# A Proactive Cache Privacy Attack on NDN

Alberto Compagno
*Cisco Systems*
Paris, France
acompagn@cisco.com

Mauro Conti
*Department of Mathematics*
*University of Padova*
Padova, Italy
conti@math.unipd.it

Eleonora Losiouk
*Department of Mathematics*
*University of Padova*
Padova, Italy
elosiouk@math.unipd.it

Gene Tsudik
*Donald Bren School of Information and Computer Sciences*
*University of California*
Irvine, US
gts@ics.uci.edu

Sebastiano Valle
*Department of Mathematics*
*University of Padova*
Padova, Italy
valle.sebastiano93@gmail.com

***Abstract*—Named Data Networking (NDN) is a relatively new architecture, adhering to the Information-Centric Networking (ICN) paradigm, which focuses on explicitly named, routable and addressable content. While addressing and overcoming some of the current Internet issues, ICN is also affected by its own ones. Among those, content caching can be exploited, together with the Content Fetch Time (CFT), to identify the contents requested by the users. This attack is *reactive*, since the attacker infers whether a content has been requested after the victim has already sent a request for it. The defence mechanisms rely on the modification of the CFT, which, despite defending the users, also damage them from a usability point of view.**

**In this paper, we investigate an enhanced version of the attack, which is *proactive* and is still feasible even under enabled countermeasures. In the *proactive* attack, the attacker forces a router to cache a content and only afterwards checks whether the victim sends requests for that content. With respect to the *reactive* attack the *proactive* one: (i) is resilient to the existing defence mechanisms; (ii) can be applied for both popular and unpopular contents; (iii) can be used also in case of multiple users connected to the same router of the victim. After several simulations, we identified the parameters required to setup the *proactive* attack and proved its feasibility, both in terms of effectiveness and in terms of bypassing the existing countermeasures. We, finally, explored new possible countermeasures.**

## I. Introduction

The paradigm shift in the Internet usage exposed its design limitations and motivated research to explore new solutions. Among those, ICN [1] is a relatively new inter-networking paradigm where content, rather than a host or an interface, plays the central role. Among the different architectures adhering to the ICN paradigm, NDN [2] is the most well-known. NDN directly addresses content using unique network names, instead of establishing communication between a source and a destination through packets, as in IP. A consumer requests desired content by issuing an interest carrying the content name. Then, the network is in charge of finding and returning the nearest copy of requested content. To this end, NDN features the *ubiquitous content caching*, according to which any host/router can store a copy of the content it receives or forwards, and use it to satisfy subsequent requests. Despite its benefits [3], content caching has raised some privacy concerns in the past [4], [5], such as the so called *cache privacy attack (CPA)*: an attacker, that shares a first-hop router with the victim, can exploit timing information to infer if the victim requested a content $C$. To do so, the attacker can issue an interest for $C$ and compare the CFT with the Round Trip Time (RTT) from the first-hop router. If those times match, it means the victim has previously requested the content, which has been consequently cached in the router. Among the proposed countermeasures [5], [6], [7], the delay-based approaches [4], [5] apply a delay to emulate the actual CFT needed to retrieve the content from the producer.

978-1-7281-4973-8/20/$31.00 © 2020 IEEE

In this paper, we describe a new CPA, which proves that, even under enabled countermeasures, the timing information can be used to infer the content requested by the victim. While the previously identified CPA is *reactive*, i.e., the attacker acts after the victim requests a content, we propose a *proactive* attack: the attacker forces a router to cache a content and, then, she exploits machine learning techniques [8], [9] to analyze the victim network traffic and infer whether the content has been requested.

The proactive attack differs from its reactive counterpart in several ways:

- The combination of timing and traffic information makes the attack resilient to previously proposed delay-based countermeasures [5], [6], [7];
- Pre-populating a router cache allows the attack to be conducted on any content, even on an unpopular (perhaps sensitive) one, which is unlikely to be cached since usually requested only by few users;
- The attack works even in presence of many users sharing the same first-hop router since it relies on the layer-2 identifiers (e.g., MAC addresses) to identify victim's traffic.

**Contribution.**

- We investigated the first proactive CPA in NDN by exploiting the in-network caching and timing features;

- We implemented the attack and proved its feasibility by performing several experiments;
- We proved that the attack can bypass the existing countermeasures;
- We considered several defence mechanisms.

**Organization.** In Section II, we provide a brief overview on NDN, while in Section III on related work. Section IV describes our threat model and Section V our attack. In Section VI, we present the experimental results, while, in Section VII, we discuss about the effectiveness of the attack and possible countermeasures. We conclude in Section VIII.

## II. Background

### A. NDN overview

In NDN, the communication follows a *pull* approach: content (i.e., *data*) is delivered to consumers only upon (prior) explicit request (i.e., *interest*) for that content. To support this paradigm, hosts and routers have three main data structures: the Content Store (CS), which caches contents to reduce network congestion and enable faster delivery to consumers; the Pending Interest Table (PIT), which keeps track of the pending interests (i.e., not yet satisfied by the corresponding content) and of the incoming interfaces; the Forwarding Information Base (FIB), which contains a list of name prefixes and outgoing interfaces used to forward the interests. Then, the communication works as follows: to retrieve a content $C$, a consumer sends an interest $I_C$ in the network. Upon the reception of $I_C$, a router checks for matching content in its CS and returns it, if available. Otherwise, the router performs a lookup into the PIT to check for a pending interest carrying the same name of $I_C$. If a match is found, $I_C$ is collapsed into the existing PIT entry and discarded, otherwise a new PIT entry is created and $I_C$ is forwarded to the next hop. In both cases, the $I_C$ incoming interface is stored in the PIT entry. If no intermediate router can satisfy $I_C$, the interest arrives to the producer of $C$, which replies with the requested content. The content $C$ is forwarded back to the consumer following the information in the routers' PIT.

## III. Related Work

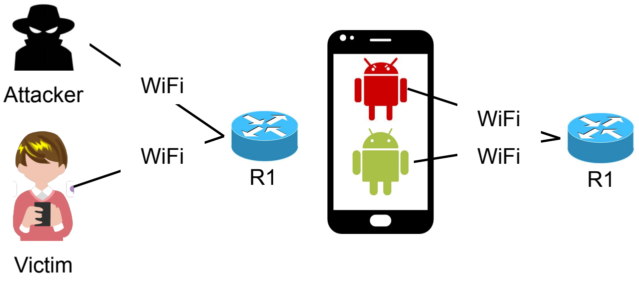
**Privacy attacks.** The reactive CPA [10] is only one of the NDN privacy issues [11], [12] known to date. Its main goal is to discover if a user has requested a specific content $C$. To the best of our knowledge, all prior CPAs in NDN are *reactive* in nature: the attack is successful only if the user has already pulled $C$ from the first-hop router. The most trivial countermeasure is for the router to withhold releasing cached content $C$ by introducing an artificial delay, which maintains the CFT equal to the RTT from the producer. Acs et al. [5] proposed Random-Cache, wherein the delay is applied only to the first $k$ requests for a content $C$, and $k$ is a value randomly chosen for each cached content. Random-Cache achieves a very high level of privacy against a reactive attacker, although it reveals that $C$ is stored in a router when a user experiences a cache hit after $k$ requests.

**Cache pollution attacks.** Cache pollution is the ability to modify the true popularity of data in caches and it is a pervasive threat in NDN since caching is a key feature. Research results mostly focused on countermeasures against this problem [13], [14], [15], without conducting a thorough study about which parameters influence this type of attack.

**Traffic analysis attacks.** Dyer et al. [8] proved that "efficient" countermeasures against traffic analysis [16] still fail when attacks based on other coarse-grained side-channels are applied. Later, Cai et al. [17] showed that HTTPOS [18] and randomized pipelining over Tor [19] can be defeated with an accuracy higher than 50% when performing a website fingerprinting attack. More recently, it has been shown how to recognize some actions of the users by analyzing the traffic generated by Android applications on mobile devices [20]. Finally, Panchenko et al. [9] showed that just by carefully choosing the training set for the classifier and using information about packet size, direction and ordering, it is possible to carry out a powerful website fingerprinting attack at Internet scale.

## IV. Threat Model and Assumptions

We considered the following attack scenarios: (i) in Figure 1a, the victim and the attacker are connected to the same router $R1$, identified by two separate interfaces; (ii) in Figure 1b, the victim and the attacker communicate with $R1$ through the same device, thus, the same interface.



(a) First attack scenario. (b) Second attack scenario.

Fig. 1: Attack scenarios.

We assume that: the attacker knows the details of $R1$, the content interest format guarantees a *weak data privacy* model [21] and the communication between the victim and the router is encrypted at layer 2. Finally, we assume that the attacker can eavesdrop on victim network traffic and measure the CFT.

## V. Design of the Proactive Cache Privacy Attack

The proactive CPA involves three steps: (i) **preparation**, in which the attacker identifies the most effective parameters' values to load the *decoy content* in the first-hop router cache (Section V-A); (ii) **content loading**, in which the attacker forces the first-hop router to cache the *decoy content* (Section V-B); (iii) **traffic analysis**, in which the attacker eavesdrops on the network traffic and determines whether the victim has requested the *decoy content* (Section V-C).

### *A. Preparation*

To complete the first step, the attacker has to perform a cache pollution attack in both its configurations: *false locality*, to cache the *decoy content* the victim might later request, and *locality disruption* [22], [13], to prevent the eviction of the *decoy content* from the cache through other legitimate requests. To achieve her aim, the attacker has to evaluate the following parameters:

- **Dimension of polluting set.** The polluting set is the set of contents used by the attacker to pollute the cache through the locality disruption attack. This set should guarantee the longest caching time for the *decoy content* and the highest percentage of polluted set in router cache.
- **Attacker traffic pattern.** To evaluate the effectiveness of the attack for different content popularity, we considered both the Zipf-like distribution [23] and the round-robin distribution [15]. The Zipf-like distribution is more suitable for the false locality because: it allows to cache contents in routers for long; it allows to cache a single content as well as a set of contents; it is less prone to detection [15], since honest consumers follow the same pattern [24]. On the other hand, a round-robin attacker sends requests for contents in a set $[0, k]$, where $k$ is the size of the polluting set, and she is able to cause locality disruption with little effort [15].
- **Attacker interest sending rate.** The attacker interest sending rate depends on the amount of cache she wants to pollute.
- **Number of honest users.** The more honest consumers are connected to the same router as the attacker, the higher will be the number of truthful requests that can cause the eviction of the *decoy content*. Previous works on cache pollution [15], [13] considered up to 15 honest users spread over their networks, while we considered a varying number between 5 and 50. When not otherwise specified, the number of honest users is equal to five.
- **Router cache size.** Since tiny cache sizes are not realistic [14], we only considered caches capable of storing at least 2000 content packets.
- **Router cache replacement policy.** We investigated the effectiveness of our attack under Least Recently Used (LRU), Least Frequently Used (LFU), and First In First Out (FIFO) policies. When not otherwise specified, routers will use LRU as a cache replacement policy.

### *B. Execution (Phase I)*

Phase I consists in the algorithm shown in Fig. 2:

- **Step 1.** The attacker sends an interest for content $C$ and measures $CFT1_c$. We assume that $C$ is not cached in the first-hop router and is retrieved from a farther node. Thus, $CFT1_c$ is the worst CFT measured by the attacker for $C$.
- **Step 2.** The attacker sends a new request for $C$ and saves $CFT2_c$.
- **Step 3.** The attacker compares $CFT1_c$ and $CFT2_c$. If they are equal, she infers the content is not cached in the first-hop router yet, ignores $CFT2_c$ and repeats *Step 2*. If they are different, the attacker cannot infer the content location and she moves forward to *Step 4*.
- **Step 4.** The attacker sends a new request for $C$ and saves $CFT3_c$.
- **Step 5.** The attacker compares $CFT2_c$ and $CFT3_c$ and concludes as follows:
  - $CFT2_c$ and $CFT3_c$ match: the content is cached and the router applies no delay-based countermeasures;
  - $CFT2_c$ and $CFT3_c$ have almost the same value: the content is cached and the router applies a static delay countermeasure, since variability on the CFT introduced by a static delay is expected to be low;
  - $CFT2_c$ and $CFT3_c$ differ: the attacker concludes that either the cached contents have changed due to requests of other consumers or the router is applying a random delay countermeasure. In both cases, the attacker goes back to *Step 2* and repeats the algorithm. If the router is applying a random delay countermeasure, after $k$ repetitions of the algorithm, the attacker reaches the threshold value after which the router stops modifying the CFT.

The attacker follows this algorithm to cache a single content both in false locality and in locality disruption attack. However, in the former, the attacker switches to Phase II as soon as the *decoy content* is cached and uses the algorithm to verify that the *decoy content* is still cached. In the locality disruption case, the attacker keeps repeating the algorithm to fill the router cache.

### *C. Execution (Phase II)*

Phase II involves monitoring the network traffic generated by the victim to infer if the victim requests the *decoy content*. This evidence can be observed if all the following conditions are satisfied: (i) the CFT for the victim to fetch a content is close to the RTT between the victim and the router; (ii) the size of data packets received by the victim match the size of the *decoy content* data packets; (iii) the number of the data packets received by the victim match the *decoy content* ones.

Several prior works [8], [9], [25], [17], [18], [19], [20] address network traffic analysis attacks and their countermeasures. Even though they consider the TCP/IP architecture, they rely on features (e.g., packet size, consumed bandwidth) that do not depend on the architecture and are available also in an encrypted and anonymous communication. Thus, the same approaches are applicable to the ICN context.

## VI. Experimental Setup and Results

As it is common practice in ICN research, we evaluated our attack in a simulated environment and we present here the experimental results concerning the preparation and Phase I of the attack.

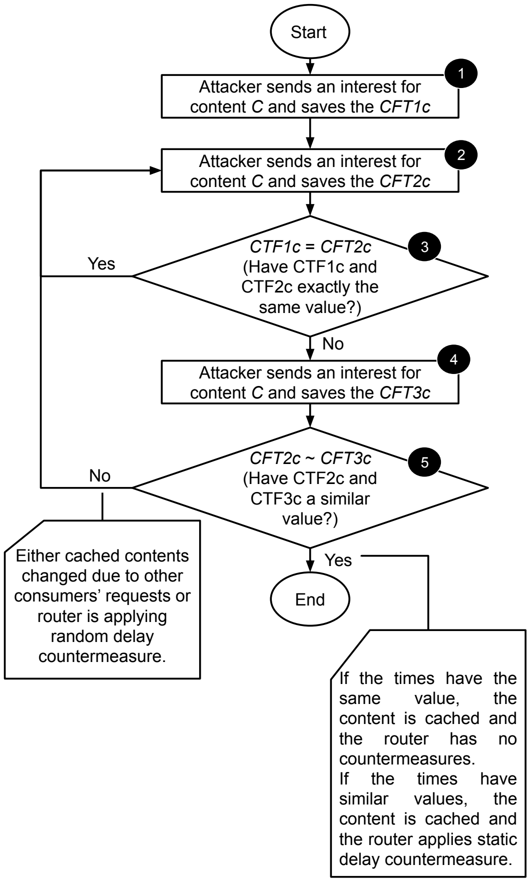

Start
Attacker sends an interest for content C and saves the CFT1c
1
Attacker sends an interest for content C and saves the CFT2c
2
CTF1c = CTF2c (Have CTF1c and CTF2c exactly the same value?)
3
Yes
No
Attacker sends an interest for content C and saves the CFT3c
4
CFT2c ~ CFT3c (Have CTF2c and CTF3c a similar value?)
5
No
Yes
End
Either cached contents changed due to other consumers' requests or router is applying random delay countermeasure.
If the times have the same value, the content is cached and the router has no countermeasures.
If the times have similar values, the content is cached and the router applies static delay countermeasure.

Fig. 2: Phase I Algorithm.

### A. Preparation

To consider the impact of different values for the parameters discussed in Section V-A, we ran several simulations in ndnSIM 2.3 [26], [27], i.e., the main NS-3 based simulator for NDN networks. All simulations were run in the 0.5-4 minute range. When not otherwise specified, the duration of a simulation is 60 seconds. The network topology involves a set of benign consumers and an attacker, all connected to the same NDN router. We assume that all parties request contents: the former as a result of a regular user activity in the Internet, while the latter to conduct a proactive CPA. Finally, the benign consumers and the attacker request content from two disjoint sets, which is the worst-case, since the attacker has to compete with the benign consumers to pollute the cache.

**Dimension of polluting set.** Figure 3 and Figure 4 show how different sizes of the polluting set, with respect to the router cache size, affect the amount of time the *decoy content* is cached and the amount of polluted cache. Figure 3 shows the most popular *decoy content*, while Figure 4 the five most popular ones. We assume to have a Zipf-like attacker, which uses the 5% of the polluting set as sensitive content and the remaining 95% for polluting the cache. The horizontal axis uses a logarithmic scale, while the vertical one shows the time spent in cache by the *decoy content* (plotted in black) and the percentage of cache polluted at the end of a simulation (plotted in red).

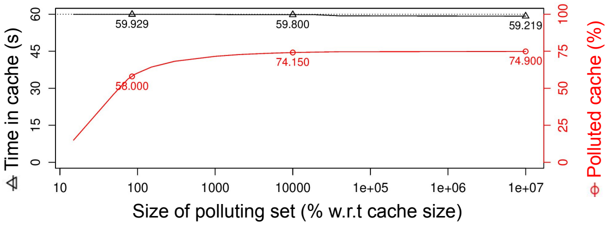


Fig. 3: Effectiveness of the attack according to different polluting set size with a Zipf-like attacker – Most popular content.

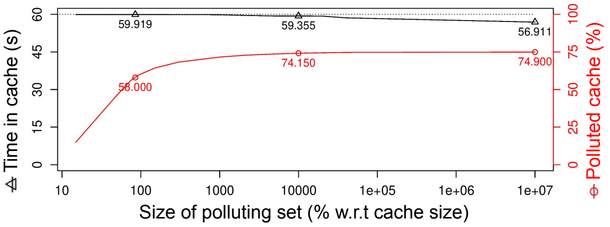


Fig. 4: Effectiveness of the attack according to different polluting set size with a Zipf-like attacker – Top 5 popular contents.

As shown in the figures, the attacker can make a small set of content stay cached approximately for the whole duration of the attack simulation, while being able to pollute large fractions of the cache.

**Attacker traffic pattern.** As shown in Figure 5, the Zipf-like attacker is more effective when the set size is large: with a polluting set size equal to the 10000% of the router cache size, the polluting set contents stay in cache for half of the whole simulation.

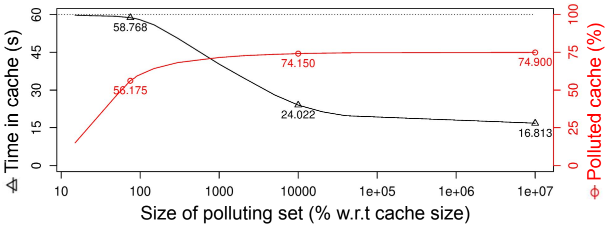


Fig. 5: Effectiveness of the attack with a Zipf-like attacker.

As shown in Figure 6, the round-robin attacker achieves good results when the polluting set is smaller than the cache size. The plot shows two straight curves caused by the criterion used to retrieve contents from the polluting set. After a threshold, for every request of a new content, the attacker can trigger the eviction of another content of the polluting set, causing the *self-eviction* phenomenon. Thus, after that threshold, each polluting content stays in cache only for a while, without being chosen again by the round-robin attacker.

Moreover, due to the *self-eviction* phenomenon, the percentage of cache polluted set cannot go above 75.25%.

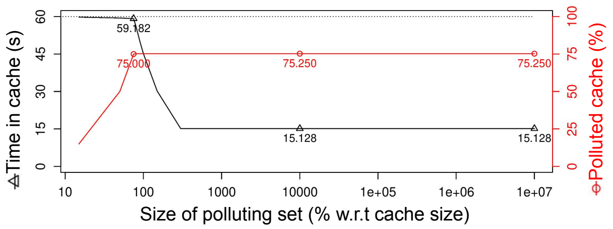
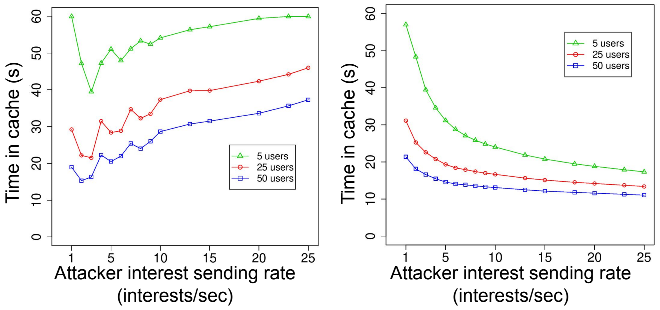


Fig. 6: Effectiveness of the attack with a round-robin attacker.

**Attacker interest sending rate.** We considered an attacker interest sending rate varying between 1 and 25.

**Number of honest users.** Figure 7a and Figure 7b show the effectiveness of a false locality attack for a round-robin and a Zipf-like attacker with different attacker interest sending rates and different number of honest users connected to the same router.



(a) Round-robin attacker

(b) Zipf-like attacker

Fig. 7: False locality with different numbers of honest users.

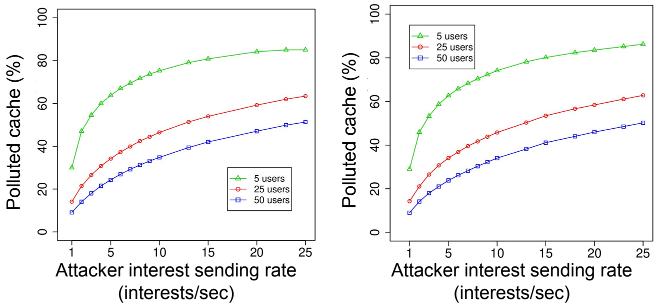
The shape of the curves in both figures is approximately the same for different number of users. However, a round-robin attacker performs better for high values of attacker interesting sending rate, whereas a Zipf-like attacker behaves in the opposite way.

Figure 8a and Figure 8b show the same scenario for a locality disruption attack. For both attacker types, the percentage of polluted cache grows monotonically and the attackers have approximately the same effectiveness: the more requests they issue, the more chances they have to evict contents requested by a honest user.

To conclude, the more users are connected to the same first-hop router, the less effective the attack is.

**Router cache size.** The cache size varies from 2000 packets (∼3MB) to 16000 packets (∼24MB), similarly to previous works [15].

Figure 9a and Figure 9b show the effectiveness of a false locality attack for a round-robin and a Zipf-like attacker with different attacker interest sending rates and different router cache sizes. Considering the round-robin attacker, all curves have the same behaviour, even if shifted along the x-axis

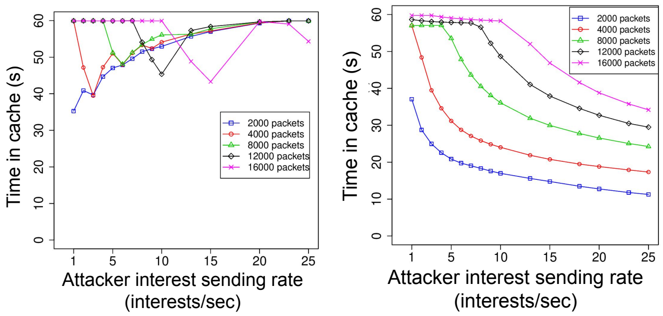


(a) Round-robin attacker

(b) Zipf-like attacker

Fig. 8: Locality disruption with different numbers of honest users.

according to the router cache size. The negative spikes correspond to the scenario in which the attacker interest sending rate is not high enough to prevent polluted content from being evicted by honest requests, and not low enough to avoid *self-eviction*. The effectiveness of a Zipf-like attacker decreases with a growing interesting sending rate, since less popular contents start being requested more than before, causing the *self-eviction* phenomenon to occur more frequently. However, when the cache size is big enough, the 10% of the contents is cached for almost the entire duration of the simulation for an attacker interest sending rate less or equal to 10. This happens because with large caches, the self-eviction starts to be noticeable at high values of attacker interest sending rate, since there is enough room to store more content in the router cache.

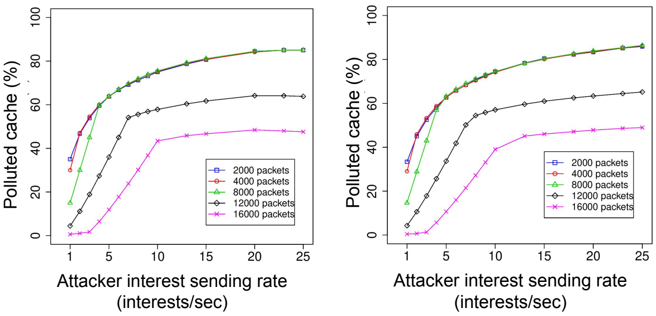


(a) Round-robin attacker

(b) Zipf-like attacker

Fig. 9: False locality with different cache sizes.

Figure 10a and Figure 10b show the same scenario in a locality disruption attack. As before, the percentage of polluted cache grows with higher attacker interest sending rate. In particular, for each cache size the increase in effectiveness is steep until a given value: looking at the corresponding false locality graphs, we can notice that this is the value before the negative spike. Hence, for an attacker it is convenient to raise her interest sending rate until a local optimum for the trade-off between false locality and locality disruption is met, regardless of the attacker traffic pattern. Similarly to the analysis on the number of users, the results for a Zipf-like attacker and a round-robin attacker are really similar under varying cache sizes.



(a) Round-robin attacker

(b) Zipf-like attacker

Fig. 10: Locality disruption with different cache sizes.

**Router cache replacement policy.** In general, the cache replacement policies do not affect our attack. Only under LFU, the effectiveness slightly decreases because it becomes more difficult to evict popular contents requested by honest users.

### *B. Execution (Phase I)*

To evaluate the precision of Algorithm I, we considered a first scenario, with a router applying no countermeasure, and a second scenario, with a router applying a static delay. For each scenario, we then have two cases: the first, in which the *decoy content* is cached when the attacker issues an interest for it; the second, in which the *decoy content* is evicted from the cache after a while. For each scenario, we ran 200 simulations and we report the results in the confusion matrices (Table I), in which '1' stands for "cached" and '0' for "evicted", respectively. In all scenarios, the attacker uses our algorithm to guess whether a content is still cached (1) or not (0), just making its decision on the timing with which a content is returned.

As shown in the confusion matrices, in the first scenario the algorithm precision is maximum, since the attacker is always able to infer whether the content is cached, while in the second, the algorithm accuracy drops 5.5% without getting any false positive.

|  |  | Predicted |  |
|---|---|---|---|
|  |  | 1 | 0 |
| Actual | 1 | 100 | 0 |
|  | 0 | 0 | 100 |

No countermeasure

|  |  | Predicted |  |
|---|---|---|---|
|  |  | 1 | 0 |
| Actual | 1 | 89 | 11 |
|  | 0 | 0 | 100 |

Static delay

TABLE I: Confusion matrices illustrating the precision of Algorithm 1 used by the attacker to determine whether a content is in router cache.

## VII. Discussion

Here, we discuss about the feasibility of the proactive CPA (Section VII-A) and about its possible countermeasures (Section VII-B).

### *A. Proactive Cache Privacy Attack Discussion*

For the preparation step, an attacker can either choose the Zipf-like distribution or the round-robin one. A Zipf-like attacker is able to cache a small set of contents for almost the whole duration of the preparation and is less prone to detection since she uses the same traffic pattern as honest consumers [15], [13] with a low interest sending rate (i.e., between 1 and 5 interests/sec). The effectiveness of the attack increases with high router cache sizes and decreases with a high number of honest consumers and an LFU eviction policy implemented in the router. The round-robin attacker can keep the router cache largely polluted for almost the whole duration of the preparation with an initial polluting set smaller than the router cache size. With respect to the Zipf-like attacker, the round-robin one needs a higher interest sending rate, which should be increased even further in case of high number of honest consumers and should be more then 10 interests/sec in case of big router cache. A round-robin is not affected by the cache replacement policy.

Concerning the Phase I, by following our algorithm, the attacker can determine whether a content is in the router cache with a precision of 100%, if there are no countermeasures, and an accuracy drop of 5.5%, if there is a static delay countermeasure.

### *B. Proactive Cache Privacy Attack Countermeasures*

A simple countermeasure against the proactive CPA could be the *Hop-Count Delay*, which consists in estimating the number of hops $h$ according to the RTT needed to satisfy an interest. Thus, a router should mimic a "$h$-hops" variance in the artificial delay when returning a content. Another possible countermeasure could be to apply specific per-face delays when returning cached content, choosing them within a range around the RTT needed to fetch that specific content in the first place. However, it might not be effective against the proactive attack when both the attacker and the victim share the same interface. Moreover, this countermeasure requires router to store per-interface state. Finally, even removing the time component as a fine-grained side-channel, the proactive attack is still feasible up to some extent. Thus, routers could add some padding to data packets or use Traffic Morphing [16], but this would need a thorough evaluation since coarse-grained information about time might still be available [8].

## VIII. Conclusion

In this paper, we propose a novel *proactive* CPA in NDN based on the fundamental in-network caching feature, which can bypass the existing countermeasures, devised for a reactive CPA. We broke our novel attack into three separate steps, and provided empirical evidence for the feasibility of its building blocks. We, finally, leave room for a full evaluation of this attack, whose Phase II could be probably performed by means of some statistical traffic analysis or machine learning technique, as well as the evaluation of the possible countermeasures we proposed in this paper.

# REFERENCES

[1] G. Xylomenos, C. N. Ververidis, V. A. Siris, N. Fotiou, C. Tsilopoulos, X. Vasilakos, K. V. Katsaros, and G. C. Polyzos, "A survey of Information-Centric Networking Research," *IEEE Communications Surveys Tutorials*, vol. 16, no. 2, pp. 1024–1049, 2014.

[2] L. Zhang, D. Estrin, J. Burke, V. Jacobson, J. D. Thornton, D. K. Smetters, B. Zhang, G. Tsudik, K. Claffy, D. Krioukov, D. Massey, C. Papadopoulos, T. Abdelzaher, L. Wang, P. Crowley, and E. Yeh, "Named Data Networking (NDN) Project," Tech. Rep. NDN-0001, October 2010.

[3] J. Samain, G. Carofiglio, L. Muscariello, M. Papalini, M. Sardara, M. Tortelli, and D. Rossi, "Dynamic adaptive video streaming: Towards a systematic comparison of ICN and TCP/IP," *IEEE Transactions on Multimedia*, vol. 19, no. 10, pp. 2166–2181, Oct 2017.

[4] G. Acs, M. Conti, P. Gasti, C. Ghali, and G. Tsudik, "Cache privacy in Named-Data Networking," in *2013 IEEE 33rd International Conference on Distributed Computing Systems*, July 2013, pp. 41–51.

[5] G. Acs, M. Conti, P. Gasti, C. Ghali, G. Tsudik, and C. A. Wood, "Privacy-aware caching in Information-Centric Networking," *IEEE Transactions on Dependable and Secure Computing*, vol. 16, pp. 313–328, 2019.

[6] S. DiBenedetto, P. Gasti, G. Tsudik, and E. Uzun, "Andana: Anonymous named data networking application," in *Proceedings of the Network and Distributed System Security Symposium, NDSS 2012, San Diego, California, USA, 5th February - 8th February 2012*, 2012.

[7] A. Mohaisen, H. Mekky, X. Zhang, H. Xie, and Y. Kim, "Timing attacks on access privacy in Information Centric Networks and countermeasures," *IEEE Trans. Dependable Sec. Comput.*, vol. 12, no. 6, pp. 675–687, 2015. [Online]. Available: https://doi.org/10.1109/TDSC.2014.2382592

[8] K. P. Dyer, S. E. Coull, T. Ristenpart, and T. Shrimpton, "Peek-a-boo, I still see you: Why efficient traffic analysis countermeasures fail," in *Proceedings of the 2012 IEEE Symposium on Security and Privacy*, ser. SP '12. Washington, DC, USA: IEEE Computer Society, 2012, pp. 332–346. [Online]. Available: https://doi.org/10.1109/SP.2012.28

[9] A. Panchenko, F. Lanze, J. Pennekamp, T. Engel, A. Zinnen, M. Henze, and K. Wehrle, "Website fingerprinting at internet scale," in *Proceedings of the Network and Distributed System Security Symposium, NDSS 2016, San Diego, California, USA, 21st February - 24th February 2016*, 2016.

[10] T. Lauinger, N. Laoutaris, P. Rodriguez, T. Strufe, E. Biersack, and E. Kirda, "Privacy implications of ubiquitous caching in named data networking architectures." Tech. Rep. TR-iSecLab-0812-001, August 2012.

[11] A. Chaabane, E. De Cristofaro, M. A. Kaafar, and E. Uzun, "Privacy in content-oriented networking: Threats and countermeasures," *SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 3, pp. 25–33, Jul. 2013. [Online]. Available: http://doi.acm.org/10.1145/2500098.2500102

[12] C. Ghali, G. Tsudik, and C. A. Wood, "When encryption is not enough: Privacy attacks in content-centric networking," in *Proceedings of the 4th ACM Conference on Information-Centric Networking*, ser. ICN '17. New York, NY, USA: ACM, 2017, pp. 1–10. [Online]. Available: http://doi.acm.org/10.1145/3125719.3125723

[13] M. Conti, P. Gasti, and M. Teoli, "A lightweight mechanism for detection of cache pollution attacks in named data networking," *Comput. Netw.*, vol. 57, no. 16, pp. 3178–3191, Nov. 2013. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2013.07.034

[14] H. Salah, M. Alfatafta, S. SayedAhmed, and T. Strufe, "Comon++: Preventing cache pollution in NDN efficiently and effectively," in *42nd IEEE Conference on Local Computer Networks, LCN 2017, Singapore, October 9-12, 2017*, 2017, pp. 43–51. [Online]. Available: https://doi.org/10.1109/LCN.2017.35

[15] M. Xie, I. Widjaja, and H. Wang, "Enhancing cache robustness for content-centric networking," in *2012 Proceedings IEEE INFOCOM*, March 2012, pp. 2426–2434.

[16] C. V. Wright, S. E. Coull, and F. Monrose, "Traffic morphing: An efficient defense against statistical traffic analysis," in *Proceedings of the Network and Distributed System Security Symposium, NDSS 2009, San Diego, California, USA, 8th February - 11th February 2009*, 2009. [Online]. Available: https://www.ndss-symposium.org/ndss2009/traffic-morphing-efficient-defense-against-statistical-traffic-analysis/

[17] X. Cai, X. C. Zhang, B. Joshi, and R. Johnson, "Touching from a distance: Website fingerprinting attacks and defenses," in *Proceedings of the 2012 ACM Conference on Computer and Communications Security, CCS 2012, Raleigh, North Carolina, USA, 16th October - 18th October 2012*, 2012.

[18] X. Luo, P. Zhou, E. W. W. Chan, W. Lee, R. K. C. Chang, and R. Perdisci, "Httpos: Sealing information leaks with browser-side obfuscation of encrypted flows," in *Proceedings of the Network and Distributed System Security Symposium, NDSS 2011, San Diego, California, USA, 6th February - 9th February 2011*, 2011.

[19] "Experimental Defense for Website Traffic Fingerprinting." [Online]. Available: https://blog.torproject.org/experimental-defense-website-traffic-fingerprinting

[20] M. Conti, L. V. Mancini, R. Spolaor, and N. V. Verde, "Can't you hear me knocking: Identification of user actions on android apps via traffic analysis," in *Proceedings of the 5th ACM Conference on Data and Application Security and Privacy*, ser. CODASPY '15. New York, NY, USA: ACM, 2015, pp. 297–304. [Online]. Available: http://doi.acm.org/10.1145/2699026.2699119

[21] C. Ghali, G. Tsudik, and C. A. Wood, "(The Futility of) data privacy in content-centric networking," in *Proceedings of the 2016 ACM on Workshop on Privacy in the Electronic Society*, ser. WPES '16. New York, NY, USA: ACM, 2016, pp. 143–152. [Online]. Available: http://doi.acm.org/10.1145/2994620.2994639

[22] Y. Gao, L. Deng, A. Kuzmanovic, and Y. Chen, "Internet cache pollution attacks and countermeasures," in *Proceedings of the 2006 IEEE International Conference on Network Protocols*, Nov 2006, pp. 54–64.

[23] R. Bain, "Human Behavior and the Principle of Least Effort: An Introduction to Human Ecology. By George Kingsley Zipf. Cambridge, Mass.: Addison-Wesley Press, Inc., 1949. 573 pp." *Social Forces*, vol. 28, no. 3, pp. 340–341, 03 1950. [Online]. Available: https://doi.org/10.2307/2572028

[24] L. Breslau, Pei Cao, Li Fan, G. Phillips, and S. Shenker, "Web caching and zipf-like distributions: evidence and implications," in *1999 Proceedings IEEE INFOCOM*, vol. 1, March 1999, pp. 126–134 vol.1.

[25] C. V. Wright, S. E. Coull, and F. Monrose, "Traffic morphing: An efficient defense against statistical traffic analysis," in *In Proceedings of the 16th Network and Distributed Security Symposium*. IEEE, 2009, pp. 237–250.

[26] A. Afanasyev, I. Moiseenko, and L. Zhang, "ndnSIM: NDN Simulator for NS-3," Tech. Rep. NDN-0005, August 2012.

[27] S. Mastorakis, A. Afanasyev, and L. Zhang, "On the evolution of ndnsim: An open-source simulator for ndn experimentation," *SIGCOMM Comput. Commun. Rev.*, vol. 47, no. 3, pp. 19–33, Sep. 2017. [Online]. Available: http://doi.acm.org/10.1145/3138808.3138812