Betweenness Centrality and Cache Privacy in Information-centric Networks

Noor Abani
University of California, Los Angeles
noorabani@cs.ucla.edu

Torsten Braun
University of Bern
braun@inf.unibe.ch

Mario Gerla
University of California, Los Angeles
gerla@cs.ucla.edu

ABSTRACT

In-network caching is a feature shared by all proposed Information Centric Networking (ICN) architectures as it is critical to achieving a more efficient retrieval of content. However, the default "cache everything everywhere" universal caching scheme has caused the emergence of several privacy threats. Timing attacks are one such privacy breach where attackers can probe caches and use timing analysis of data retrievals to identify if content was retrieved from the data source or from the cache, the latter case inferring that this content was requested recently. We have previously proposed a betweenness centrality based caching strategy to mitigate such attacks by increasing user anonymity. We demonstrated its efficacy in a transit-stub topology. In this paper, we further investigate the effect of betweenness centrality based caching on cache privacy and user anonymity in more general synthetic and real world Internet topologies. It was also shown that an attacker with access to multiple compromised routers can locate and track a mobile user by carrying out multiple timing analysis attacks from various parts of the network. We extend our privacy evaluation to a scenario with mobile users and show that a betweenness centrality based caching policy provides a mobile user with path privacy by increasing an attacker's difficulty in locating a moving user or identifying his/her route.

CCS CONCEPTS

• **Information systems → Information storage systems**; • **Security and privacy → Pseudonymity, anonymity and untraceability**; **Privacy-preserving protocols**; • **Networks → Network protocol design**;

KEYWORDS

ICN, NDN, information-centric networking, named-data networking, caching, cache privacy, timing attacks, betweenness centrality, anonymity, anonymity set

ACM Reference format:
Noor Abani, Torsten Braun, and Mario Gerla. 2018. Betweenness Centrality and Cache Privacy in Information-centric Networks. In *Proceedings of ICN '18, Boston, MA, USA, September 21–23, 2018,* 11 pages.
https://doi.org/https://doi.org/10.1145/3267955.3267964

1 INTRODUCTION

There have been several different proposals of ICN architectures in the literature [9, 12, 13, 23]. All such architectures feature content-based communication and drop the end-to-end principle that keeps end-to-end transactions oblivious to resources and content available along the path. Consequently, all ICN architectures leverage in-network caching where routers in the network cache content items rendering content retrieval faster and more bandwidth efficient.

Currently, in their default setting, ICN architectures adopt a ubiquitous mode of caching, which is more commonly known as the Leave Copy Everywhere (LCE) strategy. As the name indicates, with LCE, routers cache all content objects that pass through them. In other words, when the content is fetched from the original data publisher a copy of the object is cached on all routers in the data forwarding path back to the requester. While in-network caching can enhance performance, if done naively as described above, it can pose several privacy threats. In particular, an LCE approach to caching can make content consumers vulnerable to timing analysis attacks. In such attacks, the attacker uses the difference in data retrieval times between cached content and content retrieved from the data source to infer whether content has been requested before [3] or, if a target user's content of interest is known, to locate the user [8]. Previous work addressing these attacks aim at masking cache hits by adding delays to eliminate time differences between cache hits and cache misses [3, 17, 18]. While such mechanisms still allow for alleviating the congestion on long links to the server, they negate any latency gains in content retrieval since each cache hit should be masked with a delay that matches content retrieval from the original content server.

In this work, we advocate a different perspective to mitigating the risks of caching on user privacy. In particular, we study the effect of selective content placement to achieve higher user anonymity in comparison to LCE. In previous work [1], we proposed a Betweenness Centrality (BC)-based caching strategy, which caches sensitive content at nodes with higher BC to put their consumers in larger Anonymity Set (AS)s. Such a strategy aimed at mitigating timing attacks without the addition of delays at the routers. Based on a proof-of-concept example, in [1], we showed a correlation between BC and Anonymity Set Size (ASsize) in a simple transit-stub topology and leveraged such a correlation with a caching scheme that mitigates the risks of a timing analysis attack. In this paper, we add two contributions. First, we investigate the effect of network topology on the effficacy of BC-based caching. We consider both general synthetic and real-world Internet topologies topologies. Our privacy evaluations show that in structured synthetic networks and in real-world Internet topologies, a BC-based caching strategy guarantees consumers of sensitive content a larger AS in comparison to LCE or a random selective caching

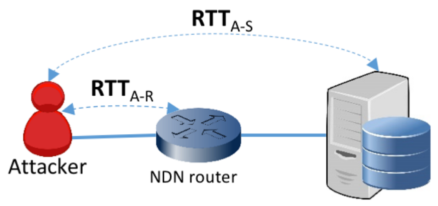

Figure 1: Timing Attack

strategy. Second, we extend our work to consider a more sophisticated attacker with access to several compromised routers in the network. It was shown in [8] that such an attacker can execute multiple timing attacks from its compromised routers to learn the path from the consumer to the data source, thus geographically locating the consumer by identifying its first hop router. We also show that a BC-based caching policy prevents the attacker from identifying a moving user's route, i.e, tracking its location through the first hop routers it connects to. In addition, cache performance results show that our BC-based caching strategy incurs minimal additional delays when compared to the aforementioned mechanisms for mitigating timing attacks.

2 BACKGROUND

2.1 Named Data Networking

In this section, we provide some background on Named Data Networking (NDN) since we use the ns3-based NDN simulator (ndnSIM) [4] to evaluate our caching policy. We will also use NDN's terminology to present our algorithms. It is worth mentioning, however, that our scheme can be integrated with any ICN architecture and is not restricted to NDN.

Proposed ICN architectures [9, 12, 13, 23] differ in their design choices but share two fundamental features, those being name-based communication and in-network caching. Similar to the other designs, NDN identifies content objects with unique names, rather than identifying hosts with unique IP addresses. A content name in NDN is represented as a "/" delimited path-like representation. For example, the name of UCLA's main homepage could be `/ndn/ucla/home/index.htm`.

A node interested in a particular content object sends an *Interest* packet specifying the name of the content. For packet forwarding, each node in the network has three main data structures: a Pending Interest Table (PIT), a Forwarding Information Base (FIB) and a Content store (CS). Upon receiving an *Interest* packet, a node checks if it has the requested content in its CS. If so, it will send a *Data* packet back to the original requester. Otherwise, the node adds a new PIT entry which includes the interest name and the interface from which the *Interest* packet was received, which is then forwarded towards the data source using the FIB. When the data is found, a *Data* packet is forwarded back to the requester using the PIT table entries and is cached according to the reactive caching policy adopted by the nodes.

2.2 Related Work

Previous work related to timing attacks can be divided into two categories. One category highlights the vulnerability of ICN to timing attacks among other threats while the other provides possible mitigation techniques.

In the first category, the surveys [2, 7, 14, 22] provide an extensive dissection of the privacy and security threats and attacks on ICN. The authors discuss ICN's vulnerability to timing attacks due to its default ubiquitous caching strategy. The authors in [8] investigate a more sophisticated threat model, where an attacker with control over several compromised routers can execute multiple timing analysis attacks to learn the path between the consumer and data source, thus learning the consumer's location (first hop router). The authors show the feasibility of the attack in identifying a consumer's first hop router for the AT&T topology.

The other category of previous work provides techniques to alleviate the privacy threat of timing attacks. The authors in [3] suggested the flagging of private content by the user notifying intermediate routers not to cache the content. However, this hinders the user from leveraging in-network caching. Therefore, to maintain the performance benefits of caching, they further suggest different mechanisms for interactive traffic and content distribution traffic. Acknowledging the fact that interactive traffic does not tolerate delay, the authors propose the addition of random suffixes to content names that are mutually agreed upon between consumer and producer making it infeasible for the attacker to carry out the attack as it does not have the content names to probe the cache. Even though this technique does not add latency, it makes cached content useful for this consumer only since other users are unaware of the content names. While acceptable for interactive traffic, applying such a technique for content distribution would render in-network caching almost obsolete.

As for content distribution, the authors propose emulating a cache miss at the routers. To do so, routers apply a random delay before satisfying interests that have resulted in a cache hit. To mitigate the latency incurred by such delays, the authors propose applying this random delay only if the number of requests for the same content is below a threshold.

Similarly, authors in [17] and [18] propose the addition of a delay to the data replies from intermediate routers. However, the delay added is not random but rather equal to the delay of retrieving the data from the original server. To keep track of this delay, routers record retrieval times of content chunks from the publisher. To reduce the amount of state stored at the router, the authors propose a more efficient algorithm where retrieval times are stored per interface rather than per content name. Authors in [7] also suggest adding random delay on all requests for cached content or on the first k requests only, which similar to the previous mitigation techniques could severely degrade performance.

While adding random delays to emulate cache misses provides strong privacy guarantees, this is done at the expense of high latency, which renders the main goal of ICN, i.e., efficient content distribution, unattainable. In addition, these delays are added to all content alike, thus affect the latency of retrieving non-sensitive content as well. In this work, we take a content placement approach to address cache privacy and introduce a centrality-based caching

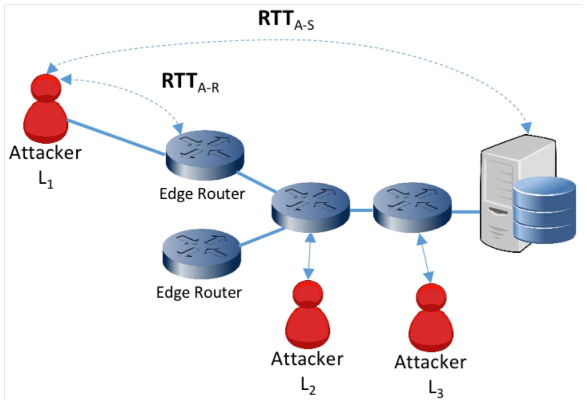

Figure 2: Multiple landmarks - Timing Attack

scheme that provides consumers of sensitive content with higher privacy by putting them in a large AS while still fully leveraging in-network caching for both sensitive and non-sensitive content.

3 THREAT MODEL AND PRIVACY METRIC

3.1 Timing Attacks

In general, with timing attacks, an adversary probes caches for content and uses the difference in retrieval time between cached content and content retrieved from the server to learn if that content was fetched from the cache or from the server. In its simplest form shown in Fig. 1, a timing attack can be performed by an attacker via the following steps:

(1) Request an unpopular content object O_1
(2) Record the Round Trip Time (RTT) for the first retrieval of O_1 (Being unpopular, O_1 is retrieved from the server with RTT_{A-S})
(3) Request the same object O_1 again
(4) Record the RTT for the second retrieval of O_1 (With LCE, O_1 is satisfied by the edge router with RTT_{A-R})
(5) Request the sensitive object of interest O_2
(6) Record the delay RTT' for the retrieval of O_2
(7) Compare RTT' with both RTT_{A-S} and RTT_{A-R}

After comparing RTT' with both RTT_{A-S} and RTT_{A-R}, the attacker can learn if its request for O_2 was retrieved from the data source or from a cache.

Previous work has shown that an attacker can carry out such attacks to do any of the following: (1) learn if content has been requested before and attribute it to a certain user [3] or (2) locate a user, i.e., locate the one-hop router to which the user is connected [8]. The authors in [3] investigate the possibility of such a timing attack and confirm its feasibility on routers that are directly connected to both the attacker and the target and also on routers that are several hops away. In this case, the attacker is interested in determining if a specific piece of content was requested and if so, attempt to attribute it to a particular user. In [8], the authors consider a more advanced adversary model, where the attacker has access to several compromised landmarks in the network, for example L_1, L_2 and L_3 in Fig. 2. They present an algorithm where such an attacker can execute multiple timing attacks from those landmarks to construct the path between the data source and the consumer, thus identifying the edge router to which the consumer is connected. In contrast to the previous scenario, an attacker in this case knows the name of a piece of content that a consumer might have requested and uses it to probe caches to locate the user or to be more precise, the one-hop router to which that consumer is connected.

3.2 Threat Model

Given that timing attacks are performed as described in the previous section, in our model, we target an adversary who is capable of executing such attacka, i.e., an adversary that is capable of performing fine time measurements to differentiate cache hits from cache misses.

The first type of adversary we consider is an adversary who is interested in identifying if certain sensitive content has been requested before and if so, identifying the user who has requested it. As shown in Fig. 1, the attacker needs to perform the attack once to probe one cache and determine if the content was requested before. It is worth noting that the content the attacker is interested in attributing to the victim is low in popularity. Content of high popularity is of no interest as it is requested often and by a large number of users. Low popularity content, on the other hand, increases the privacy risks. Such an adversary need not be very sophisticated as it suffices that it has NDN capabilities to send and to receive *Interest* and *Data* packets respectively.

The second type of adversary is one whose goal is to locate a target user. In this case, the adversary already knows the content a target user might have requested and so it probes caches for that content aiming at identifying the target's first hop router, and then identifying the user's location. As described in [8], for that purpose, the adversary is assumed to have topology and routing information, access to a fixed subset of compromised nodes in the network as shown in Fig. 2, in addition to knowing the name of the content requested by the target consumer.

3.3 Anonymity Set

For the first type of attacks, our ultimate goal is to make it infeasible for the attacker to pinpoint a particular user as the original requester of specific sensitive content when a successful timing attack is performed. For the second type of attack, the goal is to protect the user's location privacy. For both goals, we will use a metric called the AS to measure the amount of anonymity provided. In general, AS is defined as the set of all possible subjects who might cause an action [19]. Specific to our model, the AS is the set of users who could have potentially requested some content object (for the first adversary) or the set of one-hop routers that the user could be connected to (for the second adversary).

Going back to the example provided in Fig. 1, the users co-located with the attacker and connected to the same edge router are all potential originators of the previous request(s) for O_2 and hence they form the AS of the target. This is the case because, with ubiquitous on-path caching, the only users who could have requested content which was cached on this edge router are the users connected to this particular edge router. Requests from other users would not have taken this path and passed through this particular edge router.

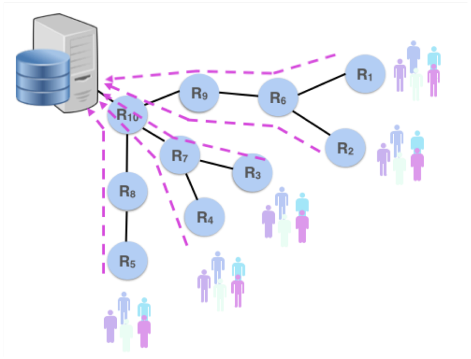

Figure 3: Motivating Scenario

Assuming that the attacker does not have additional information that differentiates one user from the other, the AS is a good indicator of how well anonymity is preserved. In a particular AS, the probability that an attacker identifies a specific user as the source of a request is $\frac{1}{\text{ASsize}}$. As a result, the larger the AS the more difficult it is for an attacker to distinguish a particular user as the requester of some content given that a larger number of users might have potentially requested it.

Similarly, if the adversary is trying to locate the user and assuming that all edge routers are equally likely of being connected to the user, the larger the AS, i.e., the number of possible such edge routers, the higher is the attacker's uncertainty in determining the location of the target.

Assuming that the number of users connected to each edge router is the same, the AS for both scenarios is then the number of first hop edge routers that could have forwarded the *Interest* packet for that content, which is the metric we will be using in our privacy evaluation in Section 5.

4 BETWEENNESS CENTRALITY-BASED CACHING

4.1 Betweenness Centrality

As we will detail next, we will use a node's BC as the measure of how much privacy is provided if content were to be cached at that node. Betweenness Centrality is a quantity that measures how often a specific node lies on the shortest path between all pairs of nodes in a network [11]. Representing a network of routers as a graph $G = (\mathcal{V}, \mathcal{E})$, the BC of a router n is:

$$BC(n) = \sum_{\forall s,t \in V \setminus n} \frac{\sigma_{st}(n)}{\sigma_{st}} \quad (1)$$

where σ_{st} is the number of shortest paths between s and t, and $\sigma_{st}(n)$ is the number of such paths that pass through n.

4.2 Motivating Scenario

Before we describe our BC-based caching strategy, we will start with a motivating scenario. Consider the topology in Fig. 3, with LCE, a *Data* packet delivered by the server in response to an *Interest* packet made by a user connected to edge router R_1 will be cached at routers R_1, R_6, R_9 and R_{10} since they all lie on the content delivery path from R_1 to the server. Consequently, an adversary that is able to successfully probe a number of those caches will be able to learn that the user is connected to edge router R_1 resulting in an ASsize of 1. On the other hand, if the content is cached at R_{10} only, which is the node with the highest BC in this topology, if an adversary is capable of probing R_{10}, it will learn that the request came from either R_1, R_2, R_3, R_4 or R_5 since R_{10} is on the content delivery path from all those routers to the data server. As a result, the ASsize is 5 providing the consumer of the content with stronger anonymity in comparison to LCE.

Algorithm 1: BC-based Caching Policy - Producer Driven

```
Input  : Data packet $P$ tagged with $C_{B_{min}}$
Output : Cache $P$ or Do Not Cache $P$
1  bool cache = false
2  if $C_B(n) \geq C_{B_{min}}$ then
3      cache = True
4  else
5      cache = False
6  end
7  return cache
```

Algorithm 2: BC-based Caching Policy - Consumer Driven

```
Input  : Interest packet $I$ tagged with $C_{B_{min}}$
Output : Cache $P$ or Do Not Cache $P$
1  Add $C_{B_{min}}$ to PIT entry
2  Use FIB to forward $I$
3  Wait for corresponding Data packet to arrive
4  Extract $C_{B_{min}}$ from PIT entry
5  bool cache = false
6  if $C_B(n) \geq C_{B_{min}}$ then
7      cache = True
8  else
9      cache = False
10 end
11 return cache
```

4.3 BC-based Caching Strategy

In ICN, the default forwarding policy is to take the shortest path towards the producer of content. Given the aforementioned definition of BC, it can be seen that, in the context of ICN, BC is a potentially good indicator of how many times a node falls on the content delivery paths from edge routers, which are the first hop receivers of *Interest* packets from consumers, to the original publishers of content. Since the goal of our work is to increase user anonymity by putting users and user locations in a larger AS, we will cache sensitive content on nodes with higher BC.

We propose both a consumer and a producer driven approach to the BC-based strategy. We assume that $BC(n)$ is pre-calculated offline for all routers and so each router n knows $BC(n)$. There are algorithms for an efficient calculation of BC, in addition to algorithms for a distributed computation at each router that does

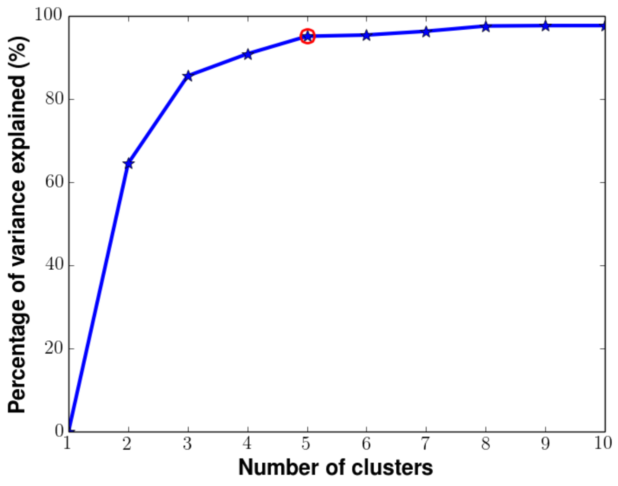

Figure 4: Distortion vs. Number of Clusters

not require a global knowledge of the topology [6, 10] but this is out of the scope of this work.

In the producer-driven approach, the producer, upon receiving an *Interest* packet, determines the popularity rank of the requested content object and its sensitivity and tags the corresponding *Data* packet with a suitable BC_{min}. As described in Algorithm 1, every router on the path back to the consumer caches the content only if its BC is higher than BC_{min}. We will elaborate more on the choice of BC_{min} in the next section but it suffices to know that the more sensitive the content is, the higher the value of BC_{min} should be.

Algorithm 2 describes the implementation of the BC caching policy in a consumer-driven approach. In such an approach, the consumer, when creating an *Data* packet, tags the packet with a BC_{min} value. The intermediate routers on the forwarding path extract this value and add it to the created PIT entry before forwarding the packet. When the corresponding *Data* packet arrives, the routers decide to cache the content object if their BC is higher than the minimum value assigned by the consumer.

4.4 Choice of BC_{min}

Popular content might be sensitive in nature; however, the fact that a large number of users is requesting it would naturally mask a particular target's interest in the content. Therefore, isensitive and not highly popular content is the content that needs to be masked and whose consumers need to be put in a larger AS. Acknowledging this fact, we choose BC_{min} for a particular content object o such that BC_{min} is inversely proportional to the popularity of o. A packet for an object with high popularity is tagged with a lower BC_{min} while that of an object of low popularity is tagged with a higher BC_{min}.

After normalization, BC assumes values between 0 and 1; therefore, to reduce the granularity of possible BC values, we leverage the fact that in structured networks nodes fall into groups of degree of connectivity, which also reflects nodes' BC. Therefore, to reduce granularity we cluster the BC values of the nodes in the network and use the centroid values as the set of finite discrete possible values for BC_{min} for each content object depending on the popularity of the object.

For example, let us consider the AT&T topology in Fig. 12. We run the k-means clustering algorithm [15] on the BC values of the nodes for different values of k. We use the elbow method and find that for $k = 5$, 95% of the variance is explained as shown in Fig. 4. As a results, we group the BC values into 5 clusters represented by $BC_1, \ldots, BC_5$ from smallest to largest BC value.

Given that the BC values fall into 5 clusters, we will split the content catalog into 5 categories of popularity. Consider content catalog C of size $|C|$ with content popularity that follows the Zipf distribution. Therefore, if Rank is the random variable representing the popularity rank of a requested object then the probability that Rank $= k$ is:

$$P(\mathsf{Rank} = k) = \frac{\frac{1}{k^{\alpha}}}{\sum_{i=1}^{|C|} \frac{1}{i^{\alpha}}} \tag{2}$$

Let rank(o) be the rank of object o. Since we know that the routers in this topology fall into 5 different tiers of BC values, we split content objects into 5 types:

- $O_1 := \{o \in C \mid P(\mathsf{Rank} \leq \mathsf{rank}(o)) \leq 0.20\}$
- $O_2 := \{o \in C \mid 0.20 < P(\mathsf{Rank} \leq \mathsf{rank}(o)) \leq 0.40\}$
- $O_3 := \{o \in C \mid 0.40 < P(\mathsf{Rank} \leq \mathsf{rank}(o)) \leq 0.60\}$
- $O_4 := \{o \in C \mid 0.60 < P(\mathsf{Rank} \leq \mathsf{rank}(o)) \leq 0.80\}$
- $O_5 := \{o \in C \mid P(\mathsf{Rank} \leq \mathsf{rank}(o)) > 0.80\}$

As a result, O_1 is the set of content objects with highest popularity i.e. they make up 20% of the bulk of requests. O_2 is the set of objects that received the next 20% of the requests, with O_5 being the set of content with lowest popularity. When the server receives requests for content that belongs to O_1 it will tag the *Data* packet with $\hat{BC}_1$, the lowest BC value in the cluster centered at BC_1. Similarly, when the requested content belongs to O_2, the server assigns it $\hat{BC}_2$. Most sensitive content, i.e, content objects in the set O_5 are tagged with the highest BC that being $\hat{BC}_5$.

5 PRIVACY EVALUATION

We will divide our performance evaluation into two parts. In the first part, we will evaluate the BC-based caching policy's effectiveness in providing better privacy by putting consumers of sensitive content in larger ASs. In the second part, we will study the performance of the caching strategies in terms of cache hit rates, content retrieval latency and server load reduction.

5.1 Methodology

5.1.1 Static Case. By definition, BC is a direct measure of the number of times a node falls on the shortest paths between all pairs of nodes in a graph. In a real network, not all shortest paths are of interest. In fact, only the shortest paths between edge routers and the content servers determine the size of an AS. It is, therefore, necessary to evaluate and quantify the privacy achieved by the BC-based caching policy in networks with a set of content delivery paths from edge routers to content servers.

For the evaluation of privacy, we will calculate the AS sizes achieved with BC-based caching in comparison to LCE and random caching. In random caching, one node on the content delivery path is chosen at random to cache the passing content object. We choose to compare with random caching because, while it might be obvious that being selective in caching does mitigate timing attacks, we would like to investigate if choosing one caching node at random would achieve the same privacy as selecting a node with high BC. We will also evaluate privacy in topologies with different structural properties because, when viewed as a graph, a network's structure highly affects the distribution of the nodes' BC values.

To compute the ASsize, we generate synthetic topologies and we choose a random producer, a random edge node and a random set of landmarks in each topology. As for caching nodes, we choose them based on the caching strategy under evaluation. For LCE, all nodes on the content delivery path are caching nodes. For random caching, we randomly choose one node on the path as a caching node and for BC caching the node with the highest BC is set as the caching node. We consider the content delivery path between that edge router and the producer, i.e., the shortest path, and check if the landmarks are able to probe any LCE caching node or the random node chosen by random caching or the node with highest BC on this path and if so, compute the ASsize of each caching policy as the number of edge routers whose content delivery path to the producer includes that caching node. We perform multiple iterations of the above to generate the results presented below.

5.1.2 Mobile User Case. To investigate the privacy achieved against an adversary attempting to track a mobile user, we use the San Francisco taxi cab mobility dataset [20], which contains the longitude and latitude coordinates of around 500 taxi cabs collected over the course of several weeks in the area of San Francisco. We divide the San Francisco area into equally sized areas (1 km × 1 km) and assume that each of these areas is covered by an access point (AP). Then, we transform the dataset into a new format, where the latitude and longitude entries are replaced by the identifier of the AP that is the closest in distance to the vehicle when it is at that GPS location. We then transform the location information into routes determined by the sequence of APs the vehicle connected to as it was moving. We then generate the network connecting these APs to the Internet by a random Barabasi-Albert (BA) graph of 500 nodes, of which 219 of those are the APs. Similar to what was described in the previous section, for computing the ASsize, we connect the producer to a random node in the network and randomly distributed the attacker's landmarks. For each route in the mobility dataset, we consider the content delivery paths from each AP in the route to the producer and determine which caching nodes (for each strategy) can be probed by the landmarks, thus computing the number of potential APs that could have been the first hop for the sent request. In addition to the ASsize, we also report the number of routes the attacker is able to identify completely, i.e., identify all the APs the vehicle connected to throughout the drive.

5.2 Results

5.2.1 Static Case.

Random Graphs. We first generate random graphs based on the Erdos-Renyi (ER) model, where a graph $G(N, p)$ has N nodes such that each pair of nodes is connected with a probability p. An example of such a graph is in Fig. 5. Fig. 6 shows the AS sizes achieved

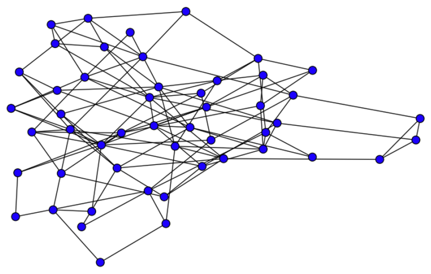

Figure 5: ER graph

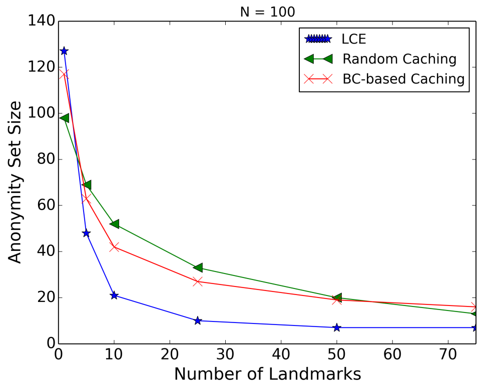

Figure 6: Anonymity Set Size - ER graph

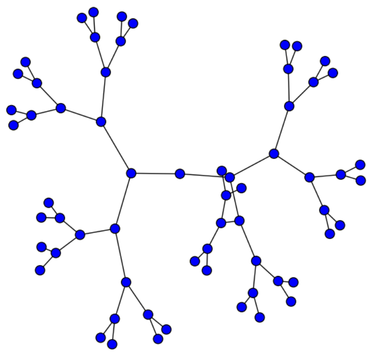

Figure 7: Tree: M = 2, D = 5

in an ER graph with $N = 100$ and $p = 0.1$. We can see that less ubiquitous caching, whether random or BC-based, does increase the ASsize by a factor of 2× in comparison to LCE. This comes naturally with the fact that there are less nodes that are caching the content making it less likely that edge routers are the nodes chosen for caching. We can also see that BC caching does not outperform random caching but rather both random caching and BC caching result in similar ASsize. This is due to the nature of ER graphs, where node connectivity does not vary over a large range, making all nodes in the network similar and of equal importance.

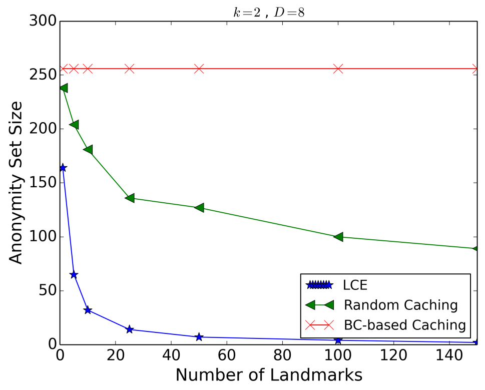

Figure 8: Anonymity Set Size - M-ary tree (M= 2, D = 8)

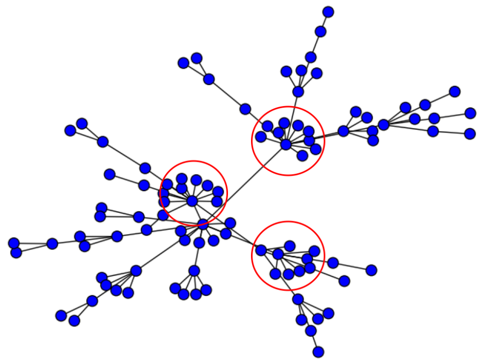

Figure 9: BA graph

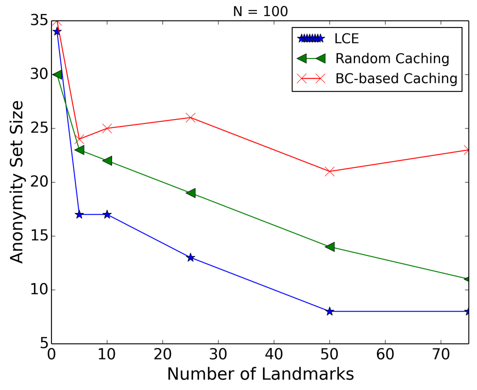

Figure 10: Anonymity Set Size - BA graph

M-ary Trees. An M-ary tree is a tree where each internal node has M children, i.e., a degree M. An M-ary tree is also defined by its depth D, which represents the number of edges between the root and the leaf nodes as in the example tree of Fig. 7. For this

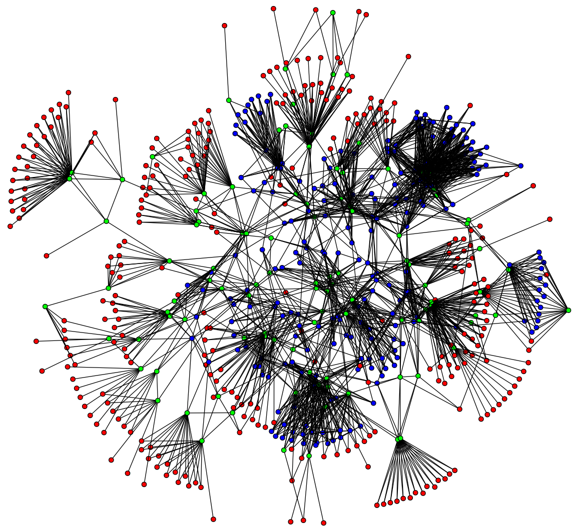

Figure 11: AT&T Topology

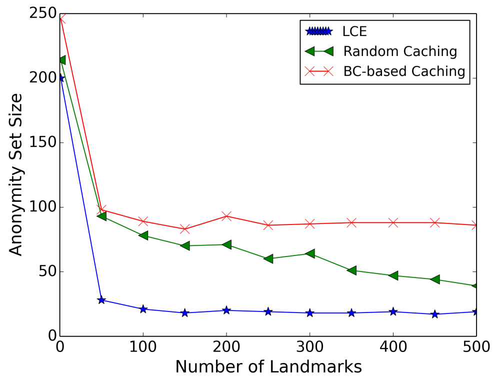

Figure 12: Anonymity Set Size - AT&T Topology

structure, we run our simulations with the producer connected to the root node and the consumers connected to the leaf nodes. For BC-based caching, content is cached at the node with highest BC, that being the root node. For LCE, content is cached on all nodes on the path from the requested consumer and the producer; whereas, the random caching strategy, as in previous cases, chooses a random node on that path to cache the requested content. Figure 8 shows the ASsize for all caching policies. We can see that while choosing a node at random increases the ASsize by ≈ 10×, a more strategic caching decision, where sensitive content is cached at the node with a high BC increases the ASsize by ≈ 25× in comparison to LCE. As the number of the adversary's landmarks increases, the ASsize of the random caching strategy decreases, showing that it becomes necessary to cache on the node with high BC to guarantee a high ASsize.

Random scale-free networks. It has been shown that real-world topologies follow a power law degree distribution. One model that

Table 1: Percentage of routes completely identified by the attacker

	10-minute routes		15-minute routes		30-minute routes	
Number of landmarks	LCE	BC	LCE	BC	LCE	BC
100	7.21%	1.47%	6.25%	1.11%	1.41%	0.19%
200	21.76%	1.66%	16.74%	1.37%	7.55%	0.28%
300	37.77%	3.54%	33.74%	1.62%	21.12%	0.56%
400	64.45%	3.57%	56.13%	2.27%	44.83%	0.34%
500	94.81%	4.23%	91.57%	3.43%	85.17%	0.42%

Table 2: Partial paths identified by the attacker (% of total path)

Number of landmarks	LCE	BC
100	37.4%	14.02%
200	59.25%	19.14%
300	79.01%	22.88%
400	92.24%	23.53%
500	98.21%	30.26%

Table 3: Average number of possible paths

Number of landmarks	LCE	BC
100	19387263	33258560661793
200	31843	3064611428045007
300	21842	246757846066123
400	2771	4074423275331
500	2	33258560661793

accurately depicts such networks is the BA model [5]. The BA model reproduces the power law degree distribution by accounting for the preferential attachment property of real networks. It says that a new vertex added to a network is more likely to connect to a vertex with a high degree rather than that likelihood being uniform among all existing vertices in the network. This results in the formation of hub nodes as shown in the example BA graph in Fig. 9. From Fig. 10, we can see that BC-based caching puts consumers in ASs of larger size in comparison to random caching and LCE. As the number of the adversary's landmarks increases in the network, the gain achieved is higher.

For the privacy evaluation, we can observe that unlike in unstructured ER graphs, in structured networks like M-ary trees and BA graphs, BC caching achieves better privacy. Selective caching, such as random caching, does increase ASsize but in the presence of a strong adversary with a large number of landmarks in the network, random caching does not maintain strong privacy while BC caching maintains a large ASsize.

Real-world Internet Topologies. For more insight into the effectiveness of BC caching in putting consumers in large ASs, we further asses the privacy achieved in real-world Internet topologies. We compare the caching strategies in the AT&T ISP topology mapped by Rocketfuel [21]. The AT&T topology shown in Fig. 11 has 625 routers and 2101 links. As expected, LCE puts consumers in ASs of smaller size in comparison to both random caching and BC caching as shown in Fig. 12. However, the ASsize achieved with random caching decreases as the number of landmarks increases; whereas, BC caching is able to maintain an ASsize of 100 edge routers even when the attacker has compromised 500 of the 625 routers in the network. This is due to the fact that BC caching is caching at the node with highest BC making even a successful timing attack highly uncertain about the user's identity or his/her location in the network.

5.2.2 Mobile Case. Table 1 shows the percentage of routes completely tracked by the attacker from start to end. A route is tracked by the attacker if the attacker is able to successfully identify every AP the vehicle has connected to throughout the drive. We present the results for different routes of 10 minutes, 15 minutes and 30 minutes of travel time. With LCE as a caching strategy, we can see that the attacker is able to track 7% of the 10-minute routes with 100 landmarks and this number increases to reach 65% when the attacker has 400 landmarks distributed in the network. With BC caching, the percentage of routes tracked reaches a maximum of 5% even when the attacker has landmarks on all nodes in the network. For routes of longer duration, these numbers are lower since it is harder for the tracker to accurately determine and track a whole route.

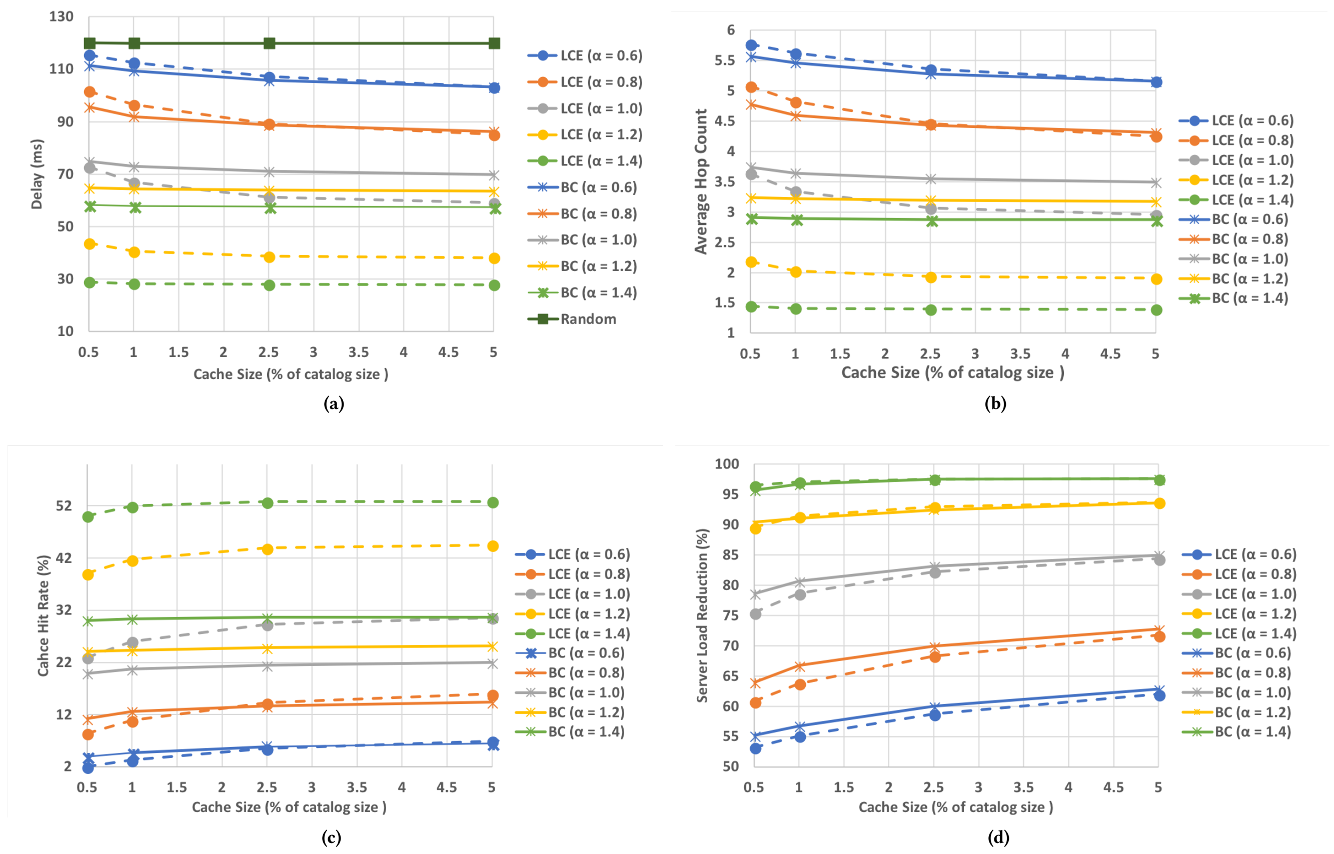

Figure 13: Performance Evaluations for the Tree Topology (M =2, D =5)

Even if the attacker is not able to completely identify the path taken by a target, it might be able to reconstruct the path from partial observations, thus weakening privacy. Table 2 shows the percentage of routes identified by the adversary, i.e., the percentage of APs identified by the attacker from the total number of APs the vehicle connected to. With LCE, an adversary with 100 landmarks is able to, on average, identify around 37% of the total route. The percentage increases to 92% for an adversary with 400 landmarks in the network. With BC caching, an adversary is able to identify only 23% of a route with 400 landmarks in the network.

Table 3 shows that the average number of possible paths produced by the attacker is much lower when LCE is used as a caching strategy in comparison to BC caching. This indicates that an attacker's uncertainty in identifying the exact route taken by the target is higher with BC caching. This is due to the fact that, with BC caching, even when the attacker is able to probe the caching node, that node will be on the content delivery path of several APs.

5.3 Performance Evaluation

We have shown that caching sensitive content on nodes of higher BC provides better privacy as it puts consumers of that content in larger ASs. In this section, we will evaluate the performance of BC-based caching in comparison to LCE and random caching. For the latency results, we will also include cache masking, as proposed in [3], where routers, even in the event of a cache hit, add a delay equal to that of retrieving the data from the original data source. We do not include the results of cache masking for other metrics because they are the same as LCE except that random delay is added to mask a cache hit. We will evaluate the three caching strategies using ndnsim [4, 16] for a tree topology and the AT&T topology with varying values of α and cache sizes at the routers.

The metrics we are interested in and will use to evaluate the caching strategies are:

(1) **Latency**: The time between when the *Interest* packet was sent and when the content was received.
(2) **Average Cache Hit Rate**: The average of the ratio of number of cache hits at the router to the total number of interests received.
(3) **Server Load Reduction**: The ratio of the number of interests satisfied by intermediate router caches to the total number of requests sent out by the consumers.
(4) **Average Hop Count**: The average of the number of hops traversed by the *Interest* packet before it reaches a router with the data in its cache or the server otherwise.

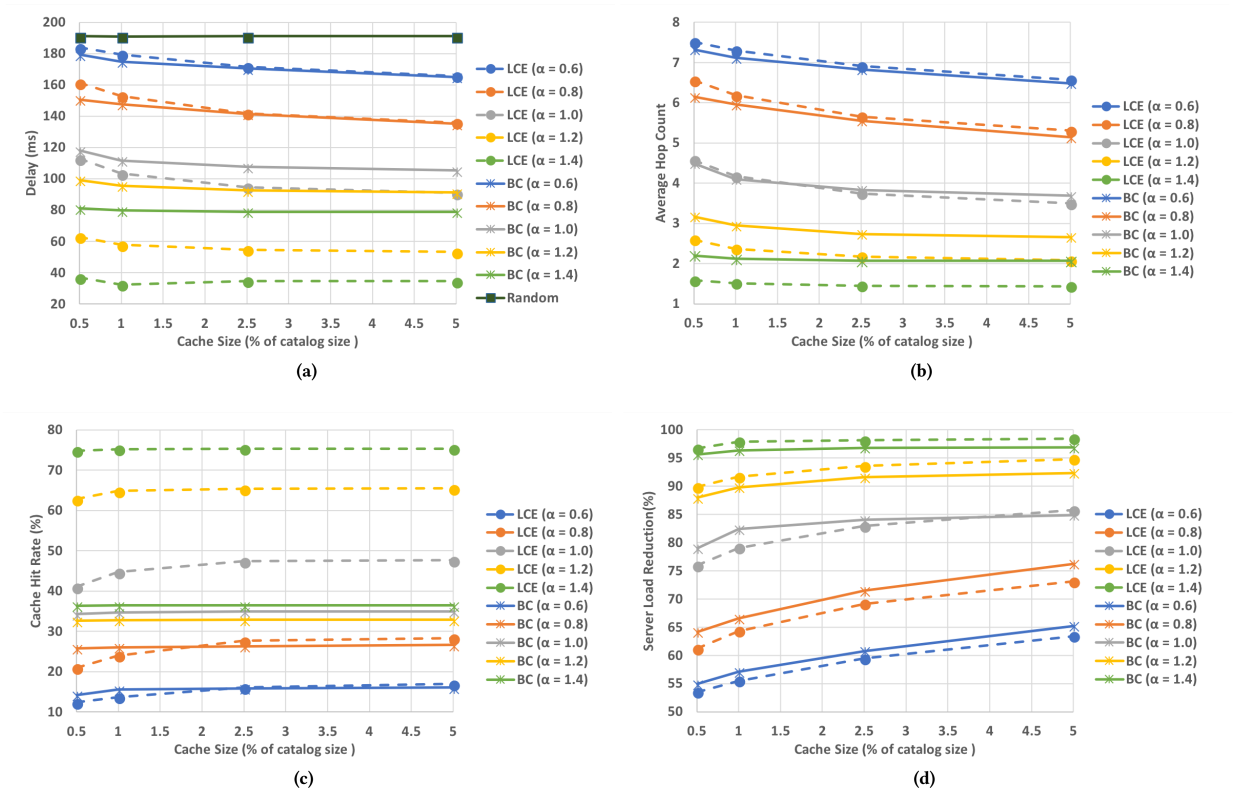

Figure 14: Performance Evaluations for the AT&T Topology

5.3.1 Tree Topology. We first show the results for a tree topology with $M = 2$ and $D = 4$. Fig. 13a shows that for α values of 0.6 and 0.8, LCE and BC caching exhibit a very similar latency for content retrieval. For higher values of α, while both exhibit lower delays, LCE starts outperforming BC caching and the gap increases to reach a 2x lower latency for $\alpha = 0.4$. This is explained by the fact that a higher value of α implies a fewer number of popular content objects which are cached at all the edge routers with extra space used for less popular (more sensitive) objects , which with LCE are also cached closer to the consumers ;whereas, with BC caching those objects are cached higher up the tree. A similar trend in average hop count and average cache hit rate is observed in figures 13c and 13b respectively. It is worth noting; however, that there is no such gap in server load reduction. BC is able to reduce up to 97% of the load on the content provider as shown in Fig. 13d. Overall, the performance penalty introduced by privacy preserving BC caching is limited.

5.3.2 AT&T Topology. We also run our simulations with the AT&T topology of Fig. 14. The trends exhibited are similar to those described earlier. For values of α = 0.6, 0.8 and 1, LCE and BC have similar latency, average hit rate and average hop count. LCE outperforms BC for higher values of α, namely $\alpha = 1.2$ and $\alpha = 1.4$. However, similar to the tree topology evaluation, both caching policies reduce the load to the content provider equally, to reach a 98% reduction when $\alpha = 1.4$.

6 CONCLUSION

A "cache everything everywhere" caching scheme like LCE renders an ICN vulnerable to timing attacks as it leaves communication traces on all nodes in the network including edge routers directly connected to consumers. In this work, we take a content placement approach to address this vulnerability and proposed a BC-based caching policy that caches sensitive content at nodes with higher BC. We showed that, in structured networks, caching content on nodes with higher BC puts consumers of that content in a larger AS, thus giving them more privacy protection against attackers. Our results also showed that a less ubiquitous mode of selective caching is necessary but not sufficient since a policy like random caching, where only one node on the content delivery path caches content, was not able to maintain a large ASsize for an attacker with a large number of compromised routers. BC caching, on the other hand, maintains a large AS as the number of routers controlled by the attacker increases in the network. We also consider more sophisticated attackers capable of locating and tracking mobile users and show that with BC caching, an attacker is not able to track user mobility. Finally, we showed that, in the overall, the performance penalty introduced by privacy preserving BC is limited as compared to LCE.

REFERENCES

[1] Noor Abani and Mario Gerla. 2016. Centrality-based caching for privacy in Information-Centric Networks. In *Military Communications Conference, MILCOM 2016-2016 IEEE*. IEEE, 1249–1254.

[2] Eslam G AbdAllah, Hossam S Hassanein, and Mohammad Zulkernine. 2015. A Survey of Security Attacks in Information-Centric Networking. *Communications Surveys & Tutorials, IEEE* 17, 3 (2015), 1441–1454.

[3] Gergely Acs, Marco Conti, Paolo Gasti, Cesar Ghali, and Gene Tsudik. 2013. Cache privacy in named-data networking. In *Distributed Computing Systems (ICDCS), 2013 IEEE 33rd International Conference on*. IEEE, 41–51.

[4] Alexander Afanasyev, Ilya Moiseenko, and Lixia Zhang. 2012. *ndnSIM: NDN simulator for NS-3.* Technical Report NDN-0005. http://named-data.net/techreports.html

[5] Albert-László Barabási and Réka Albert. 1999. Emergence of scaling in random networks. *science* 286, 5439 (1999), 509–512.

[6] Ulrik Brandes. 2001. A faster algorithm for betweenness centrality*. *Journal of mathematical sociology* 25, 2 (2001), 163–177.

[7] Abdelberi Chaabane, Emiliano De Cristofaro, Mohamed Ali Kaafar, and Ersin Uzun. 2013. Privacy in content-oriented networking: Threats and countermeasures. *ACM SIGCOMM Computer Communication Review* 43, 3 (2013), 25–33.

[8] Alberto Compagno, Mauro Conti, Paolo Gasti, Luigi Vincenzo Mancini, and Gene Tsudik. 2015. Violating consumer anonymity: Geo-locating nodes in named data networking. In *International Conference on Applied Cryptography and Network Security*. Springer, 243–262.

[9] Vladimir Dimitrov and Ventzislav Koptchev. 2010. PSIRP Project – Publish-subscribe Internet Routing Paradigm: New Ideas for Future Internet. In *Proceedings of CompSysTech*. ACM, NY, USA.

[10] Martin Everett and Stephen P Borgatti. 2005. Ego network betweenness. *Social networks* 27, 1 (2005), 31–38.

[11] Linton C Freeman. 1977. A set of measures of centrality based on betweenness. *Sociometry* (1977), 35–41.

[12] Van Jacobson, Diana K. Smetters, James D. Thornton, Michael F. Plass, Nicholas H. Briggs, and Rebecca L. Braynard. 2009. Networking Named Content. In *International Conference on Emerging Networking Experiments and Technologies (CoNEXT)*. ACM, NY, USA.

[13] Teemu Koponen, Mohit Chawla, Byung-Gon Chun, Andrey Ermolinskiy, Kye Hyun Kim, Scott Shenker, and Ion Stoica. 2007. A Data-oriented (and Beyond) Network Architecture. In *(SIGCOMM)*. ACM, NY, USA.

[14] Tobias Lauinger, Nikolaos Laoutaris, Pablo Rodriguez, Thorsten Strufe, Ernst Biersack, and Engin Kirda. 2012. Privacy risks in named data networking: what is the cost of performance? *ACM SIGCOMM Computer Communication Review* 42, 5 (2012), 54–57.

[15] James MacQueen et al. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, Vol. 1. Oakland, CA, USA., 281–297.

[16] Spyridon Mastorakis, Alexander Afanasyev, Ilya Moiseenko, and Lixia Zhang. 2015. *ndnSIM 2.0: A new version of the NDN simulator for NS-3.* Technical Report NDN-0028. NDN.

[17] Aziz Mohaisen, Hesham Mekky, Xinwen Zhang, Haiyong Xie, and Yongdae Kim. 2015. Timing attacks on access privacy in information centric networks and countermeasures. *Dependable and Secure Computing, IEEE Transactions on* 12, 6 (2015), 675–687.

[18] Abedelaziz Mohaisen, Xinwen Zhang, Max Schuchard, Haiyong Xie, and Yongdae Kim. [n. d.]. POSTER: Protecting Access Privacy of Cached Contents in Information Centric Networks. ([n. d.]).

[19] Andreas Pfitzmann and Marit Köhntopp. 2001. Anonymity, unobservability, and pseudonymity?a proposal for terminology. In *Designing privacy enhancing technologies*. Springer, 1–9.

[20] Michal Piorkowski, Natasa Sarafijanovoc-Djukic, and Matthias Grossglauser. 2009. A Parsimonious Model of Mobile Partitioned Networks with Clustering. In *The First International Conference on COMmunication Systems and NETworkS (COMSNETS)*. http://www.comsnets.org

[21] Neil Spring, Ratul Mahajan, and David Wetherall. 2002. Measuring ISP topologies with Rocketfuel. *ACM SIGCOMM Computer Communication Review* 32, 4 (2002), 133–145.

[22] Reza Tourani, Travis Mick, Satyajayant Misra, and Gaurav Panwar. 2016. Security, Privacy, and Access Control in Information-Centric Networking: A Survey. *arXiv preprint arXiv:1603.03409* (2016).

[23] Lixia Zhang, Alexander Afanasyev, Jeffrey Burke, Van Jacobson, kc claffy, Patrick Crowley, Christos Papadopoulos, Lan Wang, and Beichuan Zhang. 2014. Named Data Networking. *SIGCOMM* (Jul. 2014).