DNSSEC policies in the wild

Le, P.T.

Award date:
2017

Eindhoven University of Technology

Master Thesis

DNSSEC Policies in The Wild

Author:
Le Phuoc Tho

Supervisor:
Nicola Zannone
Luca Allodi
Roland van Rijswijk-Dijk

A thesis submitted in fulfillment of the requirements for the degree of Master of Science

in the

Security Group
Department of Mathematics and Computer Science

August 31, 2017

Acknowledgements

I would like to thank my supervisors at the Eindhoven University of Technology (TU/e), Nicola Zannone and Luca Allodi. Without their instructions and dedicated support, this thesis would not ever be accomplished. I appreciate all of your time and efforts helping me to improve the quality of my work. I also would like to take this opportunity to thank all of my lecturers at TU/e for their efforts and enthusiasm in every given lecture which passionates and inspires my research attitude.

I also would like to express my special gratitude to Roland van Rijswijk-Deij for his time and kind help. Thank you for giving me a chance to work with you on a great project. I enjoyed our weekly discussion, especially every time you push my limit. That really makes my week more "efficient". I also thank you for your arrangement allowing me to present my work at SIND and receive their feedback. In addition, I thank you for teaching me a new skill that I believe everyone living in the Netherlands should have: fixing the flat tire of a bike.

Finally, none of this could happen to me without the strong backend support from my family and my wife, Lan Nguyen. I would like to thank all of my beloveds for their complete trust on me and unconditional supports during my studying time.

Abstract

The Domain Name System (DNS) is part of the core of the Internet. However, it is vulnerable to a type of attack called cache poisoning. This attack exploits the lack of authenticity and integrity verification in DNS responses. In response, the DNS Security Extensions (DNSSEC) was developed to address those vulnerabilities via digital signatures. However, the deployment of DNSSEC is still at a dawning age. It is estimated that about 3% of all domains worldwide deploy DNSSEC. This disappointing number may indicate challenges in deploying DNSSEC. Hence, continuous DNSSEC-measurement efforts are vital to understand problems and boost its adoption.

The goal of this thesis is to perform a longitudinal study of DNSSEC policies rolling out on the Internet. We rely on recommendations of the National Institute of Standards and Technology (NIST) as the best practices to evaluate DNSSEC implementations. Furthermore, we conduct our analyses based on a unique active DNS measurement platform, OpenINTEL, which measures more than 60% of the global DNS namespace on a daily basis. In this thesis, we provide two primary contributions: (1) a thorough analysis of DNSSEC policies is presented and assessed in comparison to the best practices. This knowledge may widen the understanding of current DNSSEC employment and reveal current shortcomings. (2) We study the influence of economic incentives on the quality and the completeness of the DNSSEC deployment. This work has shown that economic incentives work. We, therefore, advocate changing the focus of economic incentives from quantity toward quality. In order to make a start with this, we have shared our results with the registries for *.nl* and *.se*, to solicit their feedback on the work and to encourage them to incorporate quality as a specific metric in future incentives. In response, the registries have acknowledged the relevance of this thesis' findings and may use our results as inputs for their discussions and planning.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Research Context

DNS is a distributed naming system for the Internet. The main purpose of DNS is to provide the mapping between IP addresses (e.g. 131.155.15.111) and human-friendly names (e.g. www.tue.nl) making resources more accessible to users. Therefore, DNS has become one of the core components of the Internet. However, since it was designed without taking into account security aspects, DNS is vulnerable to multiple attack vectors. Particularly, the cache poisoning attack [1] exploits the lack of authenticity and integrity verification to spoof DNS responses. This attack type is further extended by Kaminsky [2] allowing attackers to control a whole domain.

In response to the cache poisoning attack, DNSSEC was developed. DNSSEC adds authenticity and integrity properties to DNS responses. These additional properties allow clients to verify whether DNS responses originate from authoritative name servers and are not modified. DNSSEC employs a Public Key Infrastructure (PKI) to provide that security enhancement; hence, its operation relies deeply on proper PKI management: authoritative name servers must properly generate key pairs, sign zone data and maintain the trust chain with parent zones. Since DNSSEC deploys cryptography at the Internet scale, its deployment has been facing with many unanticipated issues for more than a decade [3]. Essentially, continuous measurement effort of DNSSEC deployment is vital to understand problems and boost its adoption.

1.2 Problem Statement

DNSSEC is relatively simple in concepts; however, its operation involves a complex stack of activities, such as maintaining the chain of trust with a parent zone,

determining key cryptographic material, performing key rollover to avoid cryptanalysis and choosing an appropriate signature lifetime to mitigate replay attacks. That operational complexity may not only hinder DNSSEC adoption but also introduce vulnerabilities in its deployments. Hence, understanding the current state of DNSSEC usage on the Internet is beneficial in several ways: firstly, it reveals deployment trends of DNSSEC such as common key algorithms and key sizes; secondly, this can give an overall flavor on the security level of DNS. Finally, studying DNSSEC usage may uncover operational problems in real conditions such as broken chain of trust, malformed zone signing or signature expiration. Furthermore, a better understanding of existing problems will allow researchers to propose more comprehensive solutions. Unfortunately, there are few works that reflect the current deployment of DNSSEC. Motivated by that factor, this thesis aims to study and explore DNSSEC policies as well as its operations that are currently rolling out in the wild.

1.3 Contributions of This Thesis

Considering the small amount of existing research about DNSSEC deployment on the Internet, this thesis aims to offer three main contributions:

- The first contribution is to provide an analysis of the state-of-the-art for DNSSEC deployment in the wild through various important aspects as discussed in Chapter 4. Furthermore, we establish best practices and evaluate DNSSEC deployment. That knowledge may widen the understanding of current DNSSEC employment and present shortcomings.

- The second contribution is to investigate the influence of economic incentives on the quality and the completeness of DNSSEC deployment. Registries may use the results of this work to steer future internal discussions on the identification of more effective incentives to encourage for higher quality of DNSSEC deployment.

- The third contribution is to provide reliable methodologies to analyze collected DNS data and establish solid approaches in order to study various DNSSEC policies and operational practices applied globally. This may ease future work on updating the status of DNSSEC deployment.

1.4 Organization of The Thesis

This thesis is organized as follows:

Chapter 2 Provides background information on DNS and DNSSEC.

Chapter 3 Reviews related works.

Chapter 4 Identifies important DNSSEC policies and best practices in deploying DNSSEC

Chapter 5 Discusses research questions of the thesis

Chapter 6 Presents the research methodology, dataset and challenges when analyzing the data.

Chapter 7 presents and evaluates results.

Chapter 8 discusses our findings.

Chapter 9 concludes the research and summarizes answers to research questions. Furthermore, future works are also discussed in this chapter.

Chapter 2

Background

This chapter provides background information on the DNS and DNSSEC. Firstly the chapter covers how the DNS works and its main components. Next, this knowledge is used to illustrate how cache poisoning becomes possible, which has led to a security extension of the DNS, namely DNSSEC. Finally, this chapter presents an overview of DNSSEC.

2.1 DNS

The DNS bridges the gap between the human preference for names, and the machine need for numbered addresses. Each computing device is assigned a unique IP address (e.g. *131.155.2.3*) so that devices can be reached and communicate with others. However, such a numerical address scheme is difficult for a human to memorize. Hence, there is a need to add logically structured names associated with IP addresses. DNS addresses this issue by maintaining mappings between human-friendly names and their corresponding IP addresses. For example, the web server machine of the TU/e website has the DNS name of *www.tue.nl* which can be translated to a public IP address of *131.155.15.111*.

DNS is a tree-like distributed database of names which contains three main elements, namely the domain name space, resource records and name servers. These elements are explained in the coming sections based on the specifications of DNS described in RFC 1034 [4] and RFC 1035 [5] defined in 1987.

2.1.1 Domain Name Space

The domain name space is a tree structure. A portion of the tree is illustrated as in Figure 2.1. This structure contains nodes and leaves represented by labels. The top

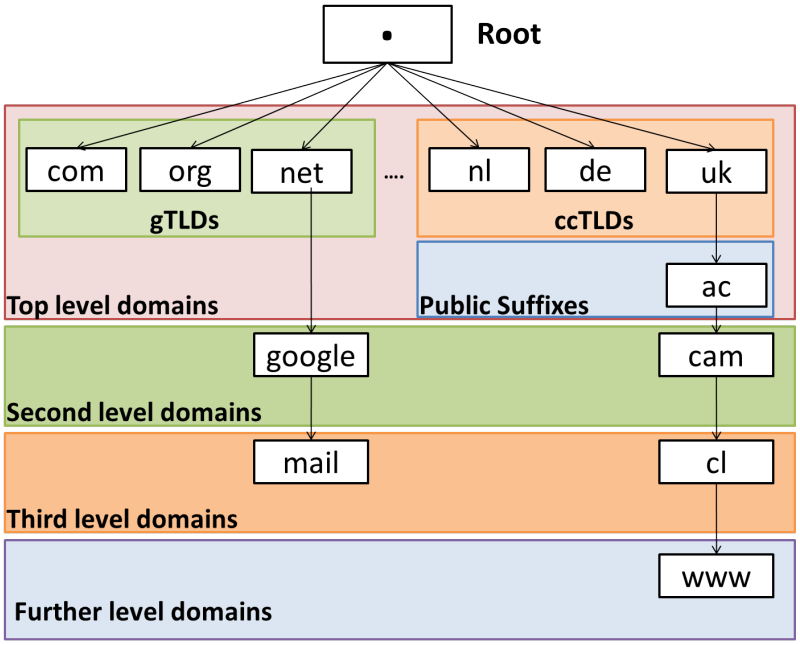


Figure 2.1: Hierarchical DNS domain name space

node is called "the root" and is usually labeled as a single dot ".". Nodes which are right below the root are called Top Level Domains (TLDs). Similarly, nodes are called second-level domains, third-level domains, and so on, based on their distance from the root. TLDs are usually classified as country-code TLDs (ccTLDs) and generic TLDs (gTLDs) in which the former one is used to indicate a specific geographic country while the latter one is for a general purpose usage. Furthermore, TLDs may be further divided into public suffixes, such as *ac.uk* for educational institutes and *co.uk* for commercial domains. A Fully Qualified Domain Name (FQDN) or domain name is a sequence of labels of a path from a node to the root separated by a dot. However, the root label is usually omitted; hence, *www.tue.nl.* and *www.tue.nl* are both considered legitimate.

With respect to the administration of the DNS, the root of the DNS is managed by the Internet Corporation for Assigned Names and Numbers (ICANN). ICANN delegates the maintenance responsibility of TLDs to organizations called *registries*. Most *registries* allow third-parties, known as *registrars*, to sell these domain names to the public and buyers or owners of domain names are called *registrants*. Hence, the *Registry-Registrar-Registrant* channel forms the core procedure for the registration and administration of domain names. Another important concept in the DNS is a DNS operator who is actually hosting and managing the content of domain names.

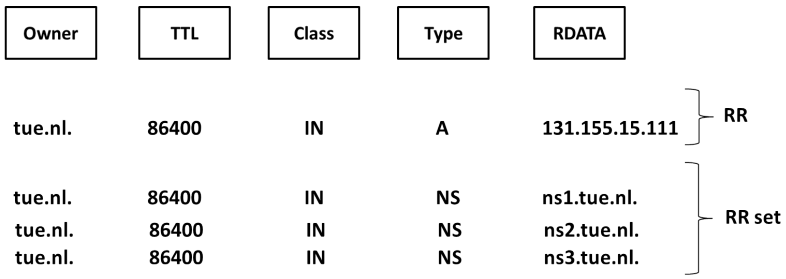


Figure 2.2: Resource record fields

Although a registrar can also be a DNS operator, these two roles are distinct and should never be confused.

2.1.2 Resource Records

Each node in the domain namespace contains a set of Resource Records (RRs) that contain data about that domain name. Each resource record consists of fields as shown in Figure 2.2.

- Owner: is the domain name that contains RR.
- Time-To-Live (TTL): is a 32-bit value indicating how long an RR is valid in a cache.
- Class: is a 16-bit value identifying a protocol. Nowadays, class IN meaning Internet is mostly used.
- Type: is a 16-bit value indicating a specific type of a resource. Table 2.1 presents the main RR types currently used.
- RDATA: is an RR specific data as shown in Table 2.1.
- Resource Record set (RRset): is a set of RRs with the same owner, class and type.

2.1.3 Domain Name Servers and DNS Resolution

Since DNS employs a client-server architecture style, its operation can be easily represented with respect to the client and server viewpoint as discussed below:

- Domain name servers are the core components of the DNS for storing and querying information. In general, there are two server roles, namely authoritative and recursive name servers.

Type	RDATA	Description
A	A 32-bit IP address (e.g. 131.155.2.3)	Maps a domain name to an IPv4 address
AAAA	A 128-bit IP address (e.g. 2001:610:1108:2::3)	Maps a domain name to an IPv6 address
CNAME	A domain name (e.g. onderwijs.tue.nl.)	Is an alias of a domain name
MX	A 16-bit preference value and mail server's domain name (e.g. 100 mx-1.mf.surf.net.)	Specifies Mail eXchange records for a name
NS	A host name (e.g. ns1.tue.nl)	Specifies authoritative name servers.
PTR	A domain name (e.g. onderwijs.tue.nl.)	Is a pointer to another part of the domain name space, mostly used for reverse DNS mapping between IP addresses and domain names
SOA	Contain metadata fields (e.g. ns1.tue.nl. hostmaster.tue.nl. 2017031202 3600 3600 1209600 3600)	Specifies metadata about a zone,including: name server, email, serial number, refresh, retry, minimum TTL
TXT	Arbitrary text data	Is a text record for general purposes.

Table 2.1: DNS resource record type and corresponding RDATA

  - The authoritative name servers contain authentic resource records of a domain name organized in data units called zones; typically an authoritative name server can host multiple zones. When receiving a query, authoritative name servers will search their zones for answers and respond with the flag Authoritative Answer (AA) set. Otherwise, they will either reply with referrals pointing to another name servers closer to the queried domain or refuse the query.
  - On the other hand, recursive name servers, also called DNS resolvers, perform DNS resolution which is the process of finding the mapping between a queried name and its corresponding value. Specifically, clients can query recursive resolvers which will then recursively travel through the DNS hierarchy to find answers, called iterative resolution, and respond to clients. In addition, recursive resolvers may also cache those answers for performance optimization; hence, they are also called recursive caching name servers.

- Clients' operating systems usually provide primitive DNS resolvers called a stub resolver which cannot query for names recursively, but instead, it takes the role of a forwarder. That means stub resolvers simply send queries to pre-configured recursive resolvers and wait for responses.

The DNS resolution process is further explained in Figure 2.3; the process comprises

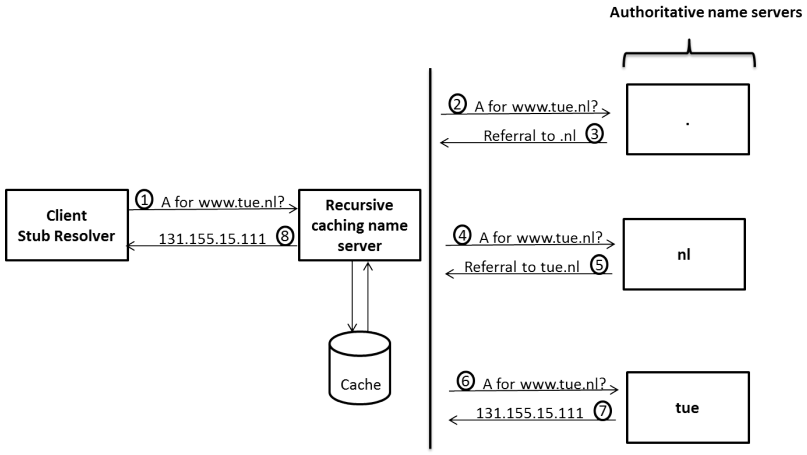


Figure 2.3: An example of DNS resolution for www.tue.nl

of eight steps for the example query of the A record for *www.tue.nl* originating from a client machine:

- **Step 1**: A client sends a query to resolve record type A for *www.tue.nl*. The stub resolver cannot find the answer in its cache and forwards the query to the recursive caching name server.
- **Step 2 and 3**: The recursive caching name server receives a query and checks in its cache for a non-expired answer. If no data is cached, the server will recursively travel through the DNS hierarchy to find an answer. Initially, it contacts a known root server for answers. Since the root server does not contain the *tue.nl* zone, it cannot reply to the query. However, the root server has a delegation for *.nl*; hence, it replies with a referral to the authoritative name servers for the *.nl* domain such as *ns1.dns.nl*. At this point, the recursive caching name server would have to start a new resolution process to find out A or AAAA records of replied referrals. Hence, to enhance recursive queries, some authoritative name servers may provide non-authoritative A/AAAA records for the authoritative name servers suggested in referrals; this is called glue data.
- **Step 4 and 5**: The recursive resolver sends the query to one of the *.nl* authoritative name servers. Similar to step 3, the authoritative name server does not have information for *tue.nl*. Hence, it replies with a referral to the *tue.nl* authoritative name servers.
- **Step 6 and 7**: The recursive resolver sends the query to one of *tue.nl* authoritative name servers. Since these are authoritative for the domain, they

will respond with an A record containing the IPv4 address associated with *www.tue.nl.*

- **Step 8**: The recursive resolver receives the answer and responds to the client's request. At the same time, it will also save the response in its cache as long as the response's TTL allows.

2.2 DNS Cache Poisoning

The lack of security mechanisms for authentication and integrity checking of DNS responses makes DNS vulnerable to a particularly dangerous attack, namely cache poisoning [1]. This attack aims to inject a malicious response into the cache of a recursive caching name server. While the attack targets DNS resolvers, its main victims are client users who ask their DNS resolvers to resolve a name and receive an injected falsified IP address. That means all traffic of services offered via that name will be transparently redirected to hackers' servers. In addition, that injected poisoning is difficult to detect and can last until the TTL expires on the cache. Otherwise, an administrator has to erase it explicitly.

Prior to explaining the cache poisoning attack, it is important to understand what is saved in the cache of a recursive caching name server. In principle, a recursive caching name server may remember all information included in the answer, such as authority and additional sections of a legitimate response to a query. A response is basically considered legitimate if it meets the following conditions:

- It arrives on the same User Datagram Protocol (UDP) port as the query was sent out.
- The question section matches with the question in the pending query.
- The 16-bit query ID in a response must match with a pending query.
- Information in the additional section must be the same as the domain in the query section which is so-called bailiwick checking.

Using the knowledge of DNS operation as presented in previous sections, we discuss how cache poisoning is possible and its most dangerous variant, namely the Kaminsky attack, as well as what countermeasures are available. Referring to Figure 2.3, the cache poisoning attack is possible whenever the recursive caching name server receives a response from an authoritative name server, namely step 3, 5 and 7.

However, for the sake of clarity, step 7 is chosen as the injecting point at which an attacker attempts to poison the recursive caching name server with a fraudulent IP address for *www.tue.nl*. In principle, the attack scenario could happen as follows:

- **Step A1**: An attacker sends a query asking for the A record for *www.tue.nl* to the recursive caching name server.
- **Step A2**: If the resolver has a valid answer in its cache, the attack fails. Otherwise, the resolution will follow steps 2, 3, 4, 5 and 6 in Figure 2.3.
- **Step A3**: At this point (step 7 in Figure 2.3), the race begins. The attacker has to send a multitude of responses with his falsified information. If one of his falsified response arrives before the response from the authoritative name server of *tue.nl* and is accepted as a legitimate response (as described above), he succeeds. Otherwise, he has to wait until the resolver flushes its cache for that record.
- **Step A4**: The resolver caches the falsified response and drops the genuine one since it cannot distinguish the falsified response from a legitimate one. Then it serves this malicious information to all of its clients.

It is important to note that in order to be successful, the attacker has to guess the source UDP port and query ID. However, before Kaminsky, the source UDP port was usually fixed or predictable and the query ID incremented for each query sent out; therefore, it was quite trivial for attackers to guess these parameters. Although the described poisoning attack appears to be harmful, it can only change one record at a time and depends heavily on TTL. That means that if an attacker fails in the first arms race, which is likely the case, he has to wait until the TTL of that record expires in the cache. However, in 2008, Kaminsky introduced a new variant that can compromise the whole domain rather than a record and does not depend on TTL [2]. The Kaminsky attack is similar to the attack described above; however, there are two key differences: query names and counterfeit information injected. In particular, the Kaminsky attack scenario can be represented by the following steps:

- **Step K1**: An attacker sends a query asking for the A record for a random name in *tue.nl* to the recursive caching name server.
- **Step K2**: Since the query name is random, the resolver likely does not have an answer in its cache. Therefore, the resolver will follow steps 2, 3, 4, 5 and 6 in Figure 2.3.

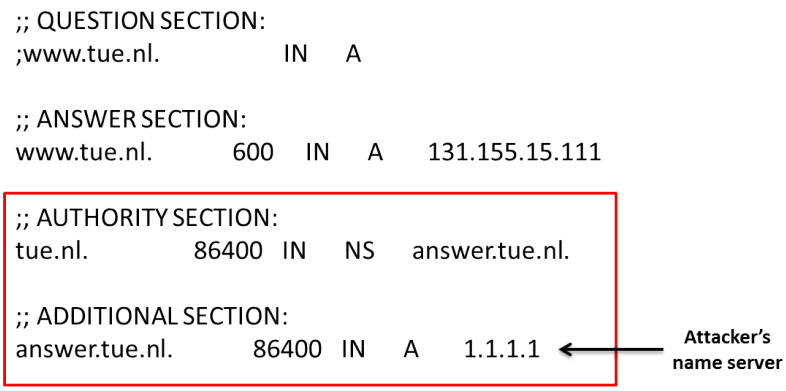


Figure 2.4: An example of Kaminsky attack

- **Step K3**: The attacker starts the race and sends out a number of responses with his falsified information in step 7 of Figure 2.3. However, instead of falsifying the answer of a response, it poisons referrals as illustrated in Figure 2.4. This is completely a valid response since the counterfeit referrals are just another delegation like what authoritative name servers of . and *.nl* have done. It simply says that "I do not know, you can ask *answer.tue.nl*". In addition, the attacker attaches a falsified "glue" IP address to that name server. If one of his falsified response arrives before the response from the authoritative name server of *tue.nl* and is accepted as a legitimate response (as described above), he succeeds. Otherwise, he can start over again with a new random name which releases him from the TTL bound time as mentioned in the cache poisoning attack described above.

- **Step K4**: Once successfully, all following requests for the *tue.nl* domain from that recursive caching name server will eventually be routed to a name server of the attacker; hence, the whole domain is compromised.

An obvious countermeasure against the cache poisoning attack is to randomize the query ID for each query sent; hence, an attacker has to guess in a pool of 2^{16} values. However, as this pool is quite small, it is likely feasible. An alternative solution is to randomize both the source port and query ID. The source port is 16 bits and normally the use of 0 to 1024 is restricted; hence, this initiative will increase the pool of guess substantially from 2^{16} to $2^{16} \times (2^{16} - 2^{10})$ values. However, it is shown that this improvement just delays an attacker from seconds to days [6]; therefore, to a committed attacker, this improvement would not be a big obstacle. Furthermore,

the "0x20" approach [7] is introduced to add few more entropy bits which may make hackers' life harder. However, it is commonly agreed that these are just temporary countermeasures since the root cause of this vulnerability is that the recursive caching name servers cannot verify that responses originate from the expected name server and are unaltered. This leads to the need for DNSSEC as a complete solution for this class of attack since it provides DNS resolvers means to verify the authenticity and integrity of each response via a digital signature.

2.3 DNSSEC

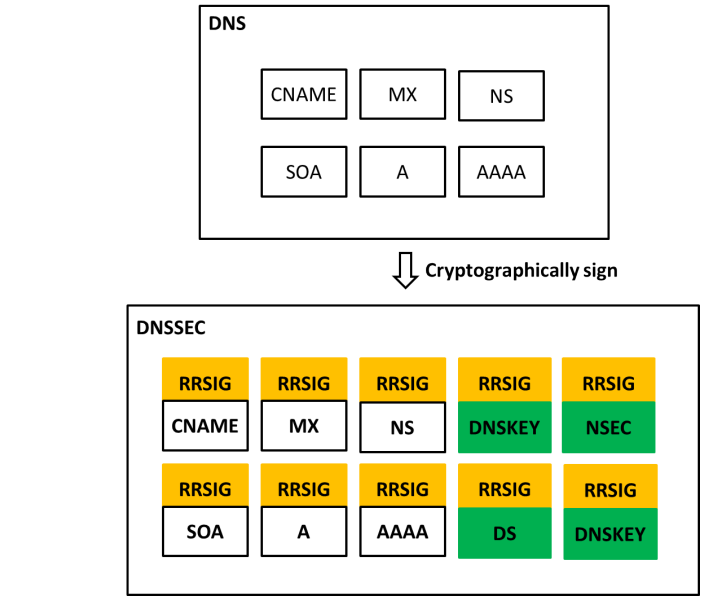
DNSSEC is a DNS security development running on top of the ordinary DNS infrastructure that provides authentication and integrity properties to DNS responses. This helps DNS to resist against attacks that depend on sending falsified data such as cache poisoning. However, it is worth noting that DNSSEC does not provide confidentiality since all DNS messages are still sent in clear text. In this section, a more detailed insight into how DNSSEC works is presented based on three RFCs: 4033 [8], 4034 [9] and 4035 [10].

2.3.1 DNSSEC Signing

The core principle of DNSSEC is that all authoritative records of a zone are digitally signed, which allows DNS resolvers to verify the integrity and authenticity of authoritative responses. Initially, a zone operator needs to generate one or more cryptographic key pairs, in which the private keys are kept secret and are used to sign the zone, while the public keys are published in the zone for validation purposes. This simple principle leads to additional new RRs to fulfill the operation of signing and validating in DNSSEC as illustrated in Figure 2.5.

As seen from the figure, DNSSEC introduces four new resource record types to DNS, namely Resource Record Signature (RRSIG), DNS Public Key (DNSKEY), Delegation Signer (DS) and Next Secure (NSEC) [8]. Firstly, each RRset is associated with at least one digital signature, called RRSIG. An example of this resource record type is shown in Figure 2.6 and contains the following fields:

- **Type covered**: specifies the type of RR covered by this signature. In this example, it covers type NS.
- **Algorithm**: specifies the signature algorithm used. Common algorithms are presented in Table 2.2 [11].



- The green box presents a new resource record type.
- The orange box presents a digital signature for each RRset.

Figure 2.5: DNSSEC additional RRs

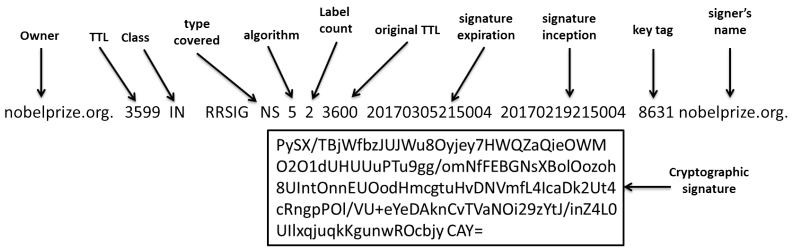


Figure 2.6: An example of RRSIG for the nobelprize.org domain

- **Label count**: specifies the number of DNS labels in the original owner name of this signature. Validators rely on this field to distinguish between "real" names and DNS wildcards (e.g. *.example.com).
- **Original TTL**: specifies the TTL of the original RRset.
- **Signature expiration**: indicates when the signature expires. It is presented in the format of YYYYMMDDHHMMSS; hence, in this example, the signature is valid until March 05, 2017 at 21:50:04 UTC.
- **Signature inception**: indicates from when the signature is valid.
- **Key tag**: specifies the 16-bit key tag of a public key that can verify the signature. However, it should be noted that this tag is not unique; hence, the key tag

Algorithm number	Algorithm	Hash	Remarks
1	RSA	MD5	With NSEC
3	DSA	SHA1	With NSEC
5	RSA	SHA-1	With NSEC
6	DSA	SHA1	With NSEC3
7	RSA	SHA1	With NSEC3
8	RSA	SHA256	With NSEC or NSEC3
10	RSA	SHA512	With NSEC or NSEC3
12	GOST R 34.10-2001	GOST R 34.11-94	With NSEC or NSEC3
13	ECDSA Curve P-256	SHA-256	With NSEC or NSEC3
14	ECDSA Curve P-384	SHA-384	With NSEC or NSEC3

Table 2.2: Common DNSSEC algorithm numbers [11]

should be used as a mean to select candidate keys for the validation process. Furthermore, this value is not available in a DNSKEY RR and hence, requires a computation. This calculation, which is the same for all DNSKEY algorithms except Algorithm 1, leverages the wire format of the RDATA of a DNSKEY RR as specified in RFC 4034 [9].

- **Signer's name**: specifies the name of zone owner.
- **Cryptographic signature**: specifies the cryptographic signature data.

Secondly, the DNSKEY record type contains public keys for the verification process. An example DNSKEY record type is shown in Figure 2.7. Notably, a zone may have one or more public keys, typically two keys, namely a Key Signing Key (KSK) and a Zone Signing Key (ZSK). The ZSK, as indicated by its name, is used to sign zone data, while the KSK is used to sign the DNSKEY set only. Key management policies are further covered in Chapter 4. A DNSKEY record contains the following fields:

- **Key flags**: is the 16-bit field specifying a role of the key. On the wire format, if bit 7 is set, it indicates the key as a zone signing key and can be used for DNSSEC validation. If bit 15 is set, it indicates that the key takes the role of Secure Entry Point (SEP) of the zone. Similar to the key tag field, this key flag field only serves as a hint; therefore, validators must not rely on this flag during the verification process.
- **Protocol**: should always be 3 to indicate the DNSSEC protocol.
- **Algorithm**: specifies the signature algorithm used and should be same as the algorithm specified in RRSIGs.

Figure 2.7: An example of DNSKEY for the nobelprize.org domain

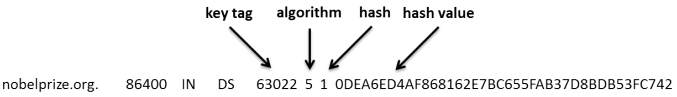


Figure 2.8: An example of DS for the nobelprize.org domain

- **Public key**: is the actual cryptographic public key data.

Thirdly, the DS record is an important concept in DNSSEC. A DS record is hosted in a parent zone and points to a SEP key for trust establishment. Figure 2.8 illustrates an example of a DS record for the *nobelprize.org* zone hosted in the *.org* zone with the following fields:

- **Key tag**: is the key tag of a DNSKEY that the DS record refers to in the child zone.
- **Algorithm**: is the signature algorithm of the DNSKEY that the DS refers to.
- **Hash algorithm**: indicates a hash algorithm that is used to create the hash value of the DNSKEY that the DS refers to.
- **Hash value**: is the actual hash data of the DNSKEY that the DS refers to.

Lastly, the NSEC record type is used to provide authenticated answers to queries asking for nonexistent records. With three new record types explained above: DNSKEY, RRSIG and DS, the authenticity of existing records in a zone is guaranteed since validators can always verify a response against its corresponding signature. However, an authoritative name server can also reply Non-Existent Domain (NXDOMAIN) to queries. Consequently, NXDOMAIN responses are required to be protected and be verifiable by validators. DNSSEC addresses this problem via a procedure, called authenticated denial of existence. That means that DNSSEC chains all records of a zone in the canonical order defined in RFC 4034 [9], then NSEC records are created for each existing record pointing to the next record in the zone like a linked list. Effectively, an NSEC record provides a proof that there are no other records existing in the interval of two consecutive DNS names.
The usage of NSEC is illustrated via a small example provided in Figure 2.9, in which

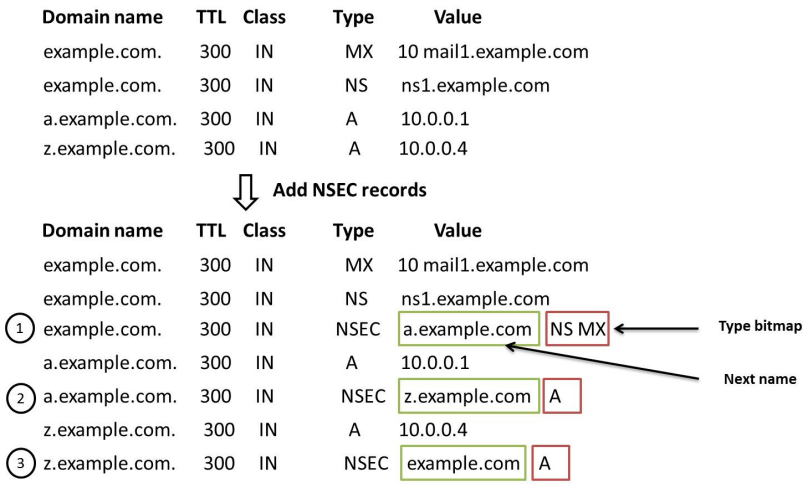


Figure 2.9: An example of NSEC record

the table on the top shows DNS records in the canonical order, while the lower table presents the added NSEC records. As seen in the figure, an NSEC record is added in response to each of three RRsets. Effectively, the first NSEC record indicates that there are MX and NS records associated with the *example.com* name and the next name is *a.example.com*. Consequently, a name server can easily prove that there is no other name possible between this name interval. Similarly, the second NSEC record shows there is only a record type A for *a.example.com* and there is no other record between *a.example.com* and *z.example.com*. In the same manner, the third NSEC record wraps around the chain by saying that there is no other record possible between the last name *z.example.com* and the first name *example.com* except an A record of *z.example.com*. It is important to mention that while it is omitted in Figure 2.9, all NSECs are signed as other zone data; otherwise, validators cannot verify their authenticity. This mechanism suffers a vulnerability, called "zone walking" that allows everyone to enumerate all records in a zone by following the linked list of existing names. Because of this zone-walking problem, NSEC3 [12] has been introduced aiming to mitigate the exposure of zone information. In principle, NSEC3 essentially replaces plain text names by their hash values. However, this approach is known to be vulnerable to dictionary attacks as mentioned in the same RFC [12]; therefore, applying additional security measure is recommended such as NSEC3 iterations and salt [13]. Furthermore, GPU-based attacks on NSEC3 hashes have recently been shown to be effective as presented in [14]. In response to this growing concern, an alternative authenticated denial of existence mechanism, called NSEC5 [15], has been proposed to address this zone enumeration cryptographically. However, this proposal is still in

the draft version; it is not further discussed in this thesis.

2.3.2 Chain of Trust

The ability to establish the trust relationship flowing from parent to child zones is one of the most vital parts of DNSSEC. Although a zone is signed and all records within that zone can be verified via the published public keys, validators need to be certain that those public keys are trustworthy. Therefore, DNSSEC has devised a mechanism called the *chain of trust* which essentially maintains the trust relationship between zones in the DNS hierarchy. In particular, a parent zone may host one or more DS records and each DS record refers to a public key in a child zone. If the parent zone is trusted, validators can reliably believe that those public keys in the child zone are trustworthy; thus, they are called SEP keys serving as the ultimate trust to all signed records in the zone.

Figure 2.10 presents the chain of trust to verify signed records in the *example.com* domain. In this example, a common key usage practice, using two different key roles named KSK and ZSK, is employed. The KSK serves as the SEP of the zone and only signs the DNSKEY set, while the ZSK is the actual key which signs zone data. The verification of DNSSEC requires a starting point called a trust anchor; validators are usually pre-configured with a trust anchor which is usually the root zone's KSK. Effectively, the trust is expanded all the way down to the targeted zone, namely *example.com* in this case, via DS records and SEP keys.

2.3.3 DNSSEC Validation

Finally, validating resolvers can verify the authenticity of records in DNSSEC-enabled zones. This validation process traverses over the DNS hierarchy from a trust anchor to a targeted record and performs two main steps as explained below at each zone. For illustration purpose, Figure 2.11 presents an example of the validation process for records in the *example.com* zone.

- **Retrieve and verify DNSKEYs**: are illustrated by step 1 and 2 in the root zone, step 5 and 6 in the *.com*, and step 9 and 10 in the *example.com* zone. Validating resolvers firstly obtain DNSKEYs in a zone and look for a match between DS records received in the parent zone and DNSKEYs in the child zone. As discussed in Section 2.3.2, such a key, if found, is the SEP which is used to verify the authenticity of public keys published in the zone. However, there is an exception: the root zone does not have a parent zone and thus does

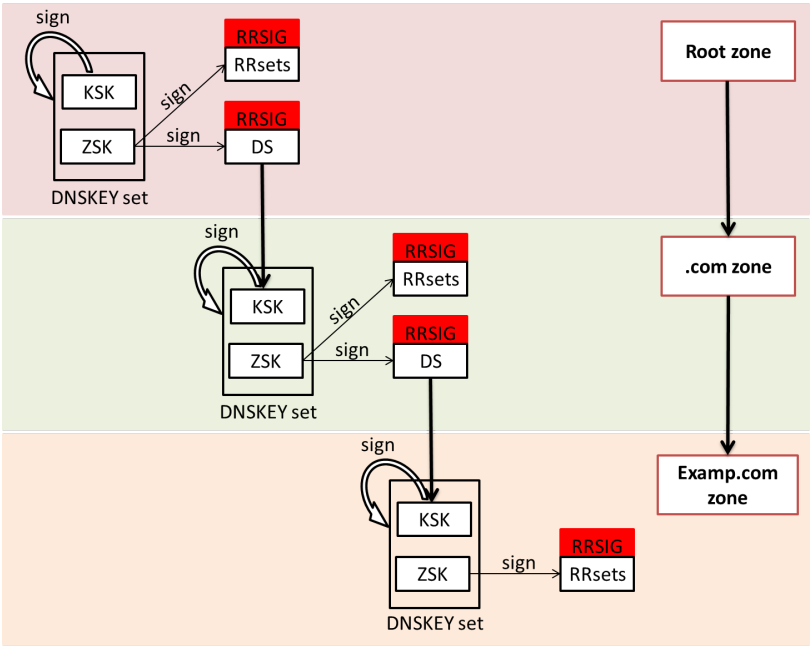


Figure 2.10: DNSSEC chain of trust

not have associated DS records. This is where the role of trust anchors comes into play since validators are pre-configured with trust anchors which are used to check the correctness of DNSKEYs in the root zone.

- **Verify RRSIGs in response to the actual query**: is represented by the last two steps at each zone, namely step 3 and 4 in the root zone, step 7 and 8 in the *.com*, and step 11 and 12 in the *example.com* zone. Validating resolvers can use keys from the DNSKEY set to validate RRSIGs. Consequently, validating resolvers can determine the resolution result of *www.example.com* as either secure, insecure, bogus or indeterminate as explained in RFC 4033 [8] and 4035 [10]:
  - **Secure**: when a resolver can follow the chain of trust and successfully verify an RRset with its corresponding RRSIG.
  - **Insecure**: when a resolver cannot follow the chain of trust from a trust anchor to an RRset due to the non-existence of a DS record.
  - **Bogus**: when a resolver should be able to follow the chain of trust from a trust anchor to an RRset, however, the validation fails for some reasons such as missing signatures or expired signatures. This may happen due to configuration errors, data corruptions, or actual attacks.

– **Indeterminate**: A resolver can not determine a particular part of the DNS hierarchy is secure due to the lack of a trust anchor.

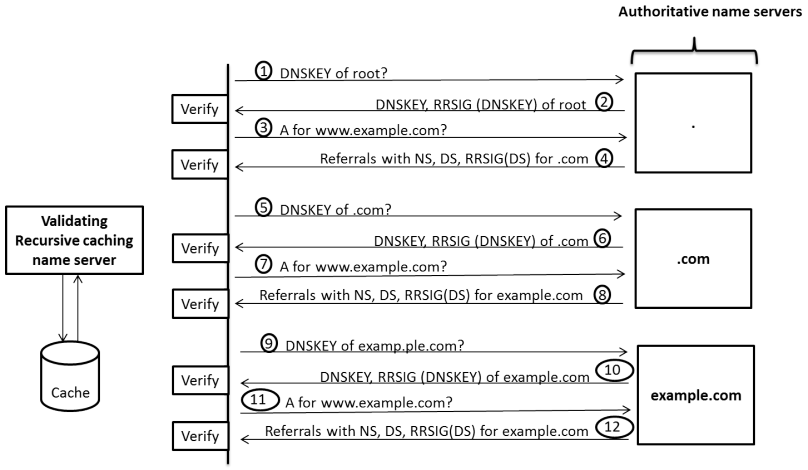


Figure 2.11: DNSSEC validation process

2.3.4 DNSSEC Economic Incentives

Although DNSSEC is expected to enhance DNS security, its adoption is still in the early stage due to the lack of DNS operators' interest. In order to encourage the DNSSEC deployment, several TLD registries offer economic incentives for each signed domain [16, 17]. In particular, the ccTLDs *.nl* and *.se* offer a small discount to registrars for every signed zone, 0.07 Euro per quarter [16] and 3 SEK per six months [18] respectively. This initiative has led to some success, for example, the ccTLD *.nl* became the largest DNSSEC zone in a short time space [19].

Chapter 3

Literature Review

There are several notable studies into DNSSEC measurement that provide insights in DNSSEC adoption and deployment. In this section, we discuss those studies and highlight the differences with this work. Aside from the server side of DNSSEC deployment measurement, there are some works that investigate the client side of DNSSEC validation adoption. Although these works are not directly related to the thesis' focus (the server side), for the sake of completeness, they are also discussed.

3.1 DNSSEC Server-Side Measurement

Shortly after the latest DNSSEC protocol revision [8, 9, 10], the SecSpider project [20] is the pioneered monitoring platform of DNSSEC deployment. This project has been tracking secure zones and collecting DNSSEC data since 2005. The work provides invaluable findings in the early age of DNSSEC deployment: firstly, the difference in the interest of DNSSEC employment between parent and child zones has resulted in islands of security; that is, a parent zone is not signed causing its child zones to be unverifiable. Secondly, while key management design is simple, its practical operation is complex causing problems in the key management. Lastly, this project has discovered that a significant number of zones have vulnerabilities in signing schedule allowing replay attacks possible. Furthermore, the SecSpider project has published their work concerning three critical metrics to measure DNSSEC deployment: availability, verifiability and validity [21]. Being different from the SecSpider project, this thesis focuses on studying various aspects of DNSSEC policies and operation in greater details based on a recent dataset [22]. In addition, this dataset is actively monitoring hundreds of thousands of secure domains, whereas that number is thousands of domains in the SecSpider project; hence, the thesis is also different with respect to the magnitude of analyzed data.

Yang et al. [3] performed a thorough discussion on challenges that DNSSEC deployments have faced and suggested some lessons learned for future designs of cryptography in a large-scale deployment. The authors explained the critical gap between cryptographic designs and Internet-scale deployments which require some properties, namely scalability, flexibility and incremental deployability. Furthermore, the study in [3] discusses various impacts of DNS caches and private-key management (i.e. storing online or offline) on the DNSSEC operation. Based on those analyses, the authors provide root causes of these challenges and emphasize the importance of DNSSEC measurement. With respect to this measurement motivation, the thesis aims to study further the current DNSSEC deployment on the Internet.

Van Rijswijk-Deij et al. [22] built a scalable infrastructure for active DNS measurements. This project initially collected DNS data from all second level domains under three TLDs: *.com*, *.net*, *.org*, which together comprise of more than 50% of the DNS name space. Since then, a number of other TLDs have been added such as *.nl*, *.se*, *.ca* and *.at*. Based on this dataset, the researchers have studied the adoption of ECDSA in DNSSEC [23]. While the result shows that only a minor number of *.com* domains employ Elliptic Curve Digital Signature Algorithm (ECDSA), it is actually growing over the 1.5 years of monitoring. In this thesis, we rely on this dataset and aim to explore further insights in DNSSEC.

Since the interest in DNSSEC has been growing recently, there are more efforts in measuring and quantifying DNSSEC deployment. The Internet Society publishes an updated state of DNSSEC deployment [24] in which they highlight that 89% of TLDs and 47% of ccTLDs are signed. Notably, over 2.5 million *.nl* registered domains are signed making it the largest signed zone in the world. Furthermore, the Internet Society reports that many DNS name server software and libraries support DNSSEC and have several years of experience [24]. In addition, Adrichem et al. [25] examined the misconfiguration of DNSSEC deployment, in which the authors use a publicly available dataset of domain names and perform DNSSEC validation in two manners: the top-down and bottom-up approaches. The result of this study shows that over 4% of evaluated domains contains misconfigurations. Similarly, Deccio et al. [26] analyzed a six-month dataset measured in 2010 to examine DNSSEC deployment issues and found out that nearly 20% of zones experienced invalid signatures. Notably, Wander [27] presented the server-side DNSSEC adoption, in which he analyzed 22-month data from April 2013 to February 2015 and provided insights into key management, NSEC/NSEC3 and signing validation in TLDs as well as in 3.4 million second-level domains. Recently, Chung et al. [28] performed a longitudinal measurement study

of the DNSSEC Ecosystem and found some critical problems. Particularly, 31% of domains fail to publish necessary records for DNSSEC validation; most ZSKs and about one-third of KSKs are weak; several DNS operators use the same DNSKEYs for most of the domains for which they are authoritative; key rollover is a critical issue when many domains do not change keys regularly. Regarding DNSSEC validation, this study found that 82% of resolvers request DNSSEC records; however, only 12% of them perform the signature verification process. The work presented in this thesis is different from previous studies in three ways: firstly, we analyze the more recent continuous dataset covering more than 50% of the DNS name space since 2015 [22]. Secondly, we consider a complete set of aspects relating to the operation of DNSSEC: key size, key algorithm, key rollover, key (algorithm) rollover approaches and signature validity periods. Furthermore, we establish DNSSEC best practices and assess how well those policies are implemented in the wild. Lastly, the thesis investigates on the differences of quality in DNSSEC deployment between large and small DNS operators. Based on that, we study the influence of economic incentives on the quality and the completeness of DNSSEC deployment.

3.2 DNSSEC Client-Side Measurement

Aside from DNSSEC server-side measurement, there are studies investigating on the DNSSEC adoption of DNS resolvers. In general, researchers have applied two approaches: (1) passive approach: researchers rely on logs from authoritative name servers or DNS queries; (2) active approach: researchers trigger DNS clients to query established domains.

Passive approach Guðmundsson and Crocker [29] observed DNSSEC validation in the wild for the *.org* namespace. According to their research outcome, it is possible to determine whether a resolver is able to validate DNSSEC responses by looking at DS queries. Furthermore, the work shows that about 10% of resolvers are DNSSEC validators. Similarly, Fukuda et al. [30] used logs of all *.jp* name servers to measure the ratio of DS queries and determined whether a DNS resolver is a DNSSEC validator. They found that only 50% of resolvers sending DNSSEC queries are DNSSEC validators.

Active approach Huston [31] presented his experiment to quantify the proportion of Internet users performing DNSSEC validation. The author employed an online

advertisement system to deliver the experiment to a random set of clients. In particular, he prepared an advertisement embedded with three links: one to a good DNSSEC signed record, one to a record with an invalid signature and one to an unsigned record. Then the advertisement was displayed in an advertisement system. The author analyzed a snapshot of data collected in May 2013 with 2,637,091 experiments. The work has shown that only 8.3% of clients seems to be performing DNSSEC validation. Lian and Rescorla [32] measured the practical impact of DNSSEC deployment on client name resolution. Applying a similar approach as Huston, the authors created an advertisement that linked to various domains classified into three classes: without DNSSEC, with good DNSSEC and with bad DNSSEC. After one week of the experiment, the work provided some valuable findings: firstly, the DNSSEC deployment increases the resolution failure rate. Specifically, one out of ten clients cannot access a DNSSEC-enabled domain; however, this rate varies among regions. Secondly, the security value offered by DNSSEC is relatively low since most resolvers cannot detect tampered data. Wander and Weis [33] performed both active and passive measurements. They configured a secure zone and conducted two test types: (1) a scripted test which requires participants to click on a test button (active); (2) a hidden test which can be embedded in websites (passive). The authors collected data over a period of seven months from May 2012 for their analysis. The outcome of this work shows that only 4.8% of resolvers are DNSSEC-capable; however, this rate varies significantly per geographic areas.

Chapter 4

DNSSEC Policies and Best Practices

This chapter presents six aspects of DNSSEC policies which zone administrators must consider carefully to maximize benefits of signed zones. All of these aspects taken together could be called a "policy". Furthermore, the best practices of these aspects are introduced based on guidelines of NIST.

4.1 DNSSEC Policies

Unlike DNS, which is usually considered as a one-shot configuration with low maintenance, DNSSEC requires a continuous effort for maintaining a stable and secure zone, especially re-signing zone data, changing signing keys and maintaining the chain of trust. All of these contribute to the complexity of DNSSEC; consequently, having a smooth operation of DNSSEC involves more considerations. RFC 6781 [13] is considered as the current best practice for DNSSEC deployment which involves multiple decisions and policies for a secure and operational zone. In this section, the important decisions that zone administrators should consider are discussed. These considerations are classified into five main categories, namely, key algorithm, key size, key scheme, key rollover and signature validity periods.

4.1.1 Key Algorithm

The core of DNSSEC is about the cryptographic operation of zone data signing and signature verification; hence, choosing a suitable public key cryptography is vital to DNSSEC. Notably, DNSSEC does not support choosing authentication and hash algorithm separately but instead offers suites of signing algorithms as in Table 2.2. At the time of writing, four types of public key algorithms, namely Rivest-Shamir-Adleman (RSA), Digital Signature Algorithm (DSA), GOST and ECDSA are standardized and

implemented. Although the Edwards-Curve Digital Security Algorithm (EdDSA) has been recently standardized in RFC 8080 [34], it is not widely implemented yet; hence, this algorithm is not discussed further in the thesis. While DSA and GOST are rarely used, RSA is, by a large margin, the most popular algorithm in DNSSEC [27]. ECDSA is emerging as a candidate to replace RSA with the benefit of smaller key sizes [27][35] while providing a similar or higher level of security [36]. Furthermore, signing with ECDSA is significantly faster than RSA [36]. This makes ECDSA particularly attractive since DNSSEC is vulnerable to the amplification-based denial of service attack due to the large size of responses when using RSA [35]. Considering those benefits, ECDSA is strongly recommended by NIST [37]. However, it is noted that with respect to signature validation, RSA outperforms ECDSA as experimented in [35][38].

The SHA1 hash algorithm is known to have cryptanalysis issues [13]. Furthermore, the Cryptology Group at the Centrum Wiskunde & Informatica (CWI) and the Google Research Security have just announced a complete approach to breaking SHA1 [39]; hence, it is recommended to move forward to a stronger hash algorithm. As a result, RFC 6781 [13], in 2012, recommended using Algorithm 8 as presented in Table 2.2, meaning RSA with SHA-256, while NIST suggests using Algorithms: 8, 10, 13 and 14 in the newer release.

4.1.2 Key Size

With respect to cryptography, the key size decides the security level of a zone. Table 4.1 [40] illustrates the key strength relation between Elliptic Curve Cryptography (ECC), RSA and symmetric key algorithms. Since GOST, ECDSA with SHA-256 and ECDSA with SHA-384 come with a pre-defined key size, this section only discusses the RSA key size.

According to RFC 6781 [13], the most-known attack against RSA is to the 700-bit length; hence, regular 1024-bit keys can be considered safe. However, higher bit keys are recommended [13][37]. It is important to note that the longer key lengths involve more workload on both authoritative servers and validating resolver sides. For example, verifying and signing with 2048-bit keys are four and ten times slower in comparison to 1024-bit keys respectively [13]. Furthermore, larger keys will produce longer signatures; consequently, longer DNS responses are generated, which may cause operational issues [37] and allow significant amplification attacks [35]. Considering the trade-off between operation and security level, NIST [37] recommends using 1024-bit ZSKs with effective time between one to three months and validity periods between

Symmetric Key Size (bits)	RSA Key Size (bits)	Elliptic Curve Key Size (bits)
80	1024	160
112	2048	224
128	3072	256
192	7680	384
256	15360	512

Table 4.1: Key size relation [40]

five to seven days for RRSIGs covering DNSKEY sets. Furthermore, KSKs are used for longer periods; hence they should be employed a stronger key size, namely 2048-bit key length.

4.1.3 Key Scheme

Although security-aware resolvers treat all keys equally, DNSSEC name servers recognize different key types for purely operational purposes; hence, a zone administrator should choose a key-type signing scheme that is suitable to the context of an organization. In principle, one can make a choice between a single-key signing scheme (also known as the Combined Signing Key (CSK) signing scheme) and a split-key signing scheme (also known as the KSK-ZSK signing scheme). RFC 4033 [8] and 6781 [13] recognize two different key types, namely the KSK and the ZSK. KSK acts as the SEP referred from the DS record in the parent zone to maintain the chain of trust as discussed in the previous section. As indicated by its name, the KSK is used to sign the DNSKEY RRset in a zone, while the ZSK is used to sign all other zone data. If a zone administrator decides to use a single key for signing, that key is called a CSK. It is worth mentioning that a zone may have multiple KSKs and ZSKs or CSKs, for example, during a key rollover.

The KSK-ZSK signing scheme

- **Benefits**: the advantages of this scheme mainly lie in benefits of the operational flexibility [13]. This operational flexibility can be argued in three points. Firstly, keys should regularly be changed for security reasons. However, rolling the SEP key is relatively expensive to the stability of the zone since it involves external parties such as the parent zone. That means that the zone administrator must interact with the parent zone to introduce a new DS record pointing to a new SEP key in such a manner that the chain of trust remains intact; in this light, the frequent rollover of SEP keys should be avoided. Considering these

respects, the KSK-ZSK split offers a trade-off between security and operational requirements. The KSK is usually chosen to be stronger than ZSK; hence, the KSK lifetime could be set longer than ZSK. Furthermore, the rollover of ZSK is relatively easy without external factor intervention. Secondly, the ease of the ZSK rollover allows zone administrators to use shorter keys without compromising the security of a zone and this is beneficial to a zone operation. Since ZSKs are used to sign all zone data, using short ZSKs minimize both computation power of signing and the data size of responses. Lastly, keys can be compromised posing risks to the security of a zone; consequently, having a strict access control to the key material or even storing them in a restricted environment (e.g. storing offline) is essential. However, this may have an adverse effect on the daily operation of a name server since it may need to access a key frequently to sign changing RRsets. For example, [3] introduces two open issues resulting from this problem, namely authenticated denial of existence and secure dynamic updates. By separating two key roles, a zone administrator can store KSKs in a highly secure place with limited access without compromising on the operational flexibility since the KSK is only used to sign the DNSKEY RRset which is only changed during key rollover time. In addition, ZSKs can be replaced relatively easily; one may choose to store them in a file system that can be accessed more easily by authoritative name servers.

- **Drawback**: the complexity of managing two different key types is the main disadvantage of this key scheme.

The CSK signing scheme

- **Benefits**: The main advantage of this key scheme is the operational simplicity. That means a zone only uses a single key type to sign both DNSKEY RRsets and other zone data; hence, the cost of management is less expensive than the cost of the previous scheme. The single-key scheme is particularly beneficial in the conditions: (1) a relatively small key size can meet operational security requirements of a zone that allows a long period of key usage and (2) key material can be accessed in a secured online environment. For example, if a zone adopts algorithm 13 or 14 in Table 2.2 and decides to store keys in an online Hardware Security Module (HSM), the zone may have more advantages by adopting the CSK scheme.

- **Drawbacks**: this scheme does not operate well with RSA keys due to large key size. Furthermore, this key scheme may require an extra investment to store keys securely online.

4.1.4 Key Rollover

Key rollover is the process of replacing an old key by a new one to mitigate the risk of cryptanalysis. Key rollover should be considered as a routine practice when deploying DNSSEC. In principle, the shorter effective period of a key lowers the risk of key compromise; therefore, scheduled key rollover events should be seriously considered. The key effective period, defined in RFC 6781 [13], is the period that a key is actually used to sign zone data. This effective period, as advised in the same RFC, is purely a policy that zone administrators should determine based on their context.

Although a zone administrator may roll keys in many fashions as long as the signing and verifying processes are ensured to operate properly, the thesis adheres to key rollover schemes that are standardized in RFC 6781 [13]. One may choose to distinguish two key types: KSK and ZSK as discussed in Section 4.1.3, and then the key rollover process is separated for each key type. While the former one involves interaction with the parent zone, the latter key type is rolled within the zone environment without communication to third parties. Furthermore, if one chooses not to distinguish between KSK and ZSK, the rollover is generally similar to the rollover of KSK. In this section, key rollover approaches for KSK, ZSK and CSK are described below.

4.1.4.1 ZSK Rollover

ZSKs can be rolled in two ways: pre-publish and double signature key rollover.

The pre-publish scheme: the main idea is to publish a new key in a zone for a reasonable amount of time, which allows a new DNSKEY set to be propagated to remote cached resolvers. Then this new key can be used to sign data as illustrated in Figure 4.1. Initially, all zone data are signed by a ZSK, ZSK_0, the figure represents this by the RRSIG of the SOA RRset. In addition, the DNSKEY RRset is signed by the KSK, KSK_1, which is referred by a DS record in a parent zone. This initial state holds for all following schemes; hence, it is not repeated for each scheme presented.

This rollover scheme involves three stages: firstly, the *New DNSKEY* stage introduces a new ZSK, ZSK_1 to a zone, and then one has to wait for at least one

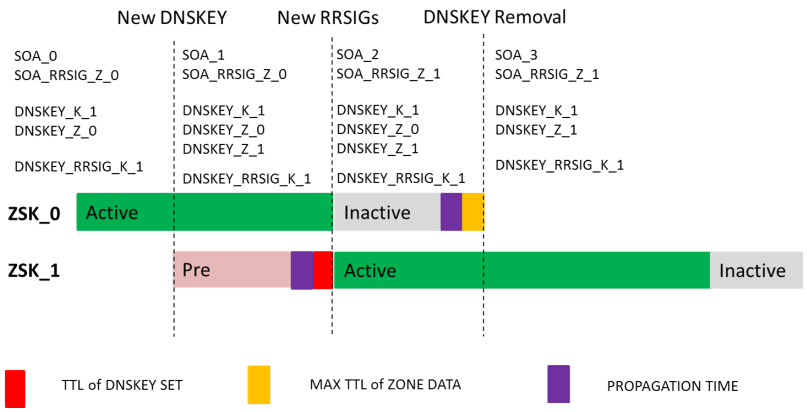


Figure 4.1: ZSK pre-publish zone signing key rollover

TTL of the DNSKEY set and the propagation time from a master server to all of its secondary servers. This is to ensure that old key material, ZSK_0, is expired from remote caches and the new DNSKEY set, including both ZSK_0 and ZSK_1, is propagated. Secondly, once this time passed, the new key is used to sign zone data and the old key becomes ineffective in the *New RRSIGs* stage. This ineffective period should last at least one propagation time and one maximum TTL of zone data to ensure that new RRSIGs are updated to secondary name servers and old RRSIGs are forgotten. Thirdly, in the *DNSKEY Removal* stage, the old key material and its associated RRSIGs can be safely removed from the zone completing a cycle of key rollover.

The double-signature scheme: is simpler than the pre-publish scheme and involves two stages, namely *New RRSIGs* and *DNSKEY Removal* as illustrated in Figure 4.2. In the *New RRSIGs* stage, the new key, ZSK_1, is introduced into the zone along to signatures signed by this new key. That means that during this period, every RRset has two signatures, one with the old key, ZSK_0, and one with the new key. Notably, this period should last at least one maximum TTL of zone data and the propagation time from a master server to all of its secondary servers. The purpose of this waiting is to be sure that the old key and all of its associated signatures are removed from remote caches. Finally, one can safely remove all material related to the old key in the zone.

Although the double-signature scheme is simpler than the pre-publish one, zone administrators should notice the consequence of double signatures, meaning double zone size during the rollover period. This may be negligible for small zones; however, it is expensive for zones with millions of records. Furthermore, double signatures

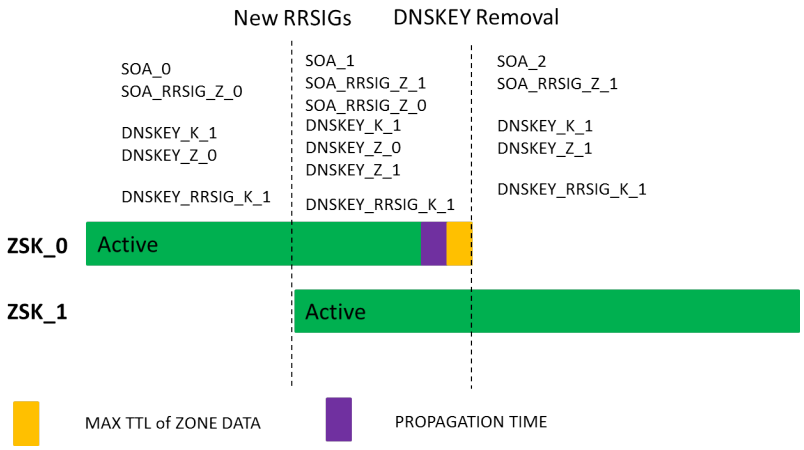
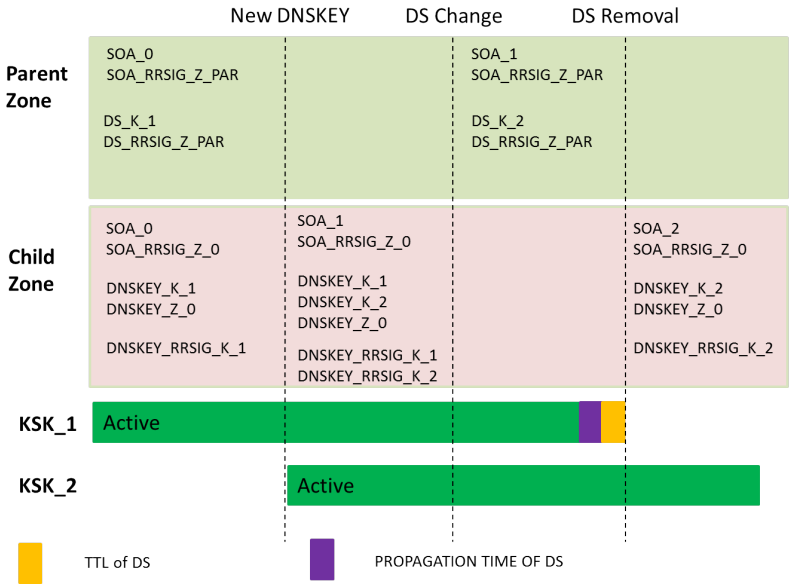


Figure 4.2: ZSK double-signature zone signing key rollover



- The blank space indicates that information in that stage is the same as the previous stage.

Figure 4.3: KSK double-signature zone signing key rollover

result in larger responses since at least two signatures are included for each query during the rollover time. Therefore, one may choose to adopt either scheme based on that consideration.

4.1.4.2 KSK Rollover

KSK rollover can be performed in two ways: double-signature and double-DS schemes.

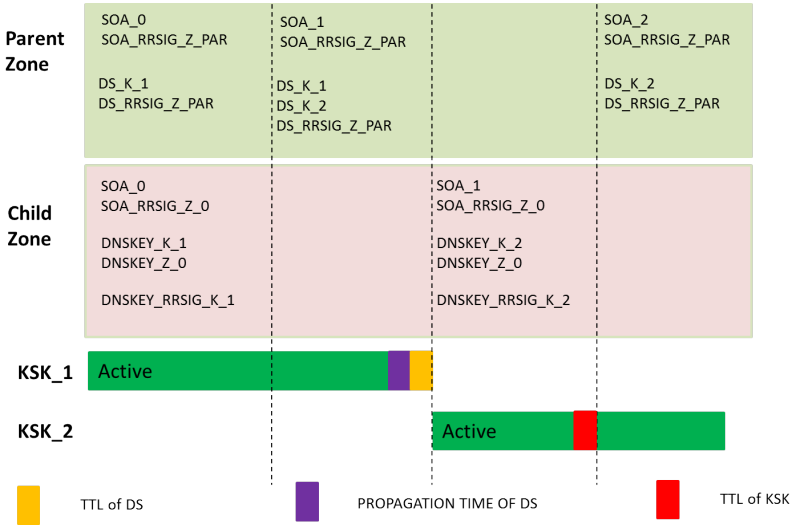
The double-signature scheme: is performed in three stages as shown in Figure 4.3. In the *New DNSKEY* phase, a new KSK key, KSK_2, as well as a DNSKEY RRSIG signed by the key are published in a zone. Since the KSK rollover requires interaction with a parent zone to update a DS record referring to the new KSK, a new DS record referring to KSK_2 is generated to replace the old one in the *DS Change* phase. This phase should last at least one TTL and the propagation time of the DS record prior to the removal of the old key material and signatures. This waiting is vital to maintain the chain of trust in two ways: firstly, it ensures that new DS records are transferred to all secondary name servers of the parent zone. Secondly, if the old key is deleted prior to the TTL of DS, remote caches may still memorize the old DS record and, hence, recognize responses as bogus. In the last stage, the old KSK and its signed DNSKEY set can be removed from the zone.

The double-DS scheme: goes through three stages as illustrated in Figure 4.4. Unlike the double-signature scheme, this rollover mechanism firstly introduces a new DS record, DS_K_2, for KSK_2 in a parent zone; however, it is worth noting that the corresponding key is not yet published in a child zone for at least one TTL and the propagation time of DS record. The rationale behind this is to prevent from breaking the chain of trust because one needs to be sure that the old DS RRset is expired in caches and the new DS RRset has been common in all zone authoritative name servers. Next, in the *New DNSKEY* stage, the new KSK and RRSIG of the DNSKEY set are introduced in the zone to replace old material. Finally, after at least one TTL of the KSK to ensure the old KSK expired in remote caches, the old DS record, DS_K_1 can be safely removed in the parent zone.

In comparison, the double-signature approach seems to be more beneficial than the double-DS scheme due to less interaction required to the parent zone. In the former one, it only requires one communication to the parent zone while the latter one needs two communications: one for adding a new DS record and one for deleting an old DS record. Furthermore, the argument of increasing zone size due to double signatures as in the ZSK rollover does no longer hold since KSKs are usually used to sign for DNSKEY RRsets; hence, the increase of size only occurs for the DNSKEY RRset.

4.1.4.3 CSK Rollover

If one decides not to distinguish KSK and ZSK roles, the CSK rollover is standardized in two manners that are quite similar to the rollover of KSK.

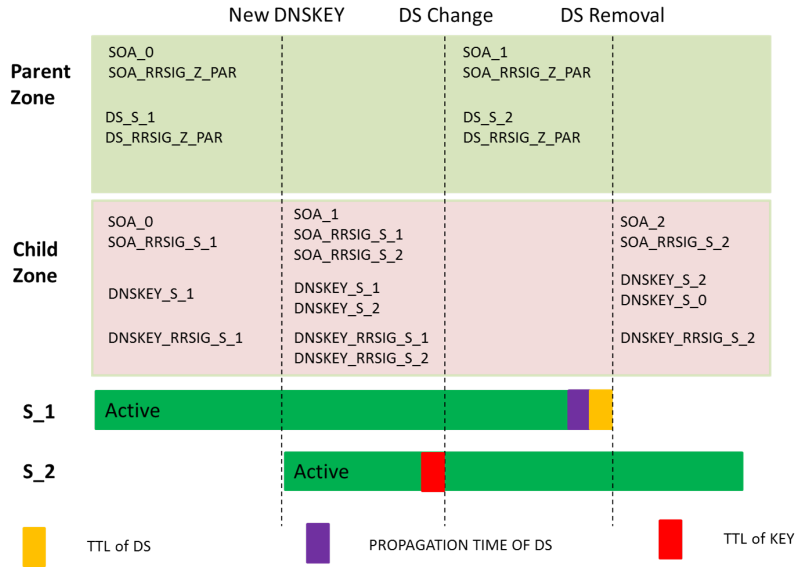


- The blank space indicates that information in that stage is the same as the previous stage.

Figure 4.4: KSK double-DS zone signing key rollover

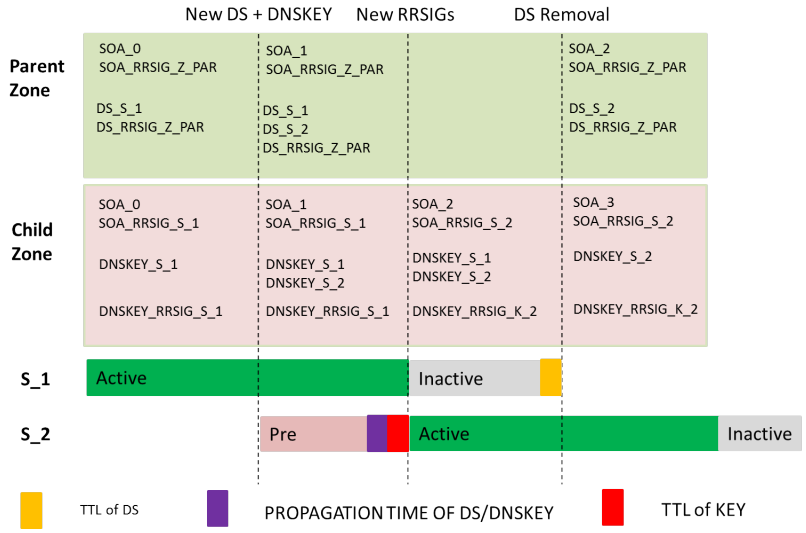
The double-signature scheme includes three stages as shown in Figure 4.5. In the *New DNSKEY* stage, the new key, S_2, is introduced and all zone data are signed by two keys, S_1 and S_2. This period should remain for at least one TTL of DNSKEY that means remote caches would expire the old DNSKEY set and start to fetch the new DNSKEY set. Next, in the *DS Change* stage, a parent zone will switch to a new DS record pointing to the new key, namely S_2. After the new DS record is propagated and the old DS record is expired from caches, the old key and signatures can be removed from the zone.

The double-DS scheme is another variant of the CSK rollover that is a combination of the pre-publish scheme for ZSK and double-DS scheme for KSK. As presented in Figure 4.6, in the first stage, a zone administrator introduces a new key, S_2, to the zone; however, it is not used to sign data yet. At the same time, the administrator interacts with a parent zone to produce a second DS record, DS_S_2, referring to S_2. This stage should remain until the DS_S_2 and S_2 are properly propagated and that the old key, S_1, should be expired from remote caches; otherwise, cached resolvers may try to use the old key to verify signatures signed by the new key. Then, in the *New RRSIGs* stage, signatures of the old keys are swapped by signatures of the new key. Once, the old DS record, DS_S_1, is expired from remote caches, the old DS and key can be safely removed, finishing a round of the key rollover.



- The blank space indicates that information in that stage is the same as the previous stage.

Figure 4.5: CSK double-signature zone signing key rollover



- The blank space indicates that information in that stage is the same as the previous stage.

Figure 4.6: CSK double-DS zone signing key rollover

4.1.5 Algorithm Rollover

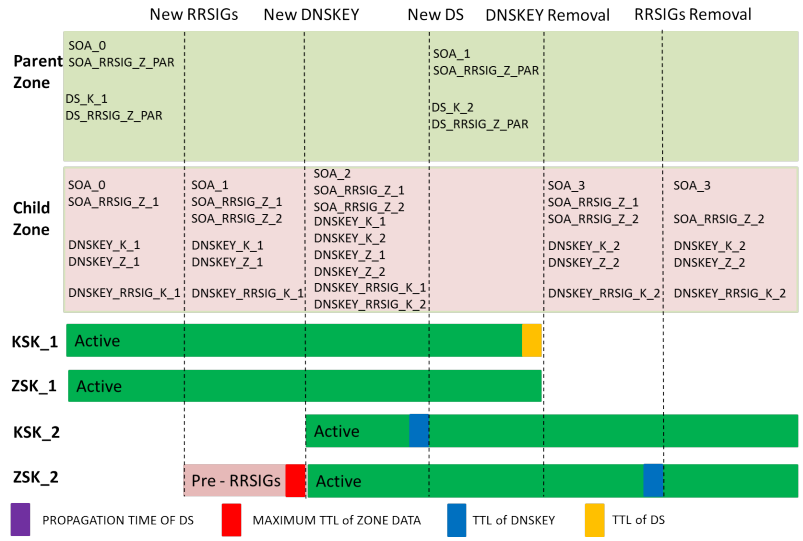
Algorithm rollover is a special class of key rollover when one needs to change from one key algorithm to another. This rollover may be motivated by various reasons, one of which is security concerns. For example, SHA1 has been just announced to be completely broken [39]; hence, zones that are signed by Algorithms 3, 5 and 7 (in Ta-

ble 2.2) would need to move forward to a stronger algorithm. Section 2.2 of RFC 4035 [10] requires that all zone data must be signed by each algorithm presented in the zone apex and that the DNSKEY set must be signed by each algorithm presented in DS records in a parent zone. This requirement can be interpreted in two ways: conservative and liberal [13]. With respect to the conservative approach, that means all signatures of a zone must be signed by each algorithm and that rule also applies to signatures which are in remote caches. On the other hand, the literal approach lessens the restriction to RRSIGs in the zone at authoritative name servers only. In this case, the algorithm rollover can be performed similarly as the KSK double-signature rollover. However, there are validating resolvers implemented in the conservative way, and in such a situation, a zone appears as bogus if it adopts the KSK double-signature rollover for the algorithm rollover. In the following paragraphs, algorithm rollover is described with respect to the conservative approach. Since the algorithm rollover for both KSK-ZSK and CSK schemes are similar as presented in Figure 4.7 and 4.8, only the former one is detailed. However, the latter rollover approach can be easily inferred because the differences are that only one key functions as a signer and that all RRSIGs signed by KSK and ZSK are replaced by signatures signed by that single key.

The KSK-ZSK algorithm rollover that adheres to the conservative approach involves five stages as presented in Figure 4.7. The key principle is that signatures should be introduced into a zone prior to corresponding keys and in a reverse manner, keys should be removed from a zone before their signature removal. This ensures that at any time, signatures are always signed by each algorithm appearing in the zone apex.

Initially, all zone data are signed by a ZSK, ZSK_0, the figure represents this by the RRSIG of the SOA RRset. In addition, the DNSKEY RRset is signed by the KSK, KSK_1, which is referred by a DS record in a parent zone.

It is worth mentioning that all keys use the same algorithm. In the first stage, *New RRSIGs*, all zone data are signed by a new ZSK and are pre-published in the zone; however, the key itself does not appear yet. This step is essential to propagate new signatures to remote caches; hence, it should last till all old signatures expire, that means a maximum TTL of zone data. If this is not followed properly, validators may have old signatures in caches and retrieve the new DNSKEY set containing the new key with a different algorithm; consequently, this violates the conservative approach. Furthermore, it is worth mentioning that the new DNSKEY RRSIG is not yet introduced in the zone since a key and its associated RRSIG are fetched together.



- The blank space indicates that information in that stage is the same as the previous stage.

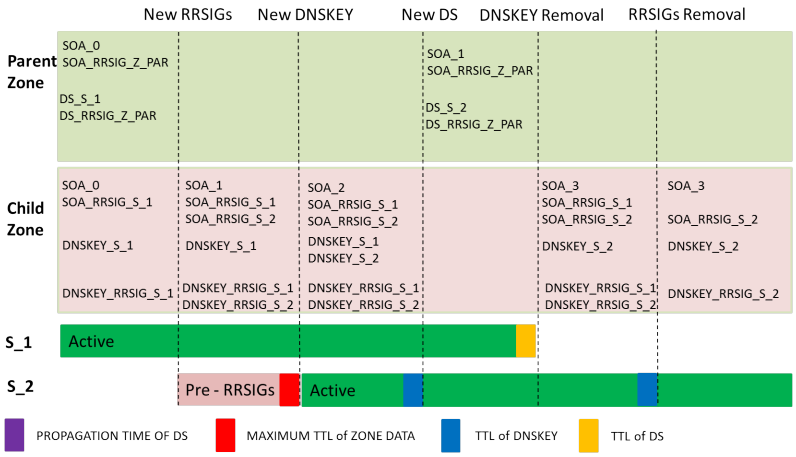
Figure 4.7: KSK-ZSK algorithm rollover

Therefore, it does not require the pre-publishing step; however, it can be done as desired. In the *New DNSKEY* stage, all new keys with a different algorithm (K_2 and Z_2) and the DNSKEY RRSIG signed by the new KSK are introduced. After the old DNSKEY set is expired from remote caches meaning a TTL of DNSKEY, ones can interact with a parent zone to create a new DS record referring to the K_2 key and remove the old one in the *New DS* stage. This stage should last at least a TTL of DS in order to make the old DS record expired from distant caches. Next, old keys can be safely removed in the DNSKEY Removal stage. Finally, one should wait until the old DNSKEY set has expired, that is a TTL of DNSKEY, then old signatures can be removed from the zone.

4.1.6 Signature Validity Periods

Another important factor in DNSSEC that zone administrators should determine is the validity period of signatures. This decision is again a policy based on an organization's context, for example, low-value and stable resources may suffer lower risks; thus, signature validity periods may be set for a long time (e.g. one year). However, if resources are updated frequently and highly sensitive, a reasonable short period should be considered.

In accordance with RFC 6781 [13], this section discusses the upper and lower bounds of signature validity periods. Generally, the upper bound of the period should be considered by taking into account the risk of the replay attack. Particularly, when



- The blank space indicates that information in that stage is the same as the previous stage.

Figure 4.8: CSK algorithm rollover

RRsets are changed or keys are compromised, RRSIGs are in need to be updated. In those situations, if the validity of old signatures is longer than the time of change, attackers may perform replay attacks by poisoning validators with the old signatures. On the other hand, the lower bound of signature validity periods can be determined by identifying the needed time to address operational issues, such as signer failure.

In a more practical way, one may decide the lower bound based on a zone's signing schedule as illustrated in Figure 4.9 [13]. That is, a signer re-visits zone data regularly, this is called the re-sign period, and refreshes only signatures that are going to expire in an identified amount of time, called the refresh period. Obviously the refresh period should be greater than the re-sign period to avoid operational issues. Furthermore, a sophisticated signing plan may prefer to avoid all signatures expired at the same time, that means that a signer would have to re-sign all zone data. Furthermore, a number of validators may need to fetch new RRSIGs from that zone due to expired signatures. As a result, authoritative name server workload may peak during re-signing periods. This operational issue can be mitigated by introducing a jitter interval to the expiration time, which helps to avoid all signatures expiring at the same time. In that light, one may set the lower bound period based on the refresh period, re-signing period and jitter interval. Figure 4.10 illustrates an example of this signing strategy with the given signature validity period, re-sign period, refresh period and jitter. The right-most column represents times (every two days) when signing software regularly checks a zone's data to re-sign signatures. In the first check on February 28, 2017, both signatures are not due to expire in four days; hence, no signature is resigned. In the next check on March 2, 2017, the signature

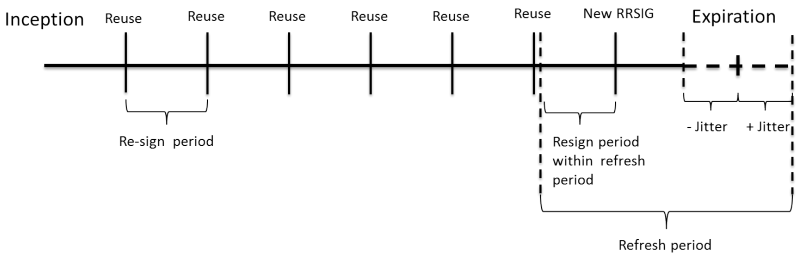


Figure 4.9: Signature validity periods

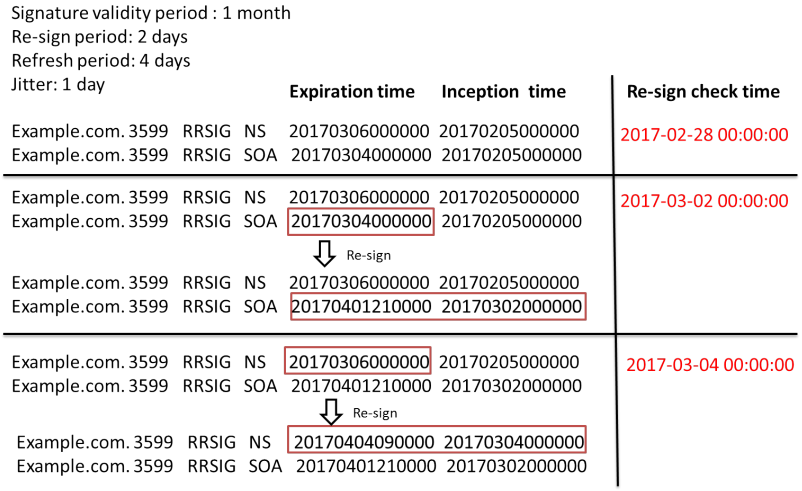


Figure 4.10: An example of signing signatures

of the SOA RRset is expired in two days which is within the refresh period; hence, this RRSIG is resigned. It should be noted that the expiration of this new RRSIG is not on April 2, 2017 for a month validity period. Instead, a jitter is added making the new signature expire on April 1, 2017 21:00:00. Similarly, in the third check, the RRSIG of the NS RRset is due to expire in two days and is refreshed with a jitter added to the expiration time.

4.2 DNSSEC Best Practices

Although there is no universal agreement on DNSSEC best practices, several works propose good guides for DNSSEC deployment: RFC 6781 [13], the Good Practices Guide for Deploying DNSSEC by European Network and Information Security Agency (ENISA) [41] and two guides by NIST, namely Secure Domain Name System (DNS) Deployment Guide [42] and Recommendations for Key Management (part 3) [37]. However, the ENISA's document is outdated (2010); therefore, in this section,

Components	**RFC 6781**	**NIST**
Key size	Using RSA with the key size. large than 1024 bits.	Using ECDSA (P-256, P-384). Using RSA with the key size: - 2048 bits for KSKs. - 1024 bits for ZSKs.
Key algorithm	Using Algorithm 8.	Using Algorithm 8,10,13 and 14.
Key rollover	**KSKs/CSKs:** - Rolling over every 12 months. **ZSKs:** - More frequently than KSKs.	**KSKs/CSKs:** - 2048-bit RSA keys: rolling over every 12-24 months. - ECDSA (P-256 and P-384) keys: rolling over every 12-24 months. **ZSKs:** - 1024-bit RSA keys: rolling over every 30-90 days. - ECDSA (P-256 and P-384) keys: rolling over every 12-24 months
Key rollover approaches	**KSKs/CSKs:** Double signature and double DS **ZSKs:** Pre-publish and double signature	**KSKs/CSKs:** Double signature **ZSKs:** Pre-publish
Key algorithm rollover	Conservative approach	Conservative approach
Signature validity period	Depend on organizations' policies	2-7 days for RRSIGs covering DNSKEY RRsets

Table 4.2: DNSSEC Recommendations

recommendations for each policy discussed in the previous sections are presented based on RFC 6781 and NIST's guides. While RFC 6781 establishes the foundation of DNSSEC deployment considerations and provides generic advice, NIST presents more detailed guidelines and thus can be considered best practice. Table 4.2 shows recommendations to maximize benefits of DNSSEC.

In this thesis, we base on NIST's recommendations to establish the best practices for DNSSEC deployment. Although NIST's recommendations are quite specific, there is still room for various interpretations. Hence, Table 4.3 presents in greater details what configurations we consider as recommended and unrecommended.

Aspects	**Recommended**	**Unrecommended**
Key size	ECDSA keys. RSA: - KSKs >= 2048 bits. - ZSKs >= 1028 bits.	DSA, GOST keys with any key size. RSA: - KSKs <2048. - ZSKs <1024.
Key algorithm	Recommended: Algorithms 8 and 10. Highly recommended: Algorithms 13 and 14.	Algorithms 1,3,5,6,7 and 12.
Key rollover	**KSKs/CSKs:** Rollover within 24 months for: - ECDSA keys. - RSA keys with key size >=2048 bits. **ZSKs:** 1024-bit RSA keys: rollover within 90 days. 1024 <RSA keys' size <2048 bits: rollover within 12 months. ECDSA keys and RSA keys (with key size >= 2048 bits): rollover within 24 months.	**KSKs/CSKs:** DSA, GOST and RSA keys (with key size <2048 bits) with any key effective periods. RSA keys with key size >=2048 bits having key effective period >24 months. **ZSKs:** DSA, GOST and RSA keys (with key size <1024 bits) with any key effective periods. 1024-bit RSA keys having effective periods >= 90 days. 1024 bits <RSA keys' size <2048 bits having key effective periods >12 months. RSA keys with key size >= 2048 bits having key effective period >24 months.
Key rollover approaches	**KSKs/CSKs:** Double signature. **ZSKs:** Pre-publish.	**KSKs/CSKs:** Double DS. **ZSKs:** Double signature.
Key algorithm rollover	Conservative approach.	Conservative approach.
Signature validity period	Within 7 days for DNSKEY RRSIGs.	Greater than 7 days for DNSKEY RRSIGs.

Table 4.3: DNSSEC Best Practices

Chapter 5

Research Questions

The previous chapter provides aspects of DNSSEC policies and best practices which zone administrators should consider carefully to maximize benefits of DNSSEC. Based on that, we aim to study and evaluate the DNSSEC deployment on the Internet. Furthermore, we examine the influence of economic incentives on the quality and the completeness of DNSSEC deployment between small and large DNS operators.

In this chapter, we discuss three research questions as well as the accompanying hypotheses.

Research Question 1

DNSSEC deployment involves a set of critical considerations that directly determine the quality and security of signed zones. Those considerations are presented in Table 4.3. Recent efforts [24, 27] have shown that there are a non-trivial amount of signed zones deployed with weak decisions on those aspects; hence, in this thesis, we aim to investigate the security status of DNSSEC deployment on a large scale. To extend previous work, we rely on the NIST's recommendations as the best practices (Table 4.3) to study and analyze the quality of signed zones. In particular, we aim to study the following research question:

RQ1: How are signed zones deployed in the wild compared to NIST's recommendations?

Research Question 2

In order to further investigate the deployment of DNSSEC, we aim to study the effect of economic incentives on the quality of signed zones. Although DNSSEC is expected to enhance DNS security, its adoption is still in the early stage due to the lack of DNS operators' interest. In order to encourage the DNSSEC deployment,

several TLD registries offer economic incentives for each signed domain [16, 17]. In particular, the ccTLDs *.nl* and *.se* offer a small discount to registrars for every signed zone. This initiative has led to some success, for example, the ccTLD *.nl* became the largest DNSSEC zone in a short time space [19]. However, aside from that positive response, it is suspected that considering DNSSEC as a commodity may cause DNS operators focusing more on the business profit than the quality of signed zones [19].

ENISA has performed a survey on the costs of DNSSEC deployment and concluded that the deployment costs do not depend on the number of zones that a DNS operator manages [43]. While the economic incentive is based on the number of signed zones, the cost of DNSSEC deployment is correlated to other factors. Therefore, it may be assumed that large DNS operators (in terms of the number of domains) may receive more benefits from economic incentives than small DNS operators, since the discount, which they receive for each signed zone, may outweigh the cost of DNSSEC deployment and maintenance. On the other hand, it is suspected that small DNS operators may deploy DNSSEC for other motivations rather than the economic incentive such as market competitive advantages and security considerations. With respect to this concern, we aim to study the differences of quality of signed zones in *.nl* and *.se* between small and large DNS operators:

RQ2: Do small DNS operators deploy DNSSEC with higher quality than large DNS operators for domains under .nl and .se?

To answer the second research question, two hypotheses are defined based on the difference of DNSSEC deployment motivations between small and large DNS operators. In particular, large DNS operators are suspected to deploy DNSSEC for economic reasons. Furthermore, previous works [24, 27, 28] have suggested that a considerable number of signed zones are employing unrecommended implementations of the key algorithm, key size and key rollover policies. In that sense, we hypothesize that:

H2.1: Large DNS operators fail more often than small DNS operators at implementing NIST's key algorithm recommendations for domains under .nl and .se.

H2.2: Large DNS operators fail more often than small DNS operators at implementing NIST's key size recommendations for domains under .nl and .se.

H2.3: Large DNS operators fail more often than small DNS operators at implementing NIST's key rollover recommendations for domains under .nl and .se.

Research Question 3

In order to further study the influence of economic incentives on DNSSEC, the third research question focuses on the completeness of DNSSEC deployment. In particular, a signed zone is considered complete when it has established a secure delegation with its parent zone. Recall from Section 2.3.2, that a secure delegation is the DS record, which maintains the chain of trust between child and parent zones. According to Wander's work [27], missing DS record is the most common issue in the DNSSEC deployment. As discussed in research question 2, small and large DNS operators may have different motivations for deploying DNSSEC. For example, a large DNS operator may maintain DS records properly for only domains under *.nl* and *.se* to earn incentives and does not deploy DS records for domains under *.com* where there is no economic incentive applied. On the other hand, due to a considerable financial investment [43] demanded by the DNSSEC deployment, it is unlikely that small DNS operators invest on DNSSEC when they cannot deploy it fully for managed zones. In this research question, we investigate the differences in the full deployment of DNSSEC between small and large DNS operators. Hence, we present the third research question and its accompanied hypothesis as below:

RQ3: Do small DNS operators deploy signed zones fully more often than large DNS operators?

H3: The missing DS record rate in small DNS operators is lower than the missing DS record rate in large DNS operators

Chapter 6

Approach

6.1 Research Methodology

This section aims to provide an overview of approaches that the thesis applies to analyze the dataset [22] in order to answer the questions discussed in Chapter 5. In particular, the section presents methods to distinguish between different key schemes (KSK-ZSK and CSK schemes) and extract related data to study the configurations of key management in the wild. These methods provide a reliable mean to address research question 1. In addition, we also define a criterion to classify small and large DNS operators, which facilitates the study of research questions 2 and 3.

6.1.1 Distinguish between KSK-ZSK and CSK Scheme

As explained in Section 4.1.3, the DNSSEC validation process does not distinguish different key roles; however, a signed zone might decide to use KSK-ZSK or CSK schemes for operational purposes. In that respect, a DNSSEC validator must perform the DNSSEC validation process to identify whether a zone distinguishes between KSK and ZSK key roles. That is, a zone adopts the CSK scheme if it uses a SEP DNSKEY to sign the DNSKEY RRset as well as zone data. On the other hand, a zone adopts the KSK-ZSK scheme if it uses two different DNSKEYs to sign the DNSKEY RRset and zone data. However, the main drawback of this approach to distinguish between key schemes is that the validation process is expensive.

An alternative approach is to utilize the key tag field. Particularly, a zone adopts the CSK scheme if the key tag included in a DS record is the same as the key tag in the RRSIG of a SOA RRset. If the two key tags are different, the zone employs the KSK-ZSK key scheme. Although this method is vulnerable to the key tag duplication within a key set (which is quite rare), the further process of key management discussed

in the following paragraph will detect the key tag duplication event and handle it accordingly. Since this alternative approach offers better performance than the former approach, the thesis employs this method to distinguish between the two key schemes.

6.1.2 Extracting Data Related to The Configuration of Key Management

Signed domains may have many keys published in their zones. However, some of them may not be actually used to sign data. In this study, we only consider keys that are used to sign data and take roles in the operation of DNSSEC. That means that we do not consider KSKs that do not have corresponding DS records and ZSKs that are not used to sign zone data. Particularly, SOA RRs are considered as the representative of a zone's data since every zone should have exactly one SOA RR [5]; hence, the RRSIGs of SOA RRs are verified against ZSKs.

Furthermore, in the KSK-ZSK scheme, it is important to emphasize that the thesis does not rely on the flags to acknowledge key roles. Instead, the actual verification is performed. In particular, keys are recognized as KSKs when they are used to sign DNSKEY RRsets and there are DS records in a parent zone referring to them. Similarly, keys are recognized as ZSKs when they are used to sign SOA RRsets. This verification approach offers two benefits over the key-flag-dependent approach: firstly, it filters out unused keys and therefore, makes the result more reliable; secondly, it is immune to mis-configured key flags.

When processing data for domains applying the CSK scheme, a special caution should be given to the key tag duplication event because domains using the KSK-ZSK scheme will have the same key tag for both signatures and DS records; hence, these domains may be misread as applying the CSK scheme. In such situations, keys are verified against both DS records and signatures of DNSKEY sets. The success of both validations indicates the usage of the CSK scheme. Otherwise, those domains are applying the KSK-ZSK scheme and should be handled accordingly.

Figure 6.1 presents an overview of how key information is obtained. The process involves five primary steps:

- **Step 1**: firstly, key information for one month is collected. Since keys used in a domain may have duplicated key tags, a key-tag duplication check is performed for all domains. The "data" field indicates any other related data; in this experiment, we store key algorithms and partial public keys. Secondly, depending on the key type, corresponding data are obtained:

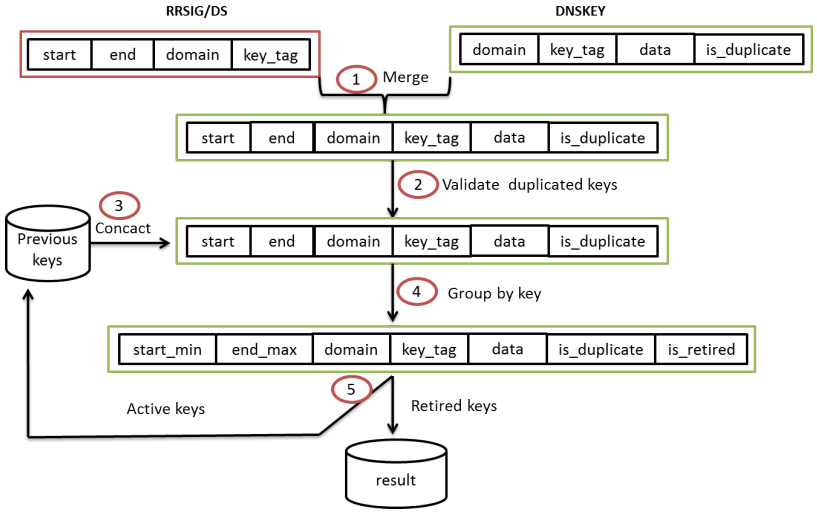


Figure 6.1: Extracting key information in the monthly-basic approach

- ZSKs: signatures of SOA RRsets are collected.
- KSKs: we collect both signatures of DNSKEY RRsets and DS records.
- CSKs: signatures of SOA RRsets and DNSKEY RRsets, as well as DS records, are obtained.

Furthermore, each record is associated with a "start" and "end" time, which indicates its usage time. Finally, collected information is merged together to provide the usage time of keys. It is noted that a key may appear multiple times since it may be used to signed data frequently in that month.

- **Step 2**: keys with duplicated key tags are verified to ensure that only valid ones are taken into account.
- **Step 3**: remaining keys of previous months are added to the key pool.
- **Step 4**: keys with multiple records are grouped together to identify the usage period of those keys, that is the minimum of the "start" time and the maximum of the "end" time. In other words, a key effective period is checked by its first and last usage. Considering the last day of a checking month, keys can be verified whether they are retired in that month.
- **Step 5**: if keys are retired, their information is stored for further analysis to answer research questions. Otherwise, they are added to the "previous keys" dataset. This finishes a process of a month data.

6.1.3 Distinguishing between Small and Large DNS Operators

Since a clear-cut definition of large and small DNS operators is not available, we define a criterion to classify DNS operators. Particularly, we apply the Pareto principle (also known as the 80/20 rule) to distinguish types of DNS operators. That is, a small number of DNS operators account for a significant number of domains, those are considered large DNS operators. All others are classified as small DNS operators.

6.2 Dataset

The thesis uses the data from the large-scale OpenINTEL active measurement platform [22]. It is a unique long-term dataset that crawls daily DNS records from all domains under various TLDs together comprising more than 50% of the global DNS namespace. The project has been measuring DNS for over two years since February 2015. Initially, the measurement platform monitored all domains under three primary TLDs, namely *.com*, *.net* and *.org* by issuing queries to authoritative name servers and obtaining different types of DNS responses: SOA, A, AAAA, NS, MX, TXT, Sender Policy Framework (SPF), DS, DNSKEY and NSEC [22]. Later, OpenINTEL has added more TLDs into the dataset; Table 6.1 presents data from the measurement that we use in this thesis.

We analyze data from 12 TLDs with different measurement periods to study the DNSSEC deployment on the Internet as shown in Table 6.1. In this thesis, we only consider zones having functional DNSKEYs; those zones are called *signed zones*. That means that zones are not counted if they only publish DNSKEYs and do not apply those keys in DNSSEC operations. As presented in the table, about 3% of domains are signed, in which the KSK-ZSK scheme is by far the most popular scheme. Furthermore, TLDs *.nl* and *.se* achieve the highest proportions of signed zones, about half of their domains implement DNSSEC.

The dataset has two limitations: (1) it only covers data longer than two years for TLDs: *.com*, *.net* and *.org*; hence, we do not have sufficient data to study some aspects of DNSSEC policies related to KSKs and CSKs in other TLDs since these keys are advised to be used up to two years; (2) since OpenINTEL measures domains daily, the dataset may miss DNSSEC operations happening within a day, such as some key rollover operations.

TLDs	Start date	End date	#Domains	#Signed domains			%
				KSK-ZSK scheme	CSK scheme	Total	
			On March 31, 2017				
.com	28/2/15	31/3/17	123,965,154	900,806	3,143	903,949	0.73
.net	28/2/15	31/3/17	14,567,892	138,060	661	138,721	0.95
.org	28/2/15	31/3/17	10,147,085	103,026	486	103,512	1.02
.nl	9/02/16	31/3/17	5,685,813	2,647,052	53,235	2,700,287	47.49
.mobi	6/4/16	31/3/17	562,301	3,481	0	3,481	0.62
.info	6/4/16	31/3/17	5,048,847	43,674	230	43,904	0.87
.nu	7/6/16	31/3/17	307,459	98,605	229	98,834	32.15
.se	7/6/16	31/3/17	1,392,007	723,874	392	724,266	52.03
.fi	23/11/16	31/3/17	417,570	5,563	7	5,570	1.33
.ca	7/7/16	31/3/17	2,481,305	868	16	884	0.04
.at	12/1/17	31/3/17	1,235,310	5,901	60	5,961	0.48
.dk	6/2/17	31/3/17	1,308,530	19,964	11,189	31,153	2.38
Total			**167,119,273**	**4,690,874**	**69,648**	**4,760,522**	**2.85**

Table 6.1: Dataset

6.3 Challenges

We have encountered two main challenges when analyzing such a large dataset to study the configurations of DNSSEC deployments in the wild.

Firstly, the number of measured domains and records imposes a significant challenge on the methodology to process and extract meaningful information. For example, an experiment to measure the key effective period is estimated to take about one month for the execution based on the daily analysis approach. In addition, row-based methods to process the hundred-million-record dataset would likely fail to meet an acceptable execution time. Furthermore, the size of processed data also poses an additional difficulty. Therefore, several principle rules are followed to overcome this challenge: (1) data are always processed on a monthly basis. Since DNSSEC-related data are often active for more than several days, analyzing such data in a short period frame would provide not only local information (not the general and meaningful one) but also suffer a high volume of computational redundancy; (2) vectorized computation is prioritized and applied as many times as possible.

Secondly, the diversity of signed zone management behaviors poses another challenge to analyze the dataset. For example, when processing the dataset, we find malformed DNSSEC operations of more than 30,000 zones; particularly, those zones respond to DNSSEC-related queries intermittently. Consequently, it is hard to measure the effective periods of keys since we do not have continuous data of those zones. This problem is caused mostly by a single DNS operator, namely **.domainmon-*

ster.com. As a result, this obstacle requires an extra effort to track those problematic DNSSEC zones.

Chapter 7

Results

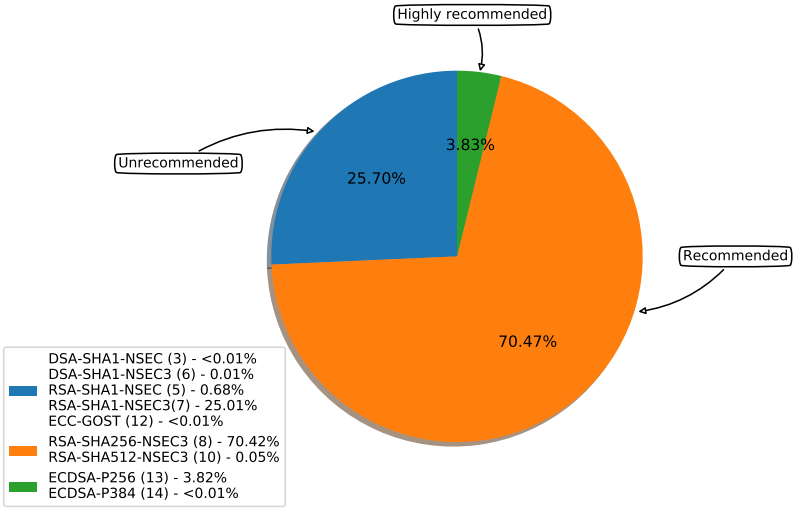
This chapter presents an analysis of DNSSEC policies implemented in the wild based on the best practices in Table 4.3 and the dataset as discussed in the previous chapter (from Section 7.1 to Section 7.6). Essentially, this information allows for a detailed insight into the quality of signed zones deployed on the Internet and hence, provides solid answers for the first research question. Further, the obtained knowledge is used to analyze the differences in the quality of signed zones managed by large and small DNS operators. Based on this analysis, we present the answers to the second and third research questions (Section 7.7 and Section 7.8 respectively).

7.1 Key Algorithm

This section firstly provides information about the key algorithm aspect in both KSK-ZSK and CSK schemes by using a snapshot data in March 2017, then trends over time are presented for each TLD in the dataset.

7.1.1 KSK-ZSK Scheme

Figure 7.1 provides an overview of the implementation of key algorithms for the KSK-ZSK scheme for all investigated TLDs (the absolute numbers can be found in Table A.1, Appendix A). As shown in the pie chart, about a quarter of domains are using unrecommended algorithms, mainly Algorithm 7 (RSA with SHA1). Interestingly, it can be easily identified that more than 90% of domains use RSA, while ECDSA only accounts for a small amount. This status may suggest two near-future trends since ECDSA offers equal or higher security comparing to RSA with shorter key lengths: (1) since SHA1 is completely broken [39], a significant number of domains will perform the key algorithm rollover to a more secure algorithm; (2) since



- Legends show full key algorithm descriptions and their associated numbers inside parentheses
- Percentages present the contribution of each key algorithm to the total population.

Figure 7.1: Pie chart for KSK-ZSK key algorithm in March 2017

ECDSA is highly recommended by NIST and offers equal or higher security comparing to RSA with shorter key lengths, its growth rate is expected to increase rapidly in the time to come. Furthermore, 410 domains use multiple key algorithms in parallel. This may be an indication of key algorithm rollover processes.

Table 7.1 provides a closer look at the key algorithm details for each TLD. Highlighted lines indicate TLDs that have a high proportion of unrecommended key algorithm usage. In general, more domains under ccTLDs have adopted suggested algorithms than domains under gTLDs. Especially, four popular TLDs, namely *.com*, *.net*, *.org* and *.info* are performing quite poorly since more than half of DNSSEC-enabled zones are signing with unrecommended algorithms.

On the opposite side, ccTLDs are doing significantly better. Particularly, only about 22% of domains adopt unrecommended algorithms in *.nl*, the largest DNSSEC-enabled TLD in the world, and less than 1% in *.se* as shown in Table 7.1.

Figure 7.2 shows trends of each TLD over the monitoring time. These trends are consistent with the analysis above demonstrating that ccTLDs are adopting better key algorithms than gTLDs. In addition, in the four popular gTLDs: *.com*, *.net*, *.org* and *.info*, although the number of domains using recommended algorithms has increased, less secure choices still form a majority. Interestingly, we observe a similar peak pattern shared between these four TLDs in October 2016 for recommended algorithms. That sudden increase is made by a single DNS operator, namely **.hyp.net*.

TLDs	Highly recommended (1)	Recommended (2)	Unrecommended (3)	% (1)	% (2)	% (3)
.com	103,377	336,529	461,104	11.47	37.35	51.18
.net	19,490	44,854	73,791	14.11	32.47	53.42
.org	13,914	35,073	54,101	13.50	34.02	52.48
.nl	5,960	2,067,510	573,589	0.23	78.11	21.67
.mobi	590	2,564	333	16.92	73.53	9.55
.info	4,755	14,586	24,346	10.88	33.39	55.73
.nu	1,321	91,346	5,943	1.34	92.63	6.03
.se	17,774	699,534	6,582	2.46	96.64	0.91
.fi	1,475	3,844	278	26.35	68.68	4.97
.ca	374	226	276	42.69	25.80	31.51
.at	112	2,928	2,866	1.90	49.58	48.53
.dk	10,409	6,955	2,613	52.10	34.82	13.08

* Recommended = (1) + (2)

Table 7.1: KSK-ZSK key algorithm per TLD in March 2017

Prior to October 2016, this operator only signed zones by Algorithm 8. However, they newly signed a significant number of zones, over 118,000 in total, with Algorithm 13 causing the peak as seen in the figure. Moreover, the number of domains signed with Algorithm 8 decreased rapidly in following months and caused a decreasing trend. This may be an indication that **.hyp.net* performed key algorithm rollovers from Algorithm 8 to Algorithm 13 during this period.

7.1.2 CSK Scheme

Compared to the KSK-ZSK scheme, there are significantly fewer domains using this key scheme. However, the fraction of domains using unrecommended algorithms is much lower as illustrated in Figure 7.3. In addition, although RSA is still the dominant algorithm, more than a quarter of domains use ECDSA to sign their zones. This trend is anticipated to continue in the coming time since the CSK scheme and ECDSA match well together and offer both a high level of security and a simplicity of key management for signed zones.

Analyzing in further detail, Table 7.2 shows that domains under most of the measured TLDs adopt recommended key algorithms. However, it is noted that while TLDs tend to adopt better key algorithms in the CSK scheme than in the KSK-ZSK one, domains under *.se* are doing the opposite. Furthermore, domains under *.at* are choosing consistently weak key algorithms for both KSK-ZSK and single-key schemes. Moreover, trends in the key algorithm adoption over the measurement periods for TLDs are also analyzed. A clear improvement has been observed in almost all of TLDs. The details can be found in Figure A.1, Appendix A.

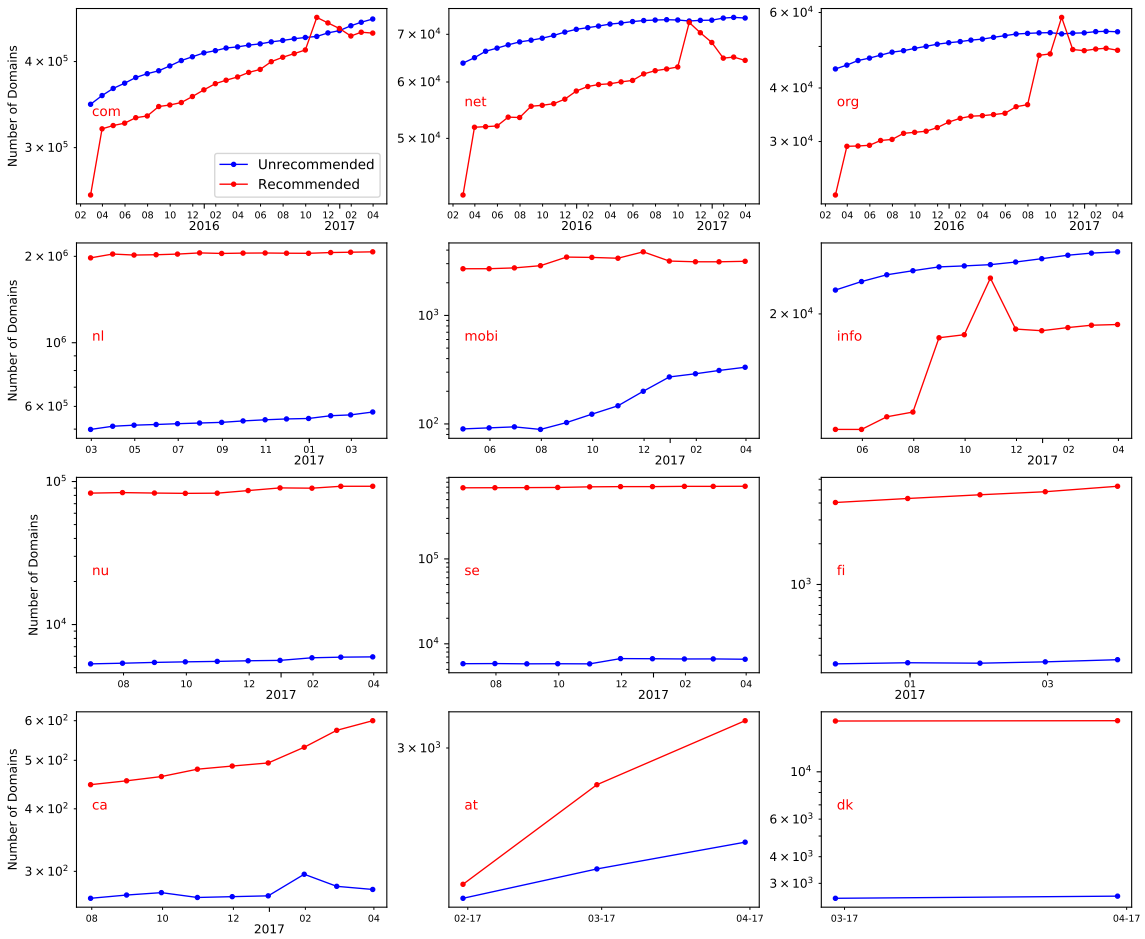
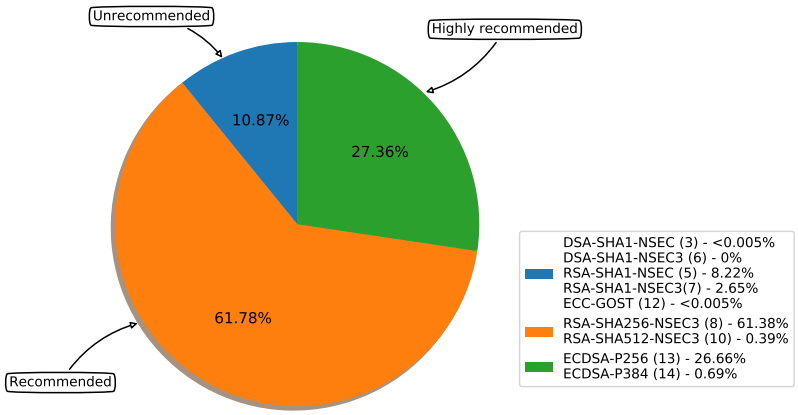


Figure 7.2: KSK-ZSK key algorithm per TLD over measurement periods

TLDs	Highly recommended (1)	Recommended (2)	Unrecommended (3)	% (1)	% (2)	% (3)
.com	2,029	374	756	64.23	11.84	23.93
.net	302	134	236	44.94	19.94	35.12
.org	235	91	169	47.47	18.38	34.14
.nl	4,784	42,397	6,058	8.99	79.64	11.38
.mobi	0	0	0	0.00	0.00	0.00
.info	135	18	77	58.70	7.83	33.48
.nu	214	14	1	93.45	6.11	0.44
.se	156	15	235	38.42	3.69	57.88
.fi	7	0	0	100.00	0.00	0.00
.ca	9	5	3	52.94	29.41	17.65
.at	16	10	36	25.81	16.13	58.06
.dk	11,183	4	4	99.93	0.04	0.04

* Recommended = (1) + (2)

Table 7.2: CSK key algorithm per TLD over measurement periods



- Legends show full key algorithm descriptions and their associated numbers inside parentheses
- Percentages present the contribution of each key algorithm to the total population.

Figure 7.3: Pie chart for CSK key algorithm in March 2017

7.2 Key Size

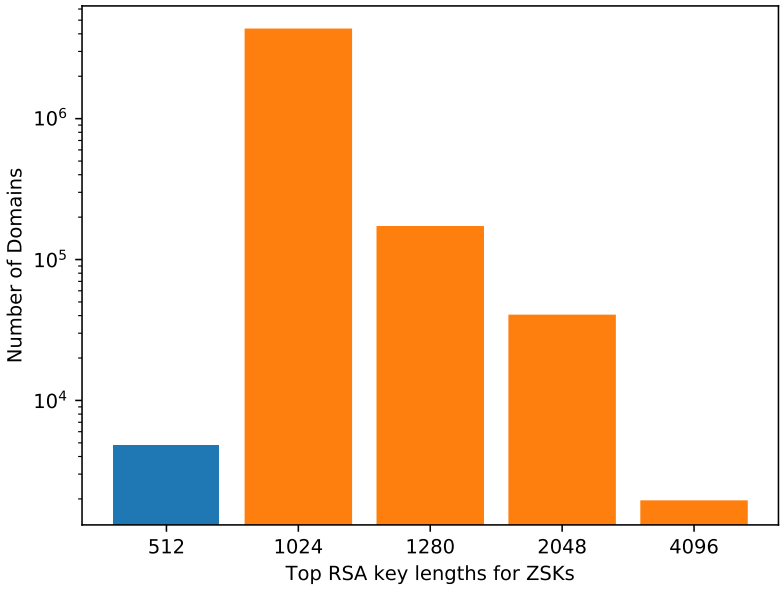
Since DSA and GOST keys are rarely used, and ECDSA keys come with fixed key lengths, this section only presents the key sizes for RSA keys.

7.2.1 ZSK

Figure 7.4 shows the most popular RSA key sizes used for ZSKs. Although the use of the 1024-bit key length is acceptable in NIST's best practice, using longer key lengths is strongly recommended. However, in this measurement, the 1024-bit key length is the most popular accounting for approximately 95% of all ZSKs using RSA. In general, most of the zones (about 99.99%) are choosing recommended key sizes for their zones. A similar conclusion is also applied when analyzing data for each TLD. Further details can be found in Table A.3 and Figure A.3 in Appendix A.

7.2.2 KSK

Figure 7.5 presents the top five most popular key sizes used for KSKs and the current status compared to the best practices presented in Table 4.3. As shown in Figure 7.5(a), although the 2048-bit key length accounts for the largest proportion, unrecommended key sizes, namely 512, 1024 and 1536, surprisingly appear in the top key lengths. Consequently, while recommended ZSK key sizes are deployed widely,



- The orange color indicates recommended key lengths
- The blue color indicates unrecommended key lengths

Figure 7.4: ZSK: Key size usage in March 2017

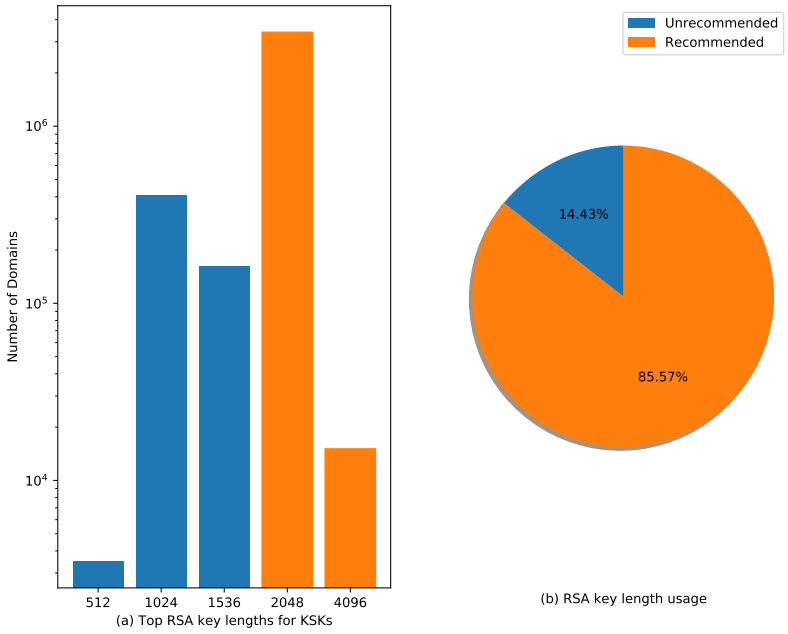


Figure 7.5: KSK: Key size usage in March 2017

there are a non-trivial amount of domains using weak key sizes for KSKs (about 15% of domains as illustrated in Figure 7.5(b)).

Table 7.3 offers more fine-grained information for each TLD. The first impression is that the number of KSKs is far fewer than the number of ZSKs since we only

TLDs	Recommended (1)	Unrecommended (2)	% (1)	% (2)
.com	235,058	298,202	44.08%	55.92%
.net	32,756	50,306	39.44%	60.56%
.org	24,045	36,767	39.54%	60.46%
.nl	2,483,309	59,477	97.66%	2.34%
.mobi	0	0	0.00%	0.00%
.info	6,386	17,265	27.00%	73.00%
.nu	76,656	4,269	94.72%	5.28%
.se	550,639	105,323	83.94%	16.06%
.fi	479	26	94.85%	5.15%
.ca	138	25	84.66%	15.34%
.at	2,063	1,624	55.95%	44.05%
.dk	713	1,861	27.70%	72.30%

Table 7.3: KSK: RSA key sizes per TLD in March 2017

consider KSKs if they are referred to by DS records. A large proportion of domains at each TLD are suffering the incompleteness of DNSSEC deployments, that is the missing of DS records. Especially, we do not observe any DS records referring to KSKs in all domains under *.mobi*. However, there is an exception for two TLDs, namely *.nl* and *.se*, where the missing DS rates are maintained relatively low, about 4% and 7% respectively.

Comparing to the best practices, more than half of domains having DS records under four popular gTLDs, namely *.com*, *.net*, *.org* and *.info*, adopt weak key sizes for KSKs, while other TLDs, except for *.dk*, are performing relatively well as shown in Table 7.3. In addition, this observation remains applicable when analyzing data of TLDs over their measurement periods (the details can be found in Figure A.4, Appendix A).

With regard to the analysis of KSK-ZSK pairs, the most common size of KSK-ZSK pairs is 2048-1024, since 2048-bit and 1024-bit lengths are the most popular key sizes for KSK and ZSK respectively. Furthermore, it is worth mentioning that due to the high rate of missing DS records, the number of domains having both keys reduces considerably. Therefore, the analyzed information of the KSK-ZSK pair is mainly affected by KSKs. Further information can be found in Appendix A.

7.2.3 CSK

Figure 7.6 presents the current status of the key length usage in the CSK scheme. While the majority of domains adopt good key size practices in the KSK-ZSK scheme, more than half of the domains employ weak key lengths in the CSK scheme. The

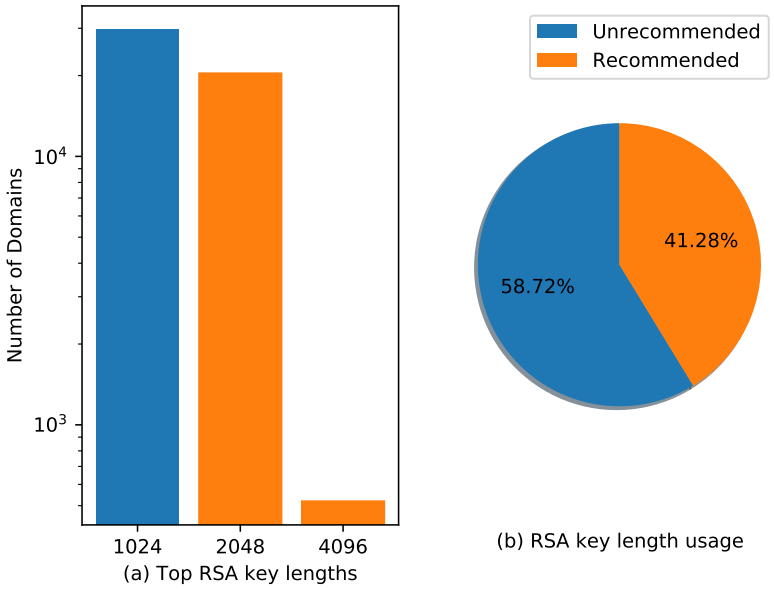


Figure 7.6: CSK scheme: Key size usage in March 2017

primary cause of this problem is the choice of 1024 bits for RSA keys. Since this key scheme is rarely used, this information may reflect the lack of understanding of this key scheme.

Table 7.4 shows that *.nl* has the highest proportion of unrecommended key size usage. Furthermore, we observe that a large part of domains under all TLDs are adopting weak decisions on the key size. Notably, most of the domains under *.se* are using unrecommended key sizes. This is surprising because both *.nl* and *.se* are operating well in the KSK-ZSK key scheme. The analysis over the whole measurement period for each TLD provides further information about trends and can be found in Appendix A.

7.3 Key Rollover

This section presents and evaluates results of key effective periods applied in the wild against best practices. Since different lifetimes of keys are recommended based on key algorithms (e.g. DSA, RSA, GOST and ECDSA) and key sizes, this section studies the common combinations of these two factors. Particularly, we present results for RSA 1024-bit and ECDSA keys for ZSK. With regard to KSKs and CSKs, we consider RSA keys with the length equal or greater than 2048 bits and ECDSA keys. Since the CSK scheme is not widely used, its results are not significant and are placed in

TLDs	Recommended (1)	Unrecommended (2)	% (1)	% (2)
.com	581	551	51.33	48.67
.net	237	132	64.23	35.77
.org	149	114	56.65	43.35
.nl	19,809	28,643	40.88	59.12
.mobi	0	0	0.00	0.00
.info	51	44	53.68	46.32
.nu	8	7	53.33	46.67
.se	24	226	9.60	90.40
.fi	0	0	0.00	0.00
.ca	2	6	25.00	75.00
.at	38	8	82.61	17.39
.dk	5	3	62.50	37.50

Table 7.4: CSK scheme: RSA key sizes per TLD in March 2017

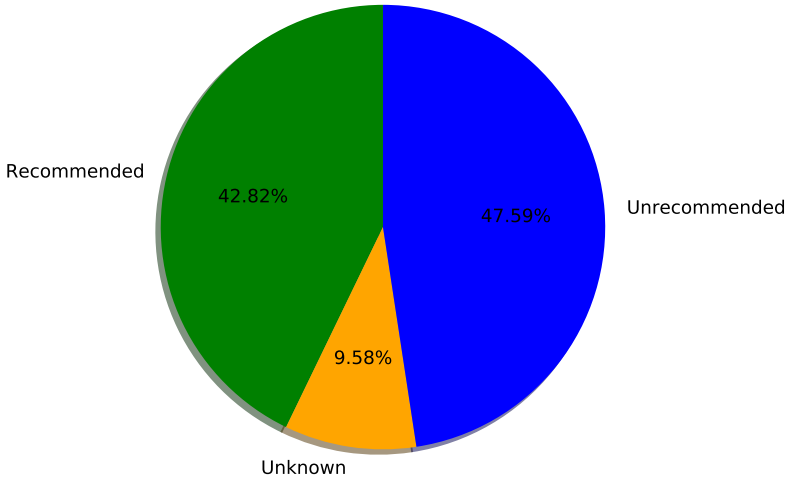
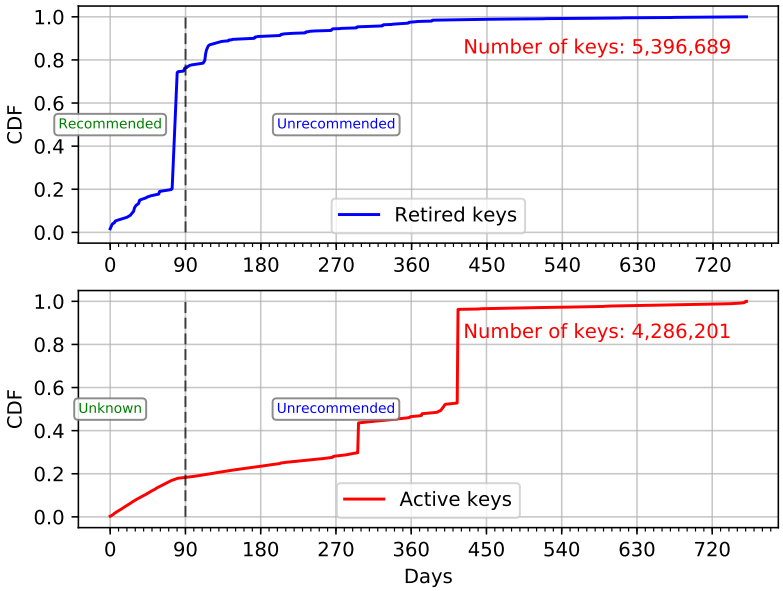


Figure 7.7: Key effective period of ZSK comparing to the best practices

Appendix A. Furthermore, it is worth mentioning that a limitation of the dataset, as discussed in Section 6.2, prevents a complete study of this policy. Specifically, RSA 2048-bit keys and ECDSA keys are suggested to be used up to two years. However, the dataset used in this thesis measures DNS for more than two years (25 months) for only three TLDs: *.com*, *.net*, and *.org*; hence, the current data may not be sufficient to evaluate the rollover of KSK and CSK types.

7.3.1 ZSK

The pie chart in Figure 7.7 provides an overview of the key rollover status for ZSKs. As shown, around half of measured keys are not replaced on time in comparison to the best practices. It is an alerting situation since it indicates that a significant number



- Retired keys are keys that we observe a start date and an end date of their usage
- Active keys are keys that we observe a start date and have seen their usage till the latest date of our measurement

Figure 7.8: CDF of ZSK effective period with RSA 1024-bit length

of domains do not apply a secure key rollover policy to protect their zones. The *Unknown* part categorizes keys that are still active and do not exceed recommended periods.

In order to study further the key rollover for ZSK, we specifically analyze in detail RSA 1024-bit keys since this is the most common key size as shown in Section 7.1 and 7.2. Figure 7.8 provides a Cumulative Distribution Function (CDF) of retired and active keys. The dashed vertical line at 90 days represents NIST's recommendation for this key type. About 79% of retired keys meet the best practices. Furthermore, there is a sharp increase around 75 days, which may indicate a common pattern for key rollover shared between domains. We will explore this further in following paragraphs. We observe a non-trivial amount of keys (about 100,000 keys) that are only used for a short period of time (less than seven days). We investigate this phenomenon and uncover two main causes: (1) domains enabled DNSSEC for a period of time and then stopped using it (75,495 domains), this may be because of DNSSEC trials; (2) domains replace their keys in an arbitrary manner (27,322 domains). Regarding active keys, an opposite situation is observed. More than 80% of active keys have already exceeded the recommendation. This is the source of the high unrecommended rate shown in Figure 7.7. Furthermore, it is noted that a large

TLDs	Recommended (1)	Unknown (2)	Unrecommended (3)	% (1)	% (2)	% (3)
.com	2,668,420	357,573	1,034,614	65.71	8.81	25.48
.net	492,699	58,753	183,967	67.00	7.99	25.02
.org	354,003	43,011	77,519	74.60	9.06	16.34
.nl	474,734	204,675	2,764,877	13.78	5.94	80.27
.mobi	649	423	3,589	13.92	9.08	77.00
.info	71,903	20,002	24,981	61.52	17.11	21.37
.nu	7,626	10,005	96,588	6.68	8.76	84.56
.se	43,336	41,375	639,064	5.99	5.72	88.30
.fi	1,007	1,046	2,771	20.87	21.68	57.44
.ca	245	89	228	43.59	15.84	40.57
.at	257	1,981	0	11.48	88.52	0.00
.dk	78	802	0	8.86	91.14	0.00

Table 7.5: Key effective period of ZSK with RSA 1024-bit length per TLD

number of active keys have been using for 297 and 416 days. This is an artifact since those numbers indicate the time when the OpenINTEL platform started to measure for domains under .se as well as .nu and .nl respectively. A breakdown into TLDs provides further detail as shown in Table 7.5. Generally, the highlighted rows show TLDs suffering high proportions of unrecommended key rollover periods. The majority of signed domains under these TLDs may not frequently change keys. Notably, two large ccTLDs, namely *.nl* and *.se*, have serious problems with the key rollover policy in both absolute and relative numbers. Section 7.4 will discuss this issue further.

The discussion above presents the effective period of keys; however, the final goal is to study how often domains are rolling their keys; hence, this paragraph explains how we infer key rollover policies of domains. This task is challenging since there are domains that do not alter their keys regularly. Furthermore, domains, which frequently roll keys, may have jitter in key effective periods due to various reasons, such as a different number of days in months, OpenINTEL measurement frequency and key changing due to emergency or operational problems. In this thesis, we define a reliable way to recognize scheduled key rollovers for domains. This method is derived from practical tests in order to infer the key effective period of a domain. Particularly, we identify domains changing keys regularly, called *regular domains*, if they meet two conditions: (1) domains have changed keys at least three times so that we can reliably study the rollover pattern; (2) the standard deviation of keys' effective periods is less than three days, which allows us to distinguish reliably domains that replace their keys regularly. Domains are considered *irregular* if they meet the first condition and miss the second one, others are classified as *Unknown*.

Table 7.6 depicts the result, in which a small number of domains that have regular

	Number of domains
Regular	359,841
Irregular	290,699
Unknown	4,734,799

Table 7.6: Regulation of key rollover for ZSK with RSA 1024-bit length

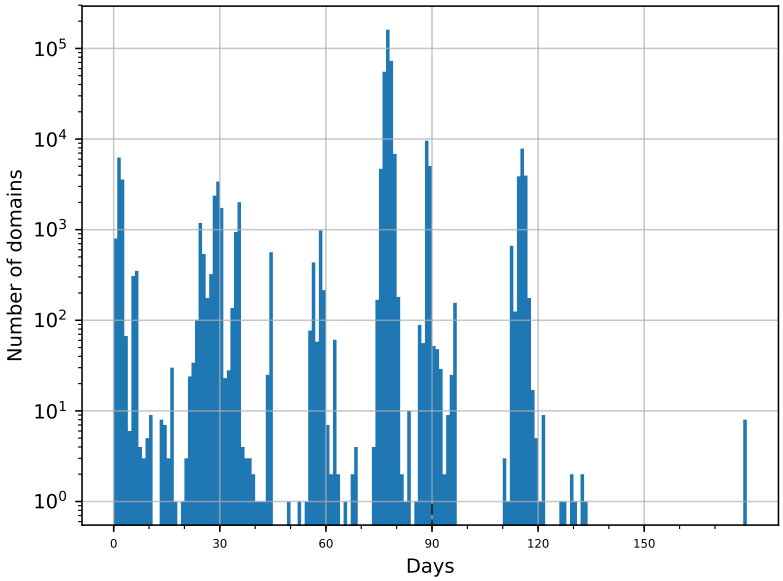


Figure 7.9: Histogram of key effective period of regular domains for ZSK with RSA 1024-bit length

key rollover policies are uncovered. For regular domains, we infer their key rollover policies by applying the mean of all keys' periods. Figure 7.9 shows common practices that regular domains apply. As seen from the figure, various peaks indicate common key rollover policies. The most popular key effective period is about 75 days, which is in line with the observation in Figure 7.8. In addition, Figure 7.10 provides a breakdown per TLD; however, information for *.fi*, *.ca*, *.at* and *.dk* is omitted due to a small number of data points. Generally, regular domains tend to roll keys in the recommended manner.

A similar study was also conducted for ECDSA keys since their use is expected to grow in the near future. However, due to the limitations of the dataset and a small number of keys, the result is not significant; hence, details related to this analysis can be found in Appendix A.

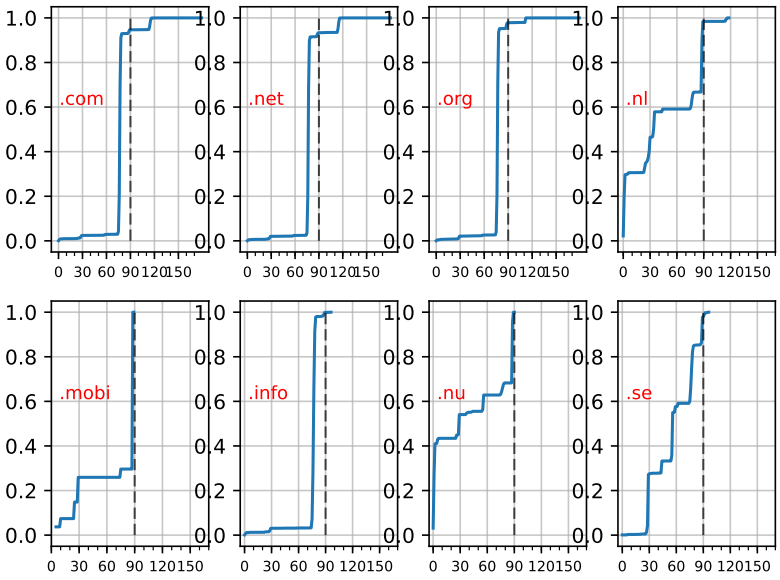


Figure 7.10: Key effective period of regular domains per TLD for ZSK with RSA 1024-bit length

7.3.2 KSK

The pie chart in Figure 7.11 provides an overview of the key rollover status for KSKs. First, we note that the *Unknown* part forms the largest proportion in the graph, which generally illustrates the effect of insufficient measurement data. Similar to ZSKs, we observe that a large percentage of keys do not meet best practices. Further investigation of this problem shows that the use of a weak key algorithm (RSA with SHA1) and unrecommended key sizes for RSA are the primary cause of this issue.

Next, we analyzed the key rollover policy for a recommended and common key usage for KSK, namely RSA keys with the length equal or greater than 2048 bits. Figure 7.12 shows that most of the domains using suitable key sizes for RSA roll their keys frequently and meet best practices. However, we note that about 12% of retired keys (49,531) is changed within a week, which is unusually short. A study conducted to explore this observation reveals three causes: (1) domains enabled DNSSEC for a period and then stopped using it (about 67%), this may be because of DNSSEC trials, similar to what we observe in the previous section; (2) domains change their keys in an arbitrary manner (about 12%) and (3) domains indeed roll their keys in a short period (about 21%). To study the surprising observation that many domains roll KSK keys in such a short time, we apply the same method as defined in Section 7.3.1 to identify regular, irregular and unknown domains as summarized in Table 7.7. As

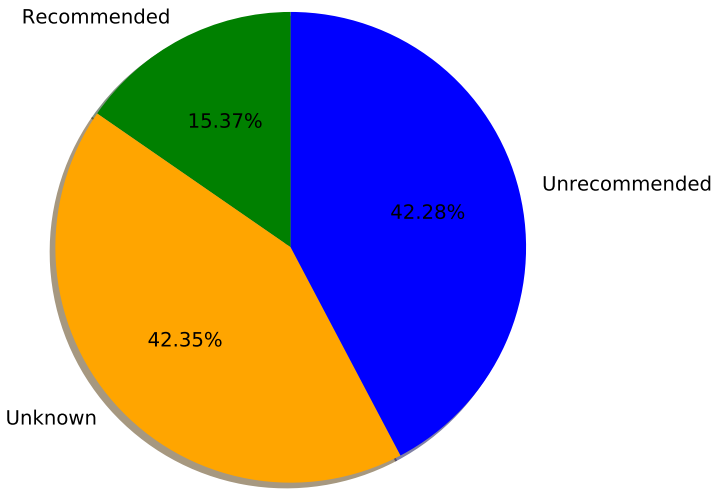


Figure 7.11: Key effective period of KSK comparing to the best practices

	Number of domains
Regular	11,105
Irregular	6,442
Unknown	1,470,914

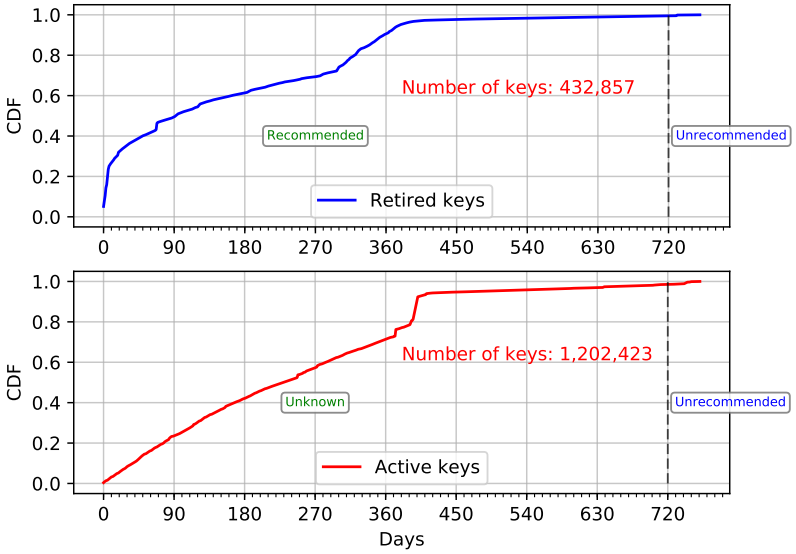
Table 7.7: Regulation of key rollover for KSK with RSA equal or greater than 2048-bit length

can be seen in Figure 7.13, the histogram shows rollover patterns for identified regular domains. As expected, there are a large number of domains changing keys within a week. This is caused mainly by a single DNS operator, namely **.sitebytes.nl*, who has replaced keys in a short time period. However, they have recently changed the policy to use keys for a longer time. Furthermore, an analysis for each TLD confirms that all regular domains of TLDs meet the best practices.

A similar study was also conducted for ECDSA keys since their use is expected to grow in the near future. However, due to the limitation of the data set and a small number of keys, the result is not significant; hence, details related to this analysis can be found in Appendix A.

7.4 Key Rollover Approaches

This section presents our analysis of the key rollover approaches for ZSK and KSK. While a similar study is conducted for the CSK scheme, its results are placed in Appendix A due to insufficient data and a small number of domains using this scheme.



- Retired keys are keys that we observe a start date and an end date of their usage
- Active keys are keys that we observe being introduced and are still active on the last day for which we have data

Figure 7.12: CDF of KSK effective period of RSA keys with key length equal or greater than 2048 bits

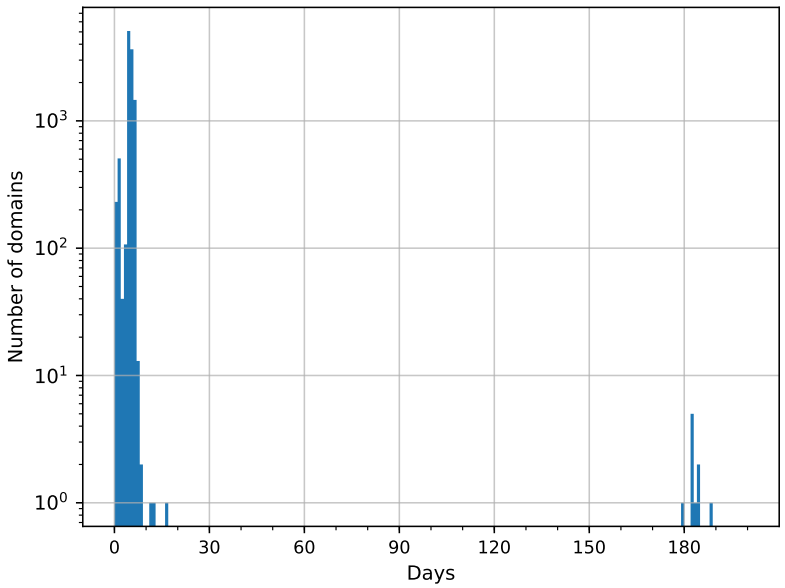


Figure 7.13: Histogram of key effective period of regular domains for RSA KSKs with key length equal or greater than 2048 bits

ZSK

Table 7.8 presents the inferred key rollover approaches for all TLDs over the entire monitoring period. Interestingly, the *no ZSK rollover* rates are critically high in

TLDs	No ZSK rollover	Abrupt	Double Signature	Double Signature with Abrupt	Pre-Publish	Pre-Publish with abrupt
com	418,812	4,837	83,788	6,832	449,231	14,107
net	55,883	901	17,137	1,574	79,955	2,354
org	43,448	529	11,294	110	57,077	1,463
nl	2,248,598	164,119	14,532	1,129	77,309	3,673
mobi	2,415	3	560	5	237	5
info	17,962	76	3,972	9	22,016	198
nu	82,415	526	1,426	106	1,189	24
se	557,639	1,931	18,925	1,352	9,597	128
fi	2,728	4	837	4	398	6
ca	202	9	18	5	126	18
at	6	23	54	25	1,643	19
dk	2,442	1	8,423	0	219	2

* No ZSK rollover means that a domain has used only one key and that key has already exceeded the key effective period recommended in the best practices. We only account for active DNSSEC domains to avoid biases due to DNSSEC trial domains

** Abrupt means that a domain replaces keys in a sudden way, which may cause validation failures during rollovers

Table 7.8: ZSK key rollover approach analysis

most of the TLDs. We observe that about 75% of all domains perform zero ZSK rollovers, in which domains under TLDs *.nl*, *.se* and *.com* are the main contributors. About 3% of domains may suffer failures during the rollover due to abrupt operations. Furthermore, we observe that both pre-publish and double-signature schemes are applied in practice. However, the former one is much more popular. In comparison to the best practice, about 12% of domains meet the recommendation.

KSK

Table 7.9 shows the analysis for KSK rollovers for all TLDs over the entire monitoring period. This result should be treated with caution since we only have sufficient data to analyze *.com*, *.org* and *.net*. Hence, it may be unreliable to make any comparison between the rollover of ZSK and KSK. Generally, about 30% of domains have never rolled their keys, and more than 3% of domains may suffer validation failures during rollovers. Furthermore, the recommended key rollover scheme, the double-signature scheme, is more popular than the double-DS scheme, about 6.7% versus 0.08% respectively.

7.5 Key Algorithm Rollover

Table 7.10 presents the results of our analysis for the rollover of key algorithms for all TLDs. It can be easily identified that switching to the elliptic curve algorithm,

TLDs	No KSK rollover	Abrupt	Double Signature	Double Signature with abrupt	Double DS	Double DS with abrupt
com	529,472	5,996	230,130	936	3,381	364
net	80,910	883	48,316	181	494	75
org	66,103	498	24,211	79	374	62
nl	590,850	166,173	32,206	2,822	16	812
mobi	0	0	0	0	0	0
info	22,528	78	5,775	14	51	12
nu	21,045	256	554	12	1	4
se	211,503	2,264	5,430	43	19	47
fi	399	8	9	0	1	4
ca	54	3	10	0	1	1
at	1,611	4	137	1	0	0
dk	2,406	1	21	1	21	0

* No KSK rollover means that a domain has used only one key and that key has already exceeded the key effective period recommended in the best practices. We only account for active DNSSEC domains to avoid biases due to DNSSEC trial domains

** Abrupt means that a domain replaces keys in a sudden way, which may cause validation failures during rollovers

Table 7.9: KSK key rollover analysis

From_to algorithm	Number of changes	Percent	Conservative approach		
			Success	Unidentified	Fail
8 to 13	118,751	76.77	1	114,860	3,890
8 to 7	18,109	11.71	10	6,061	12,038
7 to 8	11,513	7.44	8	4,669	6,836
5 to 8	1,906	1.23	1	282	1,623
7 to 13	1,196	0.77	1	172	1,023
Others	3,215	2.08	16	447	2,752

Table 7.10: Key algorithm rollover analysis

namely algorithm 13, is the dominant trend accounting for more 76% of all algorithm changes. Furthermore, rolling to a more secure algorithm is also observed. However, there is an observable phenomenon of backward rolling, from algorithm 8 to algorithm 7. The primary cause of this is the change of DNS operators; that means domain owners switch from one DNS operator to another. This may indicate that domain owners may not be aware of DNSSEC and may not consider it when switching DNS operators; hence, the failure rate of this rollover is the highest. Moreover, there are a significant number of unidentified cases which is due to limitations of the daily measurement dataset. In particular, the conservative approach, as presented in Section 4.1.5, requires signatures to be published for at least the maximum TTL of zone data prior to the appearance of keys in order to avoid downgrade attacks. If domains have TTLs less than one day, the required publish procedure can occur within a day. Hence, the daily dataset will miss capturing these operations.

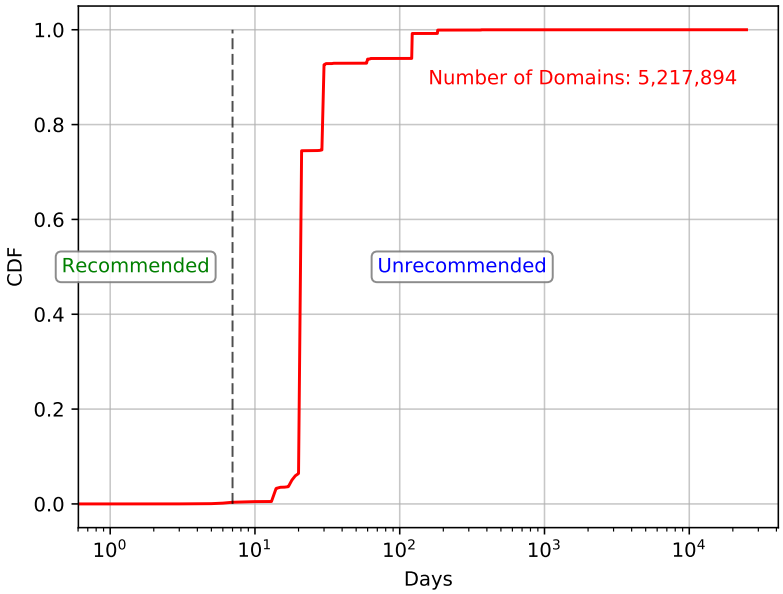


Figure 7.14: CDF of validity periods for signatures covering DNSKEY RRSets

7.6 Signature Validity Period

This section presents the analysis of the validity period of signatures covering DNSKEY RRsets. NIST recommends a 7-day validity period for this RRSIG type to minimize the impact of a key compromise incident. However, this recommendation is quite controversial since a short validity period for RRSIGs may interfere with the operational stability of a DNS operator. In that sense, Figure 7.14 illustrates that only a trivial amount of domains, about 0.003%, meet this recommendation. Instead, there are clearly three common practices for the validity period of a signature covering a DNSKEY RRset, namely 21, 30 and 122 days as shown in Figure 7.14. Furthermore, we observe the same situation across TLDs, in which only a handful of domains meet the recommendation. Figure 7.15 illustrates this observation and shows the validity period for RRSIGs of DNSKEY RRsets in *.com*. The observed distribution is also representative of other TLDs. In addition, we conduct a similar study for signatures covering SOA RRsets and conclude that most of the domains (over 98%) do not implement different validity periods for signatures of different RRsets.

7.7 Quality of Signed Zones in *.nl* and *.se*

As discussed in Chapter 5, large DNS operators (in terms of the number of domains) may receive more benefits from economic incentives than small DNS operators, since

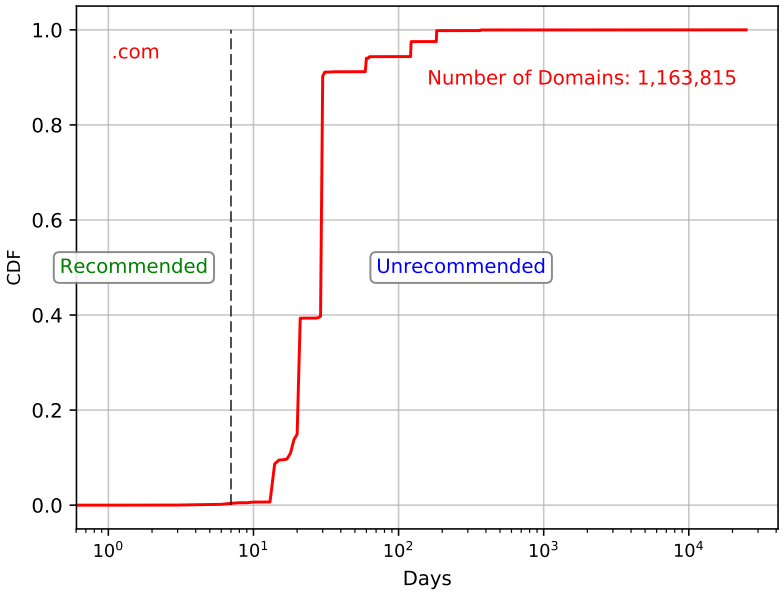


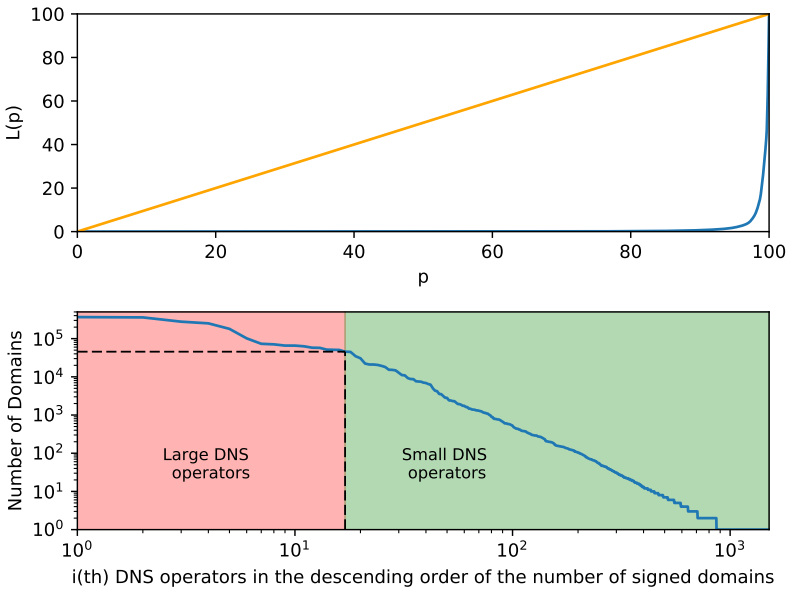
Figure 7.15: CDF of validity periods for signatures covering DNSKEY RRSets in *.com*

the discount, which they receive for each signed zone, may outweigh the cost of DNSSEC deployment and maintenance. On the other hand, it is suspected that small DNS operators may deploy DNSSEC for other motivations rather than the economic incentive such as market competitive advantages and security considerations. With respect to this concern, we aim to study the differences of quality of signed zones in *.nl* and *.se* between small and large DNS operators in this section.

Firstly, we distinguish between large and small DNS operators in TLDs *.nl* and *.se*. Next, we evaluate differences in quality of signed zones managing by these two groups of DNS operators. Results of this section offer answers for hypotheses H2.1, H2.2 and H2.3, and hence, answer to the second research question.

7.7.1 Classification of Large and Small DNS Operators for *.nl* and *.se*

As presented in Section 6.1.3, we apply the Pareto principle to classify large and small DNS operators for the *.nl* and *.se* ccTLDs. Figure 7.16 and 7.17 illustrate our classification. The top row of each figure presents Lorenz curves [44] which describe the p percent of DNS operator population that are managing the $L(p)$ percent of DNSSEC signed domains. The diagonal line presents the equality state in which each DNS operator is responsible for the same amount of signed domains. The large gaps



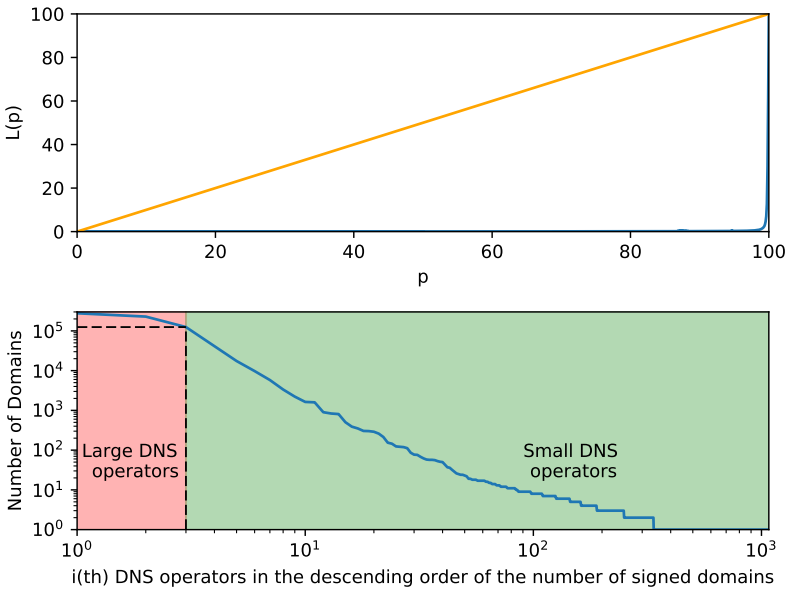
- The top: the Lorenz curve describes the p percent of the DNS operator population that manage the $L(p)$ percent of DNSSEC signed domains. About 0.01% of DNS operators manage more than 81% of signed domains.
- The bottom: the line presents the number of signed domains of each DNS operator sorted in the descending order.

Figure 7.16: Classification of Large and small DNS operators for *.nl*

between two Lorenz curves and equality lines indicate a significant inequality in the distribution of domains per DNS operator. In that light, only a small number of DNS operators accounts for most of the signed domains. The second row of each figure illustrates our clear-cut classification between large and small DNS operators. Although it is a subjective choice, we decided to distinguish large DNS operators as those that are responsible for more 80% of signed domains in *.nl* and *.se*, others are classified as small DNS operators. The full list of large DNS operators is presented in Table 7.11.

7.7.2 Comparison of Key Algorithm Implementation

Table 7.12 illustrates the comparison of key algorithm implementation between large and small DNS operators for domains under *.nl* and *.se* based on the latest snapshot of data in March 2017. As shown in the "total" lines, small DNS operators perform slightly better than large DNS operators, with 87.1% and 81.97% of domains implementing recommended key algorithms respectively. Although large DNS operators in *.se* perform very well, about 23% of all domains managed by large DNS operators in *.nl* are implemented with weak key algorithms. The high percentage of unrecom-



- The top: the Lorenz curve describes the p percentage of the DNS operator population that manage the $L(p)$ percentage of DNSSEC signed domains. About 0.003% of DNS operators manage more than 87% of signed domains.
- The bottom: the line presents the number of signed domains of each DNS operator sorted in the descending order.

Figure 7.17: Classification of Large and small DNS operators for *.se*

mended key algorithms is primarily caused by a single DNS operator, namely TransIP. This operator employs Algorithm 7 in three of its sub-platforms: **.transip.net*, **.transip.nl* and *ns0.nl*. Since small DNS operators are performing better than large ones, we conclude that Hypothesis H2.1 holds.

7.7.3 Comparison of Key Size Implementation

This section discusses the usage of RSA key size based on the latest snapshot in March 2017. Table 7.13 shows the comparison of ZSK and KSK key size between two groups of DNS operators. With regards to the ZSK key size, all DNS operators perform well, most of the zones are signed with suitable key lengths. On the other hand, large DNS operators adopt weak key lengths for KSKs more often than small DNS operators, especially in *.se*. A large DNS operator, namely *one.com*, is identified as the cause of this problem. Particularly, they adopt different key lengths for RSA keys, namely 1536 (unrecommended) and 2048 (recommended) bits. Since large DNS operators fail more often than small DNS operators at implementing NIST's key size recommendations for domains under *.nl* and *.se*, we conclude that Hypothesis H2.2 holds.

DNS operators	Name server platform	Number of signed domains
TLD .nl		
Hostnet BV Network	*.hostnet.nl.	366,787
Metaregistrar BV	*.metaregistrar.nl.	359,567
TransIP	*.transip.net.	277,869
	*.transip.nl.	180,220
	*.sonexo.eu.	73,273
	ns0.nl.	50,562
Cyso Hosting	*.firstfind.nl.	251,263
Argeweb BV	*.argewebhosting.eu.	101,724
Openprovider	*.openprovider.nl.	70,832
Village Media BV	*.webhostingserver.nl.	65,740
Hosting2GO	*.hosting2go.nl.	65,706
Flexwebhosting BV	*.flexwebhosting.nl.	62,990
One.com	*.one.com.	57,869
KPN Internedservices	*.is.nl.	57,362
Neostrada	*.neostrada.nl.	51,505
PCextreme	*.pcextreme.nl.	50,902
AXC B.V.	*.axc.nl.	45,531
TLD .se		
Loopia AB	*.loopia.se.	277,798
One.com	*.one.com.	229,075
Binero AB	*.binero.se.	124,825

Table 7.11: Large DNS operators in TLDs *.nl* and *.se*

Type	TLD	Recommended (1)	Unrecommended (2)	% (1)	% (2)
Large DNS operators	Total	2,312,725	508,675	81.97%	18.03%
	.se	631,698	0	100.00%	0.00%
	.nl	1,681,027	508,675	76.77%	23.23%
Small DNS operators	Total	525,405	77,789	87.10%	12.90%
	.se	85,781	6,817	92.64%	7.36%
	.nl	439,624	70,972	86.10%	13.90%

Table 7.12: Comparison of key algorithm implementation between large and small DNS operators in TLDs *.nl* and *.se*. Values are measured by the number of domains

7.7.4 Comparison of ZSK Rollover Implementation

In this section, we compare the ZSK key rollover implementations of large and small DNS operators. We did not consider evaluating the rollover for KSKs and CSKs due to the lack of sufficient data (the period over which we have data is too short). Table 7.14 shows the percentages of the numbers of keys in *recommended*, *unrecommended* and *unknown* categories. The *unknown* category indicates keys that are still in use and have not yet exceeded recommendations in the best practices. As can be seen, small DNS operators perform significantly better than large DNS operators in both *.se* and *.nl*. Table 7.15 analyses two root causes of the high percentages of the unrecommended

Type	TLD	Recommended (1)	Unrecommended (2)	% (1)	% (2)
ZSK RSA key size					
Large DNS operators	Total	2,821,360	0	100.00%	0.00%
	.se	631,695	0	100.00%	0.00%
	.nl	2,189,665	0	100.00%	0.00%
Small DNS operators	Total	546,957	1,748	99.68%	0.32%
	.se	90,429	1,725	98.13%	1.87%
	.nl	456,528	23	99.99%	0.01%
KSK RSA key size					
Large DNS operators	Total	2,596,101	158,646	94.24%	5.76%
	.se	500,453	102,078	83.06%	16.94%
	.nl	2,095,648	56,568	97.37%	2.63%
Small DNS operators	Total	459,634	6,157	98.68%	1.32%
	.se	67,807	3,246	95.43%	4.57%
	.nl	391,827	2,911	99.26%	0.74%

Table 7.13: Comparison of key size implementation between large and small DNS operators in TLDs *.nl* and *.se*. Values are measured by the number of domains

Type	TLD	% Recommended	% Unrecommended	% Unknown
Large DNS operators	Total	4.07%	90.21%	5.73%
	.se	1.20%	89.88%	8.93%
	.nl	4.73%	90.28%	4.99%
Small DNS operators	Total	47.20%	42.42%	10.37%
	.se	36.26%	48.35%	15.39%
	.nl	49.22%	41.33%	9.45%

Table 7.14: Comparison of ZSK rollover implementation between large and small DNS operators in TLDs *.nl* and *.se*. Values are measured by the number of keys

category, namely late key replacement or intermittent DNSSEC operation and no key replacement. While the former cause is responsible for less than 1% of signed domains, the latter one accounts for more 90% of signed domains managing by large DNS operators. This shows that not performing key rollovers is the biggest problem in DNSSEC. This may be a side effect of economic incentives. Since the key rollover process is quite complex and is not required in order to be eligible for economic incentives, large DNS operators may choose to avoid extra effort in performing the regular key rollovers. Furthermore, we observe this behavior consistently over all large DNS operators in *.se* and *.nl*. In conclusion, Hypothesis H2.3 holds and therefore, we conclude the answer to the second research question that small DNS operators deploy DNSSEC with higher quality than large DNS operators for domains under TLDs *.nl* and *.se*.

Type	TLD	% Late key replacement or intermittent DNSSEC operation	% No key replacement
Large DNS operators	Total	0.72%	91.26%
	.se	0.21%	90.09%
	.nl	0.84%	91.54%
Small DNS operators	Total	4.07%	62.89%
	.se	10.55%	55.57%
	.nl	2.69%	64.43%

Table 7.15: Root cause analysis of high percentages of unrecommended key effective periods. Values are measured by the number of domains

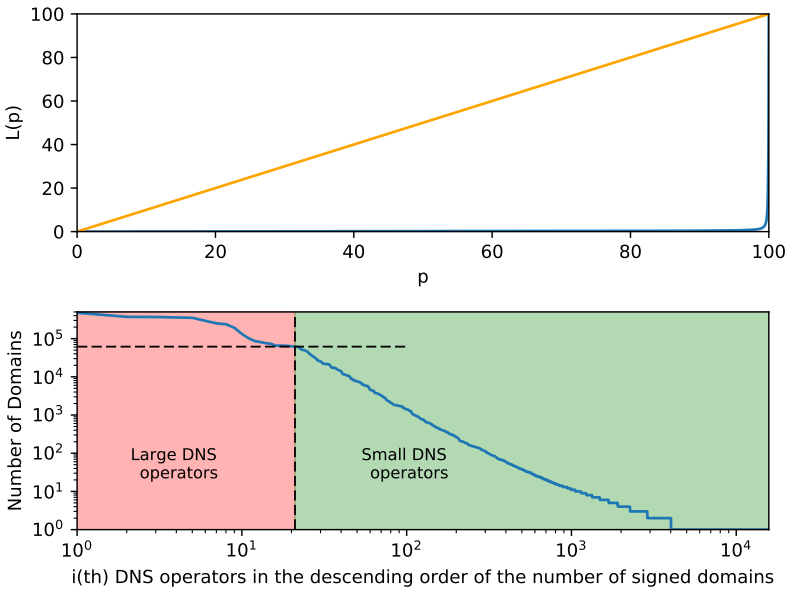
7.8 The Full Deployment of DNSSEC

As discussed in the third research question in Chapter 5, we suspect that large DNS operators may maintain DS records for domains under .nl and .se to earn incentives and do not deploy DS records for domains under other TLDs where there are no economic incentives applied. Hence, in this section, we evaluate the completeness of DNSSEC deployment between large and small DNS operators, measured by the number of signed domains in all TLDs.

Firstly, we apply a similar approach as presented in the previous section to distinguish between large and small DNS operators measured in all TLDs. Figure 7.18 presents our classification, in which a minor number, the top 21, of DNS operators is responsible for more than 80% of signed domains. Those are identified as large DNS operators as shown in Table 7.16.

Secondly, Figure 7.19 illustrates the performance comparison between two groups of DNS operators. As observed in Section 7.2, no domains under *.mobi* have DS records; hence, it is removed from the figure. Furthermore, because of a small number of domains in *.at*, *.dk* and *.fi*, information about those TLDs are omitted due to a limited number of signed zones which does not affect the outcome of this analysis. A general conclusion can be derived from Figure 7.19 that large DNS operators have been maintaining DS records significantly better than small DNS operators.

Small DNS operators have allowed relatively high proportions of miss DS records in TLDs *.com*, *.net*, *.org*, *.info* and *.nu*, more than 50% of signed domains missing DS records. Interestingly, we observe a strange "zigzag" pattern shared between TLDs *.com*, *.net*, *.org* and *.info*. This anomaly is caused by a single DNS operator, namely **.domainmonster.com* which does not maintain DS records and provides an intermittent DNSSEC service. The irregular shapes are the results of the various amount of signed zones being observed daily. With regards to the largest DNSSEC



- The top: the Lorenz curve describes the p percent of the DNS operator population that manage the $L(p)$ percent of DNSSEC signed domains. About 0.001% of DNS operators manage more than 80% of signed domains.
- The bottom: the line presents the number of signed domains of each DNS operator sorted in the descending order.

Figure 7.18: Classification of Large and small DNS operators for all TLDs

DNS operators	**Name server platform**	**Number of signed domains**
Loopia AB	*.loopia.se.	476,845
TransIP	*.transip.net.	371,992
	*.transip.nl.	238,467
	*.sonexo.eu.	73,323
	ns0.nl.	64,603
Hostnet BV Network	*.hostnet.nl.	368,357
Metaregistrar BV	*.metaregistrar.nl.	359,567
OVH	*.ovh.net.	350,717
	*.anycast.me.	61,712
One.com	*.one.com.	291,235
Cyso Hosting	*.firstfind.nl.	251,265
Binero AB	*.binero.se.	192,471
Hyp.net	*.hyp.net.	132,739
Argeweb BV	*.argewebhosting.eu.	101,725
OpenProvider	*.openprovider.nl.	86,268
Village Media BV	*.webhostingserver.nl.	80,761
Internetdservices	*.is.nl.	76,090
Pcextreme	*.pcextreme.nl.	65,865
Hosting2GO	*.hosting2go.nl.	65,706
City Network	*.citynetwork.se.	65,123
Flexwebhosting BV	*.flexwebhosting.nl.	62,990

Table 7.16: Large DNS operators in all TLDs

zone, *.nl*, there is a slight increase in the DS missing record rate with some notable spikes and drops. Particularly, the sudden increase on July 7, 2016 is caused by **.netground.nl* who newly signed 14,521 domains without DS records. Then they started to publish all DS records for signed domains on July 26, 2017. Another drop is observed a few days later on July 31, 2016 due to a significant decrease in the number of signed zones in the operator **.eatserver.nl*, who also does not maintain any DS records. Furthermore, the last notable increase happened on January 25, 2017 when **.oxilion.nl* suddenly stopped publishing DS records and only maintained secure delegations for 14 out of 20,599 of the domains they manage.

Large DNS operators have low missing DS rates, especially to TLDs *.nl* and *.se* (about 2% and 5% respectively). This may be an effect of economic incentives since both registries of *.nl* and *.se* require signed zones to be validatable in order to receive discounts.

In conclusion, since large DNS operators perform better than small DNS operators in maintaining DS records, Hypothesis H3 does not hold. Hence, the answer to the third research question is also negative. The result may be explained by the fact that large operators are often both DNS operators and registrars. This means that they control the full channel for DNSSEC, both the DNS operation itself and the communication with the parent zone operators (registries) to register DS records. Small operators, on the other hand, are likely only DNS operators, but not registrars. This means that in order to register a DS record, they need to go through their registrars and the *Registry-Registrar-Registrant* channel. This implies an extra step in the process and also means that their registrars must support registering DS records (which not all registrars do).

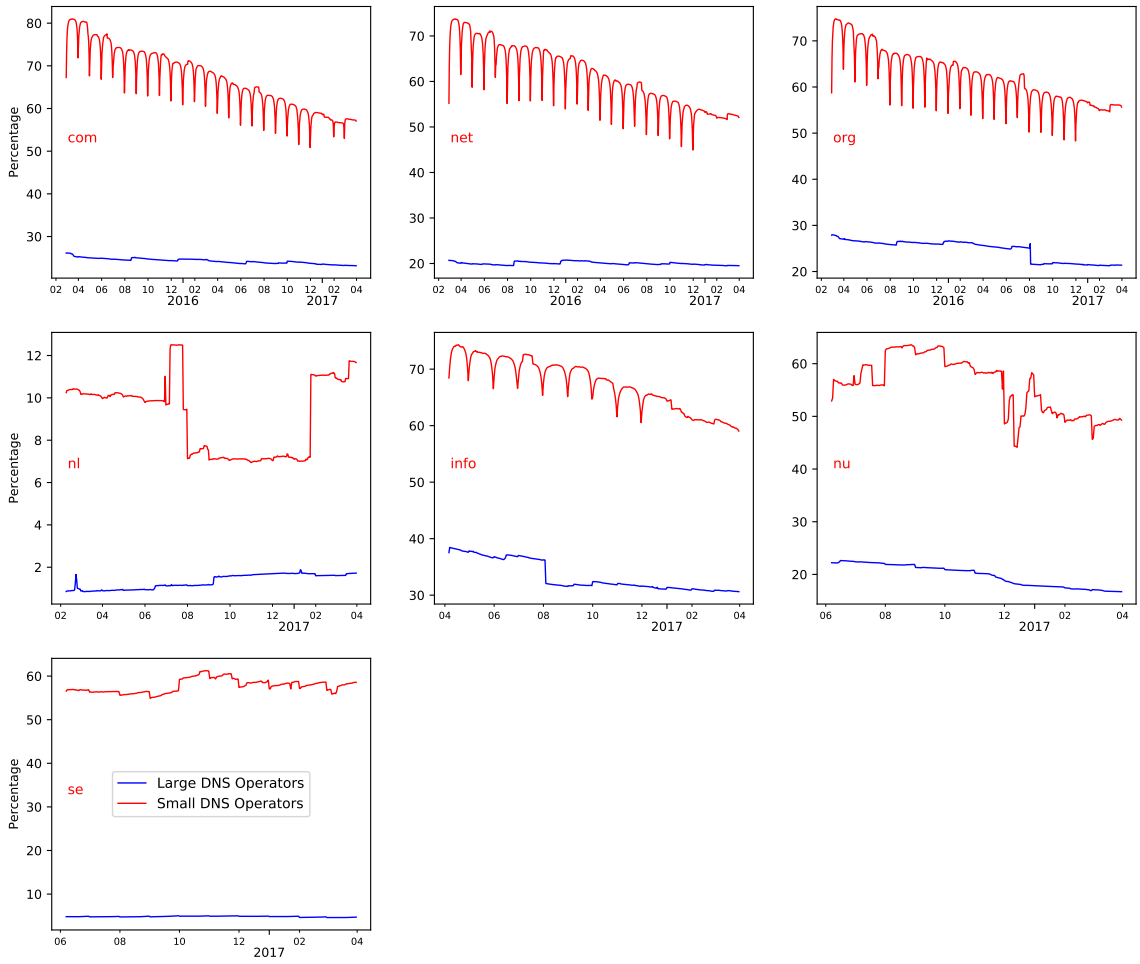


Figure 7.19: Comparison of missing DS records between large and small DNS operators in TLDs. The y-axis in each sub-figure presents the percentage of domains missing DS records in TLDs. The strange "zigzag" pattern shared between TLDs .com, .net, .org and .info is caused by a single DNS operator, namely **.domainmonster.com* which does not maintain DS records and provides an intermittent DNSSEC service.

Chapter 8

Discussion

This chapter discusses our findings and an actionable plan to improve the quality of DNSSEC deployment.

Chapter 7 reveals a poor picture of the current DNSSEC deployment. Particularly, about a quarter of signed domains adopts weak key algorithms (mainly RSA-SHA1) and more than 14% of domains sign their zones with inappropriate key sizes for KSKs. Critically, more than 75% of domains have never rolled their ZSKs. This proportion gets even higher when considering only domains under *.nl* and *.se*, where there are economic incentives applied. Furthermore, by comparing the differences in quality of the DNSSEC deployment between large and small DNS operators, we conclude that while economic incentives do help to increase the number of signed zones, they do not motivate DNS operators to improve the quality of their implementations. In that sense, we suggest that registries for *.nl* and *.se* should apply additional requirements on how DNSSEC should be deployed in their incentive programs.

We contacted the registry of *.se*, namely *Internetstiftelsen i Sverig* (IIS), to seek external feedback on our findings and suggestions. From the beginning of the bonus program in 2013, the primary goal of IIS was the number of signed zones, i.e. quantity; hence, they have not had any plan to encourage increased quality of DNSSEC deployment. However, they do have some internal discussions about quality in the sense of how DNSSEC should be deployed in general. Our discussion with IIS highlighted several aspects of DNSSEC deployment that is in this same direction, and the registry agreed that further incentivizing quality is a long term goal. Furthermore, IIS shared that they may use their bonus program as a tool to spur on quality by launching two DNSSEC bonus levels: (1) as currently and (2) with higher rewards and stricter requirements to improve the quality of signed domains.

"We also reached out to SIDN, the operator for the *.nl* ccTLD. They indicated that they will use the results of the research discussed in this report as input for

discussions on the economic incentive when it comes up for renewal in 2018."[1]

Since bonus programs have shown their effectiveness in encouraging the growth of DNSSEC, we suggest an actionable plan based on incentive programs to improve the current security status in DNSSEC. Specifically, key rollover is the most critical problem as discussed in Chapter 7; hence, addressing this issue will significantly enhance the deployment of DNSSEC. However, it is challenging for registries, such as IIS, to measure key rollover for domains under their management. On the other hand, it is easier for registries to incorporate the key size and key algorithm requirements in their bonus programs; therefore, we suggest registries to include good guides for those two policies as a mandatory part of their incentives. As a result, keys with better key algorithms and key sizes can be used safely for a longer time, which will essentially address the current key rollover issue. However, for a long-term plan, the key rollover should be a part of incentive programs. One possible way is to utilize the dataset used in this thesis since we have already established the methodology to perform the measurement.

[1]This is the text that SIDN allows us to include in this report.

Chapter 9

Conclusion and Future Work

In this chapter, we summarize our answers to the three research questions discussed in Chapter 5. Moreover, we present our recommendations for future work. A study of these recommendations may further extend the knowledge of the current status of DNSSEC on the Internet.

9.1 Answers to Research Questions

RQ1: How are signed zones deployed in the wild compared to NIST's recommendations?

We perform our analysis to study the implementations of six DNSSEC Policies in the wild and evaluate them against the best practices identified by NIST's recommendations. Results of this work provide insights into how well DNSSEC is deployed globally. A summary of the results is presented as below:

- **Key algorithm**: about 25% of measured domains adopt weak choices for the key algorithm, in which Algorithm 7 (RSA-SHA1) is the main culprit. Furthermore, we also observe that while NIST strongly recommends ECDSA key algorithms, most of the domains are employing RSA keys and only a minority of domains (about 4%) are in line with this recommendation.

- **Key size**: most of the domains choose good key lengths for ZSKs; with respect to RSA keys, the key lengths of 1024, 1280 and 2048 bits are significantly popular and account for more 90% of domains adopting RSA keys. On the other hand, many domains make weak choices for the key length of KSKs and CSKs, about 15% and 60% respectively. The primary cause of the high percentage of the unrecommended key size is due to the utilization of 1024 and 1536-bit RSA keys.

- **Key rollover and key rollover approach**: key rollover is probably the major problem in the DNSSEC deployment. About 75% of domains have never replaced keys, and about 3% of domains may suffer validation failures during a rollover due to abrupt operations. Regarding the rollover approach, we observe that the recommended approaches (pre-publish for ZSKs and double signature for KSKs) are applied more often than the unrecommended approaches (double signature for ZSKs and double DS for KSKs).

- **Key algorithm rollover**: there is a clear trend to migrate from Algorithm 8 to Algorithm 13 (from RSA-SHA256 to ECDSA-P256-SHA256), which is highly recommended by NIST. Furthermore, there is also a trend to roll to more secure algorithms in general. However, we also observed backward rolling, from Algorithm 8 to Algorithm 7 due to changes from one DNS operator to another. This may indicate that domain owners may not be aware of DNSSEC and may not consider it when switching DNS operators.

- **Signature validity period**: most of the domains do not meet the recommendation since the 7-day validity period for RRSIGs covering DNSKEY RRsets may interrupt the operation of DNS operators. Instead, there are clearly three common alternatives for the validity period of a signature covering a DNSKEY RRset, namely 21, 30 and 122 days.

RQ2: Do small DNS operators deploy DNSSEC with higher quality than large DNS operators for domains under *.nl* and *.se*?

Since large DNS operators may receive more benefits than small DNS operators from economic incentive programs, we expect them to deploy DNSSEC for economic purposes while small DNS operators may have different motivations. In that context, we perform a comparison of the quality of DNSSEC deployment between large and small DNS operators to study the influence of economic incentives. Our results show that small DNS operators implement DNSSEC consistently with higher quality than large DNS operators. For instance, about 90% of domains managed by large DNS operators have never rolled ZSKs. This finding may be an indication of the side effect of economic incentives. In conclusion, economic incentives help to increase the quantity of DNSSEC deployment; however, they have not yet offered a similar influence regarding quality.

RQ3: Do small DNS operators deploy signed zones fully more often than large DNS operators?

This research question concerns the influence of economic incentives to the full deployment of DNSSEC. Hence, a comparison of missing DS records is conducted between large and small DNS operators. The result shows that large DNS operators maintain DS records far better than small DNS operators. We conclude that economic incentives do not have much influence on the partial deployment of DNSSEC; we speculate that the *Registry-Registrar-Registrant* channel may be the main cause for this observation.

9.2 Future Work

In the final section, we provide actions for further studies to improve our understanding of DNSSEC on the Internet.

KSK and CSK Rollover

Since KSKs and CSKs with appropriate algorithms and key sizes are recommended to be safely used up to two years, we do not have sufficient data to study KSK and CSK rollovers in the wild fully. Particularly, many TLDs, such as *.nl* and *.se*, are measured less than two years. Hence, for future work, we suggest to re-perform our analysis and concentrate on KSK and CSK rollovers on the Internet.

Influence of Economic Incentives

Our findings in Chapter 7 show that economic incentives help to increase the quantity of DNSSEC deployment; however, they have not yet offered a similar influence regarding quality. Hence, we speculate that registries, in the near future, may use economic incentives as tools to spur on the quality of DNSSEC deployment. In that context, we suggest to re-analyse the influence of economic incentives on both quantity and quality aspects of DNSSEC deployment.

DNSSEC Signing Software

The goal of this study is to investigate whether it is possible to fingerprint the DNSSEC signing software that was used to sign a domain. Each signing software may be delivered with a set of default policies or distinct characteristics that may be used

as signatures to fingerprint a signing software. The study will provide a clear distribution of signing software employed in the wild. This result may direct researchers to focus on investigating and measuring a small set of the most popular DNSSEC signing software and hence contribute to improving the security status of DNSSEC.

DNSSEC Validation

The goal of this study is to evaluate the correctness status of DNSSEC deployments. That means that domains should not only publish all DNSSEC-related records but also maintain them correctly for the DNSSEC validation process to be successful. A longitudinal study on this aspect will provide a clearer view of the DNSSEC deployment and its contribution to improving the DNS security.

Appendix A

DNSSEC Policies

A.1 Key Algorithm

Recommendation	**Number of Domains**
Unrecommended	1,205,822
Recommended	3,305,949
Highly Recommended	179,551

Table A.1: KSK-ZSK key algorithm in March 2017

Recommendation	**Number of Domains**
Unrecommended	7,575
Recommended	43,062
Highly Recommended	19,070

Table A.2: CSK key algorithm in March 2017

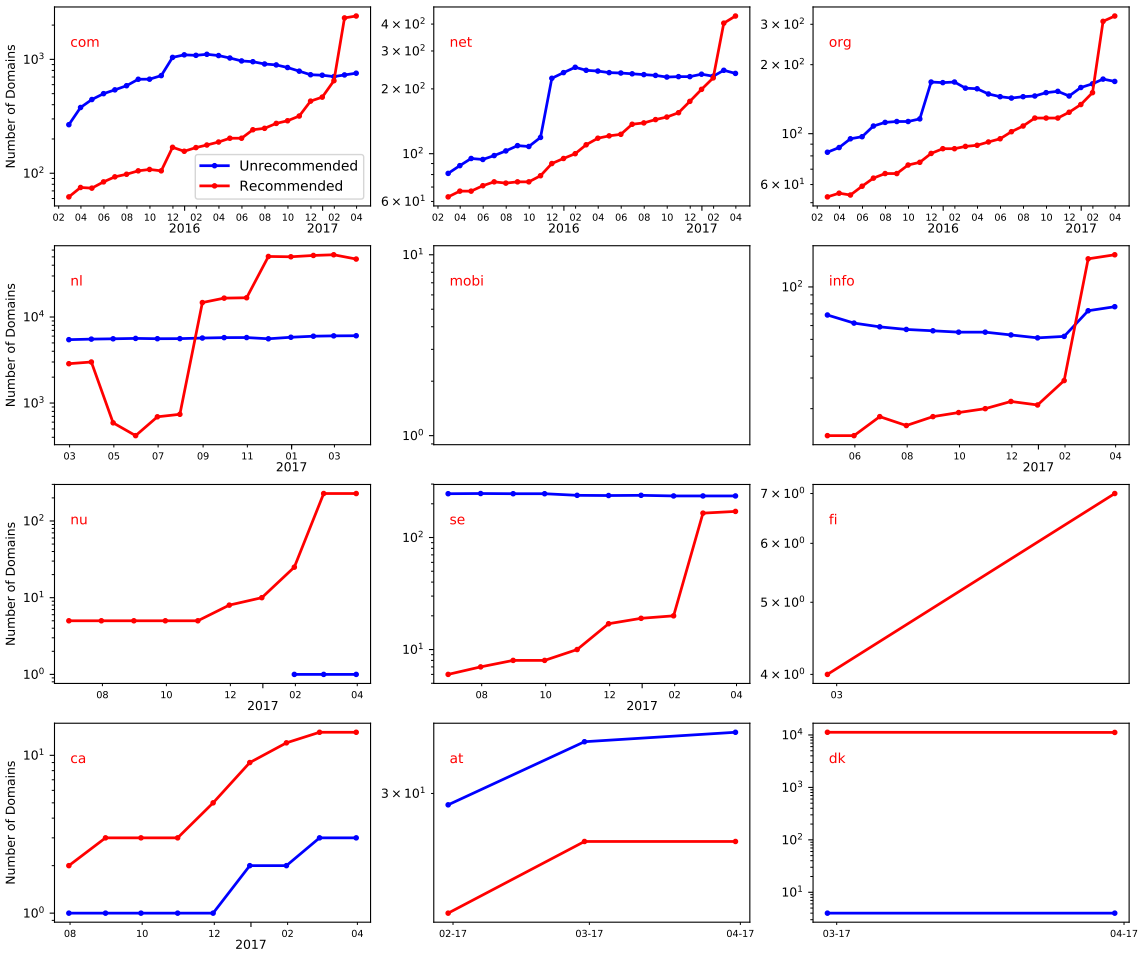


Figure A.1: CSK key algorithm per TLD over measurement periods

A.2 Key Size

TLDs	Recommended (1)	Unrecommended (2)	% (1)	% (2)
.com	796,101	356	99.96	0.04
.net	118,354	132	99.89	0.11
.org	88,972	107	99.88	0.12
.nl	2,640,237	22	100.00	0.00
.mobi	2,896	1	99.97	0.03
.info	38,907	11	99.97	0.03
.nu	97,243	42	99.96	0.04
.se	704,353	1,723	99.76	0.24
.fi	4,113	5	99.88	0.12
.ca	475	25	95.00	5.00
.at	5,780	1	99.98	0.02
.dk	7,119	2,449	74.40	25.60

Table A.3: ZSK: RSA key sizes per TLD in March 2017

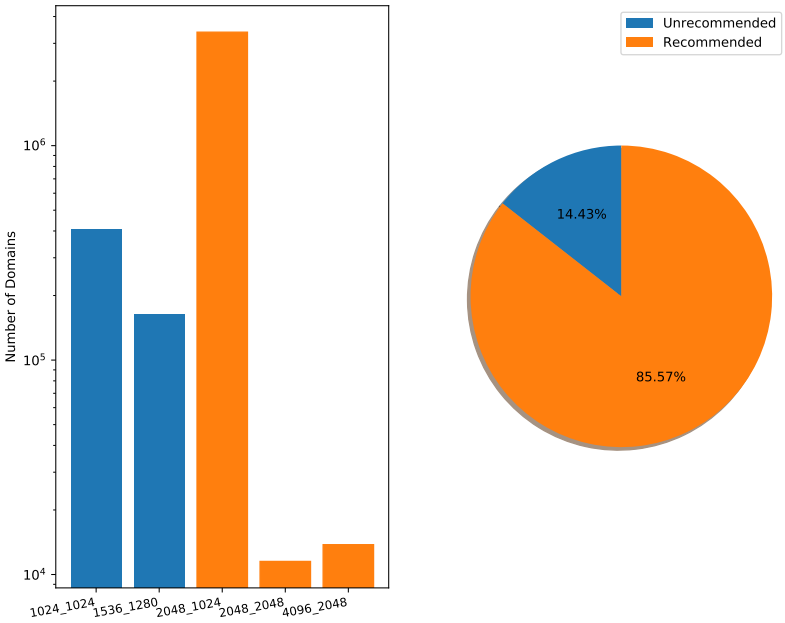


Figure A.2: KSK-ZSK: key size usage in March 2017

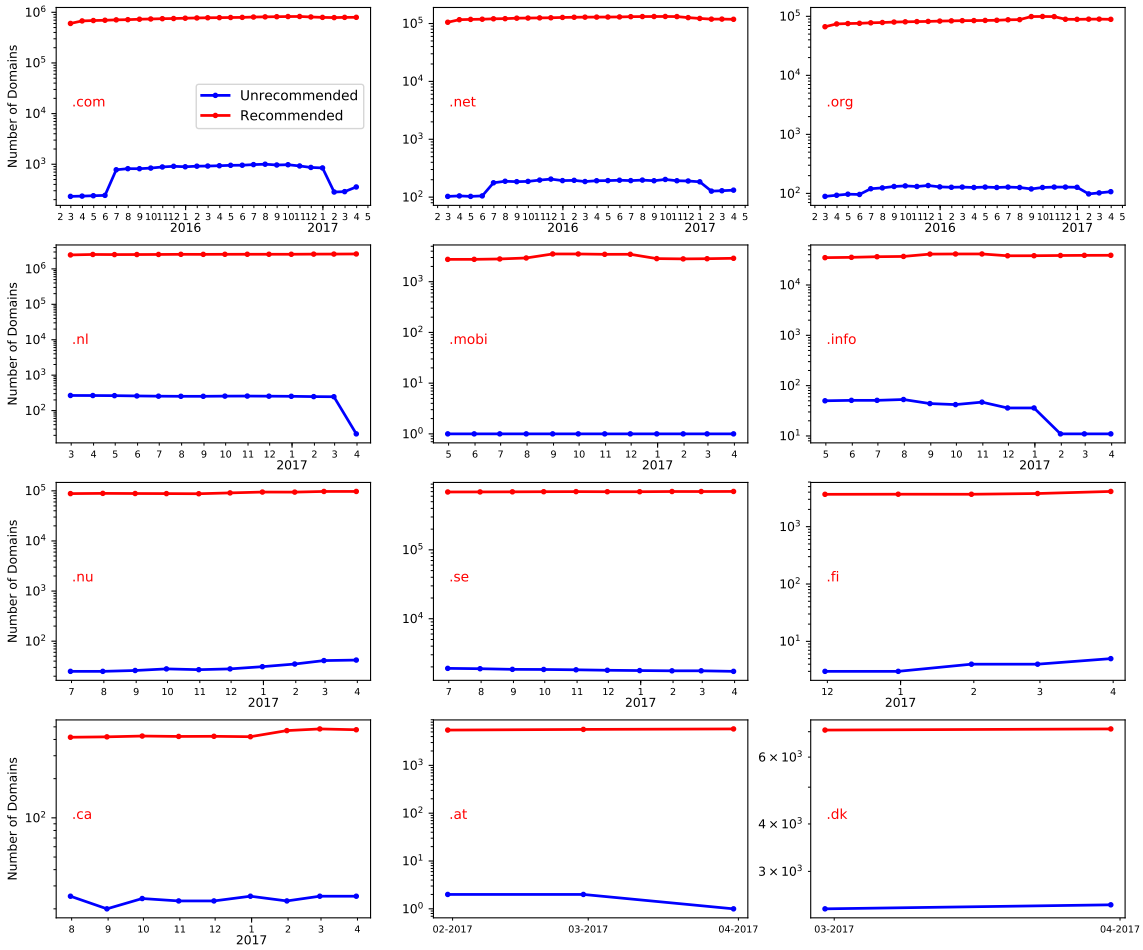


Figure A.3: ZSK: RSA key sizes per TLD over measurement periods

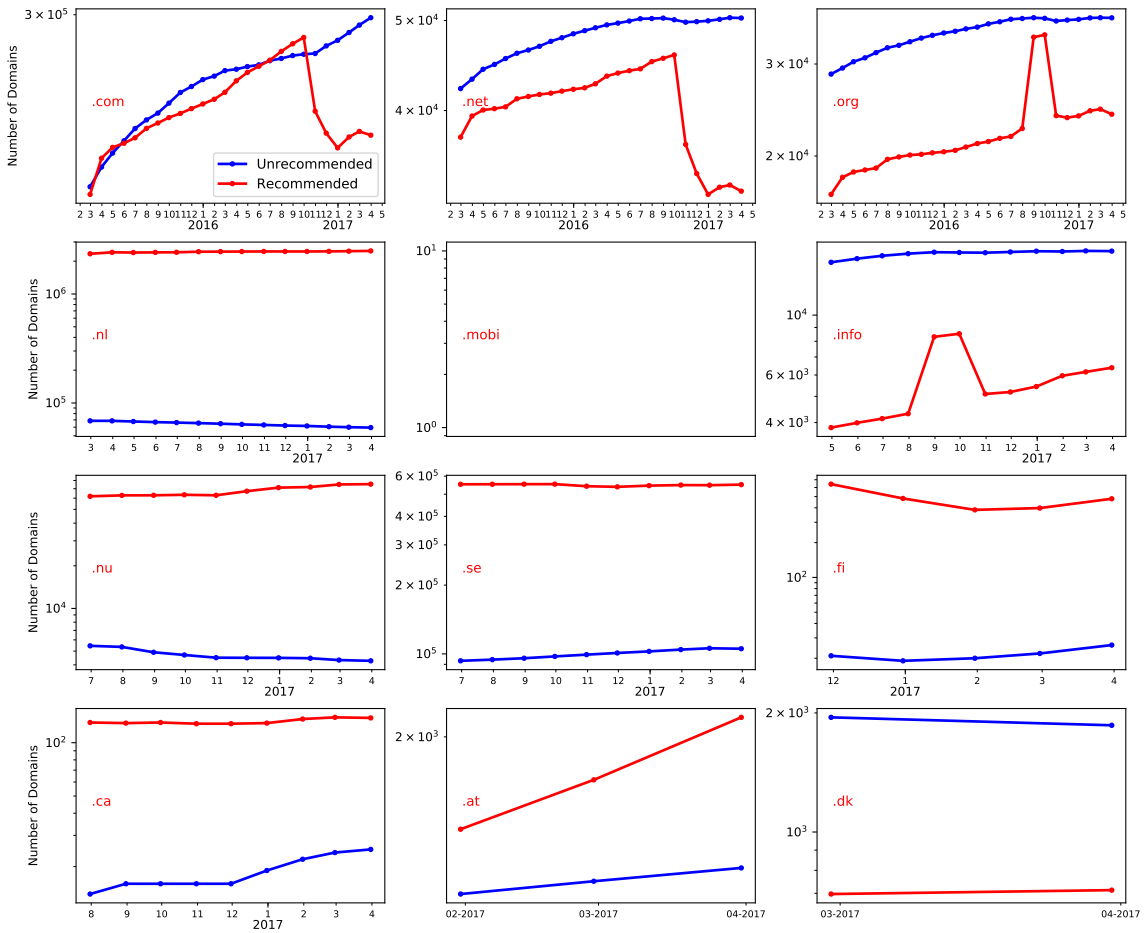


Figure A.4: KSK: RSA key sizes per TLD over measurement periods

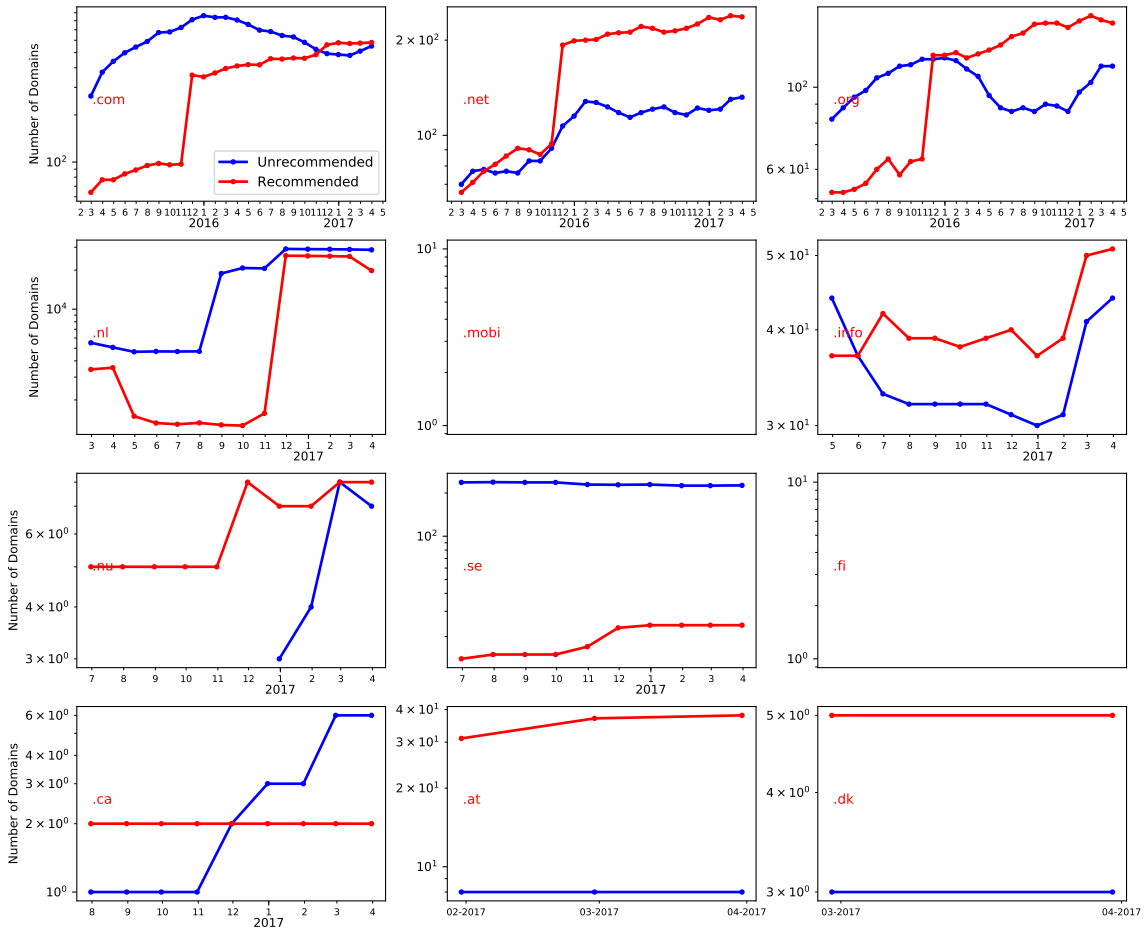


Figure A.5: CSK: RSA key sizes per TLD over measurement periods

A.3 key Rollover

ZSK with ECDSA keys

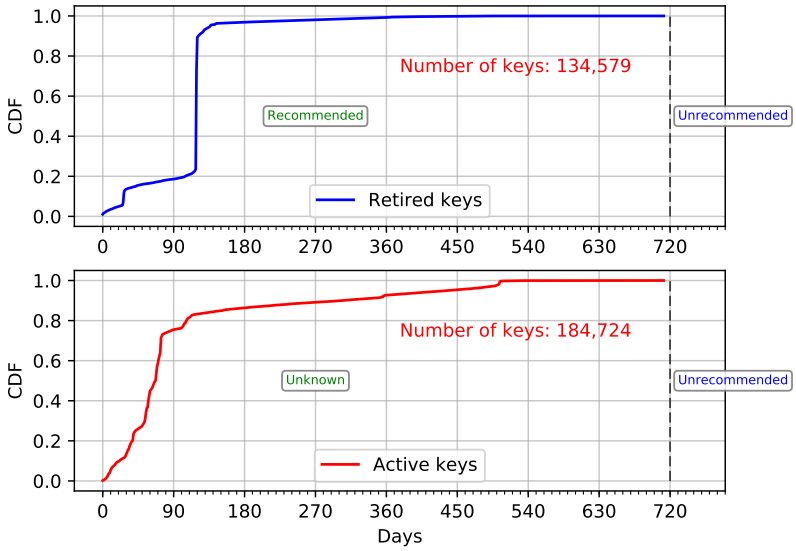


Figure A.6: CDF of ZSK effective period with ECDSA keys

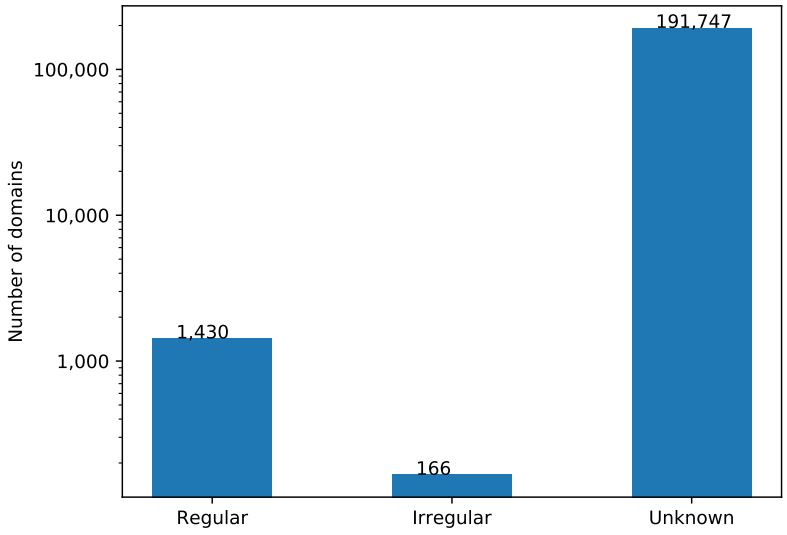


Figure A.7: Regulation of key rollover for ZSK with ECDSA keys

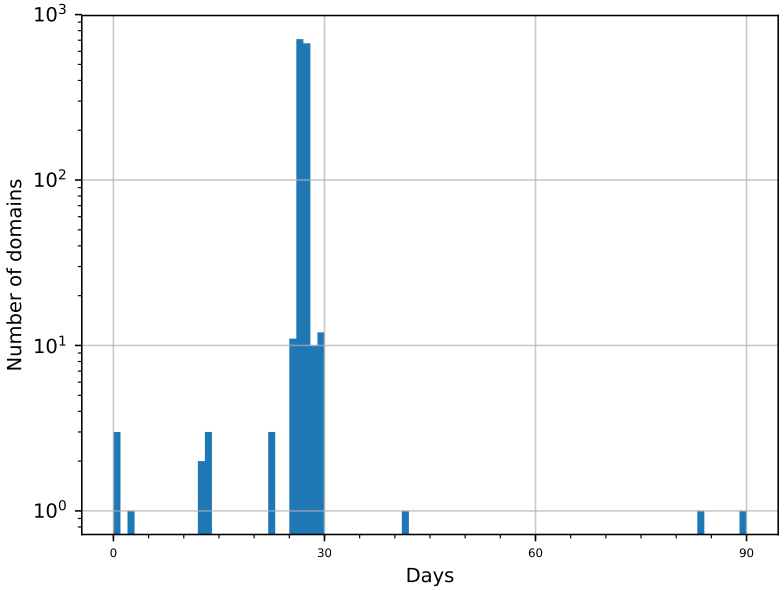


Figure A.8: Histogram of key effective period of regular domains for ZSK with ECDSA keys

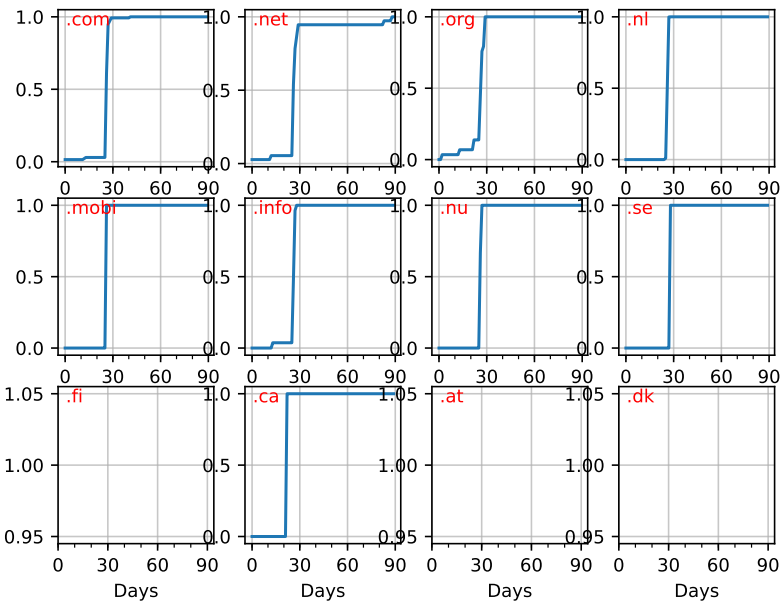


Figure A.9: Key effective period of regular domains per TLD for ZSK with ECDSA keys

KSK with RSA keys

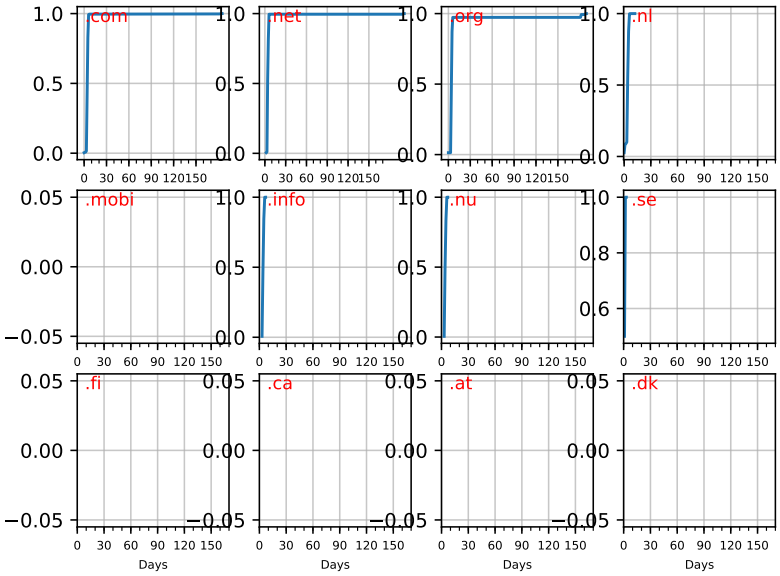


Figure A.10: Key effective period of regular domains per TLD for KSK with RSA keys having key length equal or greater than 2048 bits

KSK with ECDSA keys

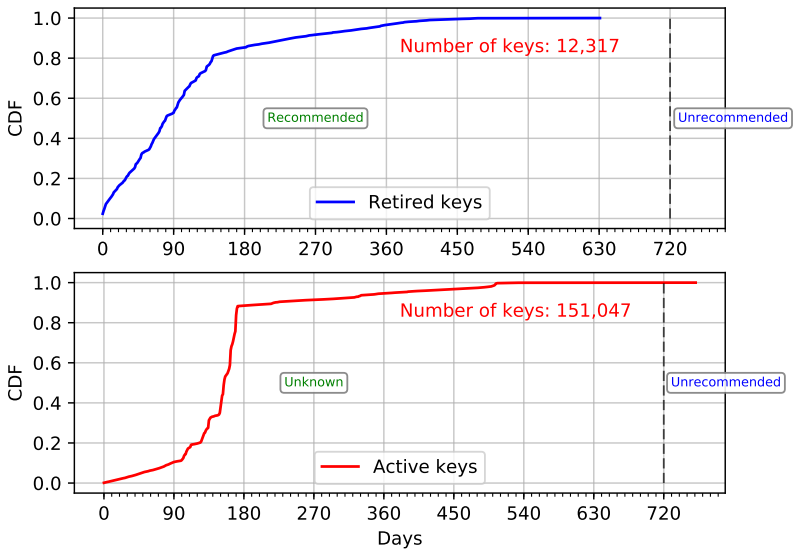


Figure A.11: CDF of KSK effective period with ECDSA keys

CSK

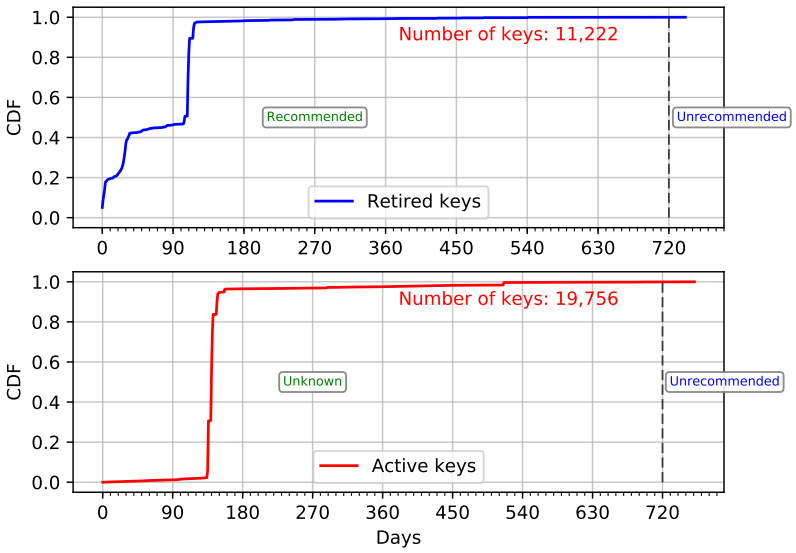


Figure A.12: CDF of effective period of single-key scheme with RSA keys having key length equal or greater than 2048 bits

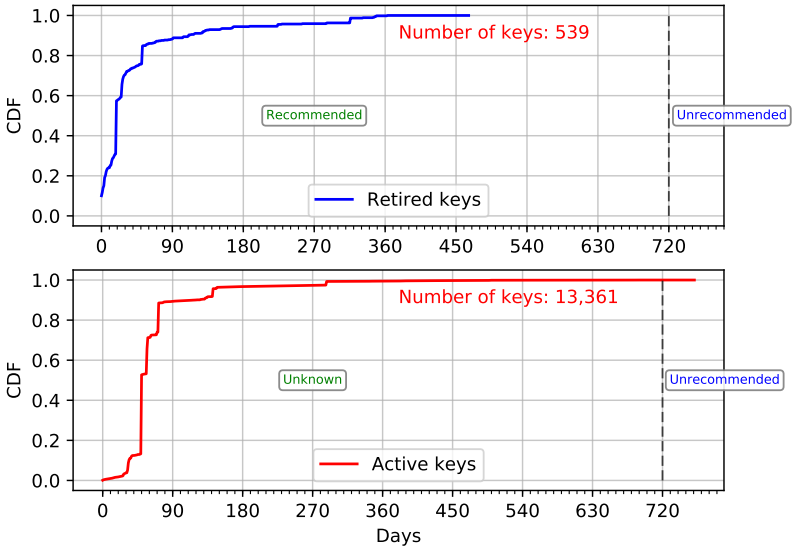


Figure A.13: CDF of effective period of single-key scheme with ECDSA keys

Bibliography

[1] S. M. Bellovin, "Using the Domain Name System for System Break-ins," in *Proceedings of the 5th USENIX Security Symposium*, (Salt Lake City, UT, USA), 1995.

[2] D. Kaminsky, "Black Ops 2008: It's The End Of The Cache As We Know It." `https://www.blackhat.com/presentations/bh-jp-08/bh-jp-08-Kaminsky/BlackHat-Japan-08-Kaminsky-DNS08-BlackOps.pdf`, 2008.

[3] H. Yang, E. Osterweil, D. Massey, S. Lu, and L. Zhang, "Deploying Cryptography in Internet-Scale Systems: A Case Study on DNSSEC," *IEEE Transactions on Dependable and Secure Computing*, vol. 8, no. 5, pp. 656–669, 2011.

[4] P. Mockapetris, "RFC 1034 - Domain Names - Concepts and Facilities." `https://www.ietf.org/rfc/rfc1034.txt`, Nov 1987.

[5] P. Mockapetris, "RFC 1035 - Domain Names - Implementation and Specification." `https://www.ietf.org/rfc/rfc1035.txt`, Nov 1987.

[6] O. Filip and E. Petr, "DNS Cache Poisoning, presentation at ICANN 37." `https://ccnso.icann.org/files/9415/presentation-dnssec-dns-attacks-in-cz-08mar10-en.pdf`, 2010.

[7] D. Dagon, M. Antonakakis, P. Vixie, T. Jinmei, and W. Lee, "Increased DNS Forgery Resistance Through 0x20-bit Encoding: Security via Leet Queries," in *Proceedings of the 15th ACM Conference on Computer and Communications Security*, CCS '08, (New York, NY, USA), pp. 211–222, ACM, 2008.

[8] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose, "RFC 4033 - DNS Security Introduction and Requirements." `https://www.ietf.org/rfc/rfc4033.txt`, Mar 2005.

[9] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose, "RFC 4034 - Resource Records for the DNS Security Extensions." https://www.ietf.org/rfc/rfc4034.txt, Mar 2005.

[10] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose, "RFC 4035 - Protocol Modifications for the DNS Security Extensions." https://www.ietf.org/rfc/rfc4035.txt, Mar 2005.

[11] "Domain Name System Security (DNSSEC) Algorithm Numbers." http://www.iana.org/assignments/dns-sec-alg-numbers/dns-sec-alg-numbers.xml. Accessed: February 2017.

[12] B. Laurie, G. Sisson, R. Arends, and D. Blacka, "DNS Security (DNSSEC) Hashed Authenticated Denial of Existence." https://tools.ietf.org/html/rfc5155, Mar 2008.

[13] O. Kolkman, W. Mekking, and R. Gieben, "RFC 6781 - DNSSEC Operational Practices, Version 2." https://tools.ietf.org/html/rfc6781, Dec 2012.

[14] M. Wander, L. Schwittmann, C. Boelmann, and T. Weis, "GPU-Based NSEC3 Hash Breaking," in *Proceedings of the 2014 IEEE 13th International Symposium on Network Computing and Applications*, NCA '14, (Washington, DC, USA), pp. 137–144, IEEE Computer Society, 2014.

[15] S. Goldberg, M. Naor, D. Papadopoulos, L. Reyzin, S. Vasant, and A. Ziv, "NSEC5 : Provably Preventing DNSSEC Zone Enumeration," in *Proceedings of the Network and Distributed Systems Symposium 2015 (NDSS 2015)*, (Internet Society), 2015.

[16] "ISDN Verlengt DNSSEC Kortingsregeling tot 1 Juli 2018." https://www.ispam.nl/archives/38957/sidn-verlengt-dnssec-kortingsregeling-tot-1-juli-2018/.

[17] "DNSSEC Deployment in Sweden." https://london50.icann.org/en/schedule/wed-dnssec/presentation-dnssec-deployment-sweden-25jun14-en.pdf,, 2014.

[18] Jimmy Brännlund, Internetstiftelsen i Sverig. Personal Communication.

[19] M. Davids, "DNSSEC in .nl." https://www.sidnlabs.nl/downloads/presentations/SIDN-Labs-InternetNL-20160316.pdf, Mar 2016.

[20] E. Osterweil, D. Massey, and L. Zhang, "Observations from the DNSSEC Deployment," in *Proceedings of IEEE ICNP Workshop on Secure Network Protocols (NPSec)*, October 2007.

[21] E. Osterweil, M. Ryan, D. Massey, and L. Zhang, "Quantifying the Operational Status of the DNSSEC Deployment," pp. 231–242, 2008.

[22] R. van Rijswijk-Deij, M. Jonker, A. Sperotto, and A. Pras, "A High-Performance, Scalable Infrastructure for Large-Scale Active DNS Measurements," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 7, 2016.

[23] R. van Rijswijk-Deij, M. Jonker, and A. Sperotto, "On the Adoption of the Elliptic Curve Digital Signature Algorithm (ECDSA) in DNSSEC," in *Proc. of the 12th International Conference on Network and Service Management (CNSM 2016)*, (Montréal, Canada), IFIP, 2016.

[24] "State of DNSSEC Deployment 2016." `https://www.internetsociety.org/sites/default/files/ISOC-State-of-DNSSEC-Deployment-2016-v1.pdf`.

[25] N. L. M. van Adrichem, N. Blenn, A. R. Lua, X. Wang, M. Wasif, F. Fatturrahman, and F. A. Kuipers, "A Measurement Study of DNSSEC Misconfigurations," *Security Informatics*, vol. 4, no. 1, p. 8, 2015.

[26] C. Deccio, J. Sedayao, K. Kant, and P. Mohapatra, "A Case for Comprehensive DNSSEC Monitoring and Analysis Tools," *Securing and Trusting Internet Names (SATIN)*, 2011.

[27] M. Wander, "Measurement Survey of Server-Side DNSSEC Adoption." `http://schd.ws/hosted_files/icann562016/d7/Wander-ICANN56-DNSSEC-Adoption-v2.pdf`, June 2016.

[28] T. Chung, R. van Rijswijk-Deij, B. Chandrasekaran, D. Choffnes, D. Levin, B. M. Maggs, A. Mislove, and C. Wilson, "A Longitudinal, End-to-End View of the DNSSEC Ecosystem," in *26th USENIX Security Symposium (USENIX Security 17)*, (Vancouver, BC), USENIX Association, 2017.

[29] Ó. Guðmundsson and S. Crocker, "Observing DNSSEC validation in the wild," *Proc. of SATIN 2011 Workshop*, 2011.

[30] K. Fukuda, S. Sato, and T. Mitamura, "A Technique for Counting DNSSEC Validators," *Proceedings - IEEE INFOCOM*, pp. 80–84, 2013.

[31] G. Huston, "Measuring DNSSEC Use." https://labs.apnic.net/presentations/store/2013-08-27-dnssec-apnic.pdf, 2013. in APNIC 36 Conference, Xi'An, China.

[32] W. Lian, E. Rescorla, H. Shacham, and S. Savage, "Measuring the Practical Impact of DNSSEC Deployment," *Presented as part of the 22nd USENIX Security Symposium (USENIX Security 13)*, pp. 573–588, 2013.

[33] M. Wander and T. Weis, "Measuring Occurrence of DNSSEC Validation," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 7799 LNCS, pp. 125–134, 2013.

[34] O. Sury and R. Edmonds, "RFC 8080 - Edwards-Curve Digital Security Algorithm (EdDSA) for DNSSEC." https://tools.ietf.org/html/rfc8080, Feb 2017.

[35] R. van Rijswijk-Deij, A. Sperotto, and A. Pras, "Making the Case for Elliptic Curves in DNSSEC," *ACM Computer Communication Review (CCR)*, vol. 45, no. 5, 2015.

[36] P. Hoffman and W. Wijngaards, "Elliptic Curve Digital Signature Algorithm (DSA) for DNSSEC." https://tools.ietf.org/html/rfc6605, Apr 2012.

[37] E. Barker and Q. Dang, "NIST Special Publication 800-57 Part 3 Revision 1 Recommendation for Key Management Part 3: Application-Specific Key Management Guidance,"

[38] R. van Rijswijk-Deij, K. Hageman, A. Sperotto, and A. Pras, "The Performance Impact of Elliptic Curve Cryptography on DNSSEC Validation," *IEEE/ACM Transactions on Networking*, vol. 25, no. 2, 2017.

[39] "We Have Broken SHA-1 in Practice." https://shattered.io/. Accessed: May 2017.

[40] "The Case for Elliptic Curve Cryptography." http://web.archive.org/web/20090117023500/http://www.nsa.gov/business/programs/elliptic_curve.shtml. Accessed: May 2017.

[41] "Good Practices Guide for Deploying DNSSEC." https://www.enisa.europa.eu/publications/gpgdnssec, 2010.

[42] R. Chandramouli and S. Rose, "NIST Special Publication 800-81-2 - Secure Domain Name System (DNS) Deployment Guide," tech. rep., National Institute of Standards and Technology, Gaithersburg, MD, sep 2013.

[43] "The Costs OF DNSSEC Deployment." `http://www.enisa.europa.eu/act/res/technologies/tech/dnsseccosts`, December 2009.

[44] "Lorenz Curve." `https://en.wikipedia.org/wiki/Lorenz_curve`. Accessed: June 2017.