What If Routers Are Malicious? Mitigating Content Poisoning Attack in NDN

Danye Wu[†‡], Zhiwei Xu[†‡], Bo Chen[§♯], Yujun Zhang[‡]
[†]University of Chinese Academy of Sciences, Beijing, China
[‡]Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
[§]Department of Computer Science, University of Memphis, Tennessee, USA
[♯]Center for Information Assurance, University of Memphis, Tennessee, USA
Email: {wudanye,xuzhiwei2001,zhmj}@ict.ac.cn, bchen2@memphis.edu

***Abstract*—Named data networking (NDN) shifts today's host-centric Internet architecture to a new data-centric network architecture. This well suits the increasingly mobile and information-intensive applications that dominate today's Internet. NDN allows routers to cache named content, leading to a significant improvement of content retrieval which, however, opens a door for many new attacks. In this work, we focus on content poisoning attack – the content poisoned by an attacker will be cached and propagated in the NDN network. Existing solutions unfortunately cannot work when the NDN routers are compromised by attackers.**

We propose ROM, the Router-Oriented Mitigation of content poisoning attack in NDN, offering security guarantees even when the routers are malicious. ROM defends against content poisoning attack by temporarily excluding the malicious routers from transmission path, eliminating (or significantly reducing) the possibility that the content will be poisoned during transmission. However, localizing malicious routers in NDN is very challenging due to NDN's distributed nature and the lack of global identifiers. We attack this issue from a new angle. We introduce reputation for each NDN router, and forward content based on the reputation. A router with a better reputation will be more likely honest and has a higher probability to be included into the transmission path. In addition, we design a novel mechanism to quantify the reputation value by utilizing our unique observations for NDN. Security analysis and simulations performed in ndnSIM demonstrate that ROM can mitigate the content poisoning attack with high efficiency and excellent accuracy.

***Index Terms*—Named Data Networking, Content Poisoning Attack, Router-oriented, Reputation Evaluation, Efficiency**

I. Introduction

The existing IP-based Internet architecture relies on IP addresses to guide packet transmissions. This architecture gradually becomes a bottleneck as network services and applications today are becoming mobile and data-centric. Many proposals have been developed to improve the IP-based Internet architecture. As a promising architectural design for future Internet, Named Data Networking (NDN) [3] aims to bridge the gap between the existing IP-based architecture and the content-centric applications. In NDN, the content is forwarded based on its name instead of its destination IP address. In addition, the content will be cached in routers along the transmission path (*i.e.*, in-network caching). These novel designs of NDN significantly improve transmission efficiency. However, they also open a door for new security flaws, which cannot be simply resolved by the existing mechanisms in current Internet.

One security flaw arises in NDN transmission: To ensure *integrity*, *authenticity*, and *correctness* of the content, NDN requires each content (together with its name) to be cryptographically signed by the content producer, and the content consumer should verify each received content [3]. However, NDN does not mandate the routers to verify the content due to various concerns like overhead. As NDN routers will cache content, and use them to satisfy subsequent requests, poisoned content (i.e., the content is corrupted or faked by an attacker) will be cached in the routers along the transmission path and eventually propagated to the consumers. Although the consumers can detect such poisoned content and re-issue requests, without appropriate mechanisms, they may keep receiving the poisoned content, because: 1) the poisoned content is cached in the routers along the transmission path; 2) a malicious router on the transmission path keeps poisoning the content passing by. This is known as *content poisoning attack* [2].

To mitigate content poisoning attack, some existing work [4], [13], [18] required routers to perform expensive signature verification (i.e., *in-network verification*). Although they proposed various optimizations to reduce overhead caused by the signature verification, the in-network verification nonetheless imposes a large burden on routers and thus significantly degrades network throughput. To eliminate in-network verification, some other work [13] proposed to prevent the poisoned content from being propagated by using consumers' feedback. As NDN requires consumers to verify the received content, the consumers usually have the knowledge on which content has been poisoned and can complain to the upstream routers about the poisoned content. By using the consumers' feedback, routers can exclude the poisoned content from their cache while eliminating the need of in-network verification.

We observed that all the existing mechanisms for mitigating content poisoning attack are "content-oriented", i.e., only the content will be verified and an honest router will always avoid caching the poisoned content. These content-oriented approaches, however, may be problematic when the attackers control NDN routers on the transmission path: the malicious routers may keep poisoning content, and without temporarily excluding those malicious routers from the transmission path, a

2324-9013/16 $31.00 © 2016 IEEE
DOI 10.1109/TrustCom.2016.99

consumer may keep receiving poisoned content (if the routers on the transmission path do not verify content passing by), or keep waiting for the content (if the routers verify and exclude the poisoned content). It thus becomes highly desirable to localize and temporarily exclude those malicious routers on the transmission path, which is unfortunately a very challenging task in NDN due to the highly distributed nature[1] of NDN: an NDN router usually does not possess a global identifier, and only exchanges messages with its neighboring routers.

In this work, we successfully design a novel "router-oriented" mechanism to address the aforementioned concern. Our basic idea is fourfold: First, we require each router to rank the reputation of each neighboring router. Specifically, if a neighboring router always propagates poisoned content, it will have a "poor reputation"; otherwise, it will have a "good reputation". Second, when passing a content packet to the next hop, the router prefers the neighboring routers who have good reputation. Third, to avoid in-network verification, the reputation values will be updated using customers' feedback [2], [4]. Four, a router's reputation is adaptive to its behavior. Specifically, a router with good reputation may become poor when it performs malicious behavior, and a router with poor reputation may turn good when it stops malicious behavior for a certain amount of time. This can help defend against a dynamic attacker who can control different routers at different time. As most of the routers and consumers are honest, in a high level, the reputation information in the entire NDN network should be properly maintained and updated, and thus the content will always transmit along the path with routers who are believed to be trustworthy.

Contributions. We summarize our contributions in the following:d

1) We propose a novel reputation evaluation protocol for NDN, in which we design a novel mechanism to quantify the reputation of a router based on several key observations in NDN. This reputation value is used to guide content forwarding, and can be updated asynchronously with content transmission, by evaluating the most recent transmission.
2) We design ROM, the first Router-Oriented mechanism which can Mitigate content poisoning attack in named data networking, offering security guarantees even when routers are malicious. Different from all the previous work, ROM defends against the content poisoning attack by temporarily excluding the malicious routers on the transmission path. ROM relies on reputation values for content forwarding, eliminating the need of localizing malicious routers.
3) We analyze the security of ROM. We also implement ROM in ndnSIM, and evaluate its efficiency and accuracy.

Organization. In Section II, we introduce the background knowledge about NDN and outline our related work. Section III presents our threat model and assumptions, and Section IV presents our main design. We provide security analysis in Section V and experimental evaluations in Section VI. We discuss our design in Section VII and conclude in Section VIII, respectively.

[1]This nature also makes it much easier for an attacker to deploy a malicious router in NDN.

II. Background and Related Work

In the following, we first introduce the background knowledge for NDN. We then outline the related work.

A. Named Data Networking (NDN)

Unlike an IP-based architecture (e.g., Internet), NDN uses hierarchically structure names rather than IP addresses to guide packet routing and forwarding [6]. NDN usually consists of three types of entities: *producers*, *consumers*, and *routers*. Producers publish content so that consumers can request and consume them (e.g., a video producer publishes videos in order to gain profits). Consumers who want to obtain certain content, issues an interest request for them. Routers are responsible to route and forward interest and content packets.

In general, a consumer requests target content following these steps: 1) The consumer sends an interest request identified by the name of the target content; 2) After having received the interest, the router stores both the interest and its ingress interface into a *Pending Interest Table* (PIT). The router will aggregate interests that request the same content into a single entry by appending them to the same ingress interface list. If the target content is cached in the router's *Content Store* (CS), the router will satisfy this interest request without forwarding it to next hop. Otherwise, it will search a *Forwarding Information Base* (FIB), longest-prefix matching on the content name to find the egress interfaces, and forward the interest from these interfaces to the next hop routers. If the interest cannot be satisfied by all the routers along the transmission path, it will reach the content producer who is able to respond with the target content. 3) The content packet will trace in reverse the path created by the interest back to the consumer. Specifically, when it reaches a router on the transmission path, the router will search its PIT table to obtain the corresponding PIT entry. After forwarding the content packet to all the ingress interfaces listed in the PIT entry, it will cache the content packet in its CS.

The clean-slate design of NDN well suits the increasingly mobile and information-intensive applications that dominate today's Internet. However, it opens a door for a large number of new security flaws including interest flooding attack [7], cache pollution attack [9], content poisoning attack [13], etc. In this work, we mainly focus on mitigating content poisoning attack.

B. Related Work

Mitigating content poisoning attack. The first line of work which targets at mitigating content poisoning attack relies on signature verification in routers. Gasti et al. [2] proposed probabilistic verification, in which a random subset of cached content will be verified. It is unclear what security guarantee can be obtained by verifying such a random subset. PPKD [4]

reduced the overhead for obtaining public keys. However, the routers still need to perform expensive signature verification for each content. Kim et al. [18] further reduced the overhead for signature verification by only performing verification on popular content. All of the aforementioned work has two limitations: 1) rely on in-network verification; 2) *cannot handle malicious routers*.

Another line of work which aims at mitigating content poisoning attack relies on consumer feedback [2], [13]. As consumers always need to verify the received content, by using the verification feedback sent by the consumers, routers can distinguish valid and poisoned content without performing expensive signature verification. Although this line of work can successfully eliminate in-network verification, they still *cannot handle malicious routers*. Simultaneously with our work, DiBenedetto et al. [14] proposed to utilize the feedback of consumers to mitigate content poisoning attack by somehow excluding the malicious routers during forwarding. They proposed two solutions, "Immediate Failover" and "Probe First". In the first solution, they simply make the next hop that returned bad data the least preferred option for future interests. This simple solution may be problematic because: first, what if the next hop router will turn good afterwards; second, this cannot differentiate the malicious behavior between creating the poisoned content and forwarding the poisoned content. In ROM, we use a global "reputation" value to evaluate how trusted a router can be, and a bad router can slowly recover its reputation if it stops performing malicious behavior. In the second solution, the nodes need to verify the feedback of consumers and to use the interest that retrieved bad content as a probe to detect bad content, which requires verifying the returned data packets and resuming forwarding to next hops upon a successful verification, leading to extra verification work. Different from their work, our design excludes the malicious routers based on multi-hop reputation values without extra verification work.

Content integrity and authenticity. Several pieces of work [5], [19], [21] related to content integrity and authenticity in NDN. However, they were not designed to mitigate content poisoning attack in routers.

Cache pollution attack. A cache pollution attack (CPA) is a different type of attack targeting the NDN caches. CPA attacks aim to preserve the unpopular content in NDN caches, leading to a significant decrease of NDN network throughput. To perform CPA attacks, an attacker repeatedly requests the unpopular content which can then fill the NDN caches with unpopular content [10], violating content locality in the caches. Various countermeasures were proposed to mitigate the CPA attacks [12], [11].

Web of trust. Many trust management solutions [24], [28] were proposed for web applications. Guha et al. [24] developed a global matrix of trust to obtain a global trust value for each website. The trust management was also well investigated for P2P applications [25], [26], [27]. To obtain the global trust information of network nodes in P2P systems, [26] took advantage of the theorem that global trust values should correspond to the left principal eigenvector of a matrix of normalized local trust values. All the aforementioned trust management solutions, centralized or decentralized, cannot be directly applied in NDN, as NDN routers do not possess global identifiers.

III. Threat Model and Assumptions

We assume most of the consumers and routers are non-malicious. Such an assumption is reasonable, as in practice, the attacker usually can only control a small portion of the entire Internet due to limitation of computational resources. Note that if a legal producer is controlled by the attacker and publishes malicious content, there is no technical way to mitigate such an attack, as the attacker will gain control over the legal publisher's private key, and become indistinguishable from the legal publisher.

We only consider an attacker who aims to perform content poisoning attacks, e.g., it may control routers and tamper content passing by, or pretend to be a content producer and publish malicious content (such a producer is usually illegal and can be viewed as a malicious router). Specifically, the malicious nodes controlled by attackers may pretend to honestly follow NDN protocol, *e.g.*, correctly forward interest packets and send back content packets. However, they may poison the content passing through by corrupting a portion of the content or replace the content with a faked one. As each content needs to be accompanied with a signature, the malicious nodes will either re-use the old signature or compute a new signature using their own private keys. In both cases, the poisoned content is detectable as the verifier will rely on a legal content producer's public key to verify the signature (there is a way to retrieve the public key for the corresponding producer [13], which is not the focus of this work). To maximize the longevity of an attack, the attacker usually sets the freshness field of the poisoned content as maximum.

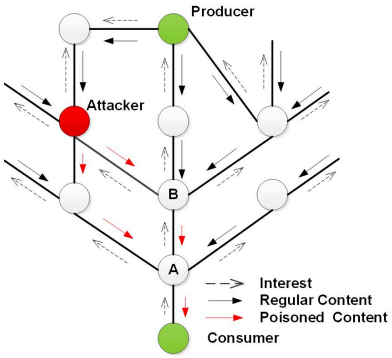

Fig. 1. An attack process

A concrete attack process is shown in Figure 1: The consumer sends out an interest packet to request certain content. If the content is not cached in the NDN network, routers will eventually forward the interest to the content provider, who will send back the desired content. A router controlled by the attacker may poison and propagate the content to its next-hop router. Without appropriate mechanisms, the poisoned content will be cached in the routers along the transmission path and eventually reach the consumer.

IV. ROM: A Router-Oriented Mitigation of Content Poisoning Attack

In this section, we present ROM, a Router-Oriented mechanism which can Mitigate content poisoning attack in NDN. ROM protects content against malicious routers by temporarily excluding them from the transmission path, so that the content will unlikely be poisoned during transmission. As localizing malicious routers is difficult in NDN, we introduce a novel reputation value to quantify the trustiness degree of a router. The reputation value is possible to be calculated based on two key observations for NDN: 1) A router who is far from the attacker should have a small probability to be the malicious router, and verse vice. 2) As the poisoned content will be propagated along multiple paths, the routers who are far from the attacker should receive more copies of poisoned content compared to the routers who are near the attacker.

Following the aforementioned observations, we quantify the reputation value of a router by evaluating the number of poisoned copies being received for certain content. In addition, we adapt the reputation value of a router over time. A router with "poor reputation" can turn good if it does not perform malicious behavior for a certain period; a router with "good reputation" will turn poor if it performs malicious behavior. Finally, we design a novel distributed protocol to update the reputation values asynchronously with content transmission.

The resulting design, ROM, can defend against content poisoning attack and ensure secure transmission even when routers are malicious, because ROM can ensure the content will be transmitted along routers who are unlikely to perform malicious behavior. In addition, ROM is very efficient as it does not rely on in-network verification. In the following, we first introduce the primitives required for ROM design (Sec. IV-A). We then present the details of ROM (Sec. IV-B).

A. Primitives

To ensure secure transmission and mitigate content poisoning attack, we need to exclude the malicious routers from transmission path. However, unlike channel-level security in IP-based networks, NDN implements content-level security and has no mechanisms for localizing a specific router. We thus address this problem from a new angle. We require a router to forward the content packets based on the reputation of its adjacent routers. In general, a larger reputation value means a router is more likely honest. In this way, the routers can always forward the content packets to the honest neighbors, and eventually the content packets will be transmitted along a transmission path filled with routers who are less likely to perform malicious behavior.

Now the question becomes, how can we measure the reputation of a neighboring router? Intuitively, a router who is closer to the attacker will have a higher probability to be the malicious router. Since this router is less likely honest, it should have a small reputation value. Similarly, a router who is far from the attacker will have a small probability to be the malicious router, and should have a large reputation value, as it is more likely honest.

The question further turns, given a router, how can we know whether it is close to the attacker (i.e., the malicious router), and how can we quantify this distance? We have another observation: In NDN, the content (including the poisoned content) is forwarded from multiple paths. In other words, if a lot of neighboring routers receive the same piece of poisoned content, this poisoned content should have been spread broadly and the malicious router who poisoned the content should be far away. We provide a concrete example in Figure 2 to illustrate this observation. An attacker forged a piece of content and used it to satisfy the requests sent by consumers. As the attacker is distant from router A, this forged content will have been propagated broadly before it reaches router A. As shown in the figure, router A received two copies of the forged content. On the contrary, router B only received one copy of the forged content, as it is closer to the attacker. *This observation makes it possible to quantify the distance between a given router and the malicious router by using the number of poisoned content being received for certain content.*

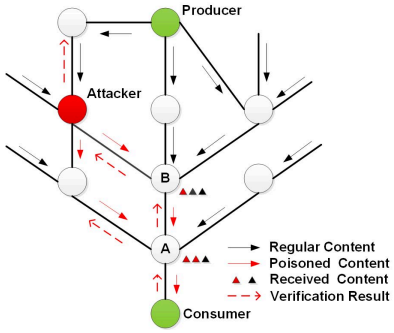

Fig. 2. A concrete example: the impact of distance from the attacker

We consider a multi-path NDN network, which contains a set of routers ($R = \{r_1, r_2, ..., r_i, ..., r_n\}$). Each router r_i has a set of k neighboring routers ($RN = \{r_{i1}, r_{i2}, ..., r_{ik}\}$). The set of content packets forwarded by routers are denoted as $C = \{C_1, C_2, ..., C_i, ..., C_m\}$. For each content packet C_i, we record the content name C_i, $hash(C_i)$, and IF_i which is the ingress interface of C_i. The verification results from consumers about content group C are represented as $VR = \{vr_1, vr_2, ..., vr_m\}$. $E_{K_s}(C_i)$ represents that the producer uses the private key K_s to sign the content C_i, and $D_{K_p}(E_{K_s}(C_i))$ means the verification process of content C_i using the public key K_p. Following these notations, we introduce three primitives supporting the design of ROM: poisoning ratio, feedback reputation score, and reputation value.

Poisoning ratio. We introduce *poisoning ratio* (PR) to indicate the percentage of poisoned copies being received for certain content. Poisoning ratio is defined as: upon receiving verification result vr_i, the router will calculate the ratio between the number of the poisoned copies for C_i and the number of all the copies for C_i received from various neighbors. We calculate $PR(C_i)$ for the upstream router that retrieved the poisoned content back as:

$$PR(C_i) = \frac{F_i(C_i)}{I_i(C_i)} \tag{1}$$

$F_i(C_i)$ is the number of the poisoned copies of C_i; $I_i(C_i)$ is the total number of the copies being received for C_i.

As analyzed previously, the PR value can somehow be used to determine the distance between the router and the attacker. A large PR value implies a high probability that the router is far from the attacker, and is thus less likely malicious. A small PR value implies a high possibility that the router is near the attacker, and is thus more likely malicious.

Feedback reputation score. An issue arises here that how we can punish the routers according to their relevance (captured in PR value) with the attacker. To decrease the reputation of the malicious routers, we design a punishment mechanism, following the principle of "linear increase, exponential decrease" [2]. We introduce *feedback reputation score*, FRS, representing the strength of the punishment on the routers who have forwarded poisoned content. If a router forwards the poisoned content, a corresponding negative FRS will be added to its reputation to prevent it from forwarding poisoned content again.

FRS indicates the strength of the punishment enforced on the neighboring routers who have retrieved and forwarded poisoned content. Updating FRS will follow the 'linear increase, exponential decrease' principle [23]. In other words, we will significantly decrease the reputation of a suspicious neighboring router, and slowly recover its reputation if it does not forward poisoned content afterwards. Specifically, the FRS can be computed as:

$$FRS_{kj}(C_i) = \begin{cases} 1 & C_i \quad is\ not\ poisoned \\ -\theta \cdot e^{\alpha(1-PR(C_i))} & C_i\ is\ poisoned \end{cases} \tag{2}$$

The content C_i is forwarded from router j to router k. θ is a penalty factor, and a larger θ indicates a more serious punishment enforced on router j. α is a adjustment parameter which adjusts the intensity of punishment.

Reputation value. Our countermeasure relies on the reputation of neighboring routers, based on which a router can determine whether it should retrieve content from these neighbors. To estimate how trusted an neighboring router can be, we accumulate the feedback reputation scores, computing the *reputation value* (RV). The reputation value from router k to its neighboring router j can be calculated in the following:

$$RV_{kj}(t) = \delta \cdot RV_{kj}(t-1) + \sum_{h=1}^{N} FRS_{kj} \tag{3}$$

N represents the content that comes from router j to router k in time interval $[t-1,\ t]$ and $t \geq 1$. $h \in [1, N]$ denotes the verification results of the content that pass through this router in this time interval. The reputation value RV at time t can be computed by using the RV value in time $t-1$ being multiplied by a time decay factor δ, to add the accumulation of FRS obtained within $[t-1,\ t]$. Note that $RV_{kj}(0) = 0$.

[2]A similar control mechanism can be found in TCP-Tahoe [23] transmission: slow start and congestion avoidance

B. The Design Details of ROM

In this section, we present our router-oriented mitigation (ROM). According to the content verification results from the consumer, the routers on the transmission path can calculate the content poisoning ratio and feedback reputation score. In addition, the routers update the reputation values of the upstream routers from which this poisoning content is forwarded. In this way, when a router needs to forward a specific interest, it can choose the honest neighboring routers as the next hop for forwarding this packet.

ROM includes four phases:

(1) The consumer sends an interest for content C_i to the network, and the first-hop router r_{kj} forwards this interest to some of its trusted neighbors in terms of the corresponding $RV_{kj}(t)$.

(2) If a piece of the target content C_i is forwarded from a router or producer to a router, this router stores the pair of $hash(C_i)$ and IF_i into the corresponding entry in *Pending Interest Table* (PIT) and delays erasing this entry. Note that multiple pieces of the requested content will be sent back, and the aforementioned pairs for these content pieces will be recorded. In addition, all of them have been signed by using the public key of the producer.

(3) After a piece of C_i arrives at the consumer, the consumer authenticates C_i, *i.e.* performing the signature verification process ($D_{K_p}(E_{K_s}(C_i))$). Once having obtained a negative verification result, this consumer will send this negative verification result (vr_i) and $hash(C_i)$ back to the upstream routers along the same transmission path with C_i.

(4) After a router r_k receives the vr_i and $hash(C_i)$ from the consumer, it compares the $hash(C_i)$ with all of the $hash(C_i)$ stored to count the number of the same malicious pieces. In this way, to punish the neighbor r_{kj} who sent this malicious piece back, router r_k will calculate the PR and FRS and update the present reputation value $RV_{kj}(t)$. Note that the router r_k will automatically add by a positive FRS (i.e., 1) to update $RV_{kj}(t)$ after a fix pending period after responding content C_i.

A detailed description of the updating process of the RV is provided in Algorithm 1. In line 1-4, after the router receives some pieces of content C_i, it stores their $hash(C_i)$ in the corresponding PIT entry. From line 5 to line 18, the router obtains the RV_{kj} on the upstream adjacent router. Specifically, in line 6, if the vr_i from the downstream neighbor is false, we evaluate the cached C_k pieces to find and count the same hash value (line 7-11). Then the $PR(C_i)$ can be obtained (line 12). According to whether the vr_i is true or false, the $FRS_{kj}(C_i)$ could be computed (line 13-16). In line 17, FRS_{kj} is accumulated to obtain the RV_{kj}. From the line 19 to 21, the RV_{kj} decreases every constant time interval by multiplying the time decay factor δ in order to eliminate the influence of the feedback reputation scores accumulated before.

Algorithm 1 The update of reputation value

Require: The content C_i; Verification result vr_i;
Ensure: The Reputation Value RV_{kj};

while receiving a piece of content C_i from r_{kj}
    $I_i(C_i)$++;
    Record the content;
end while
while receiving a vr_i of C_i from the downstream **do**
    if $!vr_i$ **then**
        For each C_k **do**
            if $hash(C_k)$==$hash(C_i)$ **then**
            $F_i(C_i)$++;
            end if
        end for
        $PR(C_i) = \frac{F_i(C_i)}{I_i(C_i)}$;
        $FRS_{kj} = -\theta \cdot e^{(1-PR(C_i))}$;
    else
        $FRS_{kj} = 1$;
    end if
    $RV_{kj} = RV_{kj} + FRS_{kj}$;
end while
while a time interval ends **do**
    $RV_{kj}(t) = \delta \cdot RV_{kj}(t-1)$
end while

V. Security Analysis

The security goal of ROM is to eliminate (or significantly reduce) the impact of malicious routers who may poison the content passing by. ROM achieves this goal by temporarily excluding the malicious routers on the transmission path. As routers forwards the content to a neighbor with good reputation, if the reputation values in the network can be correctly maintained and updated, the aforementioned security goal should be able to be achieved.

In ROM, routers update the reputation values of its neighbors according to the verification results generated by the consumers. There are two possible attacks here: 1) As the verification results need to be forwarded by the routers on the transmission path, malicious routers can modify them to disturb the protocol; 2) The malicious consumers may send back misleading verification results. Note that we assume that most of the routers and consumers are trusted.

We first show that the malicious routers cannot manipulate the verification results passing by without being detected. In ROM, only negative verification results will be forwarded back to the routers following the reverse transmission path. We analyze all the potential malicious behavior on negative verification results. The honest routers along the transmission path will always correctly forward the verification results. When a malicious router (i.e., a router controlled by the attacker) receives a *negative verification result* vr_i for content C_i, it will have three options to handle it: forwarding it correctly, modifying it to hide the attack, or discarding it. We analyze all the malicious behavior as follows:

- For the first option, the malicious router forwards vr_i to its upstream neighbor and tries to accuse its upstream neighbor. However, as this upstream neighbor honestly satisfied the interest requests for C_i with the same copy in the cache, it can detect difference between the vr_i received from the malicious router and the vr_i received from others, and then detect the malicious behavior of this malicious router (Note that we assume most of consumers are trusted in NDN).
- For the second option, the attacker changes vr_i to a positive verification result, aiming to hide the attack. This will not work as ROM does not propagate positive verification results.
- For the third option, the attacker discards this vr_i. However, such malicious behavior will actually expose the attacker, because the upstream routers probably have received the vr_i from other transmission paths, and can detect the abnormal behavior of the malicious router (Note that we leverage the multi-path forwarding to propagate vr_i and most routers are trusted in NDN).

We then show that malicious consumers cannot create a significant impact on the reputation values of the entire NDN network by sending back misleading verification results. The reasons are: 1) It is costly to send out a misleading verification result. A malicious consumer can only successfully send out a verification result upon receiving a piece of content, and such a result will trace back in reverse the transmission path. 2) The number of malicious consumers is small.

VI. Evaluation

A. Experimental Setup

We evaluated ROM by comparing it with PPKD [4], a known scheme which can reliably mitigate content poisoning attack. PPKD works as follows: To retrieve a content, the consumer first obtains the producer's public key digest and capsules it in the PPKD field of the interest. When forwarding the interest, the routers will record the digest from the PPKD field into the corresponding PIT entry. When sending back the corresponding content, the producer will store its public key in the KeyLocator field of the content packet. In this way, the router can simply hash the public key stored in the KeyLocator field and check whether it matches the PPKD in the corresponding PIT entry, and use the public key to verify the signature. If matching, the router knows that the content packet is sent by the authentic producer. It will forward and cache this content packet. Otherwise, it will discard this content packet. Note that the consumer should obtain the PPKD before issuing an interest, which leads to extra time overhead. The PPKD scheme ensures the security of the content by binding the interest with the producer's public key, preventing the attacker from modifying the content and the signature.

We used ndnSIM[16] for our simulations. All of our simulations were performed on a local machine, equipped with an Intel Core i5 2.5G CPU and 3.75G RAM, running Ubuntu

14.04. In our simulations, the size of content store is 1000 and the number of content items is 10,000. Also, the maximum PIT size are set as 15,000 entries and the rate of legitimate users is set as 50 interest/s. The duration for interests is 0-60s. We deployed the network topology of the Rocketfuel [17], which is the mapping engine part of the European backbone. The topology provided by the Rocketfuel has been widely used in ndnSIM-based simulations. In our network, there are totally 277 nodes, including 169 leaf nodes, 65 border gateway nodes and 45 gateway nodes. We deployed 159 consumers, among which 3 consumers are malicious.

To simulate background traffic, each legitimate consumer sends 100 regular interests per second. The name of an interest consists of a name prefix (*i.e.*, the first component of the name) and a distinct random number (*i.e.*, the second component of the name). We used ten optional name prefixes throughout our simulations: "/hds.univ-compiegne.fr", "/slocs.k12.ca.us", "/cs-bu.edu", "/york.ac.uk", "/accel.com", "/calypso.oit.unc.edu", "/andromeda.mis.net", "/gnn.com", "/sina.com" and "/readfar.com". We set the PIT size as 15,000, and the cache size as 1000. Five content producers will send malicious content to the network.

We use RSA to sign the content, and use SHA1 to hash the producer's public key for PPKD.

To evaluate the efficiency of ROM, we focused on three metrics, detection rate (DR), false positive rate (FPR), and false negative rate (FNR), respectively. They are obtained by the following formulas.

$$DR = avg(\frac{|r_{kj} : P_{kj} > \beta, RV_{kj} < \gamma|}{|r_{kj} : P_{kj} > \beta|}) \quad (4)$$

$$FPR = avg(\frac{|r_{kj} : P_{kj} < \beta, RV_{kj} < \gamma|}{|r_{kj} : RV_{kj} < \gamma|}) \quad (5)$$

$$FNR = avg(\frac{|r_{kj} : P_{kj} > \beta, RV_{kj} > \gamma|}{|r_{kj} : P_{kj} > \beta|}) \quad (6)$$

P_{kj} denotes the number of poisoned contents recently transmitted from the neighboring router r_{kj} to the router r_k. RV_{kj} is the reputation value from r_k to r_{kj}. β is the threshold of the number of poisoned contents. If P_{kj} is larger than β, the corresponding r_{kj} must have sent poisoned content to r_i recently, which is malicious. γ is the threshold for the RV_{kj}. If the RV_{kj} of r_{kj} is larger than the γ, r_{kj} will be considered as non-malicious according to the ROM scheme. According to method proposed in [20], we configure the above thresholds, including $\beta = 0.02$ and $\gamma = -0.1$.

B. Experimental Results

1) Efficiency Evaluation: We evaluated the efficiency of ROM by comparing it with PPKD, in terms of latency and throughput. Latency is the time from consumer's sending interest to receiving the corresponding content. And throughput is the rate at which contents can be processed in the whole network. Throughput is measured in terms of packets per second, *i.e.*, pkts.

Figure 3 compares the latency of ROM and PPKD. We observed that the PPKD scheme needs approximately $110ms$ to complete a secure content transmission, while ROM only needs around $16ms$. This confirms that ROM causes significantly less transmission delay compared to ROM.

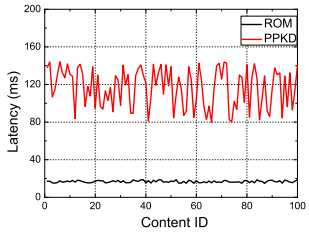

Fig. 3. Latency of ROM and PPKD

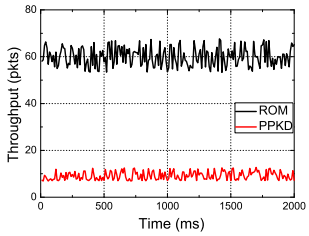

Fig. 4. Throughput of ROM and PPKD

Figure 4 compares the throughput of ROM and PPKD. We observed that the PPKD scheme can only process approximately 10 packets per second, when ROM can process around 64 packets per second.

All these results show that ROM has much better efficiency compared to the Hop by Hop scheme and PPKD. Potential reasons are: In PPKD, the consumer needs to fetch the public key digest of the corresponding provider in advance. In addition, PPKD sill needs to perform per-hop verification and digest. This will cause additional delay and reduce throughput.

2) Accuracy Evaluation: In the following, we evaluated the accuracy of ROM, in terms of detection rate (DR), false positive rate (FPR), and false negative rate (FNR). The DR, FPR and FNR values calculated at a specific moment are the statistical average of all the nodes in the network.

From Figure 5 and 6, we can observe that the DR is near 99.9%, the FNR is close to 0%, and FPR has a slight fluctuation but no more than 4%. The experimental results indicate that ROM can achieve a good detection accuracy. With a proper parameter configuration, ROM can detect almost all the routers who have the malicious behavior recently at a small cost of false positive error.

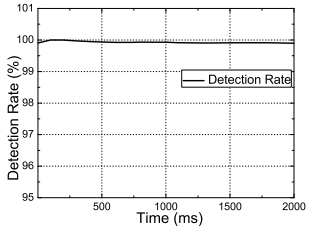

Fig. 5. Detection rate

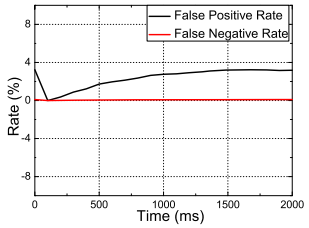

Fig. 6. False positive/negative rate

VII. Discussion

Critical routers are malicious. When a router in the critical position is compromised, it may block a lot of transmission paths and create a significant impact on the network. However, as critical routers are usually well protected, it will be very difficult for the attacker to compromise them. Even when a critical router is controlled, NDN can still function properly, as it is based on multi-path transmission. Nonetheless, just like today's Internet, critical routers being compromised still requires extra attention and may require network administrators to intervene.

Consumers' abusing of verification feedback. Ghali et al. [4] mentioned that allowing consumers to inform routers about poisoned content is problematic as the consumers may abuse it to perform various attacks. Requiring the consumers to sign the feedback cannot solve the problem as it may expose the consumers or can be abused by the consumers to perform DoS attack [4]. This issue can be mitigated in ROM, because: a consumer can only successfully send out a verification result upon receiving a piece of content. In addition, as mentioned in Section V, the poisoned verification result will either be ignored or detected by the routers.

VIII. Conclusion

In this work, we propose ROM, a Router-Oriented Mitigation of content poisoning attack in NDN. Our solution can defend against content poisoning attack even when routers are malicious, by smartly excluding the malicious routers from transmission path. To avoid verifying content signature hop by hop, we forward content based on the "reputation" of routers, and quantify this reputation by utilizing our unique observations in NDN. Security analysis and extensive simulations demonstrate the security, efficiency, and efficiency of ROM.

IX. Acknowledgement

This research is supported by the National Program on Key Basic Research Project (2012CB315804), the National Natural Science Foundation of China (61402446and61572474), the National Sciences and Technology Supporting Program of China (2012BAH45B01), the Instrument Developing Project of CAS (YZ201426). Bo Chen was supported by ARO W911NF-15-1-0576. Bo Chen would also like to thank the support from Center for Information Assurance at the University of Memphis.

References

[1] Kent S. *IP encapsulating security payload* (ESP). 2005.

[2] Gasti P, Tsudik G, Uzun E, et al. *Dos and ddos in named data networking*. 22nd International Conference on Computer Communications and Networks (ICCCN), IEEE, 2013: 1-7.

[3] Zhang L, Estrin D, Burke J, et al. *Named data networking (ndn) project*. Relatório Técnico NDN-0001, Xerox Palo Alto Research Center-PARC, 2010.

[4] Ghali C, Tsudik G, Uzun E. *Network-layer trust in named-data networking*. ACM SIGCOMM Computer Communication Review, 2014, 44(5): 12-19.

[5] Yu Y, Afanasyev A, Clark D, et al. *Schematizing and Automating Trust in Named Data Networking*. NDN Technical Report NDN-0030, 2015.

[6] Zhang L, Afanasyev A, Burke J, et al. *Named data networking*. ACM SIGCOMM Computer Communication Review, 2014, 44(3): 66-73.

[7] Afanasyev A, Mahadevan P, Moiseenko I, et al. *Interest flooding attack and countermeasures in Named Data Networking*. IFIP Networking Conference, 2013. IEEE, 2013: 1-9.

[8] Choi S, Kim K, Kim S, et al. *Threat of DoS by interest flooding attack in content-centric networking*. International Conference on Information Networking (ICOIN), IEEE, 2013: 315-319.

[9] Conti M, Gasti P, Teoli M. *A lightweight mechanism for detection of cache pollution attacks in Named Data Networking*. Computer Networks, 2013, 57(16): 3178-3191.

[10] Xie M, Widjaja I, Wang H. *Enhancing cache robustness for content-centric networking*[C]. INFOCOM, 2012 Proceedings IEEE. IEEE, 2012: 2426-2434.

[11] Xu Z, Chen B, Wang N, et al. *ELDA: Towards efficient and lightweight detection of cache pollution attacks in NDN*[C]. Local Computer Networks (LCN), 2015 IEEE 40th Conference on. IEEE, 2015: 82-90.

[12] Conti M, Gasti P, Teoli M. *A lightweight mechanism for detection of cache pollution attacks in Named Data Networking*[J]. Computer Networks, 2013, 57(16): 3178-3191.

[13] Ghali C, Tsudik G, Uzun E. *Needle in a haystack: Mitigating content poisoning in named-data networking*. Proceedings of NDSS Workshop on Security of Emerging Networking Technologies (SENT). 2014.

[14] DiBenedetto S, Papadopoulos C. *Mitigating Poisoned Content with Forwarding Strategy*[C]. Workshop on Name-Oriented Mobility (INFOCOM WKSHPS). 2016.

[15] Nakamoto S. *Bitcoin: A peer-to-peer electronic cash system*. Technical report, 2008.

[16] Afanasyev A, Moiseenko I, Zhang L. *ndnSIM: NDN simulator for NS-3*. University of California, Los Angeles, Tech. Rep, 2012.

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

[18] Kim D, Nam S, Bi J, et al. *Efficient Content Verification in Named Data Networking*. Proceedings of the 2nd International Conference on Information-Centric Networking. ACM, 2015: 109-116.

[19] Zhang X, Chang K, Xiong H, et al. *Towards name-based trust and security for content-centric network*. Network Protocols (ICNP), 2011 19th IEEE International Conference on. IEEE, 2011: 1-6.

[20] Mirkovic J, Prier G, Reiher P. Attacking DDoS at the source[C]. Network Protocols, 2002. Proceedings. 10th IEEE International Conference on. IEEE, 2002: 312-321.

[21] Hamdane B, Serhrouchni A, Fadlallah A, et al. *Named-Data security scheme for Named Data Networking*. Network of the Future (NOF), 2012 Third International Conference on the. IEEE, 2012: 1-6.

[22] Jacobson V, Smetters D K, Thornton J D, et al. *Networking named content*. Proceedings of the 5th international conference on Emerging networking experiments and technologies. ACM, 2009: 1-12.

[23] Jacobson V. *Congestion avoidance and control*[C]. ACM SIGCOMM computer communication review. ACM, 1988, 18(4): 314-329.

[24] Guha R, Kumar R, Raghavan P, et al. *Propagation of trust and distrust*[C]. Proceedings of the 13th international conference on World Wide Web. ACM, 2004: 403-412.

[25] Marti S, Garcia-Molina H. *Taxonomy of trust: Categorizing P2P reputation systems*[J]. Computer Networks, 2006, 50(4): 472-484.

[26] Kamvar S D, Schlosser M T, Garcia-Molina H. *The eigentrust algorithm for reputation management in p2p networks*[C]. Proceedings of the 12th international conference on World Wide Web. ACM, 2003: 640-651.

[27] Selcuk A A, Uzun E, Pariente M R. *A reputation-based trust management system for P2P networks*[C]. Cluster Computing and the Grid, 2004. CCGrid 2004. IEEE International Symposium on. IEEE, 2004: 251-258.

[28] Stewart K J. *Trust transfer on the world wide web*[J]. Organization Science, 2003, 14(1): 5-17.

[29] Laeeq K. RFAP, *a preventive measure against route request Flooding Attack in MANETS*[C]. Multitopic Conference (INMIC), 2012 15th International. IEEE, 2012: 480-487.