Guaranteeing the integrity of DNS records using PKIX Certificates

- OARC 40 -

Sangyoon Seok, **Hyeonmin Lee**, Taekyoung “Ted” Kwon

Seoul National University

DNS Security

- **Domain Name System (DNS)** is used to map domain names to their resources (e.g., hostnames to IP addr.)
  - The DNS lookup process is followed by most Internet activities

- However, DNS does not have any security features in its initial design
  - No mechanism to verify the authenticity and integrity of DNS responses
  - Vulnerable to attacks such as DNS cache poisoning

- **DNS Security Extensions (DNSSEC)** were introduced to provide the integrity of DNS messages

DNS Security and DNSSEC [1/2]

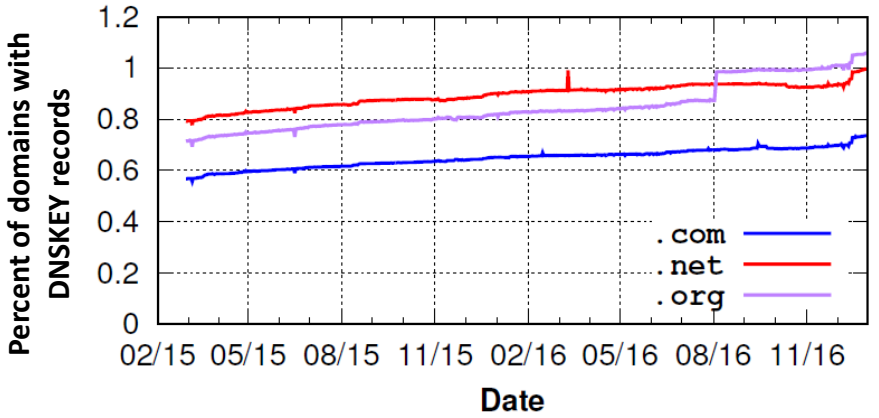
- After two decades of DNSSEC introduction..

  Top-level domains (TLDs)? 91% deployed DNSSEC [1]

  Second-level domains (SLDs)?

  - In Dec 2017 [2] .com (0.75%)
    .net , .org (~1%)

  - In Dec 2022 [1] .com (3.6%)
    .net (4.2%)
    .org (4.8%)



DNSSEC deployment rate is low..

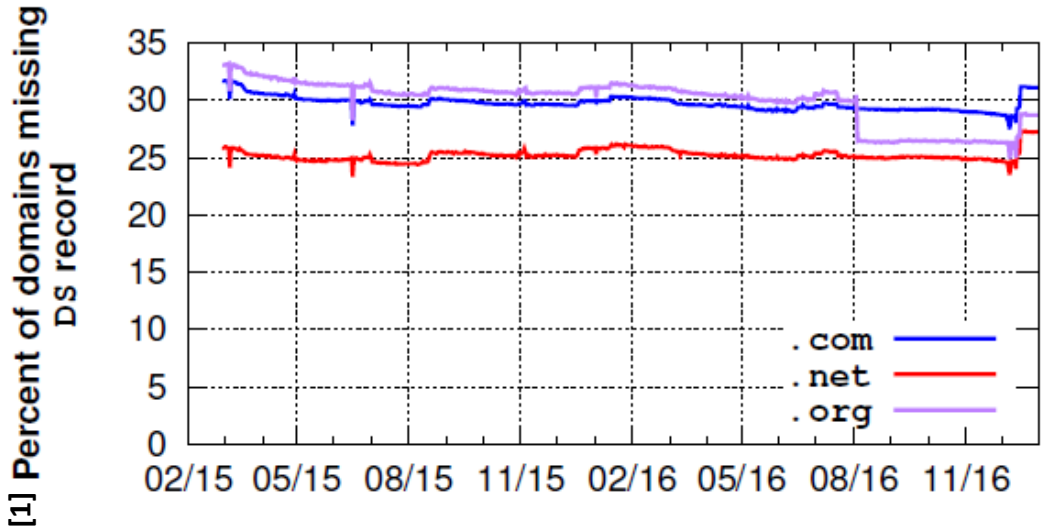
The vast majority of DNS messages in the real world are *still vulnerable*..

[1] dnssecstat (https://rick.eng.br/dnssecstat/) [2] A Longitudinal, End-to-End View of the DNSSEC Ecosystem (USENIX Security'17)

DNS Security and DNSSEC [2/2]

- Deploying/managing DNSSEC is burdensome and complex..
  - To deploy DNSSEC, a domain has to publish three DNS records (DNSKEY, RRSIG, and DS) to establish a DNSSEC chain
  - **DS records have to be uploaded to the domain's parent zone**

- Errors in the DNSSEC deployment/management



30%

Missing DS records in the parent zone

[1] A Longitudinal, End-to-End View of the DNSSEC Ecosystem (USENIX Security'17)

Objective

- Can we guarantee the integrity of DNS messages without dependencies to other zones that DNSSEC has (e.g., uploading DS records to the parent zone)?

- We need a more *practical* and *deployable* way

1. It should ***minimally require a change (or cooperation) of other entities*** **in the DNS infrastructure** such as parent zones or registrars
2. It should ***maximally reuse*** the current DNS infrastructure

Leveraging PKIX Certificates issued by CAs

- Most domains already use public keys (in certificates) – for HTTPS or TLS!
  - 94% of web traffic to Google is HTTPS [3]
  - Usually, certificates are issued by public CAs – the issuance process is well established (e.g., *Let's Encrypt*)

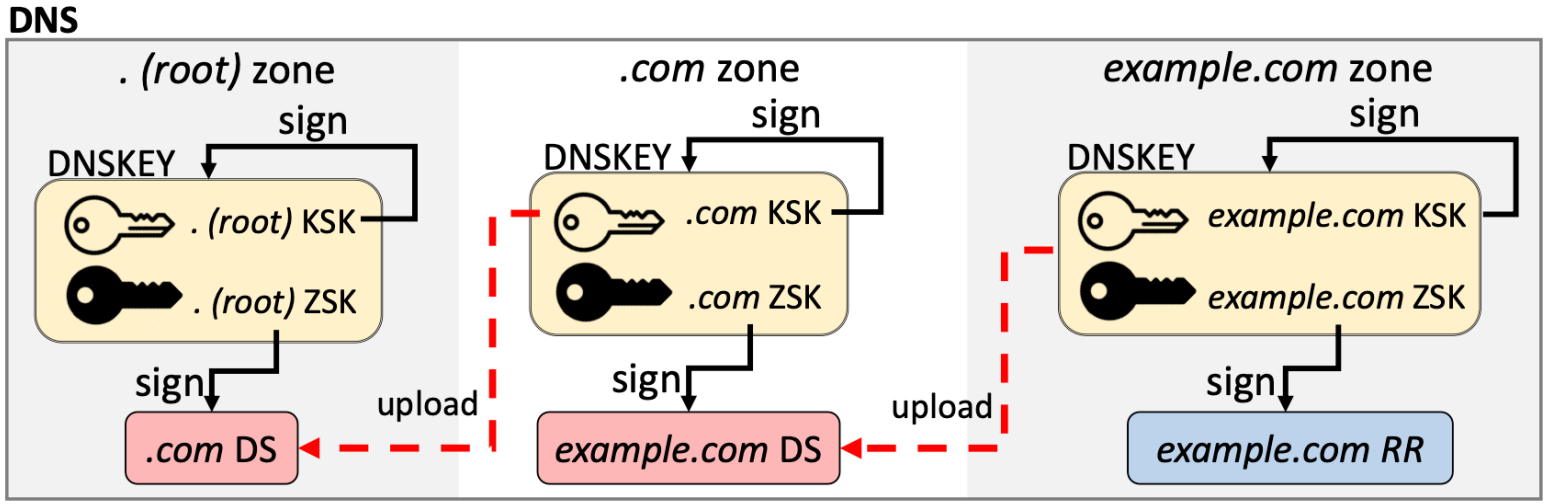
Idea

We can leverage PKIX certificates that have been successfully used by the domains

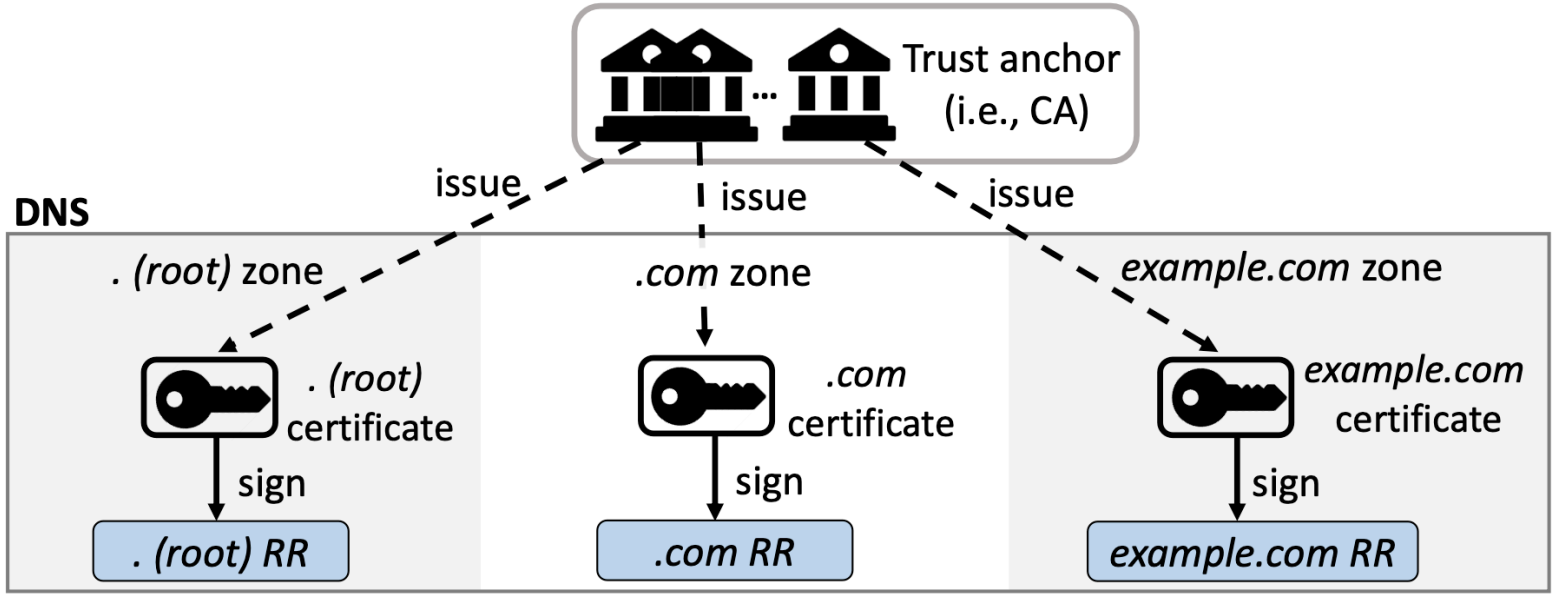
[3] https://transparencyreport.google.com/https/overview

Guaranteeing the Integrity of DNS records [1/2]

DNSSEC



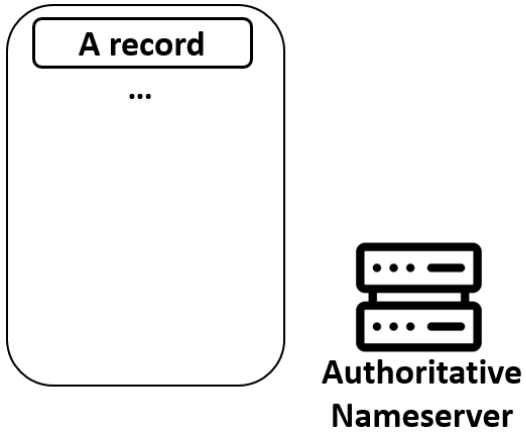
Proposal



Guaranteeing the Integrity of DNS records [2/2]

Domain

1. A domain is issued a *PKIX certificate* (or can reuse its certificate for TLS)
2. The domain generates a signature of an RRset using its private key
3. The domain uploads the signature as a DNS record (**RRSIG** record)
4. Also, the domain uploads the public key (corresponding to the private key) as a **DNSKEY** record and a certificate chain as a **CERT** record *We propose to reuse the DNSKEY, RRSIG and CERT record types



Client-side

i) A client fetches a DNS record (e.g., **A** record) and a signature (**RRSIG**) of the record

ii) The client fetches the public key (**DNSKEY**) and the certificate chain (**CERT**), and validates them through the certificate chain verification process

iii) The client verifies the signature (**RRSIG**) using the public key

Objective (Review)

- Can we guarantee the integrity of DNS messages without dependencies to other zones that DNSSEC has (e.g., uploading DS records to the parent zone)?

- We need a more *practical* and *deployable* way

1. It should ***minimally require a change (or cooperation) of other entities*** **in the DNS infrastructure** such as parent zones or registrars
2. It should ***maximally reuse*** the current DNS infrastructure

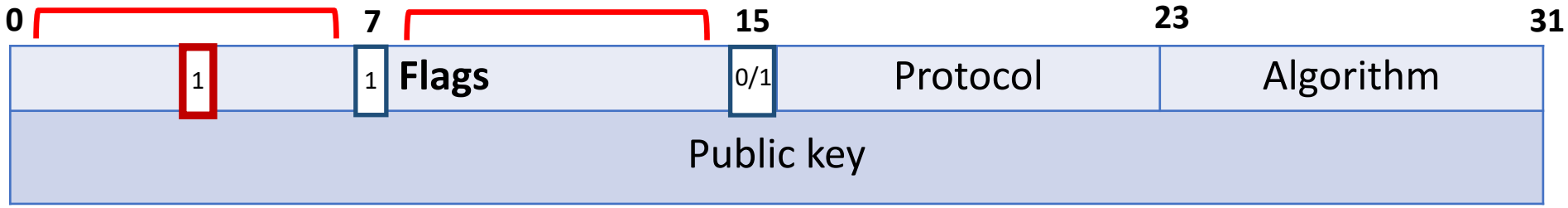
[Requirement 1] Minimum change of other entities in DNS infra.

- Our design *should* ***minimally require a change (or cooperation) of other entities in the DNS infrastructure*** in its operation
  - We leverage CA-issued PKIX certificates (and public/private keys) which are widely used by domains
  - The public key can be verified through the certificate chain verification, which does not require cooperation from *other DNS entities*
    - cf) DNSSEC requires cooperation from parent zone or registrars to establish a chain of trust according to the DNS hierarchy (e.g., uploading DS records to the parent zone)
  - Only nameservers and local resolvers need to be changed
    - Deploying a CERT record (nameservers) and verifying a certificate chain (local resolvers)

[Requirement 2] Maximum reuse of the current DNS infra.

- We suggest exploiting existing record types: **DNSKEY, RRSIG,** and **CERT** records
  1. **DNSKEY** stores a public key corresponding to the private key, which is used to generate signatures of DNS records
     - **Flags** field [4]
       - Two bits are used currently
         * bit 7 – set to 1? Holds a key for DNS zone
         * bit 15 – set to 1? KSK | set to 0? ZSK
       - Other bits (0-6, 8-14) are **reserved for future use**

         —> **We can exploit one of these bits to specify our usage**



  2. **RRSIG** stores signatures of RRsets
  3. **CERT** stores a certificate chain

[4] Resource Records for the DNS Security Extensions (RFC 4034)

Disclaimer

- We **do not** criticize or blame DNSSEC
- We try to find a practical and easier option for *domain owners* to protect their DNS message
- Our mechanism can coexist with DNSSEC
  - ex) if an upper zone does not support DNSSEC, then our mechanism can be deployed

Conclusion

- We proposed a practical way that guarantees the integrity of DNS messages
  - Most DNS messages in the real-world are not protected
  - Our mechanism minimally requires a change (or cooperation) of other entities in the DNS infrastructure
    - By leveraging PKIX certificates that are widely used by domains
  - Our mechanism is designed in a way that reuses the current DNS infrastructure
    - By exploiting existing DNS record types

Q & A

Thank you!

Hyeonmin Lee

hmlee@mmlab.snu.ac.kr