This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2017.2725229, IEEE Transactions on Information Forensics and Security

Security of Cached Content in NDN

Dohyung Kim, Jun Bi, Athanasios V. Vasilakos, Ikjun Yeom

***Abstract*—In Named-Data Networking (NDN), content is cached in network nodes and served for future requests. This property of NDN allows attackers to inject poisoned content into the network and isolate users from valid content sources. Since a digital signature is embedded in every piece of content in NDN architecture, poisoned content is discarded if routers perform signature verification; however, if every content is verified by every router, it would be overly expensive to do. In our preliminary work, we have suggested a content verification scheme that minimizes unnecessary verification and favors already-verified content in the content store, which reduces the verification overhead by as much as 90% without failing to detect every piece of poisoned content. Under this scheme, however, routers are vulnerable to *verification attack*, in which a large amount of unverified content is accessed to exhaust system resources. In this paper, we carefully look at the possible concerns of our preliminary work, including *verification attack*, and present a simple but effective solution. The proposed solution mitigates the weakness of our preliminary work and allows our work to be deployed for real-world applications.**

***Index Terms*—NDN, Network Cache, Content Poisoning Attack, Signature Verification**

I. Introduction

Named-Data Networking [2] has been proposed as a new networking paradigm. In NDN, users request content by specifying the content name rather than a location identifier such as an IP address. The request packet, referred to as the *interest*, is routed by the name and is served either by the content source or any intermediate nodes that have a copy of the content. Since routers in NDN store copies of the content that they relay, popular content is distributed over the network and users can obtain easy access to the content in terms of the content name. Obviously, the named-based access and in-network caching of NDN successfully minimize the amount of traffic over the link and provide efficient content retrieval.

However, despite these great advantages, NDN raises concerns in terms of security. Since in-network caching allows any node in the network to be a content provider, malicious users can inject poisoned content into the network. Once the poisoned content is placed on the network cache, referred to as the content store (CS), it is effectively distributed by the system itself. As a result, caches are contaminated by poisoned content, which seriously degrades the caching performance and isolates users from valid content.

To resolve this problem, NDN makes use of a digital signature that is verified either by routers or end-hosts. However, according to [3], routers cannot afford to verify all content, which can arrive at a rate above hundreds of Gbps, due to the large verification overhead. Many researchers have proposed practically feasible solutions, but these all have drawbacks/challenges. In [3], the authors presented a scheme that verifies content probabilistically. The hash values of content are used for verification instead of digital signatures. This approach significantly decreases the verification overhead, but it cannot overcome inter-packet dependency and trust management issues. In [4], a self-certifying name was discussed as a measure to mitigate content poisoning. And the authors designed a scheme that exploits users' feedback to exclude poisoned content. However, as described in [5], this scheme is at risk of excluding valid content from the CS based upon the fabricated feedback.

In the preliminary version of this paper [1], we present an efficient scheme that drastically mitigates the verification overhead while also preventing the malicious effects of cached poisoned content. In conventional research, cache integrity has been considered essential to deal with poisoned content and all content is pro-actively checked before being cached in the CS. According to our simulation study, however, over 90% of the content is evicted from the CS without serving requests, even when the cache-hit ratio is very high. This observation implies that a large amount of computation has been unnecessarily wasted. The proposed scheme sacrifices cache integrity and perform limited verification only to the cache-hit content. This limited verification helps to avoid unnecessary verification and favors already-verified content in the CS. Meanwhile, it effectively restrains poisoned content in the CS from being spread out over the network. Segmented LRU is exploited to make already-verified content remain in the CS for longer, which improves the verification efficiency by minimizing the repeated verification of popular content. Via ns-3 simulation [6], it is shown that the verification overhead is reduced to one thirtieth (when no poison content exists in the network). Alternatively, with poisoned content, the proposed scheme improves the verification efficiency by up to 20×. However, the proposed scheme has several controversial points such as increased access latency and *verification attack*.

Under the scheme in [1], verification should be done when content is accessed. Hence, verification delay is included in the service latency, which may impair service quality. We analyze the effect of verification delay and show that the verification delay is insignificant compared with the delay gain from in-network caching. Additionally, asynchronous verification is introduced to avoid verification delay and spread the computational overhead over an extended period of time. In asynchronous verification, cached content is verified in advance if extra computational resources are available. More importantly, *verification attack* is introduced as a limitation of our scheme. In *verification attack*, attackers load a large amount of unverified content into the CS and then launch requests for this unverified content to overwhelm the router's

The preliminary version of this work is presented in the 2nd ACM International Conference on Information-Centric Networking, September 2015 [1]