \in \mathcal{U}_k) \vee (F \wedge u \notin \mathcal{U}_k)$$

where $u \in \mathcal{U}_k$ means that the user u is part of the set of downstream users of the cache k.

Thus, we give our privacy definition founding it on the notion of sender anonymity introduced in [9]:

Definition 1: Anonymity of a subject from an attacker's perspective means that the attacker cannot sufficiently identify the subject within a set of subjects, the anonymity set.

From this description, we formalize our notion of δ-sender anonymity. The parameter δ, $0 \leq \delta \leq 1$, quantifies the probability that a distinguishing event happens, e.g. when a cache is compromised by the adversary, following the definition similar to CDP (Computational Differential Privacy) given in [10].

Any user u in the user space $\mathcal{U}$ of size N is equally probable to be a user served by the cache k for any PPT-adversary $\mathcal{A}$ with the capabilities previously described. This means that there exists δ such that:

$$\Pr[\mathcal{A}_{win} | u \xleftarrow{R} \mathcal{U}] \leq \frac{1}{N} + \delta \quad \forall k \xleftarrow{R} \mathcal{K} : u \in \mathcal{U}_k \tag{1}$$

$$\Pr[\mathcal{A}_{win} | u \xleftarrow{R} \mathcal{U}] \leq (1 - \frac{1}{N}) + \delta \quad \forall k \xleftarrow{R} \mathcal{K} : u \notin \mathcal{U}_k \tag{2}$$

where $x \leftarrow X(x \xleftarrow{R} X)$ means that x is drawn (uniformly at random) from the set X. It is easy to note that the smaller is δ, the more the privacy is preserved. The following Section presents a possible way to guarantee user anonymity.

B. Proposed Countermeasure: Data Perturbation

We propose to use a *data perturbation* technique in order to lower the adversary probability of winning, $Pr[\mathcal{A}_{win}]$. A data perturbation method attempts to preserve privacy by modifying values of the sensitive attributes using a randomized process.

In the ICN scenario, the cached contents are privacy sensitive information. A router node determines which contents are to be cached based on interests received from users. Obviously, the user sends interests for contents which is more interested in based on its ranking vector. Thus, the router node can profile its downstream users observing the contents it caches. In order to hide his/her preferences, a user can send interests for contents that are not really valuable for him/her.

In particular, these interests are sent based on a new ranking vector $r^p_{c,u}$ that is obtained making p permutations on the original ranking vector of the user $r_{c,u}$. The following algorithm 2 shows how to compute the perturbed ranking vector.

Algorithm 2 The data perturbation

for $1 \leq i \leq p$ **do**
1. Extract a random number w in $[0, C]$;
2. Swap $r_{c,u}[w]$ with $r_{c,u}[w+1]$

end for
return the perturbed ranking vector $r^p_{c,u}$

Thus, the interests are sent based on the new rankings $r^p_{c,u}$, which does not really represent the user u but a user with different preferences over the contents. As we show later, introducing a perturbation guarantees user privacy but lowers the performance.

VI. Results

This Section presents the results obtained from both a simulative scenario and an analytic method. The simulation scenario takes advantage of ndnSIM [11] to evaluate the perceived latency by users using the LRU and LFU policies. While, a Montecarlo simulator, implemented with Matlab, is used to evaluate proactive policies and the adversary's advantage.

From now on, we consider a tree topology, as depicted in Figure 1, where there are four leaf nodes that send interests for the contents and one root node that generates the data packets. Moreover, the network is organized into three routers' levels that can cache the contents and answer to interests.

In this scenario, the probability of winning the game defined in the section V by algorithm 1 given by equations 1 and 2 is computed as follows:

$$\Pr[T \leftarrow \mathcal{A}^{\text{CH}}] = \Pr[T | u \in \mathcal{U}_k] \leq \frac{1}{2^{L-1}} + \delta_T.$$

$$Pr[F \leftarrow \mathcal{A}^{\text{CH}}] = Pr[F | u \notin \mathcal{U}_k] \leq (1 - \frac{1}{2^{L-1}}) + \delta_F.$$

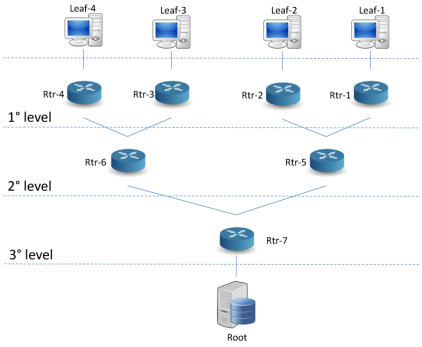

Fig. 1. The tree topology

where L is the maximum number of the tree's levels and $T \leftarrow \mathcal{A}^{\text{CH}}$ means that the adversary output is True, conversely $F \leftarrow \mathcal{A}^{\text{CH}}$, means False. Moreover, the adversary advantage δ is given by $\max(\delta_T, \delta_F)$.

In our simulations, having as reference the setup parameters used in [12], we consider a set of $I = 13.8 \cdot 10^6$ items divided into $C = 400$ classes of popularity. Each user has a rank vector $r_{c,u}$ from which the probability $p_{c,u}$ is computed according to the Zipf law. The slope of the probability distribution α is equal to 2. We consider four cases of dissimilarity with $d = 100, 1000, 10000, 100000$, where bigger d means more different users. All contents are assumed to be of the same size, i.e. $10kB$, and to be requested with a request rate $\lambda_{c,u} = \lambda p_{c,u}$, where $\lambda = 27600$ requests/sec. Moreover, we consider a tree topology with three levels of caches as shown in Figure 1. The link between each pair of nodes has a transmission delay equal to $2ms$. Each router node, rtr, has a cache (i.e. Content Store) of size s contents, which is equal to 207000, that is exactly the number of content of six classes.

A. Performance without Data Perturbation

First, we consider the mean round trip time perceived by users to retrieve a content depending on popularity classes. The results are shown in Figure 2 for dissimilarity $d = 100$ and in Figure 3 for $d = 100000$, where all the caching policies are compared.

As can be noted in Figure 2, where the dissimilarity between users is low, the prefetching policies achieve better performance in terms of Round Trip Time (RTT) than the classical LRU and LFU policies. Moreover, the three lines follow the same stepped trend because the caches contain the same content classes in the same level of the tree.

With increasing dissimilarity, as shown in Figure 3, the PxU policy guarantees the lowest latency for the six most popular classes of popularity as seen by each user. For the least popular classes, this policy has worse performance than the other ones. However this is generally not an issue since the performance seen by the user is dominated by the most popular classes, which cover the majority of the content requests. The PxP policy is independent of the popularity classes because it does not take into account the users behavior. The PxR has a similar trend as LRU policy because it reflects the mean ranking of all users.

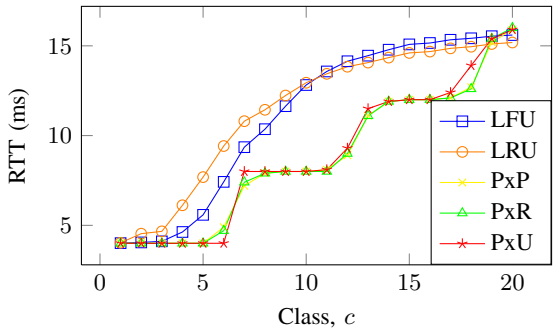

Fig. 2. Mean Round Trip Time perceived by users with $d = 100$ depending on popularity classes, c. Confidence 95%, Precision 10%

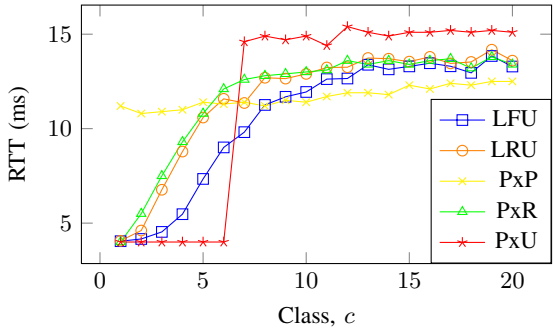

Fig. 3. Mean Round Trip Time perceived by users with $d = 100000$ depending on popularity classes, c. Confidence 95%, Precision 10%

The results relative to $d = 1000$ and $d = 10000$ are summarized in Figure 4, where the mean delay for the first six popularity classes is depicted depending on growing dissimilarity. Notice that we consider only the first six classes because they cover 90% of the total number of requests.

Figure 4 compares the mean delay for the first six popularity classes depending on dissimilarity. The prefetch by user policy achieves the lowest latency, which is constant for growing dissimilarity. The PxP and PxR performance decreases with more different users; while the LRU and LFU policies slowly increase the delay for the first classes with bigger dissimilarity. The more the performance are good, the more personal information are needed. This sentence poses a challenge that should be overcome: we need users preferences to gain the best but we should guarantee their privacy. Since the PxU policy guarantees lower latency than the classical LRU and LFU, the privacy analysis focuses on the PxU policy.

B. Performance with Data Perturbation

Now, let's see what happens if data perturbation is applied over the users' ranking. Figure 5 compares the mean delay for the first six popularity classes with growing dissimilarity depending on perturbation using a PxU policy. As can be noted, the performance are comparable to that without data perturbation with $p \leq 100$. While, the performance decreases with bigger perturbation: the mean delay grows. It can also be noted that the performance decreasing does not highly depend on dissimilarity.

Finally, Figure 6 reports the advantage δ that the Adversary $\mathcal{A}$ has in winning the game previously defined, with $i = 6$, that indicates how many of the k_c classes are given to $\mathcal{A}$.

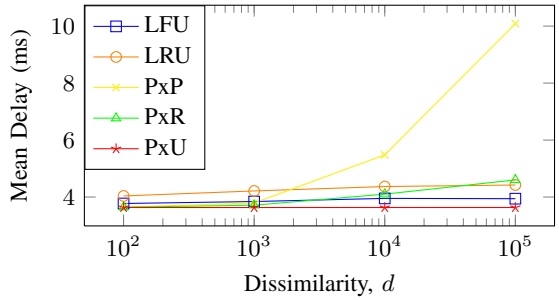

Fig. 4. Mean Delay perceived by users for retrieving a content of the first six popularity classes depending on dissimilarity, d.

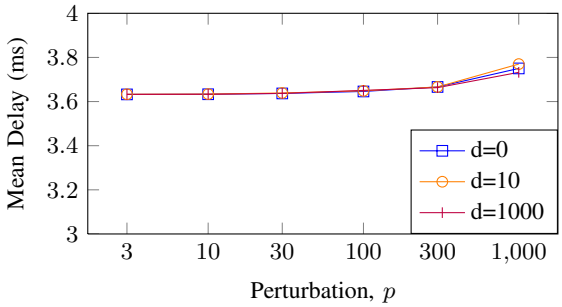

Fig. 5. Mean Delay perceived by users for retrieving a content of the first six popularity classes with growing dissimilarity, d, depending on perturbation, p.

The results are obtained repeating the challenging game until a 95% of confidence is reached. We'd like to note that the advantage in case of no perturbation, $p = 0$, is exactly 1 that means the adversary always wins the game. While if we apply a perturbation, the advantage decreases till 0.05 with any perturbation $p \geq 100$. This means that if there is this kind of perturbation the adversary randomly chooses a user and wins the game with a probability:

$$Pr[T|u \in \mathcal{U}_k] \leq \frac{1}{4} + 0.05 \vee Pr[F|u \notin \mathcal{U}_k] \leq \frac{3}{4} + 0.05.$$

Thus, the user privacy is guaranteed with a perturbation higher than 100. However, we have seen in the previous figure that the performance are not worsened with $p \leq 100$. This means that we have to choose a tradeoff between latency and privacy, and we think that using a perturbation p=100 could be the perfect choice. Indeed, the performance are comparable to the case without data perturbation and the adversary advantage is low, so the user privacy is guaranteed.

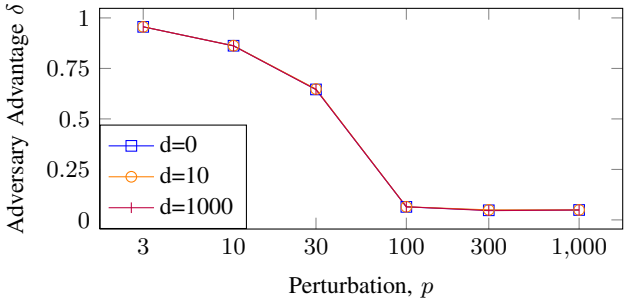

Fig. 6. Advantage that the Adversary $\mathcal{A}$ has in winning the $\mathcal{A}^{\text{CH}}$ game depending on perturbation, p, with growing dissimilarity, d

VII. Conclusion

This paper analyzes the tradeoff between network performance and user's privacy in a information centric scenario. On the one hand, low latency provides both clients and providers with advantages. On the other hand, user sensitive information are needed to gain better performance. This work poses the basis for finding a solution that simultaneously guarantees performance and respects users' needs. Indeed, we propose a prefetching policy based on users' ranking and a data perturbation technique to guarantee users' privacy.

We think that there is a lot of research to inspect in this direction. First, we need to provide a privacy analysis for LRU and LFU policies. Then, we can find the tradeoff between privacy and performance using these caching policies Moreover, we would like to extend our work considering users churn and different popularity of content chunks in order to provide better network capabilities. We believe that this change slightly modifies the optimal caching policy and also requires a deeper knowledge of user's behavior. On the other side, we would like to expand the solutions for guaranteeing user's privacy introducing an anonymization protocol.

Acknowledgment

The research leading to these results has received funding from the European Community Seventh Framework Programme FP7/2013-2015 under grant agreement no. 317762 COMBO project.

References

[1] V. Jacobson *et al.*, "Networking named content," in *Proc. of the 5th international conference on Emerging networking experiments and technologies*, ser. CoNEXT '09. ACM, 2009, pp. 1–12.

[2] L. Zhang *et al.*, "Named data networking (ndn) project," University of California and Arizona, Palo Alto Research Center and others, Tech. Rep., October 2010.

[3] T. Koponen *et al.*, "A data-oriented (and beyond) network architecture," *SIGCOMM Comp. Comm. Rev.*, vol. 37, no. 4, pp. 181–192, Aug. 2007.

[4] T. Lauinger *et al.*, "Privacy risks in named data networking: what is the cost of performance?" *SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 5, pp. 54–57, Sep. 2012.

[5] M. Xie *et al.*, "Enhancing cache robustness for content-centric networking," in *INFOCOM*, A. G. Greenberg and K. Sohraby, Eds. IEEE, 2012, pp. 2426–2434.

[6] K. Cho *et al.*, "Wave: Popularity-based and collaborative in-network caching for content-oriented networks," in *Comp. Comm. Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on*, 2012, pp. 316–321.

[7] G. Acs *et al.*, "Cache privacy in named-data networking," in *33rd IEEE International Conference on Distributed Computing Systems, Proceedings of*, 2013.

[8] V. N. Padmanabhan *et al.*, "Using predictive prefetching to improve world wide web latency," *SIGCOMM Comput. Commun. Rev.*, vol. 26, no. 3, pp. 22–36, Jul. 1996.

[9] A. Pfitzmann *et al.*, "A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management," 2009.

[10] M. Backes *et al.*, "Anoa: A framework for analyzing anonymous communication protocols," in *Computer Security Foundations Symposium (CSF), 2013 IEEE 26th*, 2013, pp. 163–178.

[11] A. Afanasyev *et al.*, "ndnSIM: NDN simulator for NS-3," NDN, Technical Report NDN-0005, October 2012.

[12] G. Carofiglio *et al.*, "Modeling data transfer in content-centric networking," in *Teletraffic Congress (ITC), 2011 23rd International*, 2011, pp. 111–118.