NDNSSEC: Namespace Management in NDN with DNSSEC

Pouyân Fotouhi Tehrani
Weizenbaum Institute / Fraunhofer FOKUS
pft@acm.org

Luca Keidel
Freie Universität Berlin
luca.keidel@fu-berlin.de

Eric Osterweil
George Mason University
eoster@gmu.edu

Jochen H. Schiller
Freie Universität Berlin
jochen.schiller@fu-berlin.de

Thomas C. Schmidt
HAW Hamburg
t.schmidt@haw-hamburg.de

Matthias Wählisch
Freie Universität Berlin
m.waehlisch@fu-berlin.de

ABSTRACT
In this demo, we showcase NDNSSEC. NDNSSEC provides a namespace management solution for named-data networking (NDN) based on the DNS ecosystem and its security extensions. Our prototype allows content consumers to verify the name ownership in commonly used NDN software.

CCS CONCEPTS
• Networks → Naming and addressing. Naming and addressing: Application layer protocols;

KEYWORDS
Namespace Management, Data Origin Authentication, ICN, NDN, DNS, DNSSEC

ACM Reference Format:

1 INTRODUCTION
Naming serves the goal of uniquely identifying things. We refer to the set of all possible names in a given context as a namespace [2, §8], and define namespace management as the task of partitioning a namespace into smaller management units, i.e., zones [3] and allocating these to zone owners. Both technical as well as non-technical aspects are integral to namespace management. On the one hand, technical means are to be provided to establish and authenticate owners of names. On the other hand, conflicting interests of different stakeholders, e.g., trademark infringements, have to be attended.

Although names constitute the cornerstone of ICN, namespace management in ICN has widely been ignored or limited to merely technical solutions [3]. By comparing the historical development of the Internet and the integral role that the DNS played, we argue that namespace management in ICN is also an enabler which can boost its deployment beyond experimental and research settings. If ICN is to reach the popularity and ubiquity of the Internet, without having to go through similar technical and legal challenges, we can rely on the existing and approved global infrastructure of DNS and ICANN.

Inherent to any ICN approach is the binding between names and data. Effective namespace management in ICN presupposes four bindings: (i) name to zone, (ii) name to producer, (iii) producer to zone, and (iv) zone to zone owner. The rationale behind these bindings is to verify whether a producer is authorized by a zone owner to publish data under its managed zone. In this demo, we present an implementation that implements all four bindings.

2 NDNSSEC PRIMER
NDNSSEC [5] leverages the already existing mappings between (digital) names and (real-world) owners provided by DNS, integrates the trust management framework of DNS security extensions (DNSSEC), and finally realizes a custom naming scheme for NDN [7] as the basis for securing names. In detail, it implements the following bindings:

(i) Name to Zone This binding is transitively authenticated by verifying name to producer and then producer to zone bindings.

(ii) Name to Producer A data packet in NDN already can bind its names to a producer through a digital signature.

(iii) Producer to Zone Using DNS, the zone owner authorizes producers by enlisting their public keys as DNSKEYs in its authoritative name server.

(iv) Zone to Zone Owner Using DNS zone delegation mechanisms to map a zone to its owner.

Before a producer can publish data under a specific zone apex, it requires authorization by the zone owner. For example if one wants to publish under /com/example/about, the owner of /com/example or respectively example.com must first authorize the publisher by enlisting its public key in its authoritative name server as a DNSKEY record. An authorized publisher in turn prefixes its content names with ndnfied [1] DNS zone apexes and signs them with the respective private key. A consumer can in turn authenticate data packets by retrieving the public key of the producer over DNS. To enable authentication, signed data packets in NDN contain the name of the publisher’s certificate in the KeyLocator field of the packet. The syntax looks as follows [6]:

/<SubjectName>/KEY/[KeyId]/[IssuerId]/[Ver]
Certificate names in NDNSSEC follow the same convention with two constraints: (i) SubjectName must denote the zone apex, and (ii) KeyId must be the digest of the producer’s public key calculated the same way DNSKEY digests are determined for DNS DS records [4, §5]. The former allows distinction between the zone apex and the rest of the NDN name. The latter enables a consumer to fetch the correct key among possible DNSKEY candidates from the authoritative name server.

An extended NDN consumer software authenticates packets by first making sure that the content name is prefixed with a zone apex matching the SubjectName segment of its KeyLocator. If not, the packet is discarded as unauthorized. Next, the DNSKEYs of authorized producers for that zone apex are fetched over DNS, their digests are computed, and finally compared with the KeyId to find the matching public key. If no key matches, the packet is discarded as unauthorized, otherwise the matching key is used to verify the signature. To simplify the key retrieval process, each domain can provide a logical public key resolver (similar to a DNS stub resolver) that proxies the DNS related tasks and is trusted by users within that domain. The resolver returns matching public keys for a given zone apex and KeyId, or an explicit NACK if none is found. A domain-local resolver can also be leveraged to manage private zones: zone apexes which are explicitly used for internal purposes (e.g., internal top-level domains).

3 SHOWCASE

We demonstrate our implementation, covering the complete cycle of configuring and deploying NDNSSEC in four phases: 1. bootstrapping keys and certificates, 2. populating DNS data, 3. signing and publishing a packet, and 4. authenticating and consuming the packet as summarized in Figure 1.

In the bootstrapping phase, two asymmetric key pairs, a zone signing key (ZSK) and a key signing key (KSK), are generated and are used to create and sign a DNS zone file using a single script sign.sh. The KSK is further utilized to create an NDN certificate through a standalone application, createcert. The certificate is then imported in NDN local KeyChain through ndnsec-import tool provided by the NDN toolchain and library. Eventually, the certificate is used by the producer to sign its content. Note that for the sake of simplicity the zone owner and the producer are the same entity in our demo. In the data publication phase, the producer registers for the names or name prefixes under which it is authorized to publish and signs all data packets using the previously imported private key generated in the bootstrapping phase prior to dispatching them. Finally, a consumer dispatches an interest and authenticates the results served by the producer using NDNSSEC. Two scenarios are identified for this phase: (i) authenticating valid data, and (ii) failing to authenticate invalid data either while the packet is signed by unauthorized key or because it is published under a not allocated zone. The latter case is realized through authenticated denial of existence of DNSSEC.

We illustrate the message exchange during the publication as well as consumption and authentication phases in Figure 1.

4 FUTURE WORK

Our future work evolves around two main objectives: (i) to integrate our approach into existing NDN-specific approaches, e.g., NDNS [1], and (ii) develop a method to access public DNS(SEC) data without having to use the DNS-specific transport mechanisms.

Acknowledgments. This work was supported in parts by the German Federal Ministry of Education and Research (BMBF) within the projects I3 and Deutsches Internet-Institut (grant no. I6DD111).

REFERENCES


1Source code available under https://gitlab.com/ndnssec

Figure 1: Simplified Overview of NDNSSEC Workflow – From Bootstrapping to Authentication