PERSIA: a PuzzlE-based InteReSt FloodIng Attack Countermeasure

Reza Tourani
Saint Louis University
Saint Louis
reza.tourani@slu.edu

George Torres
New Mexico State University
Las Cruces
gtoressz@nmsu.edu

Satyajayant Misra
New Mexico State University
Las Cruces
misra@cs.nmsu.edu

ABSTRACT

With the proliferation of smart and connected mobile, wireless devices at the edge, Distributed Denial of Service (DDoS) attacks are increasing. Weak security, improper commissioning, and the fast, non-standardized growth of the IoT industry are the major contributors to the recent DDoS attacks, *e.g., Mirai Botnet* attack on Dyn and *Memcached* attack on GitHub. Similar to UDP/TCP flooding (common DDoS attack vector), request flooding attack is the primary DDoS vulnerability in the *Named-Data Networking* (NDN) architecture.

In this paper, we propose *PERSIA*, a distributed request flooding prevention and mitigation framework for NDN-enabled ISPs, to ward-off attacks at the edge. *PERSIA*'s edge-centric attack prevention mechanism eliminates the possibility of successful attacks from malicious end hosts. In the presence of compromised infrastructure (routers), *PERSIA* dynamically deploys an in-network mitigation strategy to minimize the attack's magnitude. Our experimentation demonstrates *PERSIA*'s resiliency and effectiveness in preventing and mitigating DDoS attacks while maintaining legitimate users' quality of experience (> 99.92% successful packet delivery rate).

CCS CONCEPTS

• **Security and privacy → Denial-of-service attacks**; **Security protocols**; *Distributed systems security*; • **Networks → Denial-of-service attacks**.

KEYWORDS

Information-centric networking, distributed denial of service attack, edge-centric prevention, in-network mitigation, edge security, network edge.

ACM Reference Format:
Reza Tourani, George Torres, and Satyajayant Misra. 2020. PERSIA: a PuzzlE-based InteReSt FloodIng Attack Countermeasure. In *7th ACM Conference on Information-Centric Networking (ICN '20), September 29-October 1, 2020, Virtual Event, Canada.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3405656.3418709

ICN '20, September 29-October 1, 2020, Virtual Event, Canada

ACM ISBN 978-1-4503-8040-9/20/09…$15.00
https://doi.org/10.1145/3405656.3418709

1 INTRODUCTION

Over the last decade, with the advent of IoT applications, per capita number of devices, and network connections have sky-rocketed. Per Cisco's prediction, the number of connected devices will reach 28.5 billion in 2022, with industrial machine-to-machine (M2M) and smart home applications being the major growth contributors [6]. This colossal number of wireless and mobile devices, are harder to manage due to their dynamicity, and are more prone (than wired devices behind a firewall) to becoming compromised and commandeered to form botnet arrays to orchestrate Distributed Denial of Service (DDoS) attacks. Several recent substantial DDoS attacks, *e.g.,* the *Mirai* Botnet attack on *Dyn* DNS (peak strength of 1.2 Tbps from over 600,000 compromised IoT devices), which crippled Netflix, CNN, and Twitter [2], and the *LizardStresser* botnet attack during the 2016 Rio Olympic (peak strength of 540 Gbps), are the outcomes of this phenomenon. Notably, UDP flooding made up 56% of all attacks in the second quarter of 2018 [36].

This attack surface continues to grow with the addition of new devices to the edge of the network and the infusion of emerging technologies, such as *Information-centric networking* (ICN) [18]. ICN, one of the future Internet architecture candidates, promotes network intelligence and pervasive caching by shifting the existing "host-centric" communication model to a "data-centric" paradigm to better scale for future demands. To facilitate content caching and reliable retrieval, many ICN architectures, including *Named-Data Networking* (NDN) [18], divide each content into uniquely named data packets (chunks); each packet can be requested individually by its name. An intermediate router stores the forwarded request (Interest in NDN) in its Pending Interest Table (PIT) until the corresponding data chunk is received, at which point the data is sent downstream in the direction of the incoming request and the PIT entry is removed.

Despite the advantages of data chunking, individually requesting each data packet poses a security threat, namely the *Interest flooding attack* (IFA), which is an intensified variant of DDoS flooding. Different from IP-based DDoS attacks that target the end hosts, IFA attackers (also termed DDoS attackers) target the network infrastructure–the routers' resources–by sending a large number of Interests for non-existing contents, thus exhausting the PIT with unexpired Interests.

Motivation: Rate-limiting is the dominant technique used in NDN-based DDoS countermeasures, in which routers rate-limit their affected links to reduce the IFA attack's impact [9, 10, 14]. These approaches, despite protecting the network's resources, negatively impact the data rate of legitimate users collocated with the attackers. More importantly, they do not consider compromised infrastructure, namely compromised routers that collude with the attackers

in letting DDoS traffic through. Infrastructure compromise is fast becoming a major attack vector [3, 11]. The IP-based solutions for detecting compromised infrastructure suggest approaches based on modeling traffic behavior [4, 16, 23], intrusion detection systems [5, 42], and trust level evaluation [15, 22]. However, these approaches mainly detect compromised infrastructure without minimizing attack impacts and also suffer from high communication and computation overheads from routers' coordination and redundant traffic validation [12, 43].

To overcome these deficiencies, we propose *PERSIA*, an edge-centric puzzle-based DDoS framework for NDN-enabled ISP networks. *PERSIA* employs *Proof-of-Work* (PoW) as a plug-and-play token generation tool to mandate users to solve computational puzzles and generate tokens, which they convey to their edge routers in their Interests. These tokens enable the edge routers to validate users before processing their requests, thus preventing unbridled flooding attacks at the edge of the ISP before the malicious traffic enters the network core. Different from the assumption of trustworthy routers, we consider the situation where (edge) routers are compromised. In the presence of compromised infrastructure, *PERSIA* enables core routers to dynamically deploy a defensive strategy in the network core to mitigate the impacts of infrastructure attack on users' quality of experience (QoE).

We note that Portcullis [28], a Denial-of-Capability countermeasure using per-computation fairness, has shown that PoW is a viable and fair means of mitigating DDoS attacks (across the spectrum of devices based on capabilities). With *PERSIA* we propose a framework to utilize PoW for DDoS prevention at the network edge, and importantly, perform in-network attack mitigation from a compromised infrastructure.

Our novel **contributions** include: ***(i)*** Detailed design of *PERSIA*, an edge-centric NDN-based DDoS prevention framework, which utilizes a PoW mechanism at the edge of the network to prevent malicious requests from entering the core network. We use PoW to illustrate our framework, while *PERSIA* can easily use other means of delayed token-generation. ***(ii)*** *PERSIA* also features a novel dynamically deployable mitigation strategy for core routers to deal with compromised infrastructure and maintain legitimate users' QoE. To the best of our knowledge, our work is the first in developing a dynamic network-based defense mechanism to cope with compromised infrastructure. ***(iii)*** Augmenting the existing state-of-the-art attack mitigation strategies (rate-limiting and self-routing) for performing fair (apples-to-apples) comparison with *PERSIA* via simulations. ***(iv)*** *PERSIA*'s security analysis, discussions, and comparison with related work. ***(v)*** *PERSIA*'s implementation and ISP-scale simulations validating its effectiveness and scalability in the edge networks.

Organization: Section 2 presents our models and assumptions. Section 3 overviews *PERSIA* and its building blocks. In Sections 4 and 5, we explain *PERSIA*'s design. We evaluate *PERSIA*'s security in Section 6 and present simulation results and analyses in Section 7. Section 8 compares *PERSIA* with existing countermeasures. Section 9 concludes the paper and presents scope for future work.

2 MODELS AND ASSUMPTIONS

System Model: We consider an ISP network including wireless devices ($\mathcal{U}$), which are connected to the edge routers ($\mathcal{R}_E$) via access points (*APs*), the ISP core routers ($\mathcal{R}_C$), a set of proxies ($\mathcal{P}$), and the content providers. Routers are either edge routers ($\mathcal{R}_E$) or core routers ($\mathcal{R}_C$), where $\mathcal{R}_E \cup \mathcal{R}_C = \mathcal{R}$. In this paper, we use NDN to illustrate our framework's interactions and also for simulation purposes due to NDN's popularity.

Security Assumptions: In *PERSIA*, we assume each ISP is equipped with one or more trusted proxies that provides users ($\mathcal{U}$) with puzzles and the edge routers ($\mathcal{R}_E$) with puzzles' metadata, necessary for user validation. The edge routers only allow Interests with valid tokens. The proxy and the routers (edge and core) can interact to set the difficulty of the puzzles to define an overall system Interests rate. An ISP's infrastructure behaves rationally, that is, routers do not help attackers to orchestrate DDoS attacks. However, attackers can compromise routers to forward malicious Interests without verification to magnify the attack. We assume all communications between end-points are authenticated and encrypted.

Threat Model: In our framework, a legitimate user (hereafter user) needs to solve puzzles and use them to generate tokens for successful content retrieval. There are several attack scenarios in this context including: *(a)* attackers, requesting data without solving puzzles. *(b)* Attackers, using invalid tokens for content retrieval. *(c)* Attackers, intercepting and hijacking users' tokens to use themselves (e. g., replay attack). *(d)* Malicious users, colluding with users/attackers by sharing puzzles' solutions, tokens, and credentials. *(e)* Compromised infrastructure (malicious routers) forwarding Interests without validating the tokens. In this paper, we assume only edge routers can be compromised (refer to Subsection 6.4 for more information). In Section 6, we discuss how *PERSIA* addresses these threats and elaborate on approaches to mitigate attacks on proxies.

Design Goals: We design *PERSIA* to prevent and mitigate DDoS attacks targeting the infrastructure and data providers. Different from prior schemes [1, 9, 10, 14, 25, 40], in addition to normal flooding attack prevention, *PERSIA* enables routers to deploy a network-based mitigation strategy to tide over compromised infrastructure attacks, thus further improving legitimate users' QoE. In the presence of compromised infrastructure, *PERSIA* dynamically deploys a network-based mitigation strategy to minimize the attack impacts on legitimate users. In designing *PERSIA*, we consider achieving the following goals.

Resiliency and effectiveness: Providing effective prevention and mitigation of DDoS attacks by individual and collaborative attackers, even when routers are compromised.

Maintain users' QoE: Maintaining users' satisfaction–high data rates and low latency communication–in the presence of distributed attackers and compromised infrastructure. The majority of the existing scheme fall short in maintaining users' QoE.

Low cost: Incurring minimal cost on the infrastructure in terms of defense deployment, and routers' communication, computation, and storage overheads. Furthermore, it should not computationally overload the users for token generation.

3 BACKGROUNDS AND OVERVIEW

This section presents *PERSIA*'s building blocks, including PoW and token generation, and the two phases of *PERSIA*: attack prevention and mitigation. Table 1 introduces the notation we used to describe *PERSIA*.

Table 1: Notations Used

Notation	Description
$\mathcal{H}()$	A cryptographic hashing function
Q	Big prime number
n	Degree of polynomial $F_n(x)$
$F_n(x)$	One-dimensional n-degree polynomial
a_0	Constant of $F_n(x)$
$f(x_i)$	Evaluation of coordinate x_i on $F_n(x) \in \mathbb{Z}^*_Q$
λ_k	k^{th} Lagrangian coefficient
$ZQrand()$	Random number generator in $\mathbb{Z}^*_Q$
$\mathbb{Z}^*_Q$	Multiplicative groups of integers of order Q
E	A puzzle
E_S	Puzzle E's solution
E_M	Puzzle E's secret seed
E_{TC}	Puzzle E's token chain
E_L	Maximum number of tokens for puzzle E

3.1 Proof-of-Work (PoW) Mechanism

In *PERSIA*, we use the Lagrangian interpolation polynomial for the PoW mechanism to prevent DDoS attack by controlling users' communication.

Definition 3.1. [Lagrangian Interpolation Polynomial] *A Lagrange's polynomial of degree n (will be shown by $F_n(x)$) taking on the values $f(x_0), \ldots, f(x_n)$ for the points $x_0, \ldots, x_n$ is given by,*

$$F_n(x) = f(x_0)\frac{(x-x_1)(x-x_2)\ldots(x-x_n)}{(x_0-x_1)(x_0-x_2)\ldots(x_0-x_n)} + \ldots +$$
$$f(x_n)\frac{(x-x_0)(x-x_1)\ldots(x-x_{n-1})}{(x_n-x_0)(x_n-x_1)\ldots(x_n-x_{n-1})}.$$

The i^{th} fractional term (Lagrangian coefficient) in $F_n(0)$ is represented as, $\lambda_i = \Pi_{0\le j(\neq i)\le n}\frac{x_j}{x_j-x_i}$ resulting in $a_0 = F_n(0) = f(x_0)\lambda_0 + f(x_1)\lambda_1 + \ldots + f(x_n)\lambda_n$. □

Given $n+1$ unique coordinates $\{(x_i, f(x_i))|1 \le i \le n+1\}$ on $F_n(x)$, where x_i is a point on the X-axis and $f(x_i) = F_n(x_i)$, one can use Lagrangian interpolation to interpolate the unique polynomial ($F_n(x)$). In *PERSIA*, we consider the $F_n(x)$'s coordinates as a unique puzzle (E), the Lagrangian interpolation of $F_n(x)$ as the PoW execution, and $a_0 = F_n(0)$ as the puzzle's solution (E_S).

We note that using PoW allows the prevention phase of *PERSIA* to accomplish token generation at a set price, and is designed to have minimal impact on legitimate clients while also being scalable. We will assess the token generation cost in Section 7.2. Other PoW approaches, such as Bitcoin's *Hashcash*, *hash sequence* [19], *Diffie–Hellman-based puzzle* [41], and *Cuckoo cycle* [35] can also be easily used in our framework. We chose Lagrangian interpolation due to its standardized puzzle solving time, which enables tuning of the expected work as needed (by changing polynomial degree). We note again that we use PoW as an illustrative mechanism for generating tokens in *PERSIA*. The token generation mechanism is a plug-and-play feature of the framework and is future-proof. PoW can be replaced with any other token generation mechanism. In future, we plan to explore other token generation mechanisms to be used in conjunction with token-based access control mechanisms such as [34].

3.2 Prevention Phase Overview

In the prevention phase, *PERSIA* prevents attackers from exhausting the routers' resources, using three protocols running on users ($\mathcal{U}$), edge routers ($\mathcal{R}_E$), and the trusted proxies ($\mathcal{P}$), respectively. In the beginning, proxies generate a set of puzzles for users. Each user registers herself with a proxy and obtains a puzzle (E) and a secret seed (E_M). Secret E_M is a unique nonce, which enables generation of multiple unique token chains for a single puzzle to promote puzzle re-usability by different clients. After solving the puzzle, the user generates a *token chain* (E_{TC}) using the puzzle's solution E_S ($\equiv a_0 \equiv F_n(0)$) and its secret (E_M) (Definition 3.2).

Definition 3.2. [Token Chain] *Given a puzzle solution (E_S) and its corresponding secret seed (E_M), we define a unique token chain (E_{TC}) as: $< E_S, E_M >$: $\exists!$ $E_{TC} = \{T_1, \cdots, T_k\}$, which is an ordered sequence of k tokens, generated using a cryptographic hash function $\mathcal{H}$ as follows:*

$$T_1 = \mathcal{H}(E_S, E_M),\ T_2 = \mathcal{H}(T_1, E_M),\ \cdots,\ T_k = \mathcal{H}(T_{k-1}, E_M).$$

E_{TC} is unique for a given puzzle E and its secret seed E_M. We consider E_{TC} violated if (at least) two users, u_i and u_j, collocated under the same edge router, receive an identical puzzle E and secret seed E_M. Token chain violation prevents users from successfully retrieving content. □

We note that solving the puzzle (Definition 3.1) is the most costly operation in our PoW mechanism, which will be done at the beginning of token chain generation. With the puzzle solution, generating the subsequent tokens are relatively low cost–a series of hashing operations to generate a hash chain (Definition 3.2). Thus, generating token chain from a puzzle amortizes the PoW cost. The user includes the generated tokens from her token chain to request content (one token used per Interest). Edge routers verify the tokens in the received Interests, using the puzzles' metadata they obtain from the proxies, and forward validated Interests, or drop them otherwise.

3.3 Mitigation Phase Overview

Despite *PERSIA*'s prevention phase, there is always a possibility that attackers successfully compromise a subset of routers–*a potent threat [3, 11] that has been neglected in the majority of the existing literature.* To minimize the impact of this threat, in *PERSIA*, we design a novel attack mitigation strategy to handle compromised infrastructure–the *Bloom Filter-Assist* (BFA) strategy. In *PERSIA*, core routers ($\mathcal{R}_C$) constantly monitor the network and collect statistical information, such as Interest success and drop rates (Definition 3.3), and dynamically invoke BFA strategy when a DDoS attack is detected. Each core router autonomously decides whether to

execute BFA or not–enabling BFA's partial deployment. We extensively discuss our BFA strategy and its application in Section 5 and compare it with two state-of-the-art DDoS mitigation strategies: rate-limiting and self-routing. We note that *PERSIA*'s prevention phase and mitigation phase work independent of each other. In *PERSIA*, core routers monitor the network to initiate the mitigation phase as soon as a DDoS attack is detected regardless of status of the prevention phase (*i.e.,* puzzle distribution or puzzle solving). However, there is a period between the time that a DDoS attack is initiated and time that core routers detect the attack and initiate BFA strategy. During this short period, the network would not be protected against the compromised routers.

Definition 3.3. [Interest Drop Rate] *Similar to [27], we define the Interest drop rate $\mathcal{D}^k(i)$ for interface i and packet k in a series of packets $[1, \cdots, k, \cdots, K]$ as a weighted moving average window:* $\mathcal{D}^k(i) = \alpha \times d_k + (1-\alpha) \times \mathcal{D}^{k-1}(i)$,

in which $d_k = 0$ is set when packet k is successfully delivered and $d_k = 1$ is set if packet k times out (packet drop). We also use $\alpha = 0.833$ as [27]. □

4 *PERSIA*'S PREVENTION MECHANISM

In this section, we present *PERSIA*'s attack prevention design. In particular, we introduce the *Puzzle Generation*, *Puzzle Solving*, and *Solution Validation* procedures along with *Edge Router-Proxy*, *User-Proxy*, and *User-Edge Router* interactions in detail.

Protocol 1 presents proxy $\mathcal{P}$'s puzzle generation routine, with the polynomial degree (n), a large prime number Q, and a random number generator ($ZQrand()$) in the multiplicative groups of integers of order Q ($\mathbb{Z}_Q^*$) as inputs. It returns a puzzle (E), its solution (E_S), secret seed (E_M), and the token limit (E_L). Then $\mathcal{P}$ generates polynomial $F_n(x)$ using $n+1$ random integer coefficients (Lines 1-4). It then computes $n+1$ random unique integers as the X-axis coordinates (excluding $x_j = 0$), and the corresponding values of $F_n(x)$, as the puzzle (Lines 5-9).

The proxy uses the polynomial constant as the solution (E_S), a random number (E_M) as the secret seed, and selects the maximum number of tokens (E_L) (Lines 10-12). Puzzle E's token limit (E_L) indicates the maximum number of tokens that will be accepted for it. E_L can be decided proportional to E's difficulty, allowing $\mathcal{P}$ to control the users' requesting rates when the network is under attack. The secret seed (E_M) enables the generation of multiple unique token chains for a unique puzzle, which reduces the puzzle generation complexity.

Proxy $\mathcal{P}$ repeats Protocol 1 to generate a pool of unique puzzles. A unique puzzle can be assigned to multiple users using a unique E_M per user to reduce the storage and communication overhead of routers and proxies. $\mathcal{P}$ generates a *Solution Table* (ST) including puzzles' solutions, secret seeds, and token limits.

4.1 Edge Router–Proxy Interactions

Before discussing the edge router (r_E) and the proxy ($\mathcal{P}$) interactions, we briefly explain a few features of NDN for better understanding. NDN is a *user-driven* architecture, in which contents are composed of a set of smaller fixed size chunks (in the order of 1 kB to 9 kB). Each chunk is uniquely named and should be individually requested. In *PERSIA*, we use *manifest* [24], which includes the

Protocol 1 Puzzle Generation at Proxy $\mathcal{P}$

Input: Prime Q, n ($n < Q$), and $ZQrand()$.
Output: Puzzle E, its solution E_S, secret seed E_M, and token limit E_L.

1: **for** $0 \le i \le n$ **do**
2: $a_i = ZQrand()$. {coefficients of $F_n(x)$.}
3: **end for**
4: Generates $F_n(x)$ using the a_is.
5: **for** $0 \le j \le n$ **do**
6: $x_j = ZQrand()$, and $x_j \neq 0$.
7: $f(x_j) = F_n(x_j) \in \mathbb{Z}_Q^*$.
8: $E = E \cup (x_j, f(x_j))$.
9: **end for**
10: $E_S = F_n(0) \in \mathbb{Z}_Q^*$.
11: $E_M = ZQrand()$.
12: Selects E_L proportional to $F_n(x)$'s degree.

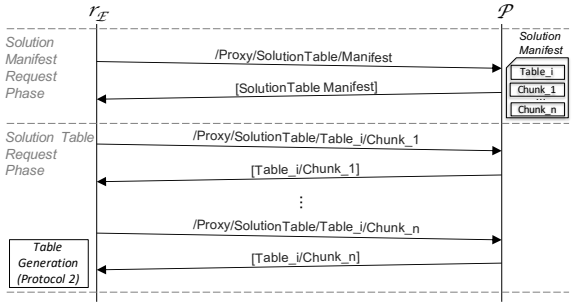


Figure 1: The edge router retrieves the proxy's solution table for verifying users' tokens.

content's metadata, such as its chunks' names and sizes, to help requesting nodes generate correct Interests [31]. Using manifest in *PERSIA* helps reduce the workload of the puzzle distribution proxy by promoting puzzle reuse. Moreover, using puzzle manifest allows the routers to cache the puzzles and reduce the communication latency. With the puzzles cached in the network, a user only need to request the puzzle manifest from the proxy and then obtain the puzzle from the cache. For successful user token verification, r_E needs to obtain the *Solution Table* (ST) from $\mathcal{P}$ (Figure 1). Thus, r_E requests ST's manifest from $\mathcal{P}$ (*Manifest Request Phase*); ST's manifest includes the table name, chunks' names, and their sizes. Then r_E enters the *Solution Table Request Phase* and requests ST's chunks from $\mathcal{P}$, generates a *Validation Table* (VT), and populates VT with the obtained puzzles' data for token verification.

Protocol 2 VT Generation at Edge Router r_E

1: Generates validation table VT as a five tuples:
$< Solution, Seed, Limit, T_1, Counter >$.
2: **for each** $E \in ST$ **do**
3: Calculates $T_1 = \mathcal{H}(E_S, E_M)$.
4: $VT \cup < E_S, E_M, E_L, T_1, 1 >$.
5: **end for**

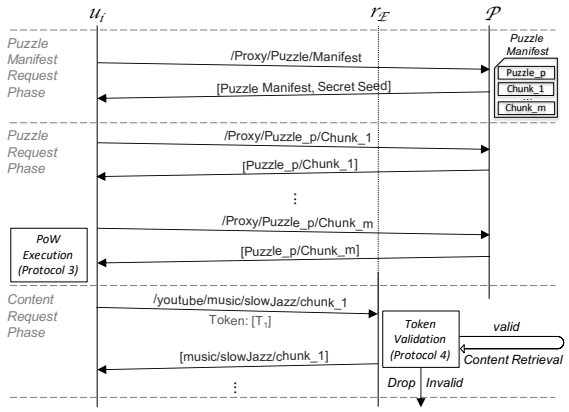


Figure 2: User u_i obtains a puzzle from $\mathcal{P}$, generates tokens that will be validated by edge router r_E.

As shown in Protocol 2, VT is a five tuples table including the puzzle's solution, E_M, E_L, the *First Token*, and a *Counter* (Line 1). For each puzzle in the solution table, r_E adds a row to VT by calculating the T_1 in the token chain as defined in Definition 3.2 (Lines 2-5). Router r_E keeps a counter for each puzzle to prevent reuse of used tokens. In our implementation, when $\mathcal{P}$ runs out of its pre-generated puzzles (users have requested all puzzles) $\mathcal{P}$ generates a new set of puzzles and advertises them to the edge routers. Edge routers follow the interactions in Figure 1 to obtain the new ST and update their VTs. Proxy $\mathcal{P}$ assigns unique puzzles to users collocated under the same r_E to prevent users from colluding to reuse tokens.

4.2 User–Proxy Interactions

In *PERSIA*, users securely obtain puzzles from the proxies, execute the PoW, and generate tokens for data retrieval. Figure 2 illustrates the interactions between user u_i and proxy $\mathcal{P}$ (*Puzzle Manifest* and *Puzzle Request* phases). User u_i also starts with requesting a puzzle manifest. On receiving the manifest, u_i requests the actual puzzle from $\mathcal{P}$. The requested puzzle can be delivered to the user either by $\mathcal{P}$ or an intermediate router caching the puzzle.

Protocol 3 Proof-of-Work Execution by User u_i

Input: Puzzle E, its secret E_M, its token limit E_L.
Output: Token Chain E_{TC}.

1: **for** $0 \leq k \leq n$ **do**
2: **for** $0 \leq j \leq n$ && $j \neq k$ **do**
3: $\lambda_k = \lambda_k \times \frac{x_j}{x_j - x_k} \in \mathbb{Z}_Q^*$.
4: **end for**
5: $\Lambda = \Lambda \cup \lambda_k$.
6: **end for**
7: Interpolates $F_n(x) = \lambda_0 f(x_0) + \cdots + \lambda_n f(x_n)$.
8: $E_S = F_n(0)$.
9: $T_1 = \mathcal{H}(E_S, E_M)$.
10: Sets $c = 1$.
11: **while** $c \leq E_L$ **do**
12: $T_c = \mathcal{H}(T_{c-1}, E_M)$.
13: $E_{TC} = E_{TC} \cup T_c$.
14: $c = c + 1$.
15: **end while**

On receiving puzzle E, u_i executes the PoW procedure (Protocol 3) to solve E. Protocol 3 takes E, its secret seed E_M, and token limit E_L and generates E_{TC}. Per Definition 3.1, u_i calculates the Lagrangian coefficients of E and interpolates polynomial $F_n(x)$ (Lines 1-7). User u_i then evaluates the polynomial's constant ($F_n(0)$) as the puzzle's solution E_S (Line 8). Per Definition 3.2, u_i generates her first token T_1 (Line 9) and the subsequent tokens (Lines 11-15) in the token chain E_{TC}. Then, u_i enters the *Content Request Phase* and initiates content retrieval. Each new Interest contains a unique token (refer Figure 2). User u_i requests and solves a new puzzle upon reaching the token limit.

Protocol 4 Token Validation at Edge router r_E

Input: Interest $\mathcal{I}_k$ and its token T_j.
Output: Forwarding or dropping decision.

1: $E \leftarrow T_j \in VT$.
2: **if** E && $C \leq E_L$ **then**
3: Process Interest $\mathcal{I}_k$. {NDN forwarding.}
4: $T_{j+1} = \mathcal{H}(T_j, E_M)$. {$E_M$ is the secret seed.}
5: Updating T_{j+1} and $C = C + 1$ in VT.
6: **else**
7: Drop $\mathcal{I}_k$.
8: **end if**

4.3 User–Edge Router Interactions

After u_i and r_E completed their interactions with $\mathcal{P}$, u_i initiates the data retrieval procedure (*Content Request Phase* in Figure 2). In this phase, u_i includes the next available token (T_j, assuming u_i has used tokens $[T_1, .., T_{j-1}]$; tokens are used in order) in the Interest $\mathcal{I}_k$, indicating the executed PoW. On receiving u_i's Interest, r_E searches token T_j in its validation table VT to fetch the corresponding tuple (Line 1 Protocol 4). On successful lookup and if the puzzle's counter (C) has not reached its limit (E_L), r_E processes the Interest using NDN's forwarding daemon, as explained in Section 2 (Lines 2-3). After Interest processing, r_E uses E_M to generate the next token (T_{j+1}), increases the counter (C), and updates VT with T_{j+1} and updated C (Lines 4-5). If token T_j lookup fails, r_E drops u_i's Interest (Lines 6-8).

5 *PERSIA* ATTACK MITIGATION STRATEGY

As discussed in the threat model, a compromised router may skip *PERSIA*'s prevention phase (Protocol 4) and forward malicious Interests. In this section, we introduce our novel Bloom Filter-Assist (BFA) defensive strategy to mitigate the impact of compromised routers on users' QoE. Unlike *PERSIA*'s PoW mechanism that runs on edge routers, BFA will be dynamically invoked by those core routers that detect an ongoing attack. Before discussing *PERSIA*'s mitigation strategy, we briefly explain two most relevant existing DDoS mitigation strategies.

The Rate-Limiting (RL) forwarding strategy has been extensively discussed in the literature [1, 9, 10, 14]. RL's primary objective is prioritizing traffic arriving from more reliable interfaces defined using statistical information such as interfaces' Interest drop rates. We enhanced the RL strategy (enhanced for fair comparison) where

core routers monitor their interfaces' drop rates to invoke RL on suspicious interfaces; those interfaces whose drop rate exceeds a pre-defined threshold z ($0 \leq z \leq 1$). The Interests arriving at suspicious interfaces will be randomly forwarded with probability $(1-z)$. We also augment RL with routers' collaboration via negative acknowledgment, which reduces falsified measurements.

The Self-Routing (SR) forwarding strategy has been proposed by Wang *et al.* [39]. In principle, upon an IFA attack detection (when the Interest drop rate exceeds a threshold), SR decouples suspicious Interests from the routers' PITs by including the routing information in the Interest and data packets (appended to by each router). However, the proposed SR strategy [39] undermines cache utilization and contradicts NDN's content name immutability [17] principle as routers change the content name. Thus, we *implemented an augmented SR strategy (SR+)*, in which routers store the routing information as Bloom filters (an efficient fixed-size structure) in the Interest and data packets to preserve the original content name and promote caching.

Bloom Filter-Assist (BFA): For *PERSIA*'s attack mitigation phase, we propose the novel Bloom Filter-Assist (*BFA*) forwarding strategy to cope with compromised infrastructure threats. In BFA, each core router autonomously monitors its interfaces drop rates. We note that one of BFA strength is that it works independent of any particular namespace. This allows BFA to detect and mitigate the impact of DDoS attacks even if the attackers use unique names for each malicious request. When an interface's drop rate reaches a pre-defined threshold, the router instantiates a counting Bloom filter (CBF) [1] for that interface (*i.e.,* suspicious interface) and uses it until the suspicious interface's drop rate falls below the threshold. It noteworthy that the router inserts all the requests arriving at the suspicious interface into the corresponding CBF–regardless of requests' names. We chose CBF for BFA due to its deletion capability, which helps reduce its false positive rate. In BFA, when an Interest arrives on a suspicious interface with an existing CBF, the router inserts the content name into the CBF and avoids PIT insertion to decouple suspicious Interests from the PIT. The PIT is used, as usual, to store the Interests arriving on the non-suspicious interfaces.

To enable Interest aggregation, a router checks the name of a newly arriving Interest over all of its CBFs and the PIT before forwarding the Interest. The Interest is suppressed (dropped due to aggregation) if it exists in more than two [2] of the CBFs or in the PIT. Otherwise, the router forwards the Interest towards the content provider. Upon receiving a data packet, the BFA strategy iterates through all the suspicious interfaces that use CBF (excluding the interface through which the data packet has arrived) and looks up the content name in their CBFs.

On a successful lookup, the router forwards the data packet over that interface, reduces the CBF count, and continues the iteration until all the CBFs are checked. The router further performs PIT lookup on the content name for honest interfaces. Routers constantly monitor the Interest drop rates of their suspicious interfaces to revert to using PITs whenever these rates fall below the pre-defined threshold.

To prevent rapid CBF saturation, BFA strategy only deletes benign Interests (those legitimate requests that elicit data) from the CBF, leaving malicious Interests in the CBFs (since they do not elicit data). The malicious Interests that remain in CBFs increase their false positive probability, which consequently result in unsolicited data delivery and higher communication overhead. Thus, in BFA, routers reset their CBFs when saturated–false positive rates reach a pre-defined threshold (system parameter). While smaller thresholds increase *PERSIA*'s precision, they require more frequent CBF resets. In contrast, higher thresholds increase the unsolicited data delivery while requiring less frequent CBF resets. It should be noted that the acceptable CBF false positive rate is a system parameter that the network administrators need to tune depending on factors, such as the network condition and the expected users' service satisfaction.

6 SECURITY ANALYSIS AND DISCUSSIONS

In this section, we review how *PERSIA* thwarts identified threats, such as attackers solving puzzles, hijacking of tokens, malicious users, compromised infrastructure, and DDoS of proxies'.

6.1 Puzzle Complexity and Solvability

In *PERSIA*, attackers can undermine the attack prevention phase by having a botnet requesting and solving puzzles and generating valid tokens for communications. There is no way to differentiate such attackers from a legitimate user. However, BFA will still detect and rate limit the suspicious interfaces, which despite negatively impacting the legitimate users collocated with the attackers (only), prevents system-wide DDoS.

We note that *PERSIA* can be enhanced if the proxies and edge/core routers communicate. In NDN, routers can maintain statistics of Interests load and satisfaction rates on each of their interfaces in the strategy layer [27, 32]. These statistics can be conveyed to the proxy(ies) to tune the PoW complexity to control overall system Interest rate. Routers and the proxies may cooperate to rate limit individual users by assigning complexity customized puzzles based on their token usage statistics.

6.2 Replay Attack & Token Hijacking

In *PERSIA*, r_E verifies the Interests' tokens to discard malicious Interests. We define a malicious Interest as one that either does not contain any token or includes a fake token (token with an invalid secret seed or an invalid puzzle solution). Edge router r_E immediately detects and drops token-less Interests. It also detects Interests with fake tokens as they are not in r_E's table. This addresses cases *(a)* and *(b)* of the threat model.

An attacker, collocated with user u_i, can eavesdrop on u_i's communication and intercept her Interests. The attacker can orchestrate a replay attack by re-forwarding those Interests or hijack the tokens and use them before u_i's Interests reach r_E. *PERSIA*'s token sequencing feature prevents replay attack as validated tokens are consumed and replaced by edge routers with subsequent tokens; replayed tokens will be dropped. Also, a replaying attacker cannot generate the subsequent tokens as it does not have access to the corresponding seed.

A hijacker may interrupt/capture u_i's communication and use the unused, intercepted token in his communication–the hijacker's

[1] CBF is the generalization of Bloom filter, which allows record deletion [13].
[2] Using two CBFs helps reduce the impact of false positive match in one CBF.

Interest consumes u_i's token at r_E. A potential solution is for $\mathcal{P}$ to send the public key of u_i in its solution table response to r_E. User u_i can sign each Interest (includes token), and r_E can verify the signature, to detect hijacking with a signature mismatch. Cost of signature and verification can be amortized with a signed manifest of a set of Interests. For user privacy, pseudonymous certificates can be used.

6.3 Malicious Users

A malicious user u_m can attack the system by obtaining and solving the puzzle and sharing the solution and seed with other users/attackers. If u_m only shares the solution with the attackers, they cannot generate new tokens without the puzzle's seed; u_m has to share the puzzle's secret seed for the attack to work. However, sharing the seed allows the attacker to hijack u_m's token chain, which consequently renders u_m's Interests useless at r_E due to token chain violation as described in Definition 3.2–undesirable for u_m. Further, if the Interests/a manifest is signed, then u_m has to share his private key with the other attackers–more inconvenience. Hence, more than one users (attackers) can be successfully prevented from token reuse, reducing DDoS potential.

In another attack scenario, malicious users may request a large number of puzzles to exhaust the routers' storage, where the puzzles' solutions (VT table) are stored. The small amount of information for token verification, the limited number of users per edge router, and the routers' extensive caching capacity, reduce the significance of this attack. To further strengthen *PERSIA*'s resiliency, each router independently associate its cached puzzles with expiry times, allowing them to purge stale puzzles from their VT. We note that a puzzle becomes stale if it *(1)* reaches its token limits or *(2)* passes their expiry time regardless of the remaining unused tokens. The puzzle expiry time is an adjustable parameter the can be set by the network administrator based on factors, such as the number of users associated with each edge router, the puzzle request rate, and the edge routers available storage.

6.4 Compromised Routers

As we discussed in the security assumptions, we expect the ISP's routers to behave rationally. That is, routers neither intentionally expose secret information to attackers nor forward the malicious Interests to the core network. However, attackers can compromise a subset of the routers, causing malicious Interests to be forwarded towards the core, in effect, degrading the users' QoE. *PERSIA* allows core routers to dynamically deploy BFA to cope with this threat. Although BFA allows all the Interests to enter the core network, it minimizes the effect of the DDoS on the network resources and maximizes the end users' QoE. In Section 7, we evaluate the effect of mitigation on users.

In *PERSIA*, we assume that only edge routers can be compromised. Our rationale behind this is two fold. First, the network's core is better monitored and managed—most ISPs deploy access control lists (ACLs) to prevent access to core network resources while only letting routine transiting flows [7, 8]. Second, in *PERSIA*, the edge routers are responsible for DDoS attacks prevention while core routers mitigate the impact of malicious traffic that bypass *PERSIA*'s prevention mechanism. DDoS attacks originating at the edges inject the major attack volume. Compromised core routers can move the attack traffic further upstream into the core, but upstream core routers (if not compromised) will autonomously execute BFA upon attack detection and can curtail the attack impact to negligible levels.

6.5 Attacks on Proxies

A potent vulnerability with proxies is the resource exhaustion attack on them [43]. Attackers can overwhelm the proxies processing or communication resources by requesting a large number of puzzles to disrupt the puzzle distribution process.

To alleviate this vulnerability, NDN's pervasive caching can be leveraged. Routers can cache the generated puzzles to optimize the puzzle delivery process and eliminate the proxy as a single point of failure. Users still need to obtain the puzzles' names and the secret seed from the proxy–can be obtained with one Interest-data communication. Resiliency can be further improved with multiple proxies. We note that compromised proxies will indeed undermine *PERSIA*'s DDoS prevention phase. However, BFA, which functions orthogonal to attack prevention, will still mitigate DDoS attacks' impacts, and quite effectively per our simulations.

7 SIMULATION RESULTS AND ANALYSES

In this section, we explain our simulation setup, the scope of implementation, and our simulation results.

7.1 Simulation Setup

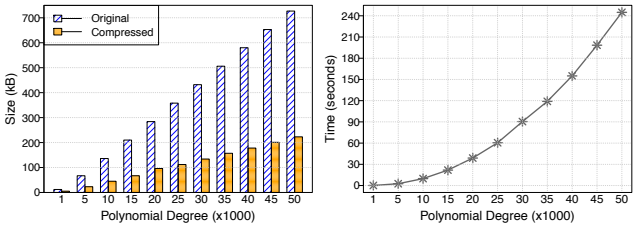
We extended the NDN simulator (ndnSIM-2.3 [21]), built on top of the ns-3. For *PERSIA*, we implemented puzzle generation (Protocol 1), validation table generation (Protocol 2), users' PoW execution (*Protocol 3*), token verification (Protocol 4), the BFA strategy (Section 5), and the improved RL and SR (SR+) strategies.In the SR+, we used 64-bits long Bloom filters with three hash functions; in the BFA, we used 10 kb long counting Bloom filter with four hash functions. *PERSIA* initiates the mitigation phase when the Interest drop rate exceeds 60%.

Table 2: Network Topologies.

	Ebone	Telstra	AT&T
Core Routers	134	295	550
Edge Routers	28	46	71
Providers	10	10	10
Proxies	5	5	5
Legitimate Users	47	98	147
Attackers	53	102	153

Network Setup: For our simulations, we used three Rocketfuel topologies [29], *Ebone*, *Telstra*, and *AT&T* with the number of routers, providers, proxies, users, and attackers shown in Table 2. We chose the number of attackers to be roughly equal to legitimate users to represent a significant botnet attack scenario (*e.g.,* a Mirai botnet attack). Each content provider serves a catalog of 1000 unique contents. Proxies were placed within 3 to 6 hops from the users and at least 3 hops from the providers.

User Setup: We implemented a *Zipf* user application in which the content popularity follows a Zipf distribution with $\alpha = 0.7$.



(a) Puzzle Sizes (b) Puzzle Solving Latencies

Figure 3: PoW Implementation: (a) puzzle sizes for various puzzle difficulties (polynomial degree); (b) puzzle solving times for various puzzle difficulties.

The users' request rates were selected randomly from the uniform distribution $\mathcal{U}[20, 30]$ Interests/sec. In *PERSIA*, the puzzle difficulty is represented by the degree of the interpolation polynomial. We experimented with a range of polynomial degrees from 1 K to 50 K and plotted the puzzles' sizes and their solution times (Figure 3); results were averaged over 100 runs.

Figure 3(a) shows the puzzles' sizes (the original and compressed puzzles using *Lempel-Ziv-Markov chain algorithm* (LZMA)) increase linearly with their degrees. We used compression (LZMA has high compression ratio) as it significantly reduces the communication overhead of the puzzle distribution. We collected the results in Figure 3(b) by running Protocol 4 on a MacBook Pro running VMware, allocated 1GB RAM and one 2.5GHz, Intel Core-i5 processor (to simulate a standard low-end mobile device–one end of the devices spectrum).

Attacker Setup: We implemented the attackers in three categories. The first category attackers (**A1**) request content either without tokens or with fake tokens. The second category (**A2**) includes the malicious users who share their puzzles' solutions and secret seeds with others. The third category (**A3**) includes compromised edge routers.

To simulate full-blown attacks, attackers request unique, non-existent content with the rates chosen randomly from the uniform distribution $\mathcal{U}[50, 100]$ Interests/sec. We randomly selected 25% of the edge routers to be compromised (no token validation) implying 7, 12, and 18 compromised edge routers in Ebone, Telstra, and AT&T, respectively. *PERSIA*'s goal, with respect to attack categories, is to guarantee that: *(a)* the Interests of the first category attackers do not pass the edge routers; *(b)* only one user successfully retrieves the content, when the puzzle information is shared; and *(c)* the compromised routers do not affect the legitimate users' QoE.

7.2 Results and Analysis

We ran our simulations for 2500 seconds on a machine (with Intel Core-i7, 4.0 GHz processor, 64 GB RAM) running Ubuntu 16.04, and averaged the results of each topology over ten runs with different seeds. We used polynomials of degree 5000 for the puzzle difficulty, which defines the users' computation and communication overhead (the number of Interests a user sends to retrieve a puzzle from a proxy). The ndnSIM simulator does not consider the time of computational operations. Thus, we benchmarked the latency distribution (identified as a normal distribution) of puzzle solving as $\sim \mathcal{N}(2.56s, 0.046s)$ for a polynomial of degree 5000. To account for puzzle solving time, users schedule sending Interests after a random wait time generated from the benchmarked latency distribution. Puzzle difficulty (polynomial degree) is a tunable system parameter, which helps adjust users' request rates. Figure 3(b) shows puzzle solving latencies of 0.1 seconds 244.8 seconds for polynomials of degree 1000 and 50000, respectively.

We activate the first two attack categories (A1 and A2) between 500 to 1000 seconds. The A1 and A2 attackers avoid PoW execution, use invalid tokens, use intercepted tokens (replay attackers), and share their credentials (malicious users). From 1000 to 1500 seconds, we stopped A1 and A2 attackers and introduced the compromised routers (A3). From 1500 to 2000 seconds, we activated the attackers and compromised routers (A1, A2, and A3) for massive DDoS orchestration. At 2000 second, we stopped all attacks to analyze *PERSIA*'s attack recovery.

Table 3: Legitimate users' success rates (%) under compromised routers (averaged per second).

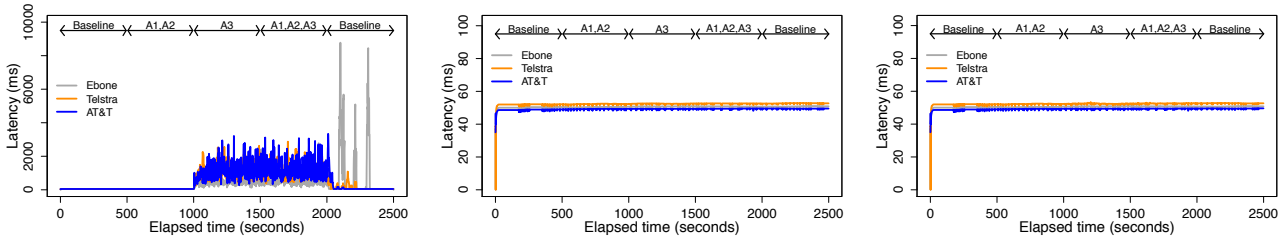
	Ebone	Telstra	AT&T
RL	85.06	75.54	80.28
SR+	99.99	99.99	99.99
BFA	99.97	99.92	99.95

During the first 1000 seconds of simulations, all strategies resulted in 100% satisfaction rate even when we enabled DDoS attackers (500 to 1000 seconds) due to *PERSIA*'s attack prevention mechanism. Introducing compromised routers (7, 12, and 18 for Ebone, Telstra, and AT&T respectively) affected the mitigation strategies averaged satisfaction rates (refer to Table 3). Table 3 only shows the per second averaged measurements for the period between 1000 to 2000 seconds. For RL, enabling compromised routers reduced the users' satisfaction rates across all topologies. This behavior is expected as routers cannot distinguish the benign from the malicious Interests and hence, rate limit all Interests.

In contrast, compromised routers imposed minimal impact on users' QoE for the SR+ and BFA. The small drop rates of these two strategies ($< 0.1\%$) are due to the Bloom filter's false positive probability. For the BFA strategy, a CBF's false positive may delete a content name from the CBF, preventing the content delivery, but that was rare.

Figure 4 shows the content retrieval latencies of all strategies. We note that the small dips in the lines are due to users obtaining their new puzzles from proxies. All strategies performed similarly when the network is under the first two attack categories (from 500 to 1000 seconds). Similar to the satisfaction rate results, enabling compromised routers affected the latency measurement of the RL strategy (Figure 4(a) from 1000 to 2000 seconds). In RL, the average communication latency increased significantly to 1000 ms due to frequent packet drops and Interest retransmissions. However, the latency in the SR+ and BFA strategies remained steady (around 50 ms). *The SR+ and BFA strategies outperformed RL in meeting users' QoE expectations.*

Figure 5 presents the average per-second throughput of the users, and core and edge routers, and the averaged per second PIT sizes of the core, edge and compromised routers for the AT&T topology



(a) RL's latencies (b) SR+'s latencies (c) BFA's latencies

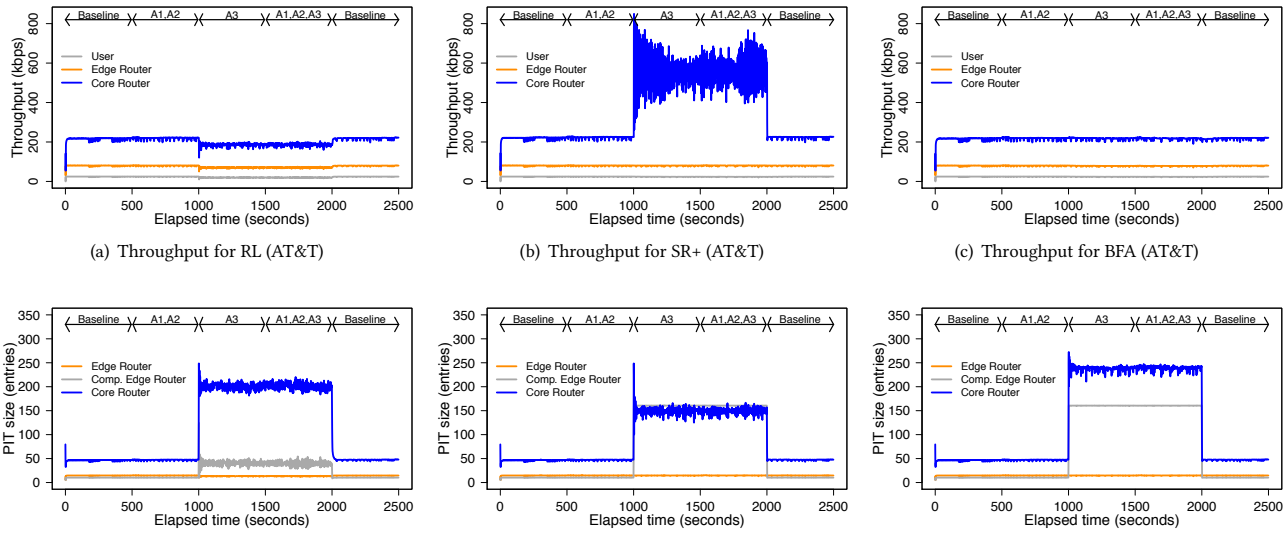
Figure 4: The averaged communication latency per second for legitimate users.

(due to space limitations, we did not include the throughput and PIT size results of Ebone and Telstra topologies but results of these topologies follow a similar trend). The average throughput of the users and routers did not change between 500 to 1000 seconds for any strategy. However, enabling compromised routers (from 1000 to 2000 seconds) affected the throughput measurements in the RL and SR+ strategies (Figure 5(a) and 5(b)) while the BFA strategy preserved its steady throughput (Figure 5(c)).

With RL, all entities achieved lower throughputs due to random dropping of Interests at routers (including benign and malicious Interests), which is also reflected in lower satisfaction rates (refer to Table 3). We note that increasing request rates and the number of attackers magnifies RL's throughput degradation. In contrast, enabling compromised routers increased the core routers' throughput in SR+ strategy–maps to its higher communication overhead (Figure 6)–which is an undesirable phenomenon. There are two reasons for the SR+'s higher throughput; first, Bloom filters' false positive causes unsolicited data delivery, which wastes the bandwidth and increases the communication overhead. Second, SR+ disables Interest aggregation resulting in redundant packets delivery.

In contrast, BFA maintained its steady throughput even in the presence of compromised routers, proving its strength in satisfying users' QoE without adversely affecting the network. We emphasize that BFA's dynamic nature–each core router autonomously deploys BFA strategy upon attack detection–resulted in 35%, 33.33%, and 28.65% of the core routers deploying BFA in Ebone, Telstra, and AT&T networks, respectively. We also tested the AT&T network with varying percentages of compromised edge routers, 25% to 55% in increments of 10%, the percentage of core routers that deployed BFA were 28.65%, 31.38%, 37.7%, and 43.01% respectively, which is a sublinear increase for linear increase in attack surface.

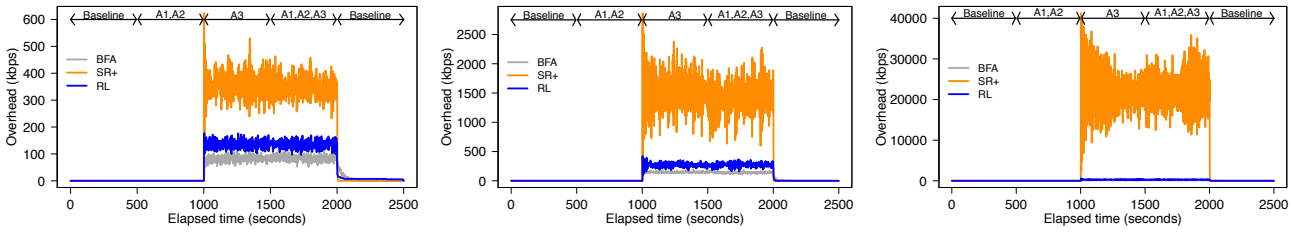
Note that the edge routers' computation overhead in the prevention phase includes a table lookup and one hash operation per Interest; a negligible overhead compared to NDN routine. The core



(a) Throughput for RL (AT&T) (b) Throughput for SR+ (AT&T) (c) Throughput for BFA (AT&T)

(d) PIT size for RL (AT&T) (e) PIT size for SR+ (AT&T) (f) PIT size for BFA (AT&T)

Figure 5: The averaged throughput and PIT sizes for the AT&T network. RL's low throughput (from packet drops) and SR+'s higher throughput (from communication overheads) are undesirable—BFA is consistent.



(a) Ebone (b) Telstra (c) AT&T

Figure 6: The averaged per second communication overhead for RL, SR+, and BFA strategies.

routers' computation overhead is in executing the BFA strategy (maintain a constant number of CBFs), which requires four hashes for insert and lookup operations.

Figure 5(d), 5(e), and 5(f) present the PIT sizes (averaged per-second) of the core, edge, and compromised edge routers in AT&T network (due to space limitations, we did not include the results of Ebone and Telstra topologies). For all strategies, the PIT sizes of the core and edge routers remained steady in the first 1000 seconds–showing *PERSIA*'s robustness DDoS prevention. However, enabling compromised routers increased the core routers' PIT sizes in all strategies as compromised routers forward the fake Interests into the core.

Among all, SR+ experiences slightly smaller PITs since it decouples all the Interests from the routers' PITs and allows the routers to implicitly coordinate their decisions. We design the BFA strategy to store 1% of the Interests in the routers' PITs to enable attack recovery (for success rate measurement), which resulted in BFA strategy experiencing a slightly higher PIT sizes (250 compared to 175 and 225 for SR+ and RL, respectively).

The legitimate edge routers maintained their PIT sizes, across all strategies, since they only forward the benign Interests that will fetch the requested data. The compromised edge routers (Comp. Edge Router in Figure 5) in the RL strategy observed smaller PITs due to leveraging the NACK feature; on receiving a NACK packet, the routers removes the corresponding PIT entry. However, these NACK packets caused the RL strategy to experience a higher communication overhead compared to the BFA strategy.

Figure 6 represents the strategies communication overheads. In our experiments, SR+ introduced the highest communication overhead followed by RL in the middle and BFA with the lowest overhead. Despite the similarity of SR+ and BFA strategies in causing unsolicited content delivery, SR+ undermines Interest aggregation, and hence, produces significantly higher overhead. For RL, the Interest retransmissions and the NACK packets are the primary sources of communication overhead. *The BFA strategy incurred minimal communication overhead while satisfying the users' QoE. We note that scaling the size of the network (Ebone to Telstra to AT&T) escalates the extent of the communication overhead in SR+.*
Insights: The RL strategy negatively impacts users' QoE–lower satisfaction rate and higher communication latency. The SR+ strategy, despite meeting QoE expectations, introduces significant communication overheads. The BFA strategy provides similar user' QoE satisfaction to SR+ with the least impact on the network resources. The trend shows that scaling the size of the network (from 162 to 341 to 621 routers) intensifies the attack's effects on RL or SR+ without perceptibly affecting BFA! These network sizes are representative of edge networks in size (especially at the lower end) and hence will apply to them.

8 RELATED WORK

In this section, we review the state-of-the-art literature (summarized in Table 4) in NDN-based DDoS countermeasures and compare them with *PERSIA* . We refer the readers to a survey on ICN security [33] for more details. In NDN, the non-existent contents are those with valid name prefixes and invalid suffixes, which are used

Table 4: Classification of DDoS (IFA) Mitigation Approaches and Their Salient Features

Mechanism	Target	Objective	Content Type	Technique
PERSIA	Router/Provider	Prevention/Mitigation	All Content	Proof-of-Work & BFA
Afanasyev [1]	Router	Mitigation	Non-Existent	Rate Limiting & Per-face Fairness
Gasti [14]	Router/Provider	Mitigation	All Content	Rate Limiting & Per-face Statistics
Compagno [9]	Router	Mitigation	Non-Existent	Rate Limiting & Per-face Statistics
Dai [10]	Router	Mitigation	Non-Existent	Rate Limiting & PIT Size Monitoring
Wang [40]	Router	Mitigation	Non-Existent	Fuzzy Logic Detection & Rate Limiting
Nguyen [25]	Router	Mitigation	Non-Existent	Statistical Hypotheses Testing Theory
Wang [39]	Router	Mitigation	Non-Existent	Self Routing for Suspicious Requests
Li [20]	Provider	Prevention	Dynamic	Per Request Proof-of-Work

by the attackers to orchestrate DDoS attacks. Providers generate and sign dynamic content upon receiving the request.

In general, the NDN-based DDoS attacks are categorized into two types. In the first type, the attackers either explicitly [9, 10, 25] or implicitly [26] flood the network with malicious requests to exhaust infrastructure's resources–preventing routers from serving the legitimate users. The second type attackers target the content providers by requesting dynamic contents to increase the content retrieval latency of other users [14, 20]. Among these, the overloading of the network infrastructure[1, 37–39] has more devastating outcomes compared to those targeting data providers. In contrast to existing works that focus on attack mitigation, *PERSIA* prioritizes DDoS prevention–by controlling users' request rates–over mitigation. *PERSIA*'s prevention method protects the infrastructure and providers independent of the content type, similar to [14].

Among existing countermeasures, rate-limiting is the predominant technique despite its inherent low user QoE (high communication latency and low satisfaction rate); it penalizes the legitimate users, who are collocated with the attacker. In *PERSIA*, similar to the majority of the existing work [1, 9, 14, 25, 40], routers collect and store statistical information such as per-interface drop rate. The collected information will be used for attack detection. Other countermeasures include theoretical modeling [25, 40], self-routing [39], and proof-of-work [20]. Different from [20], *PERSIA* prevents DDoS attacks that target the routers, which is more challenging.

In *PERSIA*, we tackled one of the most neglected assumption–the existence of malicious routers. *PERSIA*'s novel BFA strategy is an in-network mitigation scheme, which is dynamically deployed on impacted core routers to maintain flows' QoS and users' QoE in the presence of compromised routers. The BFA strategy, despite allowing the malicious traffic to enter the core network, delivers more than 99% of the legitimate users' requests (similar to self-routing [39]). The core routers dynamically invoke the BFA strategy when they sense DDoS attacks with drastically lower communication overhead compared to self-routing; a significant achievement considering the malicious traffic of an under attack network.

9 CONCLUSIONS

In this paper, we proposed *PERSIA*, a DDoS prevention and mitigation countermeasure for NDN-enabled ISP networks. *PERSIA*'s attack prevention uses a PoW and token-based mechanism to control users' communication rates. When infrastructure is compromised, *PERSIA* uses a novel mitigation strategy to preserve network resources without impacting users' QoE. Our simulations demonstrate *PERSIA*'s effectiveness and scalability in thwarting DDoS attacks compared to other work.

In the future, we augment *PERSIA* with a dynamic PoW difficulty alignment feature and further analysis with real DDoS traces. Considering *PERSIA*'s token-based approach, we will work on its integration with a token-based access control system [34] to build a secure and resilient framework towards Security-as-a-Service at the edge [30].

10 ACKNOWLEDGMENTS

Research supported by US NSF awards #1719342, #1345232, #1914635, and #2028797; EPSCoR Cooperative agreement OIA-1757207; Intel grant #34627535; and US DoE SETO award #DE-EE0008774. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the federal government.

REFERENCES

[1] A. Afanasyev, P. Mahadevan, I. Moiseenko, E. Uzun, and L. Zhang. 2013. Interest flooding attack and countermeasures in Named Data Networking. In *Proceedings of the IFIP Networking Conference*. IEEE, 1–9.
[2] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, A. Halderman, L. Invernizzi, and M. Kallitsis. 2017. Understanding the mirai botnet. In *USENIX Security Symposium*. 1092–1110.
[3] W. Ashford. [n.d.]. Russia compromised core router in US energy attacks. https://www.computerweekly.com/news/252437089/Russia-compromised-core-router-in-US-energy-attacks. Accessed: 2019-11-6.
[4] K. A. Bradley, S. Cheung, N. Puketza, B. Mukherjee, and R. Olsson. 1998. Detecting disruptive routers: A distributed network monitoring approach. *IEEE network* 12, 5 (1998), 50–60.
[5] C. J. Chung, P. Khatkar, T. Xing, J. Lee, and D. Huang. 2013. NICE: Network intrusion detection and countermeasure selection in virtual network systems. *IEEE transactions on dependable and secure computing* 10, 4 (2013), 198–211.
[6] CISCO. 2019. CISCO VNI Forecast Highlights. Retrieved August, 2019 from https://www.cisco.com/c/m/en$_{u}$s/solutions/service-provider/vni-forecast-highlights.html
[7] CISCO. 2019. ISP Infrastructure & Operational Security. Retrieved November, 2019 from https://www.cisco.com/c/dam/global/en$_{a}$e/assets/exposaudi2009/assets/docs/isp-security-routing-and-switching.pdf
[8] CISCO. 2019. Secure Network Infrastructure. https://tools.cisco.com/security/center/resources/securing$_{i}$p$_{v}$ideo#secure$_{c}$ore
[9] A. Compagno, M. Conti, P. Gasti, and G. Tsudik. 2013. Poseidon: Mitigating interest flooding DDoS attacks in named data networking. In *Proceedings of IEEE Conference on Local Computer Networks (LCN)*. 630–638.
[10] H. Dai, Y. Wang, J. Fan, and B. Liu. 2013. Mitigate ddos attacks in ndn by interest traceback. In *Proceedings of IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. 381–386.
[11] B. Donohue. 2018. Orbitz Breached, Compromising Core Cisco Routers, and More. https://mkacyber.io/news/orbitz-breached-compromising-core-cisco-routers-and-more/.
[12] C. Douligeris and A. Mitrokotsa. 2004. DDoS attacks and defense mechanisms: classification and state-of-the-art. *Computer Networks* 44, 5 (2004), 643–666.
[13] L. Fan, P. Cao, J. Almeida, and A. Broder. 2000. Summary cache: a scalable wide-area web cache sharing protocol. *IEEE/ACM Transactions on Networking* 8, 3 (2000), 281–293.
[14] P. Gasti, G. Tsudik, E. Uzun, and L. Zhang. 2013. DoS and DDoS in named data networking. In *Proceedings of International Conference on Computer Communications and Networks (ICCCN)*. IEEE, 1–7.
[15] J. Gonzalez, M. Anwar, and J. Joshi. 2011. A trust-based approach against IP-spoofing attacks. In *Annual International Conference on Privacy, Security and Trust (PST)*. IEEE, 63–70.
[16] J. Hughes, T. Aura, and M. Bishop. 2000. Using conservation of flow as a security mechanism in network protocols. In *Symposium on Security and Privacy*. IEEE, 132–141.
[17] V. Jacobson, M. Mosko, D. Smetters, and J. Garcia-Luna-Aceves. 2007. Content-centric networking. *Whitepaper, Palo Alto Research Center* (2007), 2–4.
[18] V. Jacobson, D. Smetters, J. Thornton, M. Plass, N. Briggs, and R. Braynard. 2009. Networking named content. In *Proceedings of the ACM international conference on emerging networking experiments and technologies*. ACM, 1–12.
[19] M. k. Franklin and D. Malkhi. 1997. Auditable metering with lightweight security. In *International Conference on Financial Cryptography*. Springer, 151–160.
[20] Z. Li and J. Bi. 2014. Interest cash: an application-based countermeasure against interest flooding for dynamic content in named data networking. In *Proceedings of the International Conference on Future Internet Technologies*. ACM, 2.
[21] S. Mastorakis, A. Afanasyev, I. Moiseenko, and L. Zhang. 2016. *ndnSIM 2: An updated NDN simulator for NS-3.* Technical Report NDN-0028, Revision 2. NDN.
[22] W. Meng, W. Li, C. Su, J. Zhou, and R. Lu. 2018. Enhancing Trust Management for Wireless Intrusion Detection via Traffic Sampling in the Era of Big Data. *Ieee Access* 6 (2018), 7234–7243.
[23] A. Mizrak, S. Savage, and K. Marzullo. 2008. Detecting compromised routers via packet forwarding behavior. *IEEE network* 22, 2 (2008).
[24] Named-Data Networking Manifest Embedding 2014. Manifest Embedding. https://named-data.net/publications/techreports/ndn-tr-25-manifest-embedding/.
[25] T. Nguyen, R. Cogranne, and G. Doyen. 2015. An optimal statistical test for robust detection against interest flooding attacks in CCN. In *Proceedings of IFIP/IEEE International Symposium on Integrated Network Management (INM)*. IEEE, 252–260.

[26] G. Panwar, R. Tourani, T. Mick, S. Misra, and A. Mtibaa. 2018. On implicit denial of service attack in NDN and potential mitigations. In *International Conference on Communications Workshops (ICC Workshops)*. IEEE, 1–6.

[27] G. Panwar, R. Tourani, T. Mick, A. Mtibaa, and S. Misra. 2017. DICE: Dynamic Multi-RAT Selection in the ICN-enabled Wireless Edge. In *Proceedings of the Workshop on Mobility in the Evolving Internet Architecture*. ACM, 31–36.

[28] B. Parno, D. Wendlandt, E. Shi, A. Perrig, B. Maggs, and Y. Hu. 2007. Portcullis: Protecting connection setup from denial-of-capability attacks. In *SIGCOMM Computer Communication Review*. ACM, 289–300.

[29] N. Spring, R. Mahajan, and D. Wetherall. 2002. Measuring ISP topologies with Rocketfuel. *ACM SIGCOMM Computer Communication Review* 32, 4 (2002), 133–145.

[30] R. Tourani, A. Bos, S. Misra, and F. Esposito. 2019. Towards security-as-a-service in multi-access edge. In *Proceedings of the 4th ACM/IEEE Symposium on Edge Computing*. 358–363.

[31] R. Tourani, S. Misra, and T. Mick. 2016. Application-specific secure gathering of consumer preferences and feedback in ICNs. In *Proceedings of the 3rd ACM Conference on Information-Centric Networking*. 65–70.

[32] R. Tourani, S. Misra, and T. Mick. 2016. IC-MCN: An architecture for an information-centric mobile converged network. *IEEE Communications Magazine* 54, 9 (2016), 43–49.

[33] R. Tourani, S. Misra, T. Mick, and G. Panwar. 2017. Security, Privacy, and Access Control in Information-Centric Networking: A Survey. *IEEE Communications Surveys & Tutorials* (2017).

[34] R. Tourani, R. Stubbs, and S. Misra. 2018. TACTIC: Tag-Based Access ConTrol Framework for the Information-Centric Wireless Edge Networks. In *International Conference on Distributed Computing Systems*. IEEE, 456–466.

[35] J. Tromp. 2015. Cuckoo cycle: a memory bound graph-theoretic proof-of-work. In *International Conference on Financial Cryptography and Data Security*. Springer, 49–62.

[36] Verisign. 2019. Verisign DDoS Trends Report (2018). Retrieved August, 2019 from https://www.a10networks.com/sites/default/files/a10-tps-eb-verisign-distributed-denial-of-service-trends-report-vol-5-issue-2.pdf

[37] M. Wählisch, T. Schmidt, and M. Vahlenkamp. 2013. Backscatter from the data plane–threats to stability and security in information-centric network infrastructure. *Computer Networks* 57, 16 (2013), 3192–3206.

[38] K. Wang, J. Chen, H. Zhou, Y. Qin, and H. Zhang. 2014. Modeling denial-of-service against pending interest table in named data networking. *International Journal of Communication Systems* 27, 12 (2014), 4355–4368.

[39] K. Wang, H. Zhou, Y. Qin, J. Chen, and H. Zhang. 2013. Decoupling malicious interests from pending interest table to mitigate interest flooding attacks. In *Proceedings of IEEE Globecom Workshops (GC Wkshps)*. 963–968.

[40] K. Wang, H. Zhou, Y. Qin, and H. Zhang. 2014. Cooperative-Filter: countering Interest flooding attacks in named data networking. *Soft Computing* 18, 9 (2014), 1803–1813.

[41] B. Waters, A. Juels, A. Halderman, and E. W. Felten. 2004. New client puzzle outsourcing techniques for DoS resistance. In *Proceedings of the 11th ACM conference on Computer and communications security*. ACM, 246–256.

[42] S. Zargar and J. Joshi. 2010. A collaborative approach to facilitate intrusion detection and response against DDoS attacks.. In *International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom)*. IEEE, 1–8.

[43] S. T. Zargar, J. Joshi, and D. Tipper. 2013. A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks. *IEEE Communications Surveys and Tutorials* 15, 4 (2013), 2046–2069.