Violating Consumer Anonymity: Geo-locating Nodes in Named Data Networking

Alberto Compagno[1], Mauro Conti[2], Paolo Gasti[3], Luigi Vincenzo Mancini[1], and Gene Tsudik[4]

[1] Sapienza University of Rome, Rome, Italy,
{compagno,mancini}@di.uniroma1.it
[2] University of Padua, Padua, Italy,
conti@math.unipd.it
[3] New York Institute of Technology, New York, NY, USA,
pgasti@nyit.edu
[4] University of California, Irvine, CA, USA,
gts@ics.uci.edu

Abstract. Named Data Networking (NDN) is an instance of information-centric network architecture designed as a candidate replacement for the current IP-based Internet. It emphasizes efficient content distribution, achieved via in-network caching and collapsing of closely-spaced content requests. NDN also offers strong security and explicitly decouples content from entities that distribute it. NDN is widely assumed to provide better privacy than IP, mainly because NDN packets lack source and destination addresses. In this paper, we show that this assumption does not hold in practice. In particular, we present several algorithms that help locate consumers by taking advantage of NDN router-side content caching. We use simulations to evaluate these algorithms on a large and realistic topology, and validate the results on the official NDN testbed. Beyond locating consumers, proposed techniques can also be used to detect eavesdroppers.

Keywords: Name Data Networking; Geolocation; Privacy

1 Introduction

Despite its impressive longevity, popularity and overall success, the Internet is starting to suffer from limitations of its original early 1980-s design. Current protocols (in particular, IP) were conceived when remote login, email and resource sharing were the most prominent Internet use-cases. However, a significant fraction of today's Internet traffic corresponds to content distribution. Recognizing this paradigm shift in the nature of Internet traffic, multiple large-scale research efforts [5], [19,20], [22], [32] have been trying – in the last 5-6 years – to address the shortcomings of the current Internet, with the long-term goal of replacing it with a next-generation Internet architecture. One such effort is Named Data Networking (NDN) [15].

NDN is an example of Content-Centric Networking (CCN), where content – rather than a host or an interface – plays the central role in the architecture. NDN is primarily oriented towards efficient large-scale data distribution. Rather than contacting a host at some IP address in order to request data, an NDN *consumer* directly requests desired content by name by issuing an *interest packet.* The network takes care of finding and returning the nearest copy of requested content that "satisfies" the consumer's interest. To this end, NDN features *ubiquitous content caching*, i.e., any host or router can store a copy of the content it receives or forwards, and use it to satisfy subsequent interests. NDN also provides *interest collapsing*, i.e., only the first of multiple *closely spaced* interests for the same content is forwarded by each router. Unlike IP datagrams, NDN interests and content packets do not carry source or destination addresses. One of the alleged consequences of this feature is *consumer location privacy.* In this paper we show that two fundamental NDN features (ubiquitous caching and interest collapsing) can be used to violate consumer location privacy. Specifically, we show how information leaked by caching and interest collapsing can be used to identify and locate consumers.

Assuming that the adversary can associate NDN routers with their physical location using existing methods, we focus on designing techniques that identify the router closest to the targeted consumer. We then show that proposed techniques can be used to determine consumers' location, as well as detect "eavesdroppers" that are surreptitiously requesting content for a particular set of users, e.g., in audio/video conferencing applications [14], [33]. We validate our results via experiments on the official NDN testbed [21]. Finally, we propose some countermeasure that mitigate these attacks.

We believe that this topic is both timely and important, since one of the key design goals of NDN is *security by design.* This is in contrast with today's Internet where security and privacy problems were (and are still being) identified and patched along the way. Therefore, assessing *if* and *how* geo-location and eavesdroppers identification can be implemented must be done *before* NDN is fully deployed. Furthermore, even though the research community has made significant efforts in geo-locating hosts in the current Internet [9], [13], [16,17], [23,24], [29,30,31], none of these techniques apply to locating consumers in NDN. (See Section 3.) In fact, to the best of our knowledge, all prior techniques rely on the ability to directly address the victim host. This is not possible in NDN since consumers cannot be contacted directly.

Organization: We start by overviewing the NDN architecture in Section 2. Related work is discussed in Section 3. Section 4 introduces our system and adversary models. Proposed techniques are presented in Section 5 and evaluated in Section 6. Detection of eavesdroppers is addressed in Section 7. Finally, geo-location countermeasures are presented in Section 8. We conclude in Section 9.

2 NDN Overview

NDN supports two types of packets: *interest* and *content* [4]. Notable fields in content packets are: (1) content name, (2) payload, and (3) digital signature computed by the producer. Names are intended to be human-readable, consisting of one or more components with a hierarchical structure. In NDN notation, "`/`" separates name components, e.g., `/ndn/cnn/politics`.

Consumers request desired content by name, via interests. NDN routers forward interests towards the content producer responsible for the requested name, using longest name-prefix matching for routing. If the requested content is not encountered in caches of any intervening routers, the interest eventually arrives to the producer. Upon receipt of the interest, the producer injects the content into the network, thus *satisfying* the interest. The requested content packet is then forwarded towards the consumer, traversing – in reverse – the path of the preceding interest.

Each NDN router maintains three data structures: (1) Pending Interest Table (PIT) storing interests that are not yet satisfied, (2) Forwarding Interest Base (FIB) containing routing information, and (3) Content Store (CS) where forwarded content is cached. When an NDN router receives an interest, it first looks up its PIT to check whether another interest for the same name is currently pending. There are two possible outcomes:

1. The PIT look-up succeeds, i.e., PIT entry for the same name exists and:
   - The incoming interface of the present interest is new, the router updates the PIT entry by adding the new interface to *arrival-interfaces* set. The interest is not forwarded further. This feature is called *interest collapsing.*
   - The present interest's incoming interface is already in the set of that entry's *arrival-interfaces.* In this case, the interest is simply discarded.
2. The PIT look-up fails. The router performs local cache look for the content name referenced in the interest, and:
   - The cache look-up succeeds. The content is returned on the arriving interface of the interest and no further actions are taken.
   - Otherwise, the router creates a new PIT entry and forwards the present interest out on one or more interfaces, according to its FIB.

However, note that caching of content in routers is not mandatory. Although each NDN router is expected to cache content, it is not required to do so. A router can choose whether to cache a given content based on local criteria, such as: size and occupancy rate of its cache, content name, as well as consumer or producer wishes, i.e., the interest might request caching or no caching, or the content itself might convey caching preferences.

3 Related Work

The goal of current geo-location techniques is to associate a physical location with a particular IP address. There are many studies that investigate geo-location in today's Internet [9], [13], [16,17], [23,24], [30,31].

Prior work can be divided in two classes: *measurement-based* and *database-driven* techniques. The former involve a set of geographically distributed *landmark* hosts with known locations. Their purpose is to determine the position of the target IP address using round-trip time (RTT) information as the basis for triangulation. An algorithm estimates the location of the target IP using historical data constructed using ground truth [13]. Multiple techniques can then be used to improve accuracy. For example, Wong et al. [30,31] combine delay measurements with locations of cities. [31] uses Bézier curves to represent a region containing the target IP, while [30] leverage a three-tier approach, where every tier refines results of the previous one. Finally, Eriksson et al. [9] propose a learning-based approach, where population density is used to construct a Naïve Bayes estimator.

All these techniques assume that, packets sent to a particular IP address and echoed back (e.g., via `ping`) are guaranteed to come from the same physical host. Therefore, multiple RTT measurements correspond to the same target. In contrast, requesting multiple NDN content packets created by the same producer does not guarantee that requested content will be found at the same place. Because of in-network caching, different content packets might be served by distinct entities. Thus, RTT measurements obtained by the landmarks can refer to different nodes, and cannot be immediately used to locate a single target.

Database-driven approaches determine the target IP's location using DNS LOC records, `WhoIs` lookups, Border Gateway Protocol (BGP) router tables, and/or other public databases (e.g., ARIN [3], RIPE [25], GeoTrace [12] and MaxMind GeoIP [11]). These resources either provide direct geographic information, as in DNS LOC, or reveal indirect clues, such as the organization or Autonomous System (AS) number that owns a particular IP address. For example, techniques like GTrace [24], GeoTrack and GeoCluster [23] use these public resources to locate the target IP, and then further refine the findings using RTT measurements. Recent work by Liu et al. [17] utilizes location data that users willingly disclose via location-sharing services. This technique can locate a host with a median estimation error of 799 meters – an order of magnitude better than other approaches.

Because NDN consumers have no network-layer addresses, current geo-location techniques are not directly applicable. However, it is possible to use current techniques to locate content producers. Although there are no addresses that can identify hosts in NDN, name-spaces can serve the same purpose. In fact, all producers publishing within specific namespaces (e.g., `/cnn/`, or `/microsoft/`) might be naturally located within the same Autonomous System (AS). Name prefixes could thus reveal location information. Similarly, routing tables can reveal location information for name-spaces. Although, at this stage, there are no location databases for NDN, it is not hard to anticipate these resources becoming available if and when NDN becomes wider deployed.

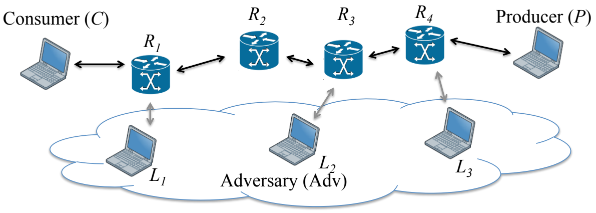


Fig. 1. Scenario considered throughout the paper.

4 System and Adversary Model

In the rest of the paper we consider the scenario illustrated in Figure 1. A consumer (C) retrieves content, composed of multiple packets, from a producer (P). We focus on the case where C requests *non-popular* content, i.e., content that is unlikely to have been recently requested by others. Thus, it is not cached in relevant routers. Each interest traverses multiple routers before being satisfied by P. The adversary (Adv) controls a set of hosts (hereafter called *landmarks*), connected to NDN routers. These hosts controlled by Adv have no special privileges and cannot eavesdrop on links between routers. We denote router i as R_i and landmark j as L_j. Adv's goal is to determine C's location in the network, i.e., identify the router closest to C.

4.1 System Model

We represent network topology as a undirected graph $G = \langle V, E \rangle$, where V is the set of vertices (routers) and E is the set of edges (links between routers). Our experiments on the official NDN testbed (see Section 6) show that NDN links are largely symmetric, i.e., bandwidth and delay are the same in either direction. For this reason, our system model also considers all links to be symmetric.

We performed experiments on the AT&T topology from Rocketfuel [26], depicted in Figure 2. It contains 625 vertexes and 2101 edges. In the experiments we assume that every router caches content packets, which is the default NDN setting. However, because NDN does not mandate a specific caching policy, we also discuss how to apply geolocation techniques when some (or all) routers do not cache content packets (see Section 5).

4.2 Adversary Model

We assume that C requests – and Adv can exploit – a large number of data packets, possibly corresponding to a single piece of content, e.g., a high-resolution video. We consider two distinct classes of adversaries: *outsiders* and *insiders*. The former cannot directly (passively) monitor packets exchanged between P and C. We assume that an outsider knows what type of applications C and P are using. Therefore it might infer the structure and naming of content packets.

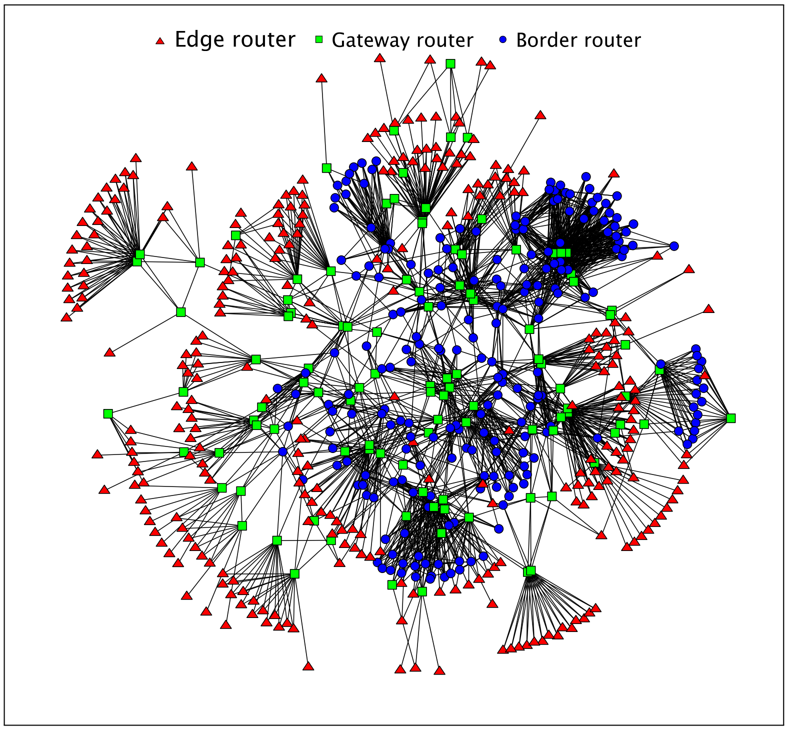


Fig. 2. AT&T topology.

However, if unique/secret naming is negotiated between P and C, outsiders cannot guess content names. Insiders can observe packets exchanged by P and C. For example, an insider could be a compromised WiFi access point to which C is directly connected, or a malicious P. Thus, countermeasures such as content name randomization are not effective.

Our analysis makes the following assumptions:

1. **Topology Information:** *Adv* knows the topology and geographic distribution of routers. Today, some AS-s already publish this information [26]. Moreover, it has been shown that it is possible to reconstruct topology even if this information is not publicly available [7].
2. **Routing Information:** *Adv* is aware of how interests are routed. Given the sheer number of routers and AS-s involved in today's Internet routing, it is unlikely that routing information can be kept secret.
3. **Distance from Sources:** *Adv* can determine the distance of a content packet (expressed in terms of number of hops) from its closest source (e.g., a cache) using Content Fetch Time (CFT) information, i.e., the time between sending an interest and receiving the related content. Our experiments on the official NDN testbed [21], reported in Section 6, confirm that this is indeed currently possible.

4. **Naming Information:** *Adv* can predict the name of content packets requested by C. As mentioned earlier, insiders and outsiders have different capabilities.
5. **Arbitrary Landmark Location:** *Adv* can connect landmarks to arbitrary routers in the network. For example, it can use a geographically distributed botnet, or purchase resources from (multiple) cloud services with machines located in different parts of the world. We allow *Adv* to select landmarks *adaptively* (i.e., the next landmark is selected after gathering information from all current landmarks) or *non-adaptively*, meaning that all landmarks are chosen at once.
6. **Upper-bound on Landmarks:** *Adv* can compromise (or purchase) up to a fixed subset of nodes in a given topology, in order to turn them into landmarks.

We refer to *Adv* with all aforementioned capabilities as *routing-aware*. As an extension, we later consider a variant *Adv* that has no knowledge of routing information. We call it *non-routing-aware Adv*.

5 Locating Consumers in NDN

To locate C, *Adv* requests cached content previously requested by C from multiple landmarks L_i. Each landmark measures CFT for each content. Since content is cached (and therefore served) by some router on the return path between C to P ($P \rightarrow C$ from here on), landmarks might learn some information about $P \rightarrow C$. Hence, *Adv* can use this information to infer the location of C.

If no intervening router caches content, *Adv* can use NDN interest collapsing feature to locate C. For the sake of simplicity, and without loss of generality, we describe *Adv*'s steps to locate a specific R_i.

Recall that, as an interest traverses routers on the path from C to P, it creates state in the form of a PIT entry. After receiving the interest, P injects requested content into the network. As the content travels back towards C, each router that forwards it flushes the corresponding PIT entry for that content. However, if an interest from a landmark reaches R_i before the corresponding PIT entry is flushed, (i.e., before the content packet requested by C arrives), the CFT measured by the landmark will be lower than the CFT for content fetched from P. This is due to interest collapsing: the landmark's interest is not forwarded by R_i since an entry for previously pending interest (for the same content) already exists in R_i's PIT. As shown in [2], this CFT difference can be easily identified by the landmark. In practice, different routers will adhere to different caching strategies. Thus, while some routers might cache all packets, others will not. Therefore, each landmark might have to probe either PIT-s, or CS-s, or both.

Regardless of caching, *Adv* can only retrieve content previously requested by C from routers, and not from C itself. *Adv*'s interests are routed toward P, and can reach C only if C is on a path $Adv \rightarrow P$. However, because C is a host and not a router, it is never part of $Adv \rightarrow P$. For this reason, we define *Adv*'s goal as identifying C's first-hop router. This allows *Adv* to accurately pinpoint the C's location, e.g., possibly within a few blocks in a densely populated city. Moreover, compared to expected errors in current geo-location techniques (on the order of 10km using state-of-the-art [17]), identifying an edge router instead of an end-host introduces only negligible errors. For this reason, in the rest of the paper, we use C to indicate the edge router closest to the actual consumer.

Routing-Aware Adversary. Knowledge of network topology and all routing tables allows landmarks to identify the source of content packets via CFT measurements. This information reveals how far the content travels in the network to reach the landmark. Given this distance, as well as topology and routing information, *Adv* can determine which router served the content. Listing 1.1 describes the steps *Adv* performs to identify C. For each L_i, *Adv* calculates path $L_i \to P$ and measures the number of hops (i.e., $hops_{L_i}$) between L_i and the cache serving the content (see lines 6-10, Listing 1.1). Then, *Adv* identifies the router at position $hops_{L_i}$ in the path $L_i \to P$ as a router on $P \to C$. N_C represents the set of candidate nodes for C (lines 11-15).

Intuitively, location of landmarks with respect to routers on $P \to C$ path affects the precision of locating C. In non-adaptive selection, *Adv* randomly selects all landmarks at once. In the adaptive case, landmark selection is performed as follows. Let R_g be a router identified by *Adv* as part of the path $P \to C$. To find the next router on the path, *Adv* selects a L_i that is far away from P, such that the path from L_i to P contains R_g (i.e., $L_i \to P = L_i \to R_g \to P$). Thus, if L_i retrieves content cached by router $R_i \neq R_g$, then R_i must (1) be on $P \to C$, and (2) be $n \geq 1$ hops closer to C compared to R_g. The larger n, the fewer landmarks are required to identify C. This process is repeated until no new landmarks are able to discover routers closer to C, or if *Adv* reached its maximum number of landmarks.

Listing 1.1. GuessPath - Routing Aware Adversary.

```
1   Input: $G$; $P$; landmarks $L$; gateway routers; edge routers
2   Output: $N_{Path}$ (nodes believed to be part of the path $P \to C$);
3            $N_C$ (nodes believed to include $C$)
4   $N_{Path} \leftarrow P$
5   $N_C \leftarrow \emptyset$
6   for each available landmark $L_i$ {
7            $path_{L_i} \leftarrow$ calculate path $L_i \to P$, ordered from $L_i$ to $P$
8            $hops_{L_i} \leftarrow$ number of hops measured when retrieving from $L_i$
9            $N_{Path} \leftarrow N_{Path} \cup$ {element at position $hops_{Li}$ in $path_{Li}$}
10  }
11  for each $n$, s.t. $n$ in $N_{Path}$, and $n$ is a gateway router {
12           for each $\bar{n}$, s.t. $\bar{n}$ is an edge router, and $\bar{n}$ is connected to $n$ {
13                    $N_C \leftarrow \bar{n}$
14           }
15  }
```

Non-Routing-Aware Adversary. The non-routing-aware adversary has no knowledge of the content of routing tables. Without this knowledge, measuring distances between the caches satisfying the landmarks' interests and the landmarks does not provide as much information as in the case of routing-aware adversaries. In fact, given a distance, *Adv* can identify a *set* of caching routers that contains the one serving her requests, instead of a single router. In this case the *Adv*'s strategy includes three phases: *Phase 1*: collecting information from landmarks to assign a *score* to each node, *Phase 2*: using scores to determine routers that are likely in the path; and *Phase 3:* further refining the selection. Pseudocode for the three phases is reported in Listing 1.2.

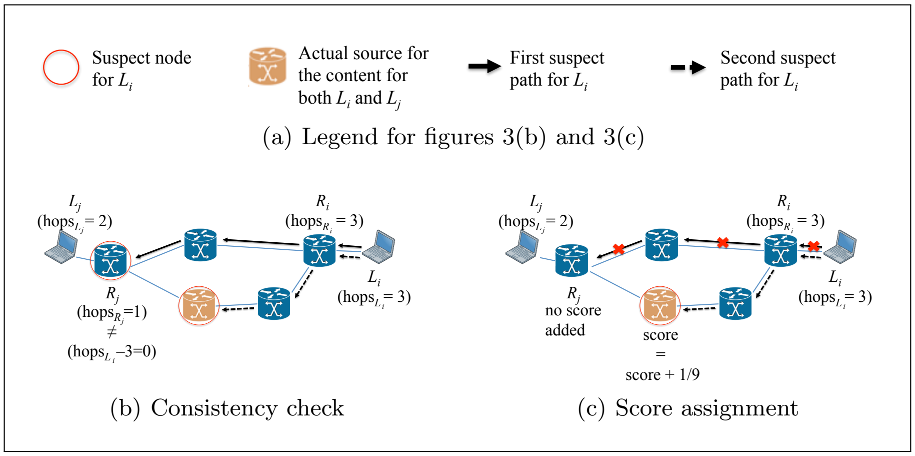
Listing 1.2. GuessPath - Non-Routing Aware Adversary.

```

Input: $G$; $P$; landmarks $L$; $threshold$; $numberOfComp$;
              gateway routers; edge routers
Output: $N_{Path}$ (nodes believed to be part of the path
              $P \to C$); $N_C$ (nodes believed to include $C$)
$N_{Path} \leftarrow P$, $N_C \leftarrow \emptyset$
for each landmark $L_i$ {
    $R_i \leftarrow$ router at one hop from $L_i$
}
PHASE 1
for i = 1 to size($L$) {
    $hops_{L_i} \leftarrow$ number of hops measured when retrieving from $L_i$
    $hops_{R_i} \leftarrow hops_{L_i} - 1$
    $suspectNodes_{L_i} \leftarrow$ all nodes $n_{L_i}$ at distance $hops_{L_i}$ from $L_i$
    $suspectPaths_{L_i} \leftarrow$ all possible paths to reach nodes
              $suspectNodes_{L_i}$ from $L_i$
    for each landmark $L_j \neq L_i$ {
        if $\exists$ $spath$ in $suspectPaths_{L_i}$, s.t. $R_j$ is in $spath$ {
            $hops_{L_j} \leftarrow$ number of hops measured when
                  retrieving from $L_j$
            if (($hops_{R_j}$) $\neq hops_{L_i} -$ (position of $R_j$ in $spath$)){
              remove $spath$ from $suspectPaths_{L_i}$
            }
        }
    }
    for each $spath$ in $suspectPaths_{L_i}$ {
        $n$ = node at position $hops_{L_i}$ in $spath$
        $Score_n = Score_n + 1/(hops_{L_i})^2$
    }
}
PHASE 2
for each $n$ in $V$ {
    if ($Score_n > threshold$) {
        $N'_{Path} \leftarrow n$
    }
}
PHASE 3
$N_{Path} \leftarrow getConnComp(N'_{Path}, numberOfComp)$
for each $n$ in $N_{Path}$ and $n$ is a gateway router {
```

```
40          for each $\bar{n}$ is an edge router and $\bar{n}$ connected to $n$ {
41                $N_C \leftarrow \bar{n}$
42          }
43    }
```

Phase 1 is based on two observations. First, estimation done independently by each landmark L_i (i.e., suspect nodes computed in line 13 in Listing 1.2) could be partially incorrect. Because L_i does not have access to routing information, it might include routers that are not on $P \rightarrow C$. However, estimates from different landmarks can be checked against each other for consistency: nodes that are not consistently considered as potential routers in the path from C to P will be assigned a zero score. This consistency check (lines 16-24 in Listing 1.2) is motivated as follows. Because each landmark L_i is connected to just one router R_i, learning the number of hops from L_i to the source also implies learning the distance from R_i to the source of the content. Moreover, because routing information is not available to Adv, every path from L_i to a "suspect" node is a candidate (suspect) path. Let us consider the situation in Figure 3(b) where R_j, one hop away from L_j, belongs to a suspect path for L_i. In this case, distance measured by L_i and L_j for R_j must be the same. If two distances differ, the suspect path for L_i is considered incorrect and no score is added to the suspect node, as shown in Figure 3(c).



(a) Legend for figures 3(b) and 3(c)

(b) Consistency check

(c) Score assignment

Fig. 3. Non-routing aware – Phase 1.

The second observation is used to add a score to the nodes selected as possible candidates to be on the $P \rightarrow C$ path (denoted hereafter by N_{Path}). In this case, the closer L_i is to N_{Path}, the more specific is the information provided by L_i. For example, if we connect L_i to a node in this path, L_i could identify the source without error – the content will be retrieved in zero hops, i.e., from the same router to which L_i is connected. Instead, if we connect L_i at a certain distance (denoted as $hops_{L_i}$) from a node on N_{Path}, L_i will consider any node that is $hops_{L_i}$-hops away from itself as a possible node in N_{Path}. As a consequence, the greater $hops_{L_i}$, the higher is the number of candidate nodes; thus, errors are more likely. In Listing 1.2, this observation is reflected in line 27 where $1/(hops_{L_i})^2$ is used to assign a score to the nodes. The intuition behind this assignment has a geometric explanation. Considering the selected node L_i as the center of a sphere and the distance $hops_{L_i}$ as the radius, the area of the sphere is a good estimator of the number of candidate nodes.

Phase 2 uses the scores provided in Phase 1 to select a number of nodes as sources of content packets. In this case, we select the nodes that exceed a predefined threshold as possible sources.

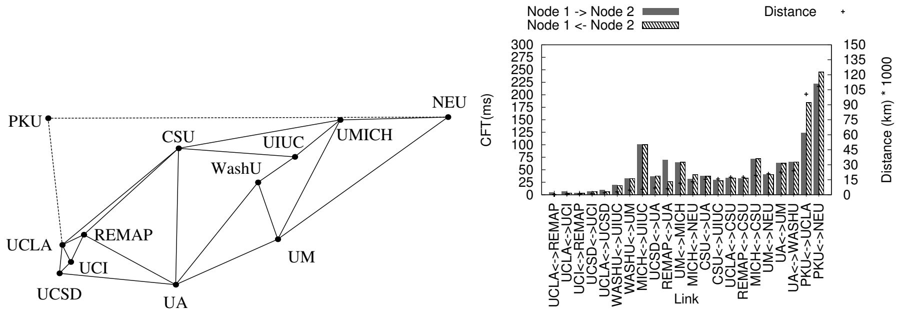
Phase 3 further refines node selection. We use the set of selected nodes from Phase 2 to create a subgraph of G. Then, we compute connected components in this new graph and we order them from the closest to the farthest from the producer. We consider the distance from a component $ConnComp[i]$ to the producer as the distance, computed in graph G, from the closest node of $ConnComp[i]$ to the producer. Therefore, Adv assumes that the nodes from $ConnComp[0]$ to $ConnComp[k-1]$ are in the path $P \rightarrow C$. Finally, we consider all edge nodes connected to gateway nodes in N_{Path} as the nodes that include C.

Landmarks are selected to minimize the difference between: (i) the score assigned to the new landmark by the previous selection step, and (ii) the average score.

6 Evaluation

In the current Internet, the relationship between RTT and distance measured in hops is subject to variation of the triangle inequality. Such variations make RTT-based distance estimation unreliable [18]. We studied this phenomenon on the NDN tested, and we evaluate how it affects the attacks discussed in this paper. To this end, we used Amazon Elastic Compute Cloud (EC2) [8] virtual machine instances. Each EC2 instance was connected to the testbed at a different router, and was used to either publish or request content. We performed exhaustive tests, including producer/consumer combinations. Figure 4(b) summarizes our findings. It also shows approximate physical straight-line distance between NDN nodes. Reported CFT is obtained after subtracting the CFT between the EC2 instance acting as C and the first-hop router. Our experiments confirm that: (1) links between routers are symmetric in terms of bandwidth and delay, except as discussed below; (2) triangle inequality violations only add a small amount of noise to distance estimation. CFT is symmetric for every link except for UA-REMAP, PKU-UCLA and PKU-NEU. In the first case, asymmetry is due to the paths UA→REMAP vs REMAP→CSU→UA. We consider asymmetry in PKU-NEU and PKU-UCLA links to be an artifact of the current NDN testbed, since it is deployed as an IP overlay, and not a property of NDN.

We ran multiple experiments in which we connected P and C to different nodes. For every experiment we measure CFT connecting landmarks to all nodes



(a) Testbed topology [21]

(b) Link Content Fetch Time (CFT)

Fig. 4. The NDN Testbed.

in the testbed. Our measurements reveal that 8% of landmarks provided an incorrect distance, likely due to violation of triangle inequality. Therefore, actual distance measurements on the testbed would be affected by "random noise" with probability 8%.

6.1 Performance of Our Algorithms

To evaluate the effectiveness of our strategies, we defined three metrics, which can be informally summarized as: (a) how effective are our strategies in identifying nodes in the path? (b) Of the selected nodes, how far from C is the closest? (c) How often do our strategies correctly identify C? Although (c) is arguably the most "natural" metric, it is also the one that provides the least amount of information, representing a simple binary outcome (identified/not identified). Therefore, we believe that (a) and (b) complement this metric by providing further details on *how close Adv* is to identifying C.

We express (a) as two quantities: *true positive* (i.e., nodes that have been correctly identified) and *false positives* (nodes that have been erroneously flagged as part of the path):

$$\text{True positive}{=}\frac{\#\text{ of output nodes in the path}}{\#\text{ of total nodes in the path}}$$

$$\text{False positive}{=}\frac{\#\text{ of output nodes not in the path}}{\#\text{ of total nodes not in the path}}$$

We compared our strategy with random guessing. This represents the best adversarial strategy if NDN truly provides consumer anonymity, i.e., if the adversary can gather no information at all about consumers. We model random guessing using the urn model without replacement [10] where the number of draws q is the number of nodes identified by our strategy in the same setting. Let N be the number of nodes in the topology, and m the length of the path $P \rightarrow C$. The probability of choosing j nodes from the path is:

$$\mathbb{P}(j) = \frac{\binom{m}{j}\binom{N-m}{q-j}}{\binom{N}{q}} \tag{1}$$

We calculate *true_pos* for our random strategy as the expected number of nodes chosen from the path, divided by the number of nodes:

$$true\_pos = \frac{\left(\sum_{j=1}^{\min(m,q)} j \cdot \mathbb{P}(j)\right)}{m} \tag{2}$$

Analogously, false positive are calculated as the expected number of incorrectly selected nodes $(q - j)$ divided by the number of nodes:

$$false\_pos = \frac{\left(\sum_{j=0}^{\min(m,q)} (q - j) \cdot \mathbb{P}(j)\right)}{(N - m)} \tag{3}$$

With respect to (b), we select as baseline the average distance to the consumer in the network. In particular, we calculate the average of the distance from every node in the network to the consumer as:

$$avg = \frac{\left(\sum_{i=0}^{N} d\,(i)\right)}{N} \tag{4}$$

where $d(i)$ is the distance of node i from the consumer.

We report results for paths of length 6. This length was selected since it is the most likely distance in several topologies (see Figure 5.)

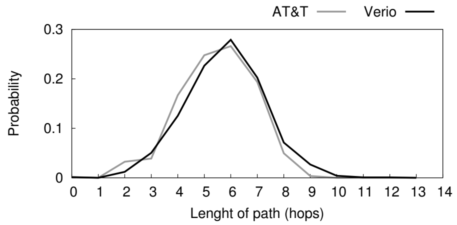


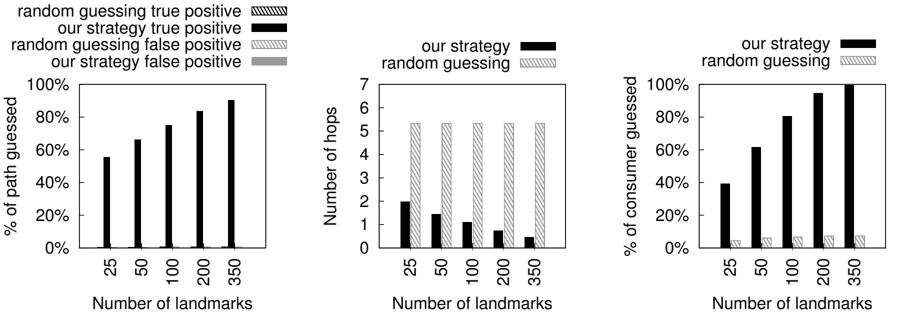
Fig. 5. Probability distribution of path lengths in the AT&T (see Figure 2) and Verio [26] topologies.

Routing-Aware Adversary – Non-Adaptive Landmarks Selection. Results in this configuration for AT&T are reported in figure 6(a). Our technique is able to keep false positive very low due to the availability of routing information.

It is interesting to note that the algorithm is not always able to guess all the nodes in the path, regardless of the number of landmarks used. The reason for this is that, sometimes, a router in the path cannot satisfy any interest from the landmarks because these interests can always be satisfied by other routers.

Figure 6(b) compares our strategy with random guessing. In this case, our guess for C is almost always at most two hops away from C, compared to five hops for random guessing.

Figure 6(c) shows how often our algorithm identifies the consumer. When our strategy is able to identify at least one node one hop away from the consumer node, it always identifies the consumer node. This is the case with 200 and 350 landmarks, where our strategy identifies C in the vast majority of our simulations.

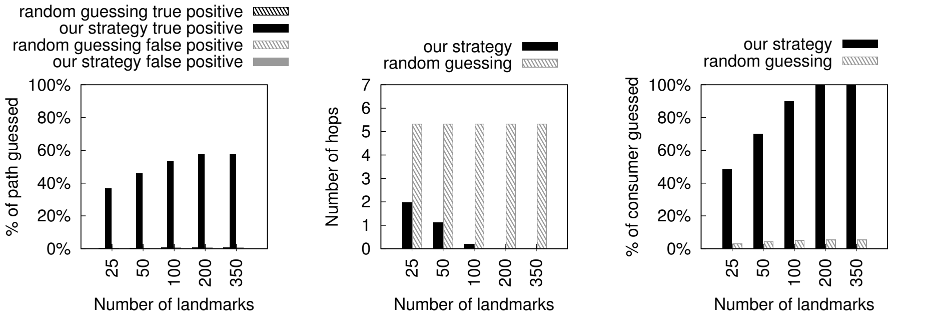


(a) AT&T - guessing path (b) AT&T - node closest to consumer (c) AT&T - guessing consumer

Fig. 6. Routing aware adversary - Non-adaptive landmarks selection.

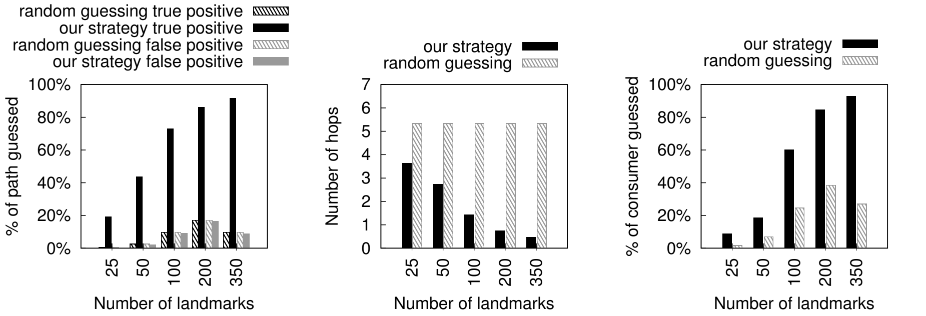
Routing-Aware-Adversary – Adaptive Landmarks Selection. Figure 7 shows the performance of our technique in this scenario. The ability to adaptively select locations within the network allows *Adv* to easily identify C in our topology. Figures 7(b) and 7(c) show that, with 100 landmarks, our algorithm is able to identify C with over 90% probability.

Non-Routing-Aware Adversary – Non-Adaptive Landmarks Selection. Figure 8 shows performance of Listing 1.2 on AT&T with respect to false positives and false negatives. Our experiments were performed with *threshold* and k set respectively to 1.5 and 2. Compared to *routing aware* adversary, the number of false positives is higher. However, overall performance is still good: Figure 8(a) shows that false positives are below 20%. Similarly to the routing-aware case, we are not able to always guess the entire path $P \rightarrow C$, as reported in Figure 8(b). A similar behavior is shown in Figure 8(c).



(a) AT&T - guessing path (b) AT&T - node closest to consumer (c) AT&T - guessing consumer

Fig. 7. Routing aware adversary - Adaptive landmarks selection.



(a) AT&T - guessing path (b) AT&T - node closest to consumer (c) AT&T - guessing consumer

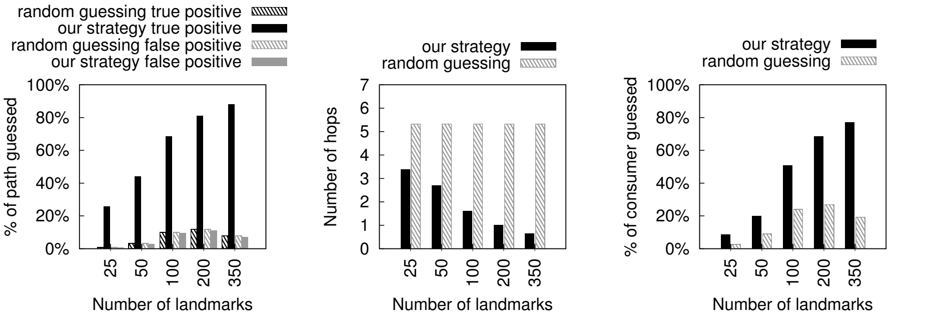
Fig. 8. Non-routing aware adversary - Non-adaptive landmarks selection.

Non-Routing-Aware Adversary – Adaptive Landmarks Selection. Performance of this scenario are reported in Figure 9. Figure 9(a) shows that our algorithm reduces the number of false positives in the AT&T topology. This strategy is able to significantly outperform random guessing strategy (figures 9(b) and 9(c)).

Table 1 summarizes the performance of all our strategies. We report performance of random guessing obtained under the same conditions.

7 Detecting Eavesdroppers

Although C might be the only intended recipient of a set of content packets from P, NDN allows any host to later retrieve these packets from routers' caches and



(a) AT&T - guessing path (b) AT&T - node closest to consumer (c) AT&T - guessing consumer

Fig. 9. Non-routing aware adversary - Adaptive landmarks selection.

Table 1. Performance of our strategies.

		number of landmarks	% of consumer guessed	
			our strategy	random guessing
Non-routing aware	non-adaptive	350	99,3%	7,4%
	adaptive	200	100%	0,5%
Routing aware	non-adaptive	350	93,0%	25,4%
	adaptive	350	77,1%	19,3%

possibly do so without either P or C being able to directly detect this action. This can be seen as an effective means of eavesdropping in NDN: in contrast with "traditional" eavesdropping, this approach does not require privileged access to the networking infrastructure and can be performed independent of the geographic location of Adv with respect to P and C.

One way to detect this type of eavesdropping is by using techniques presented in this paper. For example, P and C could "rent" a set of geographically distributed hosts while they are exchanging content packets. These rented hosts would implement the algorithms discussed in the paper. Eavesdroppers will then be consistently identified as extraneous consumers (other than C), and possibly located. We envision that such a service could be easily offered by companies such as Amazon, Microsoft, or other geographically distributed cloud providers.

8 Discussion of How to Mitigate Geo-location Attacks

One natural approach to prevent aforementioned attacks is to simply disable router content caching. Besides negating one of the main benefits on NDN, efficacy of this countermeasure is limited. In fact, an insider Adv that knows exact timing of interest packets emitted by C can implement PIT-based techniques outlined in [2]. Under normal conditions, Adv has a very small window (a few ms to a few hundreds ms) to extract information from PIT-s on a single packet.

However, it is safe to assume that P and C exchange a large number of content packets. This significantly simplifies the attack. Moreover, an insider Adv could delay injecting content packets into the network upon receiving an interest. This would force interests from C to be stored in all PITs along the path $P \rightarrow C$ for longer, thus further simplifying the attack.

A better approach involves using unpredictable names [1]: P and C can initially agree on a secret seed (e.g., via authenticated Diffie-Hellman key exchange) and use it to generate pseudo-random content names. Since the seed would be known only to the two communicating parties, no outsider can guess content names. Adv therefore cannot request content, which is necessary to locate C. Unfortunately, this solution requires both P and C to be actively engaged in the secret agreement procedure. This could generate a significant (additional) load on P, and will negating the benefit of caching and interest collapsing. Furthermore, this approach is ineffective against insider Adv who knows the seed.

Another approach is to "confuse" Adv by requesting content packets from multiple geographic locations at the same time. Intuitively, since in this case there are multiple consumers, geo-location algorithms would identify many of them with roughly the same probability, offering a weak form of privacy (i.e., k-anonymity [27]) and deniability to C.

To the best of our knowledge, the only approach completely effective against attacks discussed in this paper is the anonymizing network ANDaNA [6]. ANDaNA is an NDN equivalent of Tor [28]. It allows end host to join an anonymizing network as "onion routers", which anonymize consumers' requests. Unfortunately, the additional overhead and latency might be prohibitive for many applications.

9 Conclusion

In-network content caching, a key feature of NDN, has been shown to have unexpected privacy implications [1]. In this paper, we provided another example of how abuse of network state can lead to loss of privacy in NDN. We designed several techniques geared for adversaries with varying capabilities and evaluated proposed techniques via simulations on a realistic network topology. We then used the actual NDN testbed to validate our results.

Experiments show that plausible adversaries can locate consumers with high probability, i.e., over 90% in many scenarios. Furthermore, even adversaries with relatively little knowledge of the network can successfully locate consumers with high probability, albeit, using more resources.

We then discussed several countermeasures, showing that even disabling caches on all routers does not completely prevent this attack. Moreover, the only effective countermeasure we are aware of (ANDaNA) imposes significant overhead on the communicating parties. Finally, we sketched out how the proposed techniques can help identify eavesdroppers in NDN, which is a rather unexpected outcome of router state.

We believe that the impact of our results goes beyond geo-location. NDN has been widely assumed to provide better consumer privacy than the current IP-based Internet due to lack of source/destination addresses. However, this paper casts serious doubt on this belief. Further, we argue that our geo-location techniques apply, to some extent, not only to NDN, but to any network architecture that supports ubiquitous caching.

Acknowledgments

Alberto Compagno and Luigi Vincenzo Mancini have been partially supported by the European Commission Directorate General Home Affairs, under the GAINS project, HOME/2013/CIPS/AG/4000005057, and by the European Commission H2020 SUNFISH project, N. 644666. Mauro Conti is supported by a Marie Curie Fellowship funded by the European Commission under the agreement No. PCIG11-GA-2012-321980. This work is also partially supported by the TENACE PRIN Project 20103P34XC funded by the Italian MIUR, and by the Project "Tackling Mobile Malware with Innovative Machine Learning Techniques" funded by the University of Padua. Gene Tsudik's work was supported by United States National Science Foundation (NSF) under award: CNS-1040802: FIA: Collaborative Research: Named Data Networking (NDN).

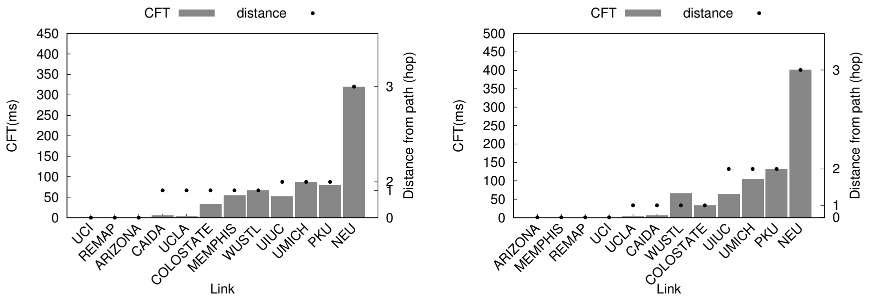
References

1. Acs, G., Conti, M., Gasti, P., Ghali, C., Tsudik, G.: Cache privacy in named-data networking. In: ICDCS. pp. 41–51. IEEE Computer Society (2013)
2. Ambrosin, M., Conti, M., Gasti, P., Tsudik, G.: Covert ephemeral communication in named data networking. In: AsiaCCS. pp. 15–26. ACM (2014)
3. American Registry for Internet Numbers (ARIN), `https://www.arin.net/`
4. CCNx protocol. `http://www.ccnx.org/releases/latest/doc/technical/CCNxProtocol.html`
5. ChoiceNet. `https://code.renci.org/gf/project/choicenet/`
6. DiBenedetto, S., Gasti, P., Tsudik, G., Uzun, E.: Andana: Anonymous named data networking application. In: NDSS. IEEE Computer Society (2012)
7. Donnet, B., Friedman, T.: Internet topology discovery: a survey. Communications Surveys Tutorials 9(4), 56–69 (2007)
8. Amazon Elastic Computing Cloud (EC2). `http://aws.amazon.com/ec2`
9. Eriksson, B., Barford, P., Sommers, J., Nowak, R.: A learning-based approach for ip geolocation. In: Krishnamurthy, A., Plattner, B. (eds.) PAM. pp. 171–180. Springer Berlin Heidelberg (2010)
10. Friedman, B.: A simple urn model. Communications on Pure and Applied Mathematics 2(1), 59–70 (1949)
11. MaxMind GeoIP database, `https://www.maxmind.com/`
12. GeoTrace, `http://www.nabber.org/projects/geotrace/`
13. Gueye, B., Ziviani, A., Crovella, M., Fdida, S.: Constraint-based geolocation of internet hosts. Transactions on Networking 14(6), 1219–1232 (2006)
14. Jacobson, V., Smetters, D.K., Briggs, N.H., Plass, M.F., Thornton, P.S.J.D., Braynard, R.L.: Voccn: Voice-over content centric networks. In: ReArch. ACM (2009)

15. Jacobson, V., Smetters, D.K., Thornton, J.D., Plass, M.F., Briggs, N.H., Braynard, R.L.: Networking named content. In: CoNEXT. pp. 1–12. ACM (2009)
16. Katz-Bassett, E., John, J.P., Krishnamurthy, A., Wetherall, D., Anderson, T., Chawathe, Y.: Towards IP Geolocation Using Delay and Topology Measurements. In: SIGCOMM IMC. pp. 71–84. ACM (2006)
17. Liu, H., Zhang, Y., Zhou, Y., Zhang, D., Fu, X., Ramakrishnan, K.: Mining checkins from location-sharing services for client-independent ip geolocation. In: INFOCOM. pp. 619–627. IEEE (2014)
18. Lumezanu, C., Baden, R., Spring, N., Bhattacharjee, B.: Triangle inequality and routing policy violations in the internet. In: PAM, pp. 45–54. Springer Berlin Heidelberg (2009)
19. MobilityFirst FIA Overview. `http://mobilityfirst.winlab.rutgers.edu`
20. Named data networking project (NDN), `http://named-data.org`
21. NDN testbed. `http://named-data.net/ndn-testbed/`
22. Nebula. `http://nebula.cis.upenn.edu`
23. Padmanabhan, V.N., Subramanian, L.: An investigation of geographic mapping techniques for internet hosts. SIGCOMM Comput. Comm. Rev. 31(4), 173–185 (2001)
24. Periakaruppan, R., Nemeth, E.: Gtrace - a graphical traceroute tool. In: USENIX LISA. pp. 69–78. ACM (1999)
25. Réseaux IP Européens (RIPE), `http://www.ripe.net/`
26. Rocketfuel. `http://research.cs.washington.edu/networking/rocketfuel/`
27. Sweeney, L.: k-anonymity: A model for protecting privacy. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 10(5), 557–570 (2002)
28. Tor Project: Anonymity Online. `https://www.torproject.org`
29. Verde, N.V., Ateniese, G., Gabrielli, E., Mancini, L.V., Spognardi, A.: No nat'd user left behind: Fingerprinting users behind NAT from netflow records alone. In: ICDCS. pp. 218–227. IEEE Computer Society (2014)
30. Wang, Y., Burgener, D., Flores, M., Kuzmanovic, A., Huang, C.: Towards street-level client-independent ip geolocation. In: USENIX NSDI. pp. 365–379. USENIX Association (2011)
31. Wong, B., Stoyanov, I., Sirer, E.G.: Octant: A comprehensive framework for the geolocalization of internet hosts. In: USENIX NSDI. pp. 313–326. USENIX Association (2007)
32. XIA - eXpressive Internet Architecture. `http://www.cs.cmu.edu/~xia/`
33. Zhu, Z., Burke, J., Zhang, L., Gasti, P., Lu, Y., Jacobson, V.: A new approach to securing audio conference tools. In: AINTEC. pp. 120–123. ACM (2011)

Appendix A: Testbed Measurements

Figures 10(a) and 10(b) show that CFT can be used to accurately estimate distance. In Figure 10(a), we connected P to University of California, Irvine (UCI) and C to University of Arizona (UA), while in Figure 10(b) we connect C to the University of Memphis node. Landmarks were connected to all nodes in the testbed. In both cases, 8% of landmarks provided an incorrect distance, likely due to violation of triangle inequality. Therefore, we added “random noise” with probability 8% in the experiments presented in this paper.



(a) From UA

(b) From University of Memphis

Fig. 10. CFT vs. distance for content published at UCI.