2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing

Detection and Prevention of Malicious Requests in ICN Routing and Caching

Eslam G. AbdAllah, Mohammad Zulkernine, and Hossam S. Hassanein
School of Computing, Queen's University
Kingston, ON, Canada
Email: {eslam, mzulker, hossam}@cs.queensu.ca

***Abstract*—Information Centric Networking (ICN) is a new communication paradigm for the upcoming Next Generation Internet (NGI). ICN is an open environment that depends on in-network caching and focuses on contents rather than infrastructures or end-points as in current Internet architectures. These ICN attributes make ICN architectures subject to different types of routing and caching attacks. An attacker sends malicious requests that can cause Distributed Denial of Service (DDoS), cache pollution, and privacy violation of ICN architectures. In this paper, we propose a solution that detects and prevents these malicious requests in ICN routing and caching. This solution allows ICN routers to differentiate between legitimate and attack behaviours in the detection phase based on threshold values. In the prevention phase, ICN routers are able to take actions against these attacks. Our experiments show that the proposed solution effectively mitigates routing and caching attacks in ICN.**

I. Introduction

In the 1970s, the Internet was designed to exchange information between well-identified hosts. Nowadays, the Internet has changed extremely into the Internet of things, Internet of services, Internet of people and Internet of media. These new Internets come with new requirements of highly scalable and efficient distribution of contents. These new requirements lead to the idea of Information Centric Networking (ICN), which is one of the alternatives for the upcoming Next Generation Internet (NGI) [1]. ICN has many unique attributes such as location independent naming, in-network caching, name-based routing and built-in security. Many architectures are proposed for ICN paradigms. The most referenced architectures are Named Data Networking (NDN), Publish Subscribe Internet Technology (PURSUIT), Data Oriented Network Architecture (DONA), and Network of Information (NetInf) [2]. These architectures have some basic components: information object, naming, routing, caching, security, and application programming interface [3], [4].

ICN is based on asynchronous publications and subscriptions. A publisher sends publication messages with content names to notify ICN network that he/she has contents to share. A subscriber sends subscription messages for contents using also content names. Publishers do not know who may be interested in their contents and subscribers do not know who have published these contents. ICN makes a delivery path between a publisher and a subscriber, when there is a match between a publication and a subscription, then ICN content is transferred from a publisher to a subscriber. If there are other requests for the same content, ICN tries to deliver this content using in-network caching [11], [12].

ICN is an open environment, which means that any user can publish or subscribe contents. An attacker can send a large number of malicious requests for available or unavailable contents to overload an ICN infrastructure and fill up ICN routing tables. ICN routers forward these malicious requests to their neighbouring routers, which in turn send the malicious requests to their neighbouring routers and so on. These types of attacks can be performed on a large distributed scale to cause Distributed Denial of Service (DDoS) for legitimate users [6].

ICN also depends on in-network caching, i.e., any node can cache contents to deliver contents to subscribers from the closest available copy. An attacker can send random and unpopular requests to force ICN caches to store unpopular contents and evicts the popular ones and hence an attacker violates ICN caching system. An attacker can also use the time difference between cached and uncached contents to acquire private information about ICN contents and his/her proximate users.

In our earlier work [2], we presented a taxonomy for ICN attacks that classifies a broad range of ICN attacks into four categories: naming, routing, caching, and other miscellaneous related attacks. In [7], we proposed detection and prevention techniques for routing related DDoS attacks. These techniques cover malicious publications and subscriptions for ICN routing. In this paper, we improve the calculation method of threshold values and the algorithms used in the detection and prevention of subscription attacks. Additionally, we consider ICN caching related attacks. By malicious requests, we mean the following: fake requests for unavailable contents; large number of requests for available contents; requests for changing contents popularities; requests for violating the privacy of ICN contents and users.

An attacker can easily send a large number of malicious requests using any ICN architecture naming and routing schemes. Hence, our solution is based on ICN generic components that can be applied into any ICN architecture. This solution detects and prevents malicious requests in ICN routing and caching. We first study the behaviours of legitimate users and calculate threshold values for the following parameters: Request Satisfaction Ratio (RSR), request rate, and cache hit ratio. RSR represents the number of satisfied requests per interface with respect to the outgoing requests from this interface. RSR depends on an ICN property that each request has only one response and there is no response without a request [7]. In non-ICN environments, a request can receive many data packets. Request rate indicates the number of outgoing requests per second for each interface. Cache hit ratio is the number of cache hits per interface with respect to the outgoing requests from this interface. In the detection phase,

978-1-5090-0154-5/15 $31.00 © 2015 IEEE
DOI 10.1109/CIT/IUCC/DASC/PICOM.2015.262

we compare users' behaviours against the identified threshold values and specify the attack case. In the prevention phase, we apply our countermeasure to mitigate these attacks.

The solution is evaluated using NS-3 based module named ndnSIM, which is a simulator for NDN architecture used as a proof of concept. Our results show that the solution mitigates these attacks and enhances ICN performance in the existence of these attacks.

The remainder of the paper is organized as follows. Section II presents the related work in ICN routing and caching attacks. Section III presents an attack model for ICN routing and caching. Section IV presents our countermeasures for mitigating these ICN attacks. Section V shows the simulation results of the proposed solutions. Finally, Section VI draws conclusions.

II. Related Work

There exist previous work that address some types of routing and caching attacks in ICN. Gasti et al. [8] present a high level classification of DDoS attacks and their solutions in NDN architecture. For routing attacks, Afanasyev et al. [9] present request flooding DDoS attack for unavailable contents. They propose three mitigation strategies based on RSR and recommend satisfaction-based pushback mechanism as a mitigation technique. Compagno et al. [10] also address request flooding DDoS attack for unavailable contents. They define threshold values for RSR and Pending Interest Table (PIT) space in NDN architecture and then limit incoming requests based on these threshold values. In both solutions, an attacker can send requests for available contents to overcome these solutions or at least decrease the solution's effectiveness. Our solution for the routing part considers malicious requests for available and unavailable contents. The solution is based on threshold values for RSR, request rate, and cache hit ratio. These three threshold values enable ICN routers to efficiently differentiate between legitimate and attack behaviours. Our rate limiting is based on attacker request rate, in addition to the threshold values.

For caching attacks, Mauro et al. [12] present a lightweight mechanism for the detection of cache pollution attacks in named data networking. The detection is based on a threshold value calculated by the number of references for each content and their variations. The Cacheshield solution [13] handles random requests for ICN caching. Cacheshield uses a shield function that determines whether to cache contents or not at ICN routers. Cacheshield is tested on small scale networks and stores content names and statistics about contents in ICN caching. Aziz et al. [14] present time analysis attack in ICN and provide countermeasures that do not detect the attacked interfaces. Our solution is different from these solutions as follows: caching contents is based on decisions coming from our detection phase according to user behaviours and threshold values; ranking cached contents is based on their popularities and evicting the least popular ones in case of cache replacements; detecting attacked interfaces when privacy violation happens in the proximity of an attacker.

Fotiou et al. [15] present a ranking algorithm for ICN contents to fight publication spam based on publisher and subscriber ranking. Ranking is based on the number of publications for publishers and the number of votes for subscribers. The authors assume many restrictions in ICN environment such as anonymity of ICN users and the existence of authentication servers. Ghali et al. [16] present a ranking algorithm for ICN cached contents to mitigate content poisoning in NDN architecture. Ranking is based on user behaviours after receiving content objects. These work focus on content poisoning from the publisher side; while in this paper, we focus on impacts of malicious requests from the subscriber side in ICN caching. Our ranking calculates cached contents popularities based on the weights of user requests and the ratio of requesting contents. Our solution is implemented in ICN routers and it does not require modifications to ICN architectures.

In the literature of DDoS attacks in general, there are many classifications for DDoS attacks and detection/prevention mechanisms [17], [18]. In current Internet architectures, most referenced countermeasures for DDoS are IP trace back, packet filtering, and rate limiting [2]. Also, there are other countermeasures for cache poisoning attacks as in DNSSEC and S-DNS [19]. Unfortunately, these techniques cannot be used in ICN architectures, because these techniques depend on IP addresses.

The proposed solution in this paper is designed specifically for ICN architectures, as it depends on unique ICN characteristics. The solution does not include any host addresses, which means that the solution does not depend on any IP-based addressing as in non-ICN environments. The solution depends on in-network caching, which is one of the major ICN components that is not available in non-ICN environments. The solution includes RSR metric, which depends on the ICN property that each request has one response and there is no response without a request. In non-ICN environments, requests can receive many data packets.

III. Attack Model

In this section, we present ICN routing and caching attacks. An attacker performs these attacks to achieve the following goals:

- DoS by increasing the request timeout for some ICN nodes to violate the consistency between ICN asynchronous publication and the subscription process.
- Cache pollution by forcing ICN caching to store unpopular contents, and consequently satisfying all requests from the original sources rather than the closest available copies.
- Privacy violations by getting private information about the requested contents and requesters in an attacker's domain.

A. Routing attacks

As shown in Figure 1, an attacker *A1* who controls many end systems, sends a large number of malicious requests to an ICN to exhaust ICN resources such as memory and processing power. An attacker aims to fill ICN routing tables to cause DDoS for legitimate users. These malicious requests can be sent for available and unavailable contents. The attacked routers try to satisfy these malicious requests and forward them to neighbouring routers, which in turn forward these malicious requests to their neighbouring routers and so on. In this case, legitimate requests take longer response times to be satisfied. If response time exceeds a certain threshold, legitimate requests

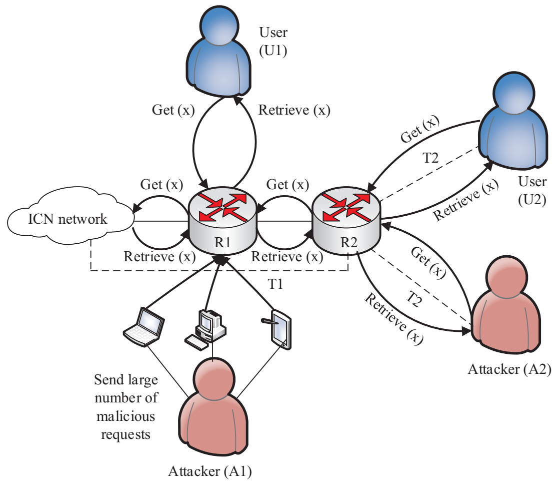


Fig. 1: ICN routing and caching related attacks

will not be satisfied. The impacts of this attack is amplified in ICN, because users retransmit unsatisfied requests, which add extra overload to ICN network. This scenario may lead to denial of service or at least long delays. In this paper, we address both malicious scenarios for available and unavailable contents.

B. Caching attacks

1) Random/unpopular requests: An attacker sends random or unpopular requests to spoil ICN caching by changing content popularities. These malicious requests force a caching system to store least popular contents and evict the popular ones. As depicted in Figure 1, normally when a user *U1* requests for a content for the first time, ICN responds with the content from the original source. If another user *U2* makes a request for the same content, the second user gets it from the closest available copy in router *R1* instead of the original source. If an attacker *A1* succeeds to violate ICN caching, and a second user *U2* requests for the same content, the second user gets the content from the original data source instead of the closest available copy. The second user's request takes the full path as the first user's request in the attack case.

2) Time analysis attack: In ICN, in-network caching enables any node to cache contents. An attacker can send requests for cached and uncached contents and measure time differences between the two cases. This time difference can be used to detect if a near-by user has requested a certain content before or not. As depicted in Figure 1, time required to send a request and receive a response between original data source and user's or attacker's edge router *R2* is T_1. Time required to send a request and receive a response between user *U2* or attacker *A2* and edge router *R2* is T_2. In a normal case, total time T_1+T_2 refers to round trip time from user *U2* or attacker *A2* to the original data source. In an attack case, if the attacker *A2* receives a response in only time T_2, he/she can use this time difference to violate ICN privacy for both contents and users. Attacker *A2* can know private information about content popularities and whether a proximate user has requested this content before or not. This attack is one of the caching related attacks that causes privacy violation. In this paper, we are concerned only with this attack, not with other ICN privacy issues related to ICN contents or users.

C. Attack scenarios

It is commonly agreed that web traffic follows Zipf-like distribution [20]. Other research shows that the Internet follows stretched exponential (SE) distribution [21]. Whichever the distribution is, an attacker performs the aforementioned attacks by changing request rate and request pattern (distribution) with respect to legitimate users. Table I shows the different scenarios and combinations of routing and caching attacks. When an attacker sends malicious requests with much higher request rate with the legitimate request pattern, this leads to routing attacks. Malicious requests can be sent for both available and unavailable contents. When an attacker sends malicious requests with different patterns with the legitimate request rate, this leads to caching attacks. Malicious request patterns can be uniform, random, and unpopular. In uniform distribution, an attacker sends the same number of malicious requests for all contents in a data set. In random distribution, an attacker sends random number of malicious requests for contents in a data set. In unpopular distribution, an attacker sends the same number of requests for just unpopular contents in a data set. When an attacker changes request rate and request pattern, he/she performs both attacks simultaneously.

TABLE I: Attack methodology (LB: Legitimate behaviour, AB: Attack behaviour)

		Request rate	
		LB	**AB**
Request pattern	**LB**	Legitimate behaviour	Routing attacks
	AB	Caching attacks	Rotuing and Caching attacks

IV. Our Countermeasures

The proposed solution is implemented in ICN routers and effectively detects and prevents malicious requests in ICN caching and routing based on threshold values as described in the following subsections.

A. Threshold values calculation

In calculating the threshold values, we assume that there are no malicious requests during this calculation. We study the normal behaviour of legitimate users and calculate threshold values for the following parameters:

- Request Satisfaction Ratio (RSR): Request satisfaction ratio for interface i is calculated by the following equation:

$$RSR_i = \frac{number\ of\ satisfied\ requests}{number\ of\ outgoing\ requests} \quad (1)$$

Then we calculate $RSR_{threshold}$ for legitimate users, which refers to minimum legitimate RSR value.

- Request Rate: The request rate for interface i is calculated by the following equation:

$$RequestRate_i = \frac{number\ of\ outgoing\ requests}{sec} \tag{2}$$

Then we calculate $RequestRate_{threshold}$ for legitimate users, which refers to maximum legitimate request rate value.

- Cache Hit Ratio: The cache hit ratio for interface i is calculated by the following equation:

$$CacheHitRatio_i = \frac{number\ of\ cache\ hits}{number\ of\ outgoing\ requests} \tag{3}$$

Then we calculate $CacheHitRatio_{thresholdL}$, $CacheHitRatio_{thresholdU}$ for legitimate users, which refer to minimum and maximum legitimate cache hit ratios, respectively.

These threshold values are calculated using the following procedure. For each time slot (sample) in a dataset, each ICN router records RSR, request rate, and cache hit ratio of different interfaces. For each sample, each ICN router calculates minimum RSR, maximum request rate, and minimum and maximum for cache hit ratio. For all samples, each ICN router calculates mean *($\bar{x}$)* and standard deviation *(s)* for minimums and maximums for each metric separately. Then threshold values are calculated as follows. In case of maximum threshold value, we add mean $\bar{x}$ to a constant multiplied by standard deviation. In case of minimum threshold value, we subtract the standard deviation s multiplied by constant c from the calculated mean $\bar{x}$. The value of constant multiplied by standard deviation detects how far our threshold value is from the calculated mean. The threshold value for RSR is:

$$RSR_{threshold} = \bar{x}_{RSR} - c_1 * s_{RSR} \tag{4}$$

The threshold value for request rate is:

$$\begin{gathered} RequestRate_{threshold} = \\ \bar{x}_{RequestRate} + c_2 * s_{RequestRate} \end{gathered} \tag{5}$$

The threshold values for cache hit ratio are:

$$\begin{gathered} CacheHitRatio_{threshold} = \\ \bar{x}_{CacheHitRatio} \pm c_3 * s_{CacheHitRatio} \end{gathered} \tag{6}$$

The above equation calculates the minimum and the maximum threshold values for cache hit ratios.

The use of more than one threshold value allows the detection and prevention phases to effectively define attacker behaviours. These values monitor attacker behaviours from different perspectives and compare it to legitimate behaviours. For an attacker, it becomes too complicated to perform an effective attack within the legitimate bounds of the threshold values, and this makes the attacks futile.

B. Detection Phase

In this phase, each ICN router is able to differentiate between legitimate and attack behaviours and detect which attack case is happening to take the appropriate actions in the prevention phase. If the request rate of an interface is above $RequestRate_{threshold}$, that means an attacker sends a large number of requests either for available or unavailable contents. In another case, when the request rate of an interface is below $RequestRate_{threshold}$ and at the same time the RSR of this interface is below $RSR_{threshold}$, that means an attacker sends a large number of requests for unavailable contents. In the third case, when the request rate of an interface is below $RequestRate_{threshold}$ and at the same time the cache hit ratio of this interface is below $CacheHitRatio_{thresholdU}$ value, that means an attacker tries to exhaust and overload ICN infrastructure and violates ICN caching to store unpopular contents. In a final case, when cache hit ratio of an interface is above $CacheHitRatio_{thresholdU}$ value, that means an attacker violates ICN privacy by monitoring his/her proximate users. For an attacker to pass these conditions, he/she needs to send malicious requests within legitimate rates, which minimize the impacts of the performed attacks.

C. Prevention Phase

By using the three threshold values, the prevention phase is able to protect ICN against the combinations of malicious requests for either available or unavailable contents in ICN routing and caching and take the appropriate actions. When an attacker sends malicious requests for available and unavailable requests, each ICN router limits the incoming requests from interface i using the following equations:

$$\begin{aligned} RequestRateLimit_i &= RSR_i * RequestRate_{threshold} \\ \textbf{If}\ RequestRate_i &> RequestRate_{threshold} \end{aligned} \tag{7}$$

OR

$$\begin{aligned} RequestRateLimit_i &= RSR_i * RequestRate_i \\ \textbf{If}\ RequestRate_i &< RequestRate_{threshold} \\ \textbf{AND}\ (RSR_i &< RSR_{threshold}\ \textbf{OR} \\ CacheHitRatio_i &< CacheHitRatio_{\mathrm{thresholdL}}) \end{aligned} \tag{8}$$

In the cache pollution case, if there is a cache hit at an ICN router, the ICN router returns content and updates cache hit ratio for the requested interface. In the case of cache miss, an ICN router only cache contents if the following conditions are true, which indicate legitimate behaviours. Otherwise, ICN routers do not cache these contents.

$$\begin{aligned} RSR_i > RSR_{threshold}\ \textbf{AND} \\ RequestRate_i < RequestRate_{threshold}\ \textbf{AND} \\ CacheHitRatio_i > CacheHitRatio_{\mathrm{thresholdL}} \end{aligned} \tag{9}$$

This countermeasure for cache pollution attacks ranks ICN cached contents at each router. ICN caching evicts least popular contents. As the number of user requests increases, the user request weight decreases. The rating depends on RSR, number of user requests, number of requesting interfaces and total number of interfaces. The request weight for cached content c *($W_{cachedcontent}$)* is calculated by the following equation:

$$W_{cachedcontent}(c) = \sum_{i=1}^{n} \frac{R_{Ui}}{number\ of\ U_i\ requests} * RSR_i \tag{10}$$

where U_i is the user who connected to interface i, R_{U_i} is the U_i requests for cached content c, and n is the number of requests. The cache request ratio for content c *($R_{cachedcontent}$)* is calculated by the following equation:

$$R_{cachedcontent}(c) = \frac{number\ of\ requesting\ interfaces}{number\ of\ interfaces} \tag{11}$$

From Equations (10) and (11), the rating for cached content popularity (c) is calculated by the following equation:

$$\begin{aligned} Rating\ for\ cached\ content\ (c) = \\ W_{cachedcontent}\ (c) * R_{cachedcontent}\ (c) \end{aligned} \tag{12}$$

Algorithm 1 Time Analysis Attack Prevention

```
Input: User ($U_i$) sends a request to ICN router
 1: Update CacheHitRatio ($U_i$)
 2: if CacheHitRatio ($U_i$) > $CacheHitRatio_{thresholdU}$ then
 3:     Mark $U_i$ as an attacker
 4:     Add random delays to $U_i$ responses
 5:     AttackedUserList = CheckAttackedUsers ($U_i$)
 6:     if Count (AttackedUserList == 1) then
 7:         Send notification message to $U_k$
 8:     else
 9:         if (1 <Count (AttackedUserList)<Max) then
10:             loop (AttackedUsersList)
11:                 Send notification message for each user
                    ($U_k$) in AttackedUserList
12:             end loop
13:         else
14:             Send broadcast notification message to all
15:         end if
16:     end if
17: else
18:     Handle request normally
19: end if
```

Algorithm 2 Check Attacked Users

```
Input: Attacker ($A_i$)
 1: Get matched requests between $A_i$ and each user $U_k$
 2: TempAttackedUserList = count matched requests group by
    user $U_k$
 3: loop (TempAttackedUserList)
 4:     if count(entry)/NoUsrReq($A_i$)>$CacheHitRatio_{thresholdL}$
    then
 5:         Add user $U_k$ to AttackedUserList
 6:     end if
 7: end loop
 8: return AttackedUserList
```

In case of time analysis attack, Algorithm 1 shows our prevention technique for this attack. Each ICN router records round trip times for cached contents. In case of attack detection, the ICN edge router connected to the requested interface responds with random delays close to the original round trip times. Also, each ICN router sends an alert message to its attacked users. We use U_i to refer to an input user to the algorithm, who will be tested to detect if he/she is an attacker or not. We use U_k to refer to a legitimate user connected to interface k. We check the attacked user list coming from Algorithm 2. Then we send an alert message to the attacked users to notify them that a near-by user is monitoring their requests. Algorithm 2 shows how we detect the attacked users by counting and grouping the common requests between an attacker and each proximate user. We use A_i to refer to an attacker connected to interface i. If the number of matched requests between an attacker A_i and a user U_k exceeds $CacheHitRatio_{thresholdL}$, then we add this user to the attacked user list. In order to prevent time analysis there are two techniques: using random delays close to original response time; generating cache misses for attacker requests. Both techniques have similar impacts from a consumer side, because he/she gets a response after approximately the same time. We prefer the first technique to save network resources instead of getting contents from the original sources everytime.

V. Experimental evaluation

In this section, we evaluate the impacts of routing and caching attacks with and without our solution. We evaluate the solution using ndnSIM, which is a simulator for NDN architecture and it is an NS-3 based module. This simulator for a specific ICN architecture is used as a proof of concept of our generic solution. We build our experiments using backbone AT&T network, which is Internet-like architecture. In our experiments, AT&T consists of 150 subscribers, 10 publishers and more than 40 routers. We use real data trace for web traffic, which contains 1.3 billion web requests recorded at servers for the 1998 World Cup [22]. The simulation parameters are detected based on the real data trace as follows: request rate for legitimate users is 20 request per second, users send 80% new requests and 20% non-modified requests, top 10% of files receive about 90% of requests, about 10% of contents are requested only one time. We set the attacker request rate to be 100 requests per second. We also use the following parameters: no. of PIT entries = 1000 entry, payload size = 1 KB, no. of cache entries = 1000 entry, request expiration timeout = 4 seconds. During the experiments, we start implementing the solution after 10% of simulation time to give users the opportunity to send their requests and then apply the proposed countermeasures. In all caching results, we record the results starting from the fifth second in order to give an opportunity to ICN users to send their requests and build their cache hit ratios before comparing ICN users' behaviours to the threshold values. In the results, the following terminologies are used:

- Baseline: measurement for a certain metric in an ideal case scenario, when there are no attacks and without any add-on solution.
- Positive likelihood rate (PLR): ratio between true positives divided by (100 - false positives). True positives (sensitivity) indicate to probability of an attacker identified truly as an attacker. False positives (specificity) indicate to probability of a legitimate user identified as an attacker. Baseline for PLR is 100%, which means that all attackers are correctly identified without any legitimate users identified as attackers.

In the following subsections *A* and *B*, we present the results and in subsection *C* we discuss these results in detail.

A. Routing attack experiments

We measure satisfied legitimate requests, which is the number of satisfied requests with respect to outgoing requests for legitimate users. This metric is chosen to study the impacts of routing attacks with and without our solution on legitimate users. The main objective of an attacker in these types of attacks is to decrease the number of satisfied legitimate requests.

Hence we use this metric as an indicator of improvement achieved using our solution in the existence of routing attacks. We change the number of attackers to be 20%, 50%, and 80% of legitimate users. Figure 2 compares satisfied legitimate requests for the following cases: baseline case; in existence of attacks; and in existence of attacks with our solution. The results show how the solution enhances satisfied legitimate requests and generate results close to baseline case. According to the real data trace, ratio for satisfied legitimate requests is about 99.5% when there are no attacks (baseline). When 20% attackers exist, satisfied legitimate requests decrease to about 55%. When 50% attackers exist, satisfied legitimate requests decrease to about 25%. When 80% attackers exist, satisfied legitimate requests decrease to about 10%. Based on our solution, satisfied legitimate requests in the existence of these attacks in the worst case is about 91%.

B. Caching attack experiments

1) Random/unpopular request experiments: We measure cache hit ratio as an indication of impacts generated by caching attacks and our solution. The attacker's main objective in these types of attacks is to decrease cache hit ratio to force ICN to retrieve data from the original source every time. We use cache hit ratio as a metric of improvement achieved by our solution in the existence of caching attacks. We change the number of attackers to be 20%, 50%, and 80% of legitimate users. We also change distribution patterns to be uniform, random, and unpopular as shown in Figure 3, Figure 4, and Figure 5, respectively. Each figure compares cache hit ratio among the following cases: baseline case; in existence of attacks; and in existence of attacks with our solution. The results show how the solution enhances cache hit ratio and generates results close to baseline case. According to the real data trace, ratio for cache hits is about 39% when there is no attack (baseline).

In uniform distribution: when 20% attackers exist, cache hit ratio decreases to about 18%. When 50% attackers exist, cache hit ratio decreases to about 13%. When 80% attackers exist, cache hit ratio decrease to about 10%. Based on our solution, cache hit ratio in the existence of these attacks is about 51%, which enhances cache hit ratio over the baseline.

In random distribution: the cache hit ratio decreases to about 19%, 16%, and 15% when there are 20%, 50%, and 80% attackers exist, respectively. The cache hit ratio using our solution and in the existence of these attacks is about 55%, which also enhances cache hit ratio over the baseline.

In unpopular distribution: without our solution and in the existence of the attacks, cache hit ratios are about 14%, 10%, and 9% when 20%, 50%, and 80% attackers exist, respectively. With our solution and in the existence of these attacks, the cache hit ratio is enhanced to about 61%, which is also better than the baseline.

2) Time analysis experiments: We measure cache hit ratio for attackers with and without the solution. In time analysis attack, we perform our experiments using only 10% attackers. The main objective for an attacker is to measure the time difference between cached and uncached contents. We are interested in caching impacts instead of number of attacker's requests. Within the time expiration period, the attacker sends his/her requests in a continues manner to monitor the neighbouring users. As shown in Figure 6, when 10% attackers exist and send specified requests for certain contents, cache hit ratio increases to about 47%. Based on our solution, cache hit ratio in the existence of these attacks is about 42%, which decreases attackers' hit ratios.

C. Discussion

As observed in the experimental results, our solution enhances ICN performance in routing and caching attack cases. With respect to related work [8], [9], [10], [12], [13], our solution achieves better or similar results. In [13], the results for cache hit ratio are better than ours, however as mentioned in [12], their experiments are performed on small topologies.

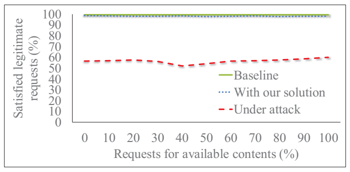
For an attacker, sending requests for unavailable contents is much easier than for available contents. In routing attack experiments, an attacker can achieve similar impacts when he/she sends malicious requests for available or unavailable requests. The reason behind this is that ICN tries to satisfy requests from different paths, which prevents legitimate requests from being satisfied. This scenario is similar to sending malicious requests for available contents preventing legitimate contents from being delivered.

In caching experiments, cache hit ratio is enhanced over the baseline as we apply our ranking technique, detection and prevention phases. ICN caching now caches the most popular contents and evicts the least popular ones. In normal ICN behaviour, each node caches any content passing through this node in on-path caching techniques. In off-path caching techniques, neighbouring nodes also can cache these contents. Using our solution, we decrease caching rate for unpopular contents and in case of any cache replacements, the least popular ones will be evicted. This is the main reason of our enhancement over the baseline case. As depicted in Figure 3, Figure 4, and Figure 5, unpopular distribution is the simplest caching attack to be detected. Uniform distribution is considered to be the most difficult caching attack to be detected. The primary reason is that cache hit ratio of an attacker in the uniform distribution is greater than the cache hit ratio in the unpopular distribution.

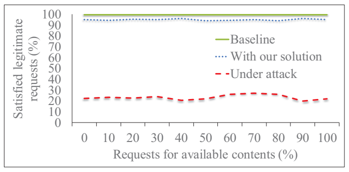
VI. Conclusion

ICN is considered as an alternative for future Internet, which comes with new features and new challenges. Any user can publish and subscribe to ICN contents, which allows attackers to perform DDoS attacks. DDoS attacks have great impacts on ICN routing that degrades ICN performance and availability. In-network caching is one of the prominent attributes in ICN. An attacker can spoil ICN caching or use time difference between cached and uncached content to violate ICN privacy.

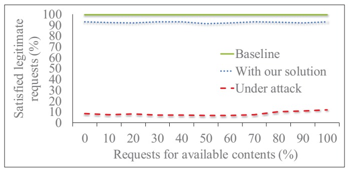
In this paper, we present different scenarios of malicious requests in ICN routing and caching. We also present our solution that detects and prevents these malicious requests. We start by calculating threshold values for RSR, request rate, and cache hit ratio. We compare user's request rate and pattern with respect to these threshold values. Then we prevent these attacks using appropriate actions based on the attack case. We evaluate our solution using real data trace and in the existence of 20%, 50%, and 80% attackers with respect to legitimate users. Our results show that the solution succeeds to mitigate these attacks and enhances ICN performance in the existence of all attack cases.



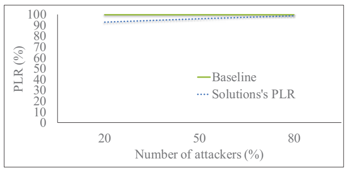
(a) 20% attackers



(b) 50% attackers

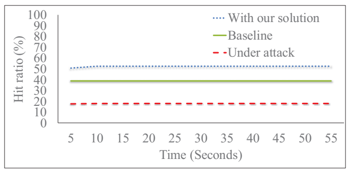


(c) 80% attackers

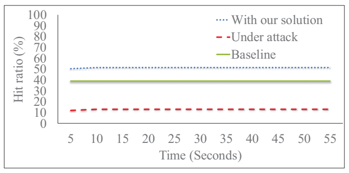


(d) Positive likelihood ratio (PLR)

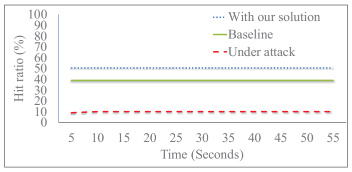
Fig. 2: Routing attacks and countermeasures



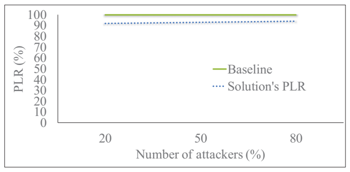
(a) 20% attackers



(b) 50% attackers

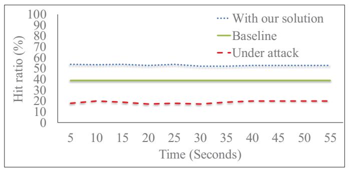


(c) 80% attackers

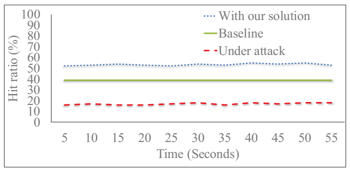


(d) Positive likelihood ratio (PLR)

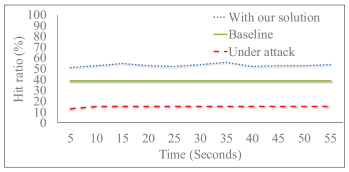
Fig. 3: Caching attacks and countermeasures: uniform distribution



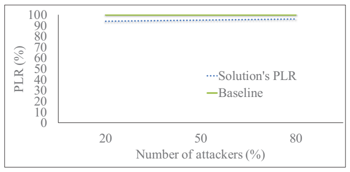
(a) 20% attackers



(b) 50% attackers



(c) 80% attackers



(d) Positive likelihood ratio (PLR)

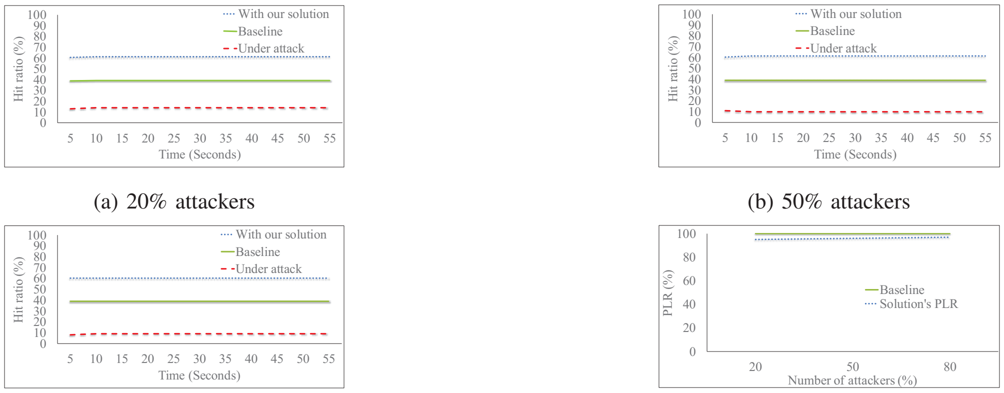
Fig. 4: Caching attacks and countermeasures: random distribution

Acknowledgment

This research has been partially funded by Mitacs Canada.

References

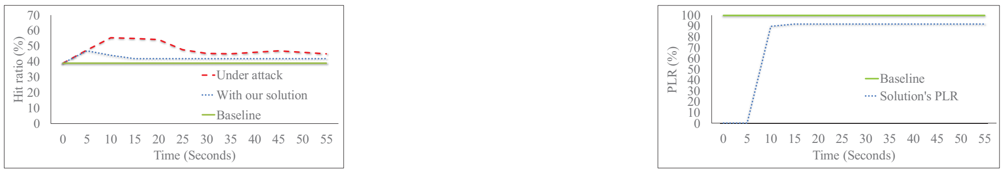
[1] J. Pan, S. Paul, and R. Jain, ”A survey of the research on future internet architectures”, *IEEE Communications Magazine*, vol. 49, no. 7, July



(a) 20% attackers (b) 50% attackers

(c) 80% attackers (d) Positive likelihood ratio (PLR)

Fig. 5: Caching attacks and countermeasures: unpopular distribution



(a) Cache hit ratio (b) Positive likelihood ratio (PLR)

Fig. 6: Time analysis attacks

2011, pp. 26-36.

[2] E. G. AbdAllah, H. S. Hassanein, and M. Zulkernine, ”A Survey of security attacks in information-centric networking”, *IEEE Communications Surveys & Tutorials*, vol. 17, no. 3, August 2015, pp. 1441-1454.

[3] Md. F. Bari, S. R. Chowdhury, R. Ahmed, R. Boutaba, and B. Mathieu, ”A survey of naming and routing in information-centric networks”, *IEEE Communications Magazine*, vol. 49, no. 12, December 2012, pp. 44-53.

[4] C. Tsilopoulos, X. Vasilakos, K. Katsaros, G. Polyzos, G. Xylomenos, C.Ververidis, V.Siris, and N. Fotiou, ”A survey of information-centric networking research”, *IEEE Communications Surveys & Tutorials*, vol. 16, no. 2, 2013, pp. 1024-1049.

[5] A. Ghodsi, T. Koponen, B. Raghavan, S. Shenker, A. Singla, and J. Wilcox, ”Information-centric networking: seeing the forest for the trees”, *in Proc. of the 10th ACM Workshop on Hot Topics in Networks*, ACM, 2011, pp. 1-6.

[6] M. Vahlenkamp, M. Whlisch, and T. C. Schmidt, ”Backscatter from the data plane—threats to stability and security in information-centric networking”, *Computer Networks*, vol. 57, no. 16, 2013, pp. 3192-3206.

[7] E. G. AbdAllah, M. Zulkernine, and H. S. Hassanein, ”Countermeasures for mitigating ICN routing related DDoS attacks”, *in Proc. of the 10th International Conference on Security and Privacy in Communication Networks (Securecomm14)*, Beijing, China, Sept. 2014.

[8] P. Gasti, G. Tsudik, E. Uzun, and L. Zhang, ”DoS & DDoS in nameddata networking”, *in Proc. of the 22nd International Conference on Computing Communications and Networks*, IEEE, 2013.

[9] A. Afanasyev, P. Mahadevan, I. Moiseenko, E. Uzun, and L. Zhang, ”Interest flooding attack and countermeasures in named data networking”, *IFIP Networking Conference*, 2013, pp. 1-9.

[10] A. Compagno, M. Conti, P. Gasti, and G. Tsudik, ”Poseidon: Mitigating interest flooding DDoS attacks in named data networking”, arXiV:1303.4823v3 [cs.NI], August 2013.

[11] F. Almeida and J. Loureno, ”Information centric networks-design issues, principles and approaches”, *International Journal of Latest Trends in Computing*, vol. 3, no. 3, September 2012, pp. 58-66.

[12] M. Conti, P. Gasti, and M. Teoli, ”A lightweight mechanism for detection of cache pollution attacks in named data networking”, *Computer Networks*, 2013.

[13] M. Xie, I. Widjaja, and H. Wang, ”Enhancing cache robustness for content-centric networking”, *in Proc. of the IEEE Infocom*, 2012.

[14] A. Mohaisen, H. Mekky, X. Zhang, H. Xie, and Y. Kim, ”Timing attacks on access privacy in information centric networks and countermeasures”, *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 99, August 2014, pp. 1-14.

[15] N. Fotiou, G. F. Marias, and G. C. Polyzos, ”Fighting spam in publish/subscribe networks using information ranking”, *Next Generation Internet (NGI), 6th EURO-NF Conference*, Paris, June 2010, pp. 1–6.

[16] C. Ghali, G. Tsudik, and E. Uzun, ”Needle in a haystack: Mitigating content poisoning in named-data networking”, *SENT 14*, San Diago, CA, USA, 2014.

[17] C. Yi, A. Afanasyev, I. Moiseenko, L. Wang, B. Zhang, and L. Zhang, ”A case for stateful forwarding plane”, *Journal of Computer Communications*, vol. 36, no. 7, April 2013, pp. 779-791.

[18] Y. You, M. Zulkernine, A. Haque, ”A Distributed defense framework for flooding-based DDoS attacks”, *in Proc. of the International Conference on Availability, Reliability and Security*, IEEE CS Press, Barcelona, Spain, 2008, pp. 245-252.

[19] H. M. Sun, W. H. Chang, S. Y. Chang, and Y. H. Lin, ”DepenDNS: Dependable mechanism against DNS cache poisoning”, *Cryptology and Network Security*, Springer, 2009, pp. 174-188.

[20] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker., ”Web caching and zipf-like distributions: evidence and implications”, *IEEE Infocom '09*, New York, USA, 1999, pp. 1-9.

[21] L. Guo, E. Tan, S. Chen, Z. Xiao, and X. Zhang, ”Does internet media traffic really follow Zipf-like distribution?”, *ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, New York, USA, 2007, pp. 359-360.

[22] ”Traces available in the Internet traffic archive”, [Online] http://ita.ee.lbl.gov/html/traces.html, accessed on 20 April 2015.