






Identifying Previously Requested Content by Side-Channel Timing Attack in NDN

Ertugrul Dogruluk^(✉) , Antonio Costa , and Joaquim Macedo 

Centro Algoritmi, Universidade do Minho, Braga, Portugal
{d7474, costa, macedo}@di.uminho.pt

Abstract. NDN is a new name-based network paradigm. It is designed to keep the contents in the cache to increase the network efficiency. However, previously requested content may put the user privacy at risk. The time difference between cached and non-cached contents of interest responses can be used by an adversary to determine previously requested contents in cache. This attack is classified as side-channel timing attack. In NDN, it is used a signature to authenticate interests and data packets. However, signed packets does not affect the performance of side-channel timing attack. Independently of being signed or not, the adversary may identify both the sensitive and non-sensitive contents, recently cached by router. In order to mitigate side-channel attacks in NDN, there are several countermeasure methods proposed by other researchers. In this work, firstly we developed an attack scenario using ndnSIM simulator. Then we evaluated the scenario under attack and without attacks. We also proposed an adversary detection algorithm that combines three different defense countermeasures in order to maximize the cache availability.

Keywords: NDN · Content privacy · Side-channel timing attack

1 Introduction

Internet is being reshaped to handle content production and distribution, as nowadays users desire, for example, to watch movies and use social networking. Moreover, the number of IoT (Internet of Things) devices over the Internet is increasing enormously. However, such activities are not the most appropriate to be done over Internet, because this network was conceived for point-to-point communications. To overcome the problems raised by this communication paradigm, content centric networks (CCNs) have been proposed. According to this new paradigm, replicas of content(s) are generated and cached. The aim of caching is to reduce the latency and data loss, thereby improving the distribution quality of popular content(s). NDN (Named Data Network) is based on cache (buffer-memory) and name-based network that has been presented as a next version of CCN networks, as proposed by [16]. Nevertheless, caching, in spite of its benefits, may threaten the privacy of NDN users. An adversary may know which content an user has requested and become motivated to proceed

with a timing attack [7] It is based on the time differences between cached and not cached contents, as discussed later in this paper. Any type of cached content may be targeted by an adversary to cause a privacy threat. So, timing attack affecting the user privacy is a major problem in NDN, since blocking such attack is a challenging problem to solve. However, there are a few limited techniques to prevent timing attacks. The proposed countermeasure methods are based on artificial network delay [7, 10], random caching [1], and request name encrypting [5]. Since these countermeasure techniques affect the cache performance, there is a trade-off between efficiency and privacy. For example, one technique is to delay the request content in the node to reduce the network throughput.

In this paper, an implemented attack topology is presented, side-channel timing attack measurements are analyzed and, based on primary findings, and is proposed an algorithm to identify the adversary node.

The rest of this paper is organized as follows. Section 2 summarizes the NDN architecture and its features. Section 3 demonstrates the operation of side-channel timing attack and how it can be used against the user privacy. Section 4 studies current countermeasures to mitigate side-channel timing attack. Section 5 shows the scenario implemented to simulate side-channel timing attacks. Section 6 summarizes attack scenario findings, and countermeasure results. Section 7 shows related researches that have been done so far. Finally, Sect. 8 presents the conclusions.

2 NDN Architecture

NDN is an ongoing project that proposes to transform the existing Internet design into a content-centric architecture, in order to improve the content distribution efficiency. NDN is based on human readable address names and keeps the contents in the cache, thus facilitating content distribution and providing low latency [17]. NDN is based on *interest* and *data* packets. Interest packets are produced by the consumers and data packets by the producers. The content name in the interest packet identifies the request of the consumer, for example, /pt/uminho/algorithmi. As shown in Fig. 1, the producer includes a signature in the data packet. Mechanisms for signing and verifying the integrity of the contents have been proposed for NDN, such as the one described in [8].

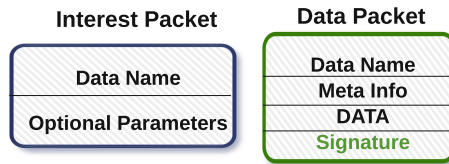


Fig. 1. Packet types used in NDN (adapted from [8])

2.1 NDN Forwarding Model

In a NDN router, the forwarding of interest and data packets is carried out by three engines: PIT(Pending Interest Table), FIB (Forwarding Information Base) and CS (Content Store) [17].

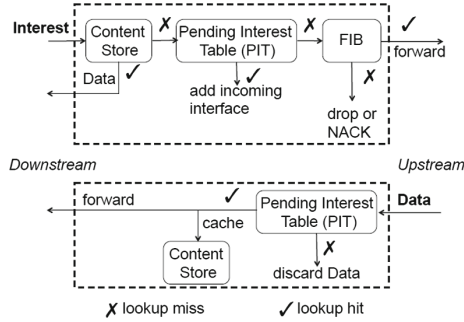


Fig. 2. NDN forwarding engine model [17].

The CS represents a cache for data packets. As illustrated in Fig. 2, the consumer interest first looks for data in the CS and, if there is a matching data, CS replies with a data packet. If the data packet is not in the CS, the router checks the data name in the PIT. If there is a matching name in the PIT, it records the incoming face (interface) of the interest. If not, the PIT forwards to the FIB the interest packet, which is then routed to the producer. For each interest packet that needs to be forwarded, the longest prefix (name) matching is searched in the FIB, which predicts when and where to forward the interest. The list of outgoing faces stored at the matched entry in the FIB is an important reference for routing.

When the data packet comes from the upstream router, the PIT is checked for a matching entry. If a match is found, the data packet is forwarded to the CS and then the entry is removed from the PIT for further incoming interest packets. If the authentication of the data packet fails, this packet is discarded by the PIT.

An NACK object informs consumers of data unavailability at the application level. Similar to NACK object, the Interest NACK is used to inform a router of its upstream router's inability at the network layer in NDN to forward an Interest packet, as described in [14].

In order to manage the packets in the CS, PIT, and FIB, the NDN developers created an engine called NFD (NDN Forwarding Daemon) [2]. This software tool is open source and keeps on evolving.

3 Side Channel Timing Attack

If a content is already cached in the CS, this replies to the user's interest with a data packet in a period of time. The RTT (Round Trip Time) is the time difference between sending the interest packet and receiving the data packet. The RTT of a cached content should be smaller than the RTT of a not-cached content. The adversary may take advantage of these RTT differences to know which contents have been or not placed in the CS. This technique is known as side-channel timing attack.

NDN struggles with four important privacy issues: *Naming*, *Content*, *Cache*, and *Signature*, as pointed out in [16]. This paper focus on the side-channel timing attack against the cached content in CS. Therefore, it mainly addresses privacy issues, but also name, content, and signature. These are an important issue to take into consideration, as it may affect the user privacy in NDN.

3.1 RTT Calculation

As shown in Fig. 3, a *timeout* occurs if the data packet is not received after a certain time interval. The timeout is related to the RTT estimation by a weight factor $\beta = 2$ (TimeOut = $\beta \times \text{RTT}_i$). Indeed, whenever an interest is forwarded to the upstream node, the router starts a timer, which will be used to measure the RTT. If the corresponding data packet arrives before the timer expiration, the router calculates the new RTT in accordance with Eq. 1, with $0 < \gamma < 1$. If γ is equal to $1 - \frac{1}{n}$ (n is the number of received data packets), then the real average RTT is obtained. If γ is close to 1, then the weighted average RTT is immune to delay changes for a short time interval. If γ is close to 0, then the weighted average RTT is very sensible to new delay changes.

If the data packet does not arrive, the router tries alternative routes until reaching the data packet. In case of non-existing data, a timeout is triggered and a NACK is sent. The router gives up of searching the data and the packet becomes an *unsatisfied interest*. But if a data packet was received, then the packet becomes a satisfied interest.

$$\boxed{\langle \text{RTT} \rangle_n = \gamma * \langle \text{RTT} \rangle_{n-1} + (1 - \gamma) * \text{RTT}_n} \quad (1)$$

On the other side, if the downstream search has also exhausted for incoming interest packets, an interest NACK will be sent to identify, through an error code, the cause of the unsatisfied interest (*e.g.*, *duplicate*, *congestion*, *no route*) [3]. Note that the interest NACKs cannot be used on a side-channel timing attack, because this attack requires the contents to be cached and retrieved.

3.2 Privacy in NDN

This sub-section explains how the side-channel timing attack is done and its effects on the CS. In this attack, an adversary node intends to break the cache

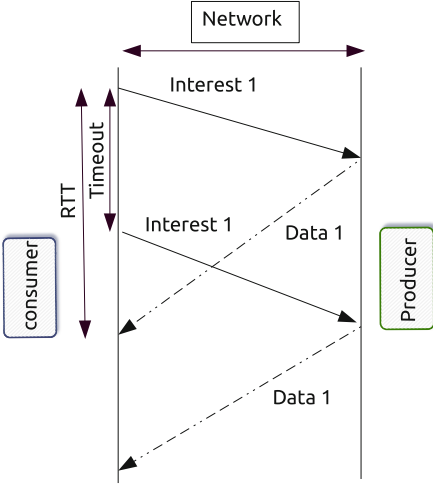


Fig. 3. Illustration of timeout and RTT in NDN

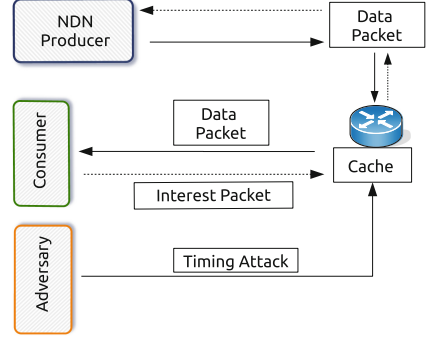


Fig. 4. Side-channel timing attack

privacy by requesting the same content from the CS that has been recently requested by a legitimate user.

Besides the cached contents, the adversary may also attack the certificate scheme. In NDN, each interest and data packets are bound with a public signature for integrity. Signatures are used for the authentication of the producer and each content is held in the CS for a time period. The public key of a certificate may also be used for a side-channel timing attack, that may jeopardize the user privacy, especially on real-time conversation applications and, trust-based communications. Next, it is presented the model based on RTTs, which may be used by an adversary to perform a side-channel timing attack in NDN.

Adversary Model. Considering the model illustrated in Fig. 4, let us suppose that the side-channel timing attack RTT measurement for a content is RTT_2 (retrieve content from NDN producer), RTT_1 is the RTT from the closest NDN router, RTT_e is the expected RTT of the intended content lookup, and ε is a negligible time difference. After collecting the timing samples, the adversary decides based on the following conditions [4]:

- If $|RTT_e - RTT_1| < \varepsilon$, the adversary node concludes that content has been cached by the closest router.
- If $|RTT_e - RTT_2| < \varepsilon$, the intended content is not held by any router, except the content producer.
- If $RTT_e > RTT_2$ and $RTT_e < RTT_1$, the adversary concludes that the lookup content has been fetched by away routers. Note that, the adversary can still predict the content location by relying on RTT_1 and RTT_2 values.
- $|RTT_2 - RTT_1| \gg \varepsilon$.

4 Countermeasures

This section presents suitable countermeasure methods to mitigate side-timing actions on the cache. These methods manipulate the RTT of the content replied by the CS. The countermeasure methods are basically classified in three groups: no caching, artificial delay, and random caching. These methods are described next.

4.1 No Caching

In NDN, the CS can be configured for not caching. Since, there is no content held in the cache, the side-channel timing attack cannot be done. However, the cache is important for the NDN, as it is required for content distribution. So, directly giving up of the caching is not a good option in NDN, as described in [1].

4.2 Artificial Delay

Data delivery in the NDN is affected by a certain delay impose by the routers. An additional artificial delay can be used as a solution to prevent side channel timing attack, as explained next. Let us consider Δ the default delay value chosen randomly by a router. In side-channel timing attack, the adversary tries to figure out the Δ value. To complicate the adversary goal, a delay τ is added to Δ , in order to increase the router response delay. By measuring an higher RTT value, the adversary supposes that the content is not retrieved from the CS. Nevertheless, this is still a challenging problem, because the adversary may find the delay τ in a period of time and have success in the attack. The value τ can be changed by proposed algorithms, as described in [11]. Instead of using a constant τ , Schinzel [11] proposed a τ value based on a cryptographic hash function. Note that the delay methods imply a trade-off between privacy and latency, as a higher τ value affects negatively the latency on NDN.

4.3 Random Cache

A NDN router can cache the contents randomly, in order to mitigate side-channel timing attack. For instance, the CS may cache one data packet and then may not cache the next incoming data packet, based on a random probability number (k), as proposed in [1]. In such a random caching design, the side-channel timing measurement would fail for the contents not cached.

5 Implementation

In order to analyze the side-channel timing attack, a simulation scenario¹ was implemented on the simulator ndnSIM v2.4 [9]. We used a part of the ISP

¹ Code at: <https://github.com/ertugd/ndnSIM-side-channel-timing-attack.git>.

(Internet Service Provider) map dataset (SIGCOMM2002) by using rocketfuel mapper [12], which is used to convert and read large data sets for ndnSIM. As shown in Fig. 5, the physical topology is formed by sixteen consumers (called leaf), eight backbones (bb), eight central gateway routers (cgw) and one producer (gw-root). Two adversary nodes (leaf 6 and leaf 13) are located randomly into the topology. The producer payload size was 1024 bytes. The bandwidth of the links were 10 Mbps, 100 Mbps and 1000 Mbps, as indicated in Fig. 5. The minimum and maximum delays of the links are presented in Table 1 and were obtained from the data set of a real topology. NFD was configured for best-route strategy, in order to have the best network paths to the consumer nodes. The contents are created by the producer (gw-root) and the adversary nodes lookup for the intended contents on the gateway CS. Once a legitimate leaf node publishes an interest, the first lookup is done in the gateway router. However, if the data is not cached in the gateway router, then the lookup will be made in the backbone routers. The adversary nodes measure the RTT for the lookup contents that the legitimate nodes have requested. In our design, the attack is targeted to the gateway routers that manage interest packets by PIT and cache contents from the backbone routers.

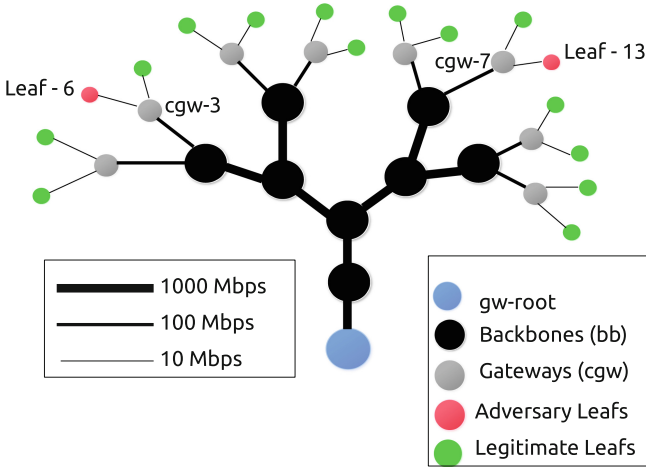


Fig. 5. Physical tree topology of the simulation scenario

Table 1. Delays (min/max) between nodes linked directly

Delays (ms)	bb	cgw	leafs	gw-root
bb	2.51/7.56	3.11/9.10	-	4.77
cgw	3.11/9.10	-	0.15/9.67	-

The adversary nodes sent interest packets at a rate of 100 packets/s with a malicious interest prefixes, and the legitimate nodes at a rate 100 packets/s sending unique (not requesting same content name again) interest prefixes. The legitimate leaf nodes were configured to generate interest traffic with a randomized uniform (7 leaves) pattern and an exponential pattern (7 leaves). The legitimate nodes sent interest packets (size = ~ 40 kB) with a randomized uniform pattern (0, 1/frequency), and exponential distribution (mean of $1/\text{frequency}$), as shown in Table 2. Every backbone and central gateway router caches the contents (data packet size = ~ 110 kB), and the popular LRU (Least Recent Used), with a capacity of 1000 data packets per node, was chosen for the CS policy.

Table 2. Interest configurations

	Frequency (s^{-1})	Traffic pattern	Prefix
Legitimate leaves	100	Uniform and exponential	/google.com/sub_prefix
Leaf-6	100	Uniform	/google.com/%FE%01
Leaf-13	100	Uniform	/google.com/%FE%07%96

6 Results

The primary results showed that the adversary may retrieve the cached contents that were recently requested by the consumer. The NFD is located on each router to analyze metrics for RTT, and cache hits or misses per node.

6.1 Cache Analysis

If a content has been requested and cached, the next request for the same content causes a cache hit. The cache hit ratio (Eq. 2) is used to indicate a side-channel timing attack, it is calculated by cache hit and misses by a predefined time interval Δ . The adversarial leaf 6 and leaf 13 sent a lookup interest packet to cgw-3, 7. The lookup is done for cached contents, which are the contents recently requested by neighbor nodes (leaf 5, leaf 14).

The legitimate leaves sent interest packets with unique prefixes. Therefore, the cache hit ratio is not possible to be calculated for the leaves, because these are not re-consuming the content from the cgw CS. As Fig. 6 illustrates, the adversary leaves hit the cache during an attack period, so their cache hit ratios may reach 100%.

$$\text{Cache hit ratio} = \frac{\sum_{i=0}^n \text{cache_hit}_i}{\sum_{i=0}^n \text{cache_hit}_i + \sum_{j=0}^m \text{cache_miss}_j} * 100\% \quad (2)$$

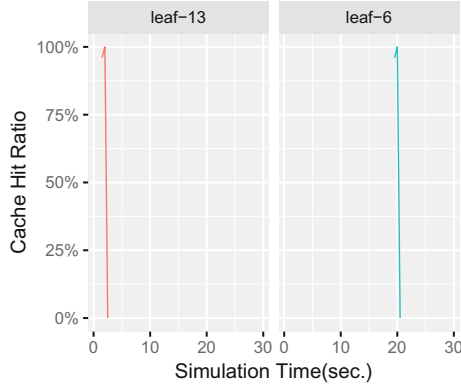


Fig. 6. Effect of a timing attack on the cache hit ratio

6.2 Countermeasure Algorithm

The countermeasure methods affects surely the CS performance. In order to improve the CS performance, we proposed a new countermeasure algorithm (Algorithm 1). The algorithm proposes a method to identify an adversary node and to apply the countermeasure mechanism. It requires that NFD is running in each router. The proposed algorithm examines the RTT and the cache hit ratio metrics to identify the adversary node. The first detection level is done by the RTT threshold. This threshold is calculated and updated for every consumer using Eq. 3, where n is the number of RTT samples obtained periodically during a predefined time interval (*e.g.* $\Delta = 0.5$ s). The detection decision is based on the RTT threshold that is expected from consumers under no attack condition. In case of a consumer RTT value being much lower than the RTT threshold, the router is set to random caching for that consumer.

The algorithm also provides a second level detection method by using the cache hit ratio, in order to identify adversary node. The cache hit ratio is calculated periodically during the same time intervals used to calculate the RTT threshold. The NFD considers the node as an adversary, if its cache hit ratio is above the cache hit ratio threshold (Eq. 4). If the cache hit ratio of a node is above the cache hit ratio threshold, the NFD sets the node for random caching. If the RTT of a node is above the RTT threshold and the cache hit ratio is considerably lower than the cache hit ratio threshold, then the NFD configures the router with the LRU policy.

$$\text{RTT threshold} = \frac{\sum_{i=0}^n \text{RTT}_i}{n} \quad (3)$$

$$\text{Cache hit ratio threshold} = \frac{\sum_{i=0}^m \text{CacheHitRatio}_i}{m} \quad (4)$$

```

while(true) {
    newRTTavailable = checkNewRTTavailable()
    newCacheHitRatioAvailable = checkNewCacheHitRatio()

    if(newRTTavailable) {
        RTT = getRTTfromNFD()
        RTTthreshold = calculateRTTthreshold(RTT) //Eq.3
        checkAttack(RTT, RTTthreshold, oldRTTthreshold, cacheHitRatio,
        cacheHitThreshold, oldCacheHitThreshold)
    }

    if(newCacheHitRatioAvailable) {
        cacheHitRatio = getCacheHitRatiofromNFD() //Eq.2
        cacheHitThreshold = calculateCacheHitThreshold(cacheHitRatio)//Eq.4
        checkAttack(RTT, RTTthreshold, oldRTTthreshold, cacheHitRatio,
        cacheHitThreshold, oldCacheHitThreshold)
        oldCacheHitThreshold = cacheHitThreshold
    }

    if(newRTTavailable) oldRTTthreshold = RTTthreshold
}

function checkAttack(RTT, RTTthreshold, oldRTTthreshold, cacheHitRatio,
    cacheHitThreshold, oldCacheHitThreshold) {
    state = NO_ATTACK

    if (RTT < RTTthreshold OR cacheHitRatio > cacheHitThreshold) {
        state = ATTACK_DETECTED
        router_random_cache() //apply CS random caching
    }

    else {
        state = NO_ATTACK
        router_LRU_cache() //apply CS LRU caching
    }

    if (RTTthreshold < oldRTTthreshold OR cacheHitRatio >
    oldCacheHitThreshold) {
        state = ATTACK_DETECTED
        router_random_cache() //apply CS random caching
    }
}

```

Algorithm 1. Adversary detection

6.3 Timing Measurements

In order to analyze the values of both the RTT samples and the RTT threshold, we ran the scenario considering both the attack and no attack. As shown in Table 3, the RTT threshold values were used to identify the attack of the adversarial node. The attack is detected when the RTT of the sample is below the

RTT threshold. Then, we ran the scenario with under attack and collected the first RTT samples. The results between threshold and first samples are showed that RTT values may change, because of real throughput delays and congestions. However, both the expected RTT under no attack and the RTT variation (regarding the RTT threshold) reduced dramatically for leaf 6 and leaf 13. Therefore, these leaves are considered adversarial nodes. The experimental results showed that the RTT of the adversarial leaves are shorter than the RTT of the legitimate leaves. When this occurs, the NFD may engage the random caching against the attack.

Table 3. RTT analysis

Leaf	Estimated RTT threshold(s) no attack	First RTT sample(s)	RTT variation (%)
1	0.03122045	0.03099957	-0.71%
2	0.03124	0.03100993	-0.74%
3	0.03123136	0.0310034	-0.73%
4	0.03122998	0.03100182	-0.73%
5	0.03122413	0.03099494	-0.73%
6	0.03122487	0.0194212	-37.80%
7	0.03122427	0.03099683	-0.73%
8	0.03122295	0.030993578	-0.73%
9	0.03123878	0.03101116	-0.73%
10	0.03123794	0.03100934	-0.73%
11	0.03123692	0.03100801	-0.73%
12	0.03123955	0.03101118	-0.73%
13	0.03123717	0.0144831	-53.64%
14	0.03123604	0.03100802	-0.73%
15	0.03123482	0.03100664	-0.73%
16	0.03123906	0.03101061	-0.73%

7 Related Work

The related works considered the side-channel attacks on NDN.

Dogruluk *et al.* [6] evaluated the privacy attack in NDN, and proposed an adversary detection method controlled by the values of cache and interest hit ratios.

Felten and Schneider [7] evaluated side-channel timing attack measurements on web privacy. They show that the malicious web cookies can be used to acknowledge user's recent visited web sites. They also concluded that web

anonymization tools and turning off the browser’s JavaScripts features do not prevent side-channel timing attack measurements.

Zhang *et al.* [15] investigated the use of side-channel timing attack on VoIP privacy and how adversary may reveal the call history attacking with legitimate Public Key Infrastructure (PKI).

DiBenetto and Gast [5] developed (ANDāNA) a tool to mitigate timing attacks in NDN. With this tool, the requested names are encrypted and afterwards are verified by the nodes and delivered to the user as data. ANDāNA is based on a TOR (The Onion Routing) paradigm [13], an anonymity browsing tool for Internet. ANDāNA makes the CS unpractical, because the data packet is only consumable for who requests it.

Mohaisen *et al.* [10] focused on cache privacy on ICNs (Information-centric networks) and proposed an user driven countermeasure method called Vanilla. For privacy-sensitive contents, an edge router caches the content from the producer and keeps the retrieval times of the first interest and delay the next coming requests. However, the per-client solution will not be feasible, because of the large number of consumers. Acs *et al.* [1] addressed side-channel timing attacks in NDN and proposed random caching and delay. This user driven method is based on flagging contents by */private* and */non-private*. Not with standing, we state that all contents have a sense of privacy, even daily visited sites such as google.com. To the best of our knowledge, per-client driven solution may not be the most appropriate approach for CS propose. For instance, the */private* content of a user could be stated as a */non-private* for other user(s).

8 Conclusions

In order to point out side-channel timing attack in NDN, we developed a scenario and show primary findings. The results show that the adversary may distinguish contents between cached and non-cached, by comparing content RTT differences. An adversary detection method algorithm is presented. The NFD based algorithm, distinguishes adversary and legitimate node by checking RTT and cache hit ratio threshold values.

The side-channel timing attack is easy to be implemented by an adversary, but the detection is a challenging issue to solve. Ideally, the cache hit and RTT metrics can be also used as an indications for side-channel timing attack. In order to keep CS performance, these indication methods can be used to identify adversary node controlled by NFD application. Through this identification method, the countermeasures (delay, random caching, and no cache) can be applied against the adversary nodes, in order to improve the CS performance.

Acknowledgment. This work has been supported by COMPETE: POCI-01- 0145-FEDER-007043 and FCT-Fundacao ao para a Ciencia e Tecnologia within the Project Scope: UID/CEC/00319/2013.

References

1. Acs, G., Conti, M., Gasti, P., Ghali, C., Tsudik, G.: Cache privacy in named-data networking. In: IEEE 33rd International Conference on Distributed Computing Systems, pp. 41–51. IEEE (2013)
2. Afanasyev, A., Shi, J., Zhang, B., Zhang, L., Moiseenko, I., Yu, Y., Shang, W., Huang, Y., Abraham, J.P., Dibenedetto, S., Fan, C., Pesavento, D., Grassi, G., Pau, G., Zhang, H., Song, T., Abraham, H.B., Crowley, P., Amin, S.O., Lehman, V., Wang, L.: NFD developer’s guide. NDN. Technical report. NDN-0021 4, pp. 1–56 (2015)
3. April, M., Report, A., Jacobson, V., Burke, J., Zhang, L., Claffy, K., Papadopoulos, C., Wang, L., Halderman, J.A., Crowley, P.: Named Data Networking Next Phase (NDN-NP) Project May 2014–April 2015 Annual Report (2015)
4. Chaabane, A., Cristofaro, E.D.: Privacy in content-oriented networking: threats and countermeasures. *ACM SIGCOMM Comput. Commun. Rev.* **43**(3), 26–33 (2013)
5. DiBenedetto, S., Gasti, P.: ANDaNA: anonymous named data networking application. In: Proceedings of the Network and Distributed System Security Symposium, pp. 1–20 (2012)
6. Dogruluk, E., Costa, A., Macedo, J.: Evaluating privacy attacks in named data network. In: Proceedings of the IEEE Symposium on Computers and Communication, vol. 2016, August 2016
7. Felten, E.W., Schneider, M.A.: Timing attacks on web privacy. In: Proceedings of the 7th ACM Conference on Computer and Communications Security, CCS 2000, pp. 25–32 (2000)
8. Jacobson, V., Smetters, D.K., Thornton, J.D., Plass, M., Briggs, N., Braynard, R.: Networking named content. *Commun. ACM* **55**(1), 117 (2012)
9. Mastorakis, S., Afanasyev, A., Moiseenko, I., Zhang, L.: ndnSIM 2.0: a new version of the NDN simulator for NS-3, pp. 1–8 (2015)
10. Mohaisen, A., Mekky, H., Zhang, X., Xie, H., Kim, Y.: Timing attacks on access privacy in information centric networks and countermeasures. *IEEE Trans. Dependable Secur. Comput.* **12**(6), 675–687 (2015)
11. Schinzel, S.: An efficient mitigation method for timing side channels on the web. In: 2nd International Workshop on Constructive Side-Channel Analysis and Secure Design, pp. 1–6 (2011)
12. Spring, N., Wetherall, D.: Measuring ISP Topologies with Rocketfuel. In: Proceedings of the 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, SIGCOMM 2002, pp. 133–145 (2002)
13. Wiangsripanawan, R., Susilo, W., Safavi-Naini, R.: Design principles for low latency anonymous network systems secure against timing attacks. In: Conferences in Research and Practice in Information Technology Series, vol. 68, pp. 183–191 (2007)
14. Yi, C., Afanasyev, A., Wang, L., Zhang, B., Zhang, L.: Adaptive forwarding in named data networking. *ACM SIGCOMM Comput. Commun. Rev.* **42**(3), 62 (2012)
15. Zhang, G., Fischer-Huebner, S., Martucci, L.a., Ehlert, S.: Revealing the calling history of SIP VoIP systems by timing attacks. In: 2009 International Conference on Availability, Reliability and Security, pp. 135–142 (2009)

16. Zhang, L., Estrin, D., Burke, J., Jacobson, V., Thornton, J.D., Smetters, D.K., Zhang, B., Tsudik, G., Massey, D., Papadopoulos, C., Wang, L., Crowley, P., Yeh, E.: Named data networking (NDN) project. NDN, Technical report NDN-0001, pp. 1–26, October 2010
17. Zhang, L., Jacobson, V., Diego, S., Crowley, P., Louis, S., Wang, L.: Named data networking. *ACM SIGCOMM Comput. Commun. Rev.* **44**(3), 66–73 (2014)