FROG: A Packet Hop Count based DDoS Countermeasure in NDN

Yoshimichi Nakatsuka
University of California, Irvine, USA
nakatsuy@uci.edu

Janaka L. Wijekoon
SLIIT, Sri Lanka
janaka.w@sliit.lk

Hiroaki Nishi
Keio University, Japan
west@sd.keio.ac.jp

***Abstract*—Named Data Networking (NDN) is a promising inter-networking paradigm that focus on content rather than hosts and their physical locations. In NDN *Consumers* issue *Interests* for *Contents*. *Producers* generate a content in response to each received interest and such content is routed back to the requesting consumer. When compared to IP, NDN brings advantages such as better throughput and lower latency, because routers are able to cache popular contents and satisfy interests for such contents locally. However, before being considered a viable approach, NDN should offer security services that are ideally better, but at least equivalent to current mechanisms in IP.**

In this regard, mechanisms to prevent DDoS are of paramount importance. In this work we propose FROG: a simple yet effective Interest Flooding Attack (IFA) detection and mitigation method. FROG runs on routers that are directly connected to NDN consumers and monitors packet hop counts. It then calculates mean and variance using stored hop counts to distinguish attackers from legitimate users. We use the NDN simulator ndnSIM to evaluate FROG's effectiveness. Our results show that FROG improves resilience against DDoS attacks. In particular, during an attack, legitimate users can still receive 75% of requested contents. Without FROG this number decreases to 50%.

***Keywords*—DDoS attack, Information-Centric Networking, Interest Flooding Attack, Named Data Networking, Packet Hop Count.**

I. Introduction

The concept of Information-Centric Networking (ICN) [1] was introduced with the motivation of filling the gap between the end-hosts' needs and the network's structure. ICN uses information as the core of all network functions and applications [2]. Several implementations of the concept of ICN [3–7] were proposed in the recent past, being Named Data Networking (NDN) [3] one of the most popular.

Security in ICNs has been well investigated [8] and Distributed Denial of Service (DDoS) resilience is one of the critical services that are needed in the architecture. In this work we focus on a particular type of DDoS: Interest Flooding Attacks (IFAs). We start by categorizing IFAs in four types and we show that, out of the four, two IFAs – Dynamically-Generated Existing Attack (DGEA) and Dynamically-Generated Non-Existing Attack (DGNEA) – are effective attacks. IFA resembles modern DDoS attacks, but many existent defense mechanisms are not effective in ICNs because of lack of IP addresses. Therefore, several works [9–11] have proposed IFA specific detection metrics and mitigation algorithms (in Sec. III, we go over some of these works in details). Most of these works target DGNEA and the only exception, which targets DGEA, is [12]. However, the work in [12] has some limitations as we discuss later on. In the present work we aim to address the limitations of [12] by introducing FROG. In summary, our contributions are the following:

- We present a novel DGEA detection metric based on packet hop count values. Our metric is implemented and evaluated using ndnSIM.
- We characterize the differences between legitimate users and attackers with respect to the proposed metric. Based on the insights from the characterization phase, we propose and evaluate the FROG algorithm: a countermeasure for DGEA.

Organization: An overview of NDN is provided in Sec. II. Related work is summarized in Sec. III. Models used for simulation and the proposed detection and mitigation method are explained in Sec. IV. Our method is evaluated in Sec. V. We discuss improvements and limitations of our algorithm in Sec. VI. Finally, Sec. VII concludes the paper.

II. Background

A. Named Data Networking (NDN)

Names: All data in NDN is hierarchically and uniquely named. A name is a combination of a prefix (used for routing the packet) and a suffix (used for identifying the data at the data producer). Consider an example of a video tagged with the name `ndn/com/videoservice/dog/video123.mp4`. The `ndn/com/videoservice/` is the prefix used for routing, and `/dog/video123.mp4` is the suffix used for the producer to identify the requested content.

Communication: The basic communication in NDN uses two types of packets: *Interest* packets or *Interests* and *Data* packets or *Contents*[1]. A consumer initiates the communication by issuing an interest for a certain content. Routers forward the packet to its destination (i.e., a given producer) via Longest Prefix Matching (LPM) on the name prefix. As the interest traverses the routers, it leaves some state that allows the content to be routed back to the requesting consumer. This is necessary because interests do not carry source addresses (which is a privacy friendly feature of NDN). The data producer sends back the requested content through the same reverse path traversed by the interest. As the content is forwarded back it removes the state that was left by the interest at each router. In addition, each router has the option of caching contents. If a content is cached, the router can directly reply for an interest

[1]Through the rest of this paper we use the terms "data packet" and "content" interchangeably

for such content, reducing latency and increasing throughput. The term *satisfied* is used in the paper to indicate that a node had received a content matching a pending interest issued in the past.

Packet Structure: The structure of the two packets used in NDN is significantly different from an IP packet [3]. A noticeable feature of an interest and a Data packet is that the packets require a content name, instead of an IP address. The content name area is used by: 1) consumers to express the content of interest, 2) data producers to identify the requested content, and 3) routers to forward packets. Another key feature is that all content is signed by it's producer to ensure authenticity.

Packet Forwarding State: Packet forwarding in NDN relies on three modules: Forwarding Interest Base (FIB), Pending Interest Table (PIT), and Content Store (CS). The FIB stores a routing table for computing LPM on the interest names, as in IP networks. The PIT keeps track of all pending interests (ones that have not yet been satisfied) until a given time limit. PIT stores the interest name and the corresponding incoming interface. When a new content arrives at a router, the router does a PIT look-up to figure out which interface should be used to forward this content (i.e., the same interface from which the interest of same name arrived). After that the the PIT entry is erased and the content optionally cached on the CS. CS caches content by creating a table that maps content with corresponding names. CS is used to reply to future interests locally with cached content, enabling better network performance and utilization.

B. Classification of Interest Flooding Attacks

IFA is conducted by sending a large number of interests to a target. Gasti et al. [13] classified IFAs into three types. Here we classify IFA into four types, instead. As discussed below, this classification is according to the type of name used in the interests and according to whether the requested content exists.

- Static/existing attack (SEA):
  In SEA, the attacker sends a large number of malicious interests for a single existing content. SEA attacks in NDN are mostly ineffective against the data producer, because, after the first interest, it is likely that the content will be cached in a router along the path. Therefore, interests after the first one will not reach the producer.
- Static/non-existing attack (SNEA):
  Attackers in SNEA request for a single non-existing content. This attack is also not effective because the PIT rejects interest with the same name to prevent duplicate entries.
- Dynamically-generated/existing attack (DGEA):
  The difference between DGEA and SEA is that the attackers' interests in DGEA never hit routers' caches, because they are dynamically-generated with different names at each time. A dynamically-generated name is made by combining an existing prefix and a random suffix, resulting in a name such as `ndn/com/videoservice/3rf3amslrx8rm9sla7`. An interest with this name will be delivered to the data producer owning the prefix `ndn/com/videoservice/`, even though the suffix `/3rf3amslrx8rm9sla7` does not exist. In addition, the dynamically-generated name causes multiple entries to be added to the PIT, as dynamically-generated names are always different from each other. In DGEA, PIT entries created by malicious interests are erased when the corresponding Data packet arrives at the node.
- Dynamically-generated/non-existing attack (DGNEA):
  DGEA and DGNEA are much alike, as both use dynamically-generated names, but the main difference is that the PIT entries created by the malicious interests cannot be erased until the entry itself expires because no content can satisfy the interests.

III. Related Work

Recent works have proposed IFA defense mechanisms [9–11, 14–17]. However, the two key limitations listed below are common to all of these works:

- DGEA is not considered. Some studies, such as [9] and [10], mention DGEA, but focus on DGNEA as it is more harmful. However, we claim that the damage of DGEA can not be ignored. As most metrics used to detect DGNEA are not applicable for detecting DGEA (PIT utilization rate used in [10] is an exception), we argue that it is necessary to propose a specific metric for DGEA detection.
- They assume that legitimate users request contents from a single producer. However, in many cases users might request contents from multiple data producers at the same time (as multiple processes run at the same time in a single computing platform).

To the best of our knowledge, the only related work that targets an attack similar to DGEA, called Collusive IFA (CIFA), is [12], which introduces a centralized controller to collect information from routers. Their detection method successfully detects CIFA and it could also be able to be used against DGEA, but it has two limitations. First, centralized controller implies an infrastructural change and a single point of failure. If the controller itself becomes the target of the attack, the defense mechanism cannot function properly. Second, there is the possibility of some delay between attack detection and attack mitigation. Our work aims at overcoming such limitations.

IV. FROG

In this section we first describe our legitimate user model and our adversary model. These models are used through the rest of our characterization and evaluation. Next we present our DGEA detection and mitigation method, FROG.

A. Legitimate User and Adversary Model

1) Legitimate user model: In this study, we assumed that multiple data producers exist in the network and that interests issued by legitimate consumers are uniformly distributed between the producers in the network. For the time between two interests issued by a given consumer (i.e., the per-consumer inter-interest time) we consider both uniform and exponential distributions.

2) Adversary model: We assume that the adversary is able to control an arbitrary number consumers. Routers are assumed to be a trusted party controlled by the Internet Service Provider (ISP).

Clearly, the adversary can attempt to mimic legitimate users' traffic patterns to remain undetected. The difference, though, is that a DDoS attack concentrates all interests to a single data producer. This implies a need for issuing interests with dynamically generated names. Otherwise routers can reply with cached content for a repeated name, preventing the flooded interests from ever reaching the producer targeted by the attack.

B. Attacker Behavior Characterization

Using the legitimate user model and adversary model described in Section IV-A, we here analyze if it is possible to tell apart attackers from legitimate users by using only hop counts of Data packets (recall that data packets are those carrying content replies issued by producers in response to interest packets). The topology and parameters used in the simulation are equivalent those used in Sec. V for evaluation.

Figure 1a shows the mean hop count of the attackers and the legitimate users. From Figure 1a, we can observe that the attacker's mean hop count value is around 5, while the legitimate users' is around 3.5. Such difference was caused because the attackers were receiving Data packets directly from the data producers, while the legitimate users received Data packets from both data producers and routers' caches.

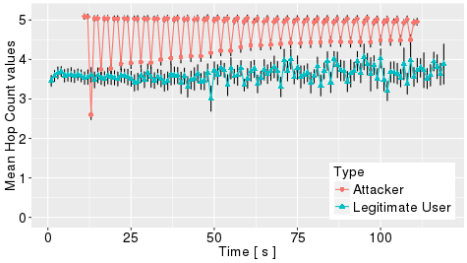
Figure 1b illustrates the hop count variance of the attackers and the legitimate users. The results of Figure 1b depict that the Data packets received by attackers have hop count variance close to zero. On the other hand, the Data packets received by legitimate users have initial hop count variance close to 1.0, which then gradually decreases. The legitimate users have higher variance for two reasons. First, legitimate users receive contents from producers and also from routers. Second, legitimate users request content from different producers, unlike attackers that target one particular producer. Legitimate users' variance decreases, because the number of data packets cached in nearby routers keeps increasing over time.

C. FROG Approach

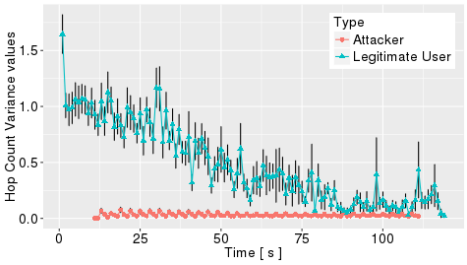
FROG uses packet hop counts gathered from Data packets. Here, packet hop count refers to the number of routers a packet passed when moving through the network. We first explain the reasons for using Data packets to gather packet hop counts. We then elaborate the principles of distinguishing attackers from legitimate users using the hop count metric.

1) Using Data packets for counting packet hop counts: Hop counts can be collected from either interests or Data packets. Hop counts are stored in the "hop count tag" in the packet header [18]. Whenever a node receives a packet, it can access the hop count value by referencing the value of the hop count tag. There are two specific reasons why we used Data packets for counting the number of packet hops.

The first reason is that Data packets are more reliable than interests. Interests cannot be certified in any way, and the attackers can modify hop count values of the interest at



(a) Average of mean hop counts based on end-host type



(b) Average of hop count variance based on end-host type

Fig. 1: Legitimate user and attacker characteristics

their will. Therefore, it is impossible to figure out whether the packet hop count of an interest is spoofed or not. On the other hand, as discussed in Sec. II, Data packets carry their producer's digital signature. Therefore, the only way for an attacker to spoof packet hop counts of Data packets is by either controlling the producer or the routers. However, that violates our adversarial model defined in Sec. IV-A2.

The second reason is that Data packets always reach a single destination while interests do not. Interests can find their matching data at either data producers or router caches. Therefore, collecting the hop count of interests is a hard task. On the other hand, there is no such problem when collecting hop count values from Data packets, because the Data packets always head back to the consumer that issued the corresponding interest.

However, recall that consumers are not trusted. Thus, routers directly connected to consumer are responsible for collecting data packets' hop counts.

2) FROG: FROG runs on client-edge routers due to two reasons. The first reason is, as aforementioned, because packet hop counts are collected at the client-edge routers. The second reason is that the client-edge routers are connected directly to the consumers. Therefore, FROG can stop the attack at its origin. In addition, a single point of failure does not exist, as FROG runs on multiple client-edge routers. At each client-edge router FROG classifies the consumer as an attacker or a legitimate user.

FROG detection algorithm runs on time windows. During each time window, FROG monitors and stores the hop count values of all received Data packets and the interface from which the packet came from. At the end of a time window, all stored hop count values are used to calculate the mean and the variance of

TABLE I: Parameters used in the simulation

Parameter	Value
Data request frequency (legitimate user)	100 [Interests/sec]
Data request frequency (attacker)	100 [Interests/sec]
Data packet size	1024 [bytes]
Number of data producers	10
Bandwidth (Backbone to Backbone)	40-100 [Mbps]
Bandwidth (Backbone to Gateway)	10-20 [Mbps]
Bandwidth (Gateway to Client-edge)	1-3 [Mbps]
Bandwidth (Client-edge to End-host)	1-3 [Mbps]
Link Delay (Backbone to Backbone)	5-10 [ms]
Link Delay (Backbone to Gateway)	5-10 [ms]
Link Delay (Gateway to Client-edge)	10-70 [ms]
Link Delay (Client-edge to End-host)	10-70 [ms]

the hop count value for each consumer connected to the client-edge router. The formula used for the calculation is shown in Equation 1, where n indicates the number of total packets received during the time window, x_i the hop count value of the i th packet, $\bar{x}$ the mean of the total hop count value, and v the variance of the hop counts stored.

$$v \equiv \frac{\sum_{i=0}^{n}(x_i - \bar{x})^2}{n}, \bar{x} \equiv \frac{\sum_{i=0}^{n} x_i}{n} \quad (1)$$

A consumer is considered an attacker whenever the mean hop count value is larger than a given threshold and the variance of the hop count values is smaller than a given threshold. In addition, FROG does not use any of the past detection results in future time windows to prevent false detection from affecting future decisions. When an IFA is detected, the incoming interface that issued the malicious interest is marked as an attack interface and then blocked.

V. Implementation and Results

We used ndnSIM [19, 20] to implement FROG. ndnSIM is a NDN package for the network simulator ns-3 [21]. This section first sets a static threshold for every FROG implementation to evaluate the effectiveness of the proposed defense mechanism. Next, different thresholds are set for each FROG implementation to show that FROG can detect and mitigate DGEA more stably. The simulations used the Telstra topology provided by Rocketfuel [22]. The original topology included 65 backbone routers, 45 gateway routers, and 169 client-edge routers. In addition to these nodes, this study added 419 consumers to the topology. The attackers and the legitimate users were randomly selected from the consumers, and the data producers were directly connected to the backbone routers in the simulation. The simulation time was set to 120 seconds due to the fact that the median duration of real life DDoS attacks is around 2 minutes [23]. The legitimate users start to send their interests at time 0. The attackers start their attack at time 10 seconds and end at time 110 seconds. The number of legitimate users was set to 17, and the attackers were set to 400 to simulate the situation of a botnet creating a DDoS attack [24]. The other parameters used in the simulation, shown in Table I, were set according to references [9–12] and Section IV.

A. Case A: Setting a static threshold for each client-edge router

1) Attack detection accuracy: Figure 1a and Figure 1b show differences between attackers and legitimate users in terms of mean and variance of hop counts. Using the threshold listed in Table II, we conducted 15 simulations to determine the appropriate threshold to effectively mitigate attacks. For simplicity, only the exponential distribution traffic pattern was used for threshold calibration. IFA detection rates for true positives, true negatives, false positives, and false negatives were computed for each simulation. Table III describes the relationship of these four rates.

TABLE II: List of mean and variance threshold choices

Mean Hop Count					
Threshold	3.0	3.5	4.0	4.5	4.9
Hop Count Variance					
Threshold	0.01	0.5	1.0		

TABLE III: Confusion Matrix and the component's meanings

	True	False
Positive	Attacker detected as an attacker (TP)	Attacker detected as a legitimate user (FP)
Negative	Legitimate user detected as an attacker (TN)	Legitimate user detected as a legitimate user (FN)

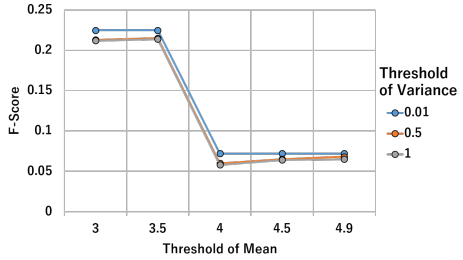


Fig. 2: F-Score values of different thresholds

We calculated F-Score using Equation 2 for each simulation and evaluated the consumer classification performance of the proposed attack detection metric. A higher F-Score indicates a higher performance in classification.

$$Fscore = \frac{TP}{TP+TN} * \frac{TP}{TP+FP} \quad (2)$$

The results of the simulations are shown in Figure 2. We can observe that the F-Score is the highest when the threshold of the mean is 3.5, and the threshold of the variance is 0.01. The confusion matrix for the two thresholds is displayed in Table IV. According to the results, FROG successfully detected 77% of the legitimate users, but only detected 41% of the attackers. The low True Positive rate is due to the mitigation algorithm. In the algorithm, previous detection results are not used to detect attackers in future time windows to prevent false detection from affecting future determinations. However, because of this feature, the attackers are able to send their interests as soon as a new time window arrives.

2) Attack mitigation effectiveness: The effectiveness of FROG is evaluated by comparing the changes in the legitimate users' satisfaction ratio. The thresholds for mean and variance of hop counts were 3.5 and 0.01 respectively.
Satisfaction ratio is calculated by dividing the number of received Data packets with the number of sent interests.

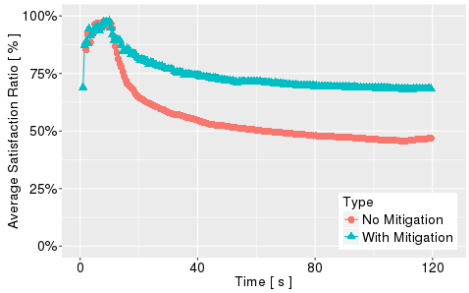
TABLE IV: Confusion Matrix (Case A)

(a) Constant

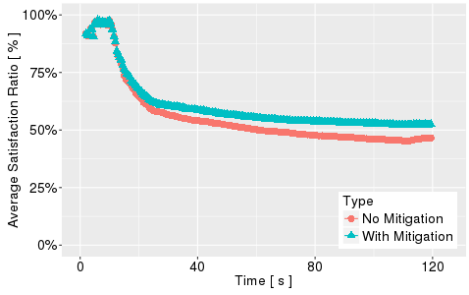
	True	False
Positive	56%	43%
Negative	12%	88%

(b) Exponential Random

	True	False
Positive	41%	59%
Negative	23%	77%



(a) Constant



(b) Exponential Random

Fig. 3: Satisfaction Ratio Improvement (Case A)

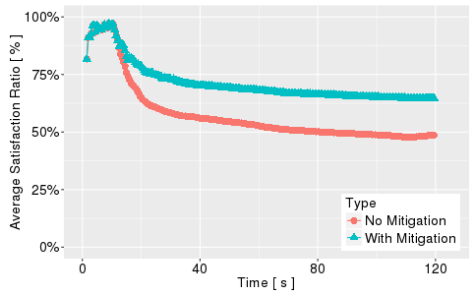
TABLE V: Confusion Matrix (Case B)

(a) Constant

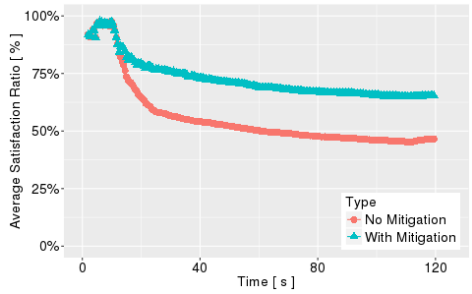
	True	False
Positive	67%	33%
Negative	32%	68%

(b) Exponential Random

	True	False
Positive	67%	33%
Negative	33%	67%



(a) Constant



(b) Exponential Random

Fig. 4: Satisfaction Ratio Improvement (Case B)

Figure 3 shows the mitigation effectiveness of FROG. We can observe that the legitimate users' average satisfaction ratio was improved by 20% for the consumers following the constant traffic pattern and around 3% for the consumers using the exponential traffic pattern. The reason the constant traffic pattern had the highest increment is that consumers receive Data packets constantly, meaning that the client-edge routers were able to collect hop counts constantly. Therefore, the client-edge routers had enough hop count data to determine whether a consumer is an attacker or not. However, if the consumers are following the exponential random traffic pattern, the amount of data the client-edge routers receive varies, and therefore the decisions made were inconsistent.

B. Case B: Setting different thresholds for each client-edge router

In Case A, we set the same mean threshold for each running FROG implementation. Next, in Case B, we set different mean thresholds for different FROG implementation as an example of an improvement of the algorithm. FROG monitors the first few packets and use the collected hop count values to calculate the mean hop count threshold. The hop count variance threshold is set to 0.01, according to the results yielded in Case A. The topology and other parameters used for the simulation were the same as Case A.

1) Attack detection accuracy: Table V summarizes the results of the attack detection accuracy. From the results, we can observe that there was a 10 to 26% improvement in the True Positive rate while the False Negative rate decreased by 10 to 20%. Compared to Case A, setting the mean threshold dynamically has affected the attack detection algorithm to become more sensitive to the attack. Another interesting result was that the detection accuracy became uninfluenced with the consumer's traffic pattern, unlike Case A.

2) Attack mitigation effectiveness: Figure 4 depicts the result of the attack mitigation effectiveness. Compared to the result from Case A, the average satisfaction ratio of the legitimate users following the constant traffic pattern decreased from 75% to 67%, while the exponential random traffic pattern increased from 50% to 67%. The results of Figure 4 also show that both constant traffic pattern and exponential random traffic pattern had the same average satisfaction ratio, as expected from the result of attack detection accuracy from Section V-B1. Results show that the improved detection algorithm became sensitive enough to detect an attack where consumers follow the exponential random traffic, which the detection algorithm in Case A failed to do. However, we can observe that the improved detection algorithm was too strict against legitimate users and limited some actions.

VI. Discussion and Limitations

FROG is a first step toward using consumer-side edge-routers' packet hop counts as a countermeasure to IFAs. Looking forward, our detection mechanism could be improved in several ways. We discuss some of them here. In addition we also discuss FROG's limitations.

- Fuzzy labels in the detection algorithm - Currently, FROG labels consumers as attacker as soon as the mean and variance values reach the threshold. It would be interesting to add intermediate labels, such as a "*suspicious*" label. A "*suspicious*" consumer would not be blocked right away. Instead the router would pay closer attention to its behavior and then decide whether to block it or not.
- Combination of different metrics - FROG uses only mean and variance of packet hop count in its decisions. However, different metrics could also be used. For instance, entropy of packet hop counts and PIT utilization rates (as used in [10]) could be used to improve FROG.

Our study assumes that the attackers do not have knowledge of FROG. Therefore, one may ask the question: What if the attackers know how FROG operates and adapts in an attempt to hide itself? The simplest way to do this would be by mixing interests that will be satisfied by caches in router CSs with the malicious interests. This would decrease the mean and increase the variance of the packet hop count, allowing the attacker to behave as a legitimate user. The first way to accomplish this is to resend an interest which was sent in the past. Since there is a high chance of receiving a copy of the content that is cached along the path, the mean hop count may decrease. However, because the past interests tend to be satisfied by caches on the same routers that are a few hops away, the hop count variance would not increase. The second way is to request popular contents within the network assuming that popular contents are more likely to be cached. However, this type of attack requires an adaptive real-time behavior from the adversary. In particular, the adversary must be aware of 1) which contents are popular; 2) how many hops away each content is; and 3) if a cached content has been deleted or not. The complexity of such adversary is much higher. Therefore, in the worst case, FROG makes it harder to perform attacks.

VII. Conclusion and Future Work

In this work we propose FROG: an approach for IFA detection and mitigation. FROG uses packet hop counts collected at the client-edge routers, where packet meta-data aggregation can be done in a reliable and simple way. FROG was implemented and evaluated in ndnSIM. Results show that, by using packet hop counts, attackers can be separated from legitimate users. Moreover, FROG improves legitimate users' satisfaction ratio by 3% and 25% by setting a static mean threshold to every client-edge router. Additionally, an average of 14% improvement is observed when a different mean threshold was set to each client-edge router.

As for future work, we plan to investigate how FROG can be extended to prevent attacks when the adversary is able to control both, producers and consumers, i.e., collusive IFA. In addition, it would also be interesting to investigate how to apply FROG as a counter-measure for DGNEAs, in addition to DGEAs.

Acknowledgment

The authors would like to thank Ivan Oliveira Nunes for his suggestions towards improving the quality of the paper.

References

[1] G. M. Brito, P. B. Velloso, and I. M. Moraes, *Information-Centric Networks A New Paradigm for the Internet*. John Wiley & Son, Inc., 2013.

[2] Named Data Networking (NDN), "Named Data Networking: Executive Summary." [Online]. Available: https://named-data.net/project/execsummary/

[3] L. Zhang *et al.*, "Named Data Networking (NDN) Project," *NDN, Technical Report NDN-0001*, 2010. [Online]. Available: https://www.parc.com/content/attachments/named-data-networking.pdf

[4] T. Koponen *et al.*, "A data-oriented (and beyond) network architecture," in *Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications - SIGCOMM '07*, vol. 37, no. 4. ACM Press, 2007, p. 181.

[5] V. Jacobson *et al.*, "Networking named content," in *Proceedings of the 5th international conference on Emerging networking experiments and technologies - CoNEXT '09*. ACM Press, 2009, p. 1.

[6] "CCNx." [Online]. Available: http://ccnx.org

[7] "CICN." [Online]. Available: https://wiki.fd.io/view/Cicn

[8] E. G. Abdallah, H. S. Hassanein, and M. Zulkernine, "A survey of security attacks in information-centric networking," *IEEE Communications Surveys and Tutorials*, vol. 17, no. 3, pp. 1441–1454, 2015.

[9] A. Afanasyev, P. Mahadevan, I. Moiseenko, E. Uzun, and L. Zhang, "Interest Flooding Attack and Countermeasures in Named Data Networking," *IFIP Networking*, pp. 26–31, 2013.

[10] A. Compagno, M. Conti, P. Gasti, and G. Tsudik, "Poseidon: Mitigating interest flooding ddos attacks in named data networking," in *38th Annual IEEE Conference on Local Computer Networks*, Oct 2013, pp. 630–638.

[11] K. Wang, H. Zhou, H. Luo, J. Guan, Y. Qin, and H. Zhang, "Detecting and mitigating interest flooding attacks in content-centric network," *Security and Communication Networks*, vol. 7, no. 4, pp. 685–699, apr 2014.

[12] H. Salah and T. Strufe, "Evaluating and mitigating a Collusive version of the Interest Flooding Attack in NDN," in *2016 IEEE Symposium on Computers and Communication (ISCC)*. IEEE, jun 2016, pp. 938–945.

[13] P. Gasti, G. Tsudik, E. Uzun, and L. Zhang, "DoS and DDoS in named data networking," *Proceedings - International Conference on Computer Communications and Networks, ICCCN*, 2013.

[14] M. Wählisch, T. C. Schmidt, and M. Vahlenkamp, "Backscatter from the data plane - Threats to stability and security in information-centric network infrastructure," *Computer Networks*, vol. 57, no. 16, pp. 3192–3206, 2013.

[15] H. Dai, Y. Wang, J. Fan, and B. Liu, "Mitigate DDoS attacks in NDN by interest traceback," *2013 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 381–386, 2013.

[16] K. Wang, H. Zhou, Y. Qin, and H. Zhang, "Cooperative-Filter: Countering Interest flooding attacks in named data networking," *Soft Computing*, vol. 18, no. 9, pp. 1803–1813, 2014.

[17] T. N. Nguyen, R. Cogranne, G. Doyen, and F. Retraint, "Detection of interest flooding attacks in Named Data Networking using hypothesis testing," *2015 IEEE International Workshop on Information Forensics and Security, WIFS 2015 - Proceedings*, no. 1, 2015.

[18] "ndnSIM: HopCountTag." [Online]. Available: http://ndnsim.net/2.3/doxygen/classHopCountTag.html

[19] "ndnsim." [Online]. Available: http://ndnsim.net/2.4/

[20] S. Mastorakis, A. Afanasyev, I. Moiseenko, and L. Zhang, "ndnsim 2.0: A new version of the ndn simulator for ns-3," *NDN, Technical Report NDN-0028*, 2015.

[21] "ns-3." [Online]. Available: https://www.nsnam.org/

[22] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson, "Measuring ISP Topologies With Rocketfuel," *IEEE/ACM Transactions on Networking*, vol. 12, no. 1, pp. 2–16, feb 2004.

[23] "Attack of Things!" [Online]. Available: http://netformation.com/level-3-pov/attack-of-things-2

[24] "Heightened DDoS Threat Posed by Mirai and Other Botnets." [Online]. Available: https://www.us-cert.gov/ncas/alerts/TA16-288A