Public Review for

Making the Case for Elliptic Curves in DNSSEC

Roland van Rijswijk-Deij, Anna Sperotto, and Aiko Pras

Like many of the Internet's protocols, DNS was designed without security in mind. Given its central role in translating human readable names into IP addresses, it constitutes an achilles heel in terms of Internet security. This fact has not gone unnoticed, with active development on DNS Security Extensions (DNSSEC) which adds integrity and authenticity to the DNS, by digitally signing DNS data. However, DNSSEC deployment has been lackluster. The authors argue that there are three problems at the core of the deployment impasse: DNSSEC responses are large which can result in (1) IP fragmentation and (2) open the door for DDoS based on DNSSEC. Third, key management complexities can result in domains becoming unreachable. The authors argue that the choice of RSA for DNSSEC is at the heart of these problems and they evaluate the potential benefits of leveraging elliptic curve cryptography (ECC) instead to ameliorate these issues. The reviewers appreciated the data-driven approach taken in this paper using data from a real network to evaluate the potential gains of ECC. They found the arguments for ECC were convincing with empirical evaluations of how ECC can help mitigate threats such as amplification attacks as well as key rollover management issues. The reviewers also raised open questions about the transition to ECC from RSA and how the two signature schemes could co-exist during the transition period. The potential overheads of verification for ECC signatures was also raised. The reviewers agree that the problem tackled in this paper is an important one, and the data-driven approach provides a convincing argument for ECC. The work also raises interesting questions about how a transition to ECC would occur in practice and whether there are other factors hindering deployment aside from the issues raised about RSA.

Public review written by
Phillipa Gill
Stony Brook University, USA

acm sigcomm

Making the Case for Elliptic Curves in DNSSEC

Roland van Rijswijk-Deij
University of Twente and
SURFnet bv
r.m.vanrijswijk@utwente.nl

Anna Sperotto
University of Twente
a.sperotto@utwente.nl

Aiko Pras
University of Twente
a.pras@utwente.nl

ABSTRACT

The Domain Name System Security Extensions (DNSSEC) add authenticity and integrity to the DNS, improving its security. Unfortunately, DNSSEC is not without problems. DNSSEC adds digital signatures to the DNS, significantly increasing the size of DNS responses. This means DNSSEC is more susceptible to packet fragmentation and makes DNSSEC an attractive vector to abuse in amplification-based denial-of-service attacks. Additionally, key management policies are often complex. This makes DNSSEC fragile and leads to operational failures. In this paper, we argue that the choice for RSA as default cryptosystem in DNSSEC is a major factor in these three problems. Alternative cryptosystems, based on elliptic curve cryptography (ECDSA and EdDSA), exist but are rarely used in DNSSEC. We show that these are highly attractive for use in DNSSEC, although they also have disadvantages. To address these, we have initiated research that aims to investigate the viability of deploying ECC at a large scale in DNSSEC.

Keywords

DNS; DNSSEC; fragmentation; DDoS; amplification attack; elliptic curve cryptography; ECDSA; EdDSA

1. INTRODUCTION

The Domain Name System (DNS) performs a critical function on the Internet, translating human readable names into IP addresses. The DNS was never designed with security in mind, though. To address this, a major overhaul of the DNS is underway with the introduction of the DNS Security Extensions (DNSSEC). DNSSEC adds integrity and authenticity to the DNS, by digitally signing DNS data. These signatures are then validated by DNS resolvers to verify that data is authentic and has not been modified in transit.

While DNSSEC can improve the security of the Internet, uptake is still lacklustre. Less than 3% of domains worldwide deploy DNSSEC[1] and at best 13% of clients are protected by DNSSEC validation[2]. We argue that this is partly due to problems with DNSSEC as a technology. Three problems stand out. First, DNSSEC responses are larger and suffer more from IP fragmentation, which impacts availability [1]. Second, DNSSEC's larger responses can be abused for potent denial-of-service attacks [2]. Third, key management in DNSSEC is often complex, which may lead to mistakes that make domains unreachable. These issues raise the question if the benefits of DNSSEC outweigh the disadvantages.

We argue that one of the root causes of these problems is the choice of RSA as default signature algorithm for DNSSEC. RSA keys and signatures are large, compared to traditional DNS messages. There are alternatives, though, based on elliptic curve cryptography (ECC). ECC keys and signatures are much smaller, while their cryptographic strength is excellent. This is attractive for DNSSEC as it reduces response sizes, addressing the first two problems (fragmentation and amplification), and their cryptographic strength makes simpler key management feasible. One particular ECC-based scheme, ECDSA, was already standardised for use in DNSSEC in 2012, but is still rarely used in practice. Given the potential benefits, we argue that this should change. Therefore, we set out to build a case for a switchover to ECDSA and other elliptic curve signature schemes.

Our contribution – We quantify, based on real-world measurements, the effect of switching DNSSEC from RSA to ECC. Our results prove that ECC can mitigate the problems outlined above. But ECC also has disadvantages. We discuss these and have initiated research to study the Internet-scale effects of switching DNSSEC to ECC. This can help guide future standardisation in this area.

1.1 Related Work

The overhead of DNSSEC on the DNS was first studied by Ager et al. [3]. They mention ECC as an alternative to RSA, albeit not in much detail. We add to their work by providing a detailed up-to-date analysis.

Yang et al. [4] performed the first systematic analysis of DNSSEC as an Internet-scale deployment of public key cryptography. They examine cryptographic aspects as well as the complexities of incremental deployment, partial trust chains and key management. What they do not touch on, though, are problems with fragmentation and amplification that we argue are a direct result of choices related to cryptography.

Herzberg & Shulman [5], like us, discuss the problem of cryptographic algorithm choices in DNSSEC. They propose a protocol for DNS clients and servers to negotiate an optimal cipher suite. Their goal is to reduce the amount of cryptographic material that needs to be exchanged, in order to reduce DNS message sizes. While this reduces fragmentation and amplification, it does not reduce the complexity of key management. Rather, it further complicates the DNSSEC protocol. We choose a different path. Instead of introducing additional complexity, we build a case for a complete switch to elliptic curve cryptography in DNSSEC.

[1] http://www.isoc.org/deploy360/dnssec/statistics/

[2] http://stats.labs.apnic.net/dnssec/XA

2. PROBLEMS WITH DNSSEC

This section provides background information on the three issues introduced in Section 1.

2.1 Fragmentation

DNSSEC responses are larger than regular DNS responses since they include digital signatures. Sometimes this leads to packet fragmentation. Moreover, some DNSSEC-specific query types are particularly at risk from this. The `DNSKEY` query type – crucial for DNSSEC – can have large responses as it includes all public keys used in signing the zone.

We showed earlier that fragmentation is a big problem [1]. Up to 10% of hosts may be unable to handle fragmented responses. Domains with fragmented DNS responses are effectively unreachable for these hosts. Based on this research, SURFnet[3] enabled "minimal responses". This tells the name server to respond with the smallest possible answer, preventing most fragmentation. Measurements show this decreases the average response size by 80%. But since it does not set a hard limit on response size, fragmentation can still occur.

To quantify fragmentation under "minimal responses", we examined traffic captured from August 2013 to April 2015 at one of SURFnet's authoritative name servers. Out of 12 million DNSSEC responses per day, 0.5% suffer fragmentation, for $\pm 15,000$ distinct query names per day. Fragmentation occurs for all common query types and response statuses (success, referral and non-existence). This underlines that fragmentation remains a problem when deploying DNSSEC.

2.2 Amplification attacks

DNSSEC's larger responses make it an attractive vector for performing distributed denial-of-service (DDoS) attacks. Like most UDP-based protocols, DNS is susceptible to IP source address spoofing. Thus, it can be abused in so-called amplification attacks, in which an attacker sends queries with a spoofed sender address (the victim's) to a DNS server. Responses – which are generally much larger than the query – are then sent to the victim. This pays off since a small amount of attack traffic results in a large attack volume.

In 2014 we studied the increased amplification potential in DNSSEC compared to regular DNS [2]. DNSSEC-signed domains yield amplification factors averaging between 40 and 55 with outliers up to 179. That is significantly higher (6- to 12-fold) than for regular DNS. Amplification is worst for so-called `ANY` queries, raising the question if this query type should be deprecated[4]. While welcome, this would not solve all amplification issues in DNSSEC. Some DNSSEC-specific query types, such as the `DNSKEY` query that is integral to the protocol, have significant amplification potential.

2.3 Complex key management

The third problem with DNSSEC is complex key management. Fig. 1 shows a typical DNSSEC trust chain (the chain required to verify the signature on `www.example.com`). The figure shows the split into two keys, used in most DNSSEC-signed domains (our measurements show, e.g., that 99.99% of `.com`, `.net` and `.org` domains use this split). The Key Signing Key (KSK) only signs the `DNSKEY` set in the zone and is the secure entry point to the zone. Trust in the KSK is expressed by including a hash of the KSK in the parent zone using a `DS` (delegation signer) record. The `DS` is then signed as part of the parent zone, creating a chain of trust. This pattern repeats all the way to the root zone of the DNS. The KSK of the root is the 'trust anchor' for the whole DNS. The second key type is the Zone Signing Key (ZSK). Its function is to sign all records inside a DNS zone. This KSK/ZSK split is complex. It requires special software to manage key rollovers, and thus introduces additional potential for failure. Furthermore, the scheme is hard to understand for system administrators and makes troubleshooting in case of problems harder. An alternative is to use a single key. This much simpler scheme is often referred to as a Combined Signing Key (CSK) because the single key combines the roles of KSK and ZSK. This scheme is currently rarely used, however, and we explain why below and in the next subsection.

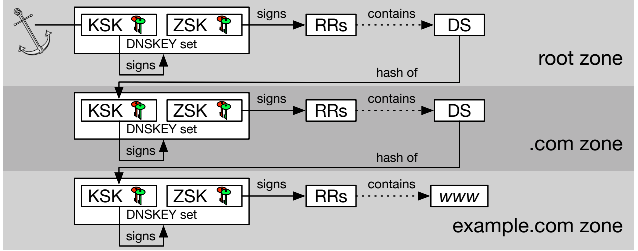

Figure 1: Example DNSSEC trust chain

As Yang et al. [4] discuss, best practices suggest that keys need to be replaced at regular intervals. The length of the interval depends on, among other things, the cryptographic strength of the key. Key rollovers in DNSSEC require interaction with the parent zone to update the `DS`. Interaction is often manual and thus should not occur too often. To limit the frequency, keys need to be strong. But there is another requirement: signatures should not be too large to limit the overhead DNSSEC imposes on DNS messages. This creates a conflict: a low rollover frequency means stronger, larger keys, but that means larger signatures. The KSK/ZSK split balances the two requirements. It allows for a large, long-lived KSK whose signatures only occur in the `DNSKEY` set, versus a small, short-lived ZSK with smaller signatures in the rest of the zone. It is easy to see why this makes a CSK unattractive; either the CSK is small and rolled frequently (which requires interaction with the parent), or it is large, resulting in large signatures and thus large DNS messages.

The KSK/ZSK split itself also introduces a problem. The `DNSKEY` set for a zone with the KSK/ZSK split always contains at least two keys. A key rollover may require having two keys of that type in the zone during the rollover. If a KSK and ZSK rollover coincide, four keys may be present concurrently. On top of that, some DNSSEC policies require standby keys, further increasing the number of keys in the `DNSKEY` set. Consequently, a `DNSKEY` response is at risk from fragmentation – particularly during rollovers – and is an attractive target for abuse in amplification attacks.

2.4 RSA: the root cause?

A big contributor to these three problems is use of the RSA cryptosystem in DNSSEC. According to the standard (RFC 4034), support for RSA is mandatory. In fact, it is the only mandatory algorithm. Almost all DNSSEC deployments make use of RSA. E.g., our measurements show that

[3] The National Research and Education Network in the Netherlands (http://www.surfnet.nl/en/).

[4] https://tools.ietf.org/html/draft-ogud-dnsop-any-notimp-00 (IETF draft for deprecating `ANY` queries).

99.99% of signed domains in .com, .net and .org use RSA.

There are two problems with RSA in the context of DNSSEC. First, DNSSEC responses contain one or more signatures. Each 1024-bit RSA signature is 128 octets, each 2048-bit signature 256 octets. That is a lot in relation to a typical DNS message. Representing an RSA public key in a DNSKEY record requires even more space (around 4 additional octets). Second, increasing the cryptographic strength of RSA has historically meant doubling the key size (from 512 to 1024 and on to 2048). While there is no cryptographic need for this (in theory, increasing the key size by a single bit doubles the burden of cryptanalysis), this has become common practice. As RSA is in widespread use and may have been implemented under the assumption that key sizes are always a power of 2, changing this may be infeasible.

The complex KSK/ZSK split discussed in the previous section comes into play here as well. KSKs are often 2048 bits to be sufficiently strong. But to keep signature sizes manageable, the ZSK is usually 1024 bits. This also illustrates why having a CSK is unattractive; either it is 2048 bits, imposing significant signature overhead on all DNS messages, or it is smaller and needs to be rolled frequently with (manual) parent interaction.

Finally, another problem is looming on the horizon: 1024-bit keys are increasingly considered weak. The National Institute of Standards and Technology (NIST), that issues guidelines for the US federal government, disallows use of 1024-bit keys beyond 2014 [6]. But that NIST makes an exemption for DNSSEC until October 2015 *"due to message size constraints [fragmentation]"* is telling ([7], §8.1.3).

3. ELLIPTIC CURVES IN DNSSEC

We argued above that RSA contributes to the three DNSSEC problems we described. Good alternatives to RSA exist. Elliptic curve cryptography (ECC) offers similar functionality with smaller keys and signatures. One ECC signature scheme, ECDSA, is standardised for use in DNSSEC, but is rarely used in practice (our measurements show less than 0.01% of domains in .com, .net and .org use it). This section builds a case for replacing RSA by ECC by quantifying the benefits for DNSSEC, based on measurements.

3.1 ECC: a brief introduction

ECC is a form of public key cryptography. It is based on algebraic groups defined by mathematical functions called elliptic curves and on operations with points on these curves. While a detailed discussion of ECC is out of scope, we summarise why ECC is an attractive alternative to RSA (for an in-depth introduction, see [8]).

Like all public key cryptosystems, ECC is based on the intractability of a number-theoretic problem. In ECC this is the Elliptic Curve Discrete Logarithm Problem (ECDLP). ECDLP is a variant of the classic Discrete Logarithm Problem (DLP), which applies in multiplicative groups like $\mathbb{Z}_N^*$. Given a group G of order N with generator $g \in G$ the DLP states that given $h = g^x, h \in G$ it is hard to find x, the *discrete logarithm* of h. ECC groups are additive rather than multiplicative. The main operation in ECC is *point multiplication*, implemented as repeated addition. The ECDLP is defined as follows: given two points P and Q that satisfy the equation $Q = dP$ for some scalar d, it is hard to find d, the *elliptic curve discrete logarithm* of Q. What makes ECC attractive is that the most efficient algorithms for breaking ECDLP are of $\mathcal{O}(\sqrt{n})$ where n is the order of the ECC group.

In practice, this means that to achieve 128-bit security, the ECC group order needs to be roughly squared that value, i.e. 256-bits. In comparison, to achieve 128-bit security with RSA requires a group size of 3072-bits [6], 12× more. As we will show in the following subsections, this smaller group size (and subsequent smaller key size) makes ECC very attractive for use in DNSSEC.

3.2 ECC signature algorithms

In this subsection, we discuss two ECC signature algorithms; ECDSA because it has been standardised by the IETF for use in DNSSEC and EdDSA because it is even more attractive for use in DNSSEC.

3.2.1 ECDSA

The Elliptic Curve Digital Signature Algorithm is a US Federal Standard published by NIST [9]. ECDSA works in conjunction with a hash function that has an output block size equal to the curve group size. E.g., with the P-256 curve (one of the curves recommended in the standard), the SHA256 hash algorithm is used. Signatures are represented using two integers – commonly represented by the tuple (r, s) – with each integer of the same bit size as the curve group. Thus, on a 256-bit curve, signatures are 512 bits in size.

ECDSA is standardised for use in DNSSEC (RFC 6605) with two NIST curves, P-256 and P-384. The RFC specifies that public keys are to be stored using the full representation of the curve point (x, y), meaning that 512 bits are required to store the key. This is suboptimal; using a technique called point compression, key storage can be reduced to storing just x with one extra bit to determine the sign of y (i.e. 257 bits for a public key on P-256). Because of fears over patent infringement, however, the authors of the RFC did not include this optimisation (more on this in Sec. 3.4).

3.2.2 EdDSA

EdDSA [10] is an alternative to ECDSA based on so-called twisted Edwards curves, a class of elliptic curves introduced by Bernstein et al. [11]. The designers claim that EdDSA significantly outperforms ECDSA, both for signing, as well as for verification, while maintaining the same level of security. Measurements of the eBACS benchmarking project[5] confirm this claim. EdDSA is not only attractive from a performance point of view, but its public keys are also smaller. EdDSA public keys are always stored as a 256-bit value, half the size of a canonically-represented ECDSA public key. EdDSA has not yet been standardised for use in DNSSEC, but the first draft standards have appeared [12, 13].

3.3 Revisiting DNSSEC problems

We revisit the DNSSEC problems from Sec. 2 and analyse to what extent switching from RSA to ECC mitigates these problems. The analysis covers the following ECC implementation choices: a) ECDSA versus EdDSA, b) which curve is used and c) the 'traditional' KSK/ZSK split versus a single Combined Signing Key (CSK). Tab. 1 provides a set of scenarios covering these choices. We note that choices in key management also affect the first two problems we identified (fragmentation and amplification) and therefore include these choices in the scenarios. The rows in Tab. 1

[5] http://bench.cr.yp.to/index.html

implementation choice	ecdsa384	ecdsa256	ecdsa384csk	ecdsa256csk	eddsasplit	eddsacsk
ECDSA vs. EdDSA	ECDSA	ECDSA	ECDSA	ECDSA	EdDSA	EdDSA
Curve	P-384	P-256	P-384	P-256	Ed25519	Ed25519
KSK/ZSK vs. CSK	KSK/ZSK	KSK/ZSK	CSK	CSK	KSK/ZSK	CSK
	most conservative		⟵	⟶		*most beneficial*

Table 1: Deployment scenarios for ECC in DNSSEC

show the implementation choices, the columns provide convenient short names for the scenarios. The scenarios are sorted from most conservative (in terms of existing standards and practices, and with respect to security and proven cryptography) to most beneficial in terms of tackling the issues we identified (but requiring implementation changes or standardisation and relying on more novel cryptographic algorithms). We will test these scenarios using measurements.

3.3.1 Fragmentation

To show the impact of the scenarios in Tab. 1 on fragmentation, we performed two measurements. First, we re-issued queries that resulted in fragmentation in our measurement from Sec. 2.1. We examined if answers to these queries would be fragmented under each of the scenarios and find that even the most conservative scenario (ecdsa384) vastly reduces the occurrence of fragmentation. Only 0.3% of previously fragmented responses would still be fragmented under this scenario. Under the most beneficial scenario (eddsacsk), less than 0.003% of responses would still be fragmented. To all intents and purposes this is a negligible number.

The second measurement examined the effect of our scenarios on DNSSEC-specific query types that earlier research [1, 2] shows suffer from fragmentation. Particularly the response to a DNSKEY query may suffer from fragmentation. We examined DNSKEY responses for the 0.5 million .com, .net and .org domains with DNSSEC and calculated the response sizes under our scenarios. Fig. 2 shows the top 10% of a CDF plot of the results. The figure shows that 6.5% of current DNSKEY responses exceed the IPv6 minimum MTU and that 0.6% exceed the MTU of Ethernet. It also shows that even switching to the most conservative scenario (ecdsa384) effectively stops fragmentation. But even more remarkable is that two CSK scenarios (ecdsa256csk, eddsadsk) are so effective that the majority of DNSKEY responses would fit in a classic DNS datagram of 512 bytes. We briefly examined the long tail that exceeds this classic DNS limit for these two scenarios and found that simple configuration changes – e.g. enabling "minimal responses" (Sec. 2.1) – can make all answers fit in a classic DNS datagram under these two scenarios.

3.3.2 Amplification attacks

To determine the impact of the scenarios in Tab. 1 on amplification, we repeated the amplification measurements we performed in earlier work [2]. We limited our measurement to the 0.5 million DNSSEC-signed domains in .com, .net and .org. Also, we did not examine all query types, but only examined queries that showed high amplification before (ANY and DNSKEY), as well as regular queries (A and AAAA). First, we examine the effect of our scenarios on amplification. Fig. 3 shows the amplification for the worst amplifier, the ANY query. The figure shows that the amplification that can be achieved with current domains has not changed compared to last year's measurement. Next, the figure illustrates that switching to a conservative ECC scenario (ecdsa256) already causes a 40% decrease in amplification. Finally, the CSK scenarios dampen amplification even more (up to 55%).

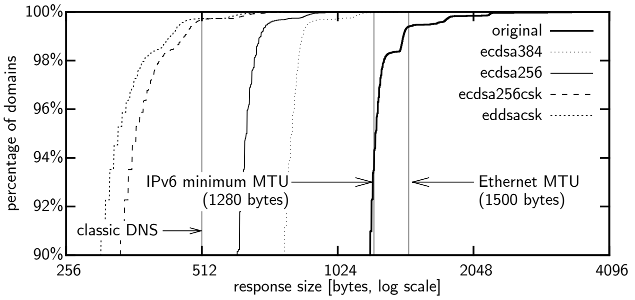

Figure 2: CDF for DNSKEY response sizes

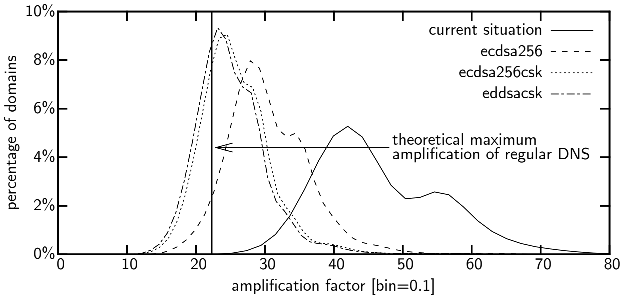

Figure 3: Effect of scenarios on ANY-amplification

It can be argued that the ANY query type can be deprecated, thus removing its dangerous amplification potential. So why spend time decreasing amplification? As we showed in [2], DNSSEC-specific query types, also have significant amplification potential. The DNSKEY query is integral to DNSSEC. Fig. 4 shows that DNSKEY queries for a significant proportion (32.3%) of domains exceed the acceptable upper limit we defined in previous work[6]. But the figure also shows that even the most conservative ECC scenario (ecdsa384) dampens amplification for the DNSKEY query to such an extent that it falls within our acceptable upper limit. If we then look at the CSK scenarios, like we demonstrated in the previous section on fragmentation, these significantly reduce the amplification potential, to such an extent that abuse becomes unattractive because of the low amplification.

Finally, we examined the effect on regular queries (A and AAAA). As we noted in earlier work [2], the amplification that can be achieved with these query types falls well within the acceptable upper limit. Unsurprisingly, applying one of our ECC scenarios further improves this situation. What is of

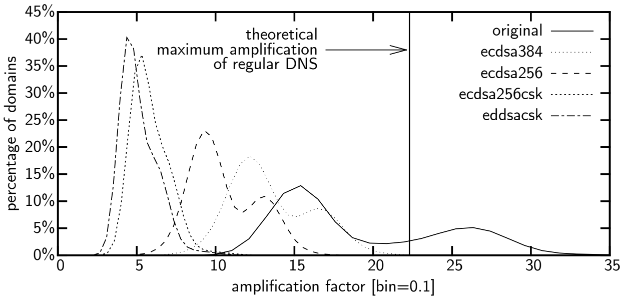

Figure 4: Effects of scenarios on DNSKEY-amplification

[6] This limit is defined as the maximum amplification that can be achieved with 'classic' DNS, i.e. where the maximum DNS message size is limited to 512 bytes.

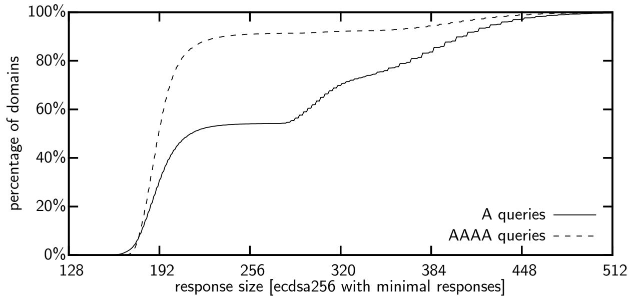

Figure 5: Combining favourable ECC scenarios with response minimisation

interest though – as Fig. 5 shows – is that if we combine the more favourable ECC scenarios (with a 256-bit group and 512-bit signatures) with response minimisation as discussed in Sec. 2.1, the size of `A` responses is reduced such that they fit in 'classic' DNS messages of 512 bytes or less.

3.3.3 *Key management*

Sec. 2.3 showed that DNSSEC key management is complex using the best current practice (the KSK/ZSK model with frequent rollovers). We argued in Sec. 2.4 that the choice for the RSA scheme plays a big part in this problem. As we discussed in Sec. 3.1, ECC has a much more favourable cryptographic strength to key size ratio. Both NIST [6] and the European ECRYPT-II project [14] claim that ECC implementations using 256-bit or larger group sizes are sufficiently strong for the next 30+ years. Based on this, there is no cryptographic reason to roll ECDSA or EdDSA keys with a high frequency. With that in mind, there is no longer a need for a ZSK, and a combined key (CSK) can be used. As we have shown, an additional benefit of this is that it also eliminates fragmentation and significantly dampens amplification. This does not mean that there are no longer valid use cases for the KSK/ZSK split. In high security environments this split is useful as it allows for a KSK that is protected by being kept offline. Arguably, however, in the vast majority of DNSSEC deployments such security measures are unnecessary (and most likely not taken). We strongly advocate that in these kinds of environments use of the much simpler CSK scheme with ECC-based signatures should be preferred, given the benefits we have shown.

3.4 Potential issues with ECC

While we have shown that there is a strong case for switching to an ECC-based signature scheme for DNSSEC, there are also drawbacks, which we examine in this subsection.

3.4.1 *Validation speed*

The most serious challenge when using ECC in DNSSEC is validation speed. As RFC 6605 mentions, validation of ECDSA signatures is slower than validation of RSA signatures. Tab. 2 compares RSA to ECC signature schemes in terms of validation speed (number of CPU cycles), based on benchmarks from the eBATS project[5]. The table lists the number of times verification under the ECC scheme is slower than the RSA scheme. Values are based on the arithmetic average over the median number of CPU cycles for four recent CPU types[7]. Tab. 2 clearly shows that ECC schemes are outperformed by RSA. This is especially true for the most commonly used RSA key size (1024-bit, the ZSK size in over 99% of DNSSEC-signed `.com`, `.net` and `.org` domains) compared to ECDSA P-256.

	RSA 1024	RSA 2048	RSA 3072
ECDSA P-256	26.3 (6.6)	13.4 (3.4)	7.6
ECDSA P-384	79.1	40.4	22.8
EdDSA	5.6	2.9	1.6

Table 2: Comparing signature verification speed

Since DNSSEC validation is performed frequently by DNS resolver systems this is potentially a serious issue. While switching to ECC solves issues that are mostly faced by operators of authoritative name servers, its introduction will impose an additional burden on DNS resolver systems. It is as yet unclear if existing DNS resolvers can deal with the increased CPU load of ECC signature verification, especially if deployment of DNSSEC accelerates. We believe this question urgently needs answering, before ECC-based signature schemes are deployed on a large scale in DNSSEC and have started research for this (Sec. 4). We note that recent advances[8] in the popular OpenSSL library speed up ECDSA by a factor of 8, and RSA by about 2. This positively affects the speed difference, which we extrapolated and include in Tab. 2 in parentheses for ECDSA P-256.

3.4.2 *Trust in NIST curves*

It is impossible to talk about ECC without addressing the elephant in the room. Since the Snowden revelations, experts have cast doubt on the trustworthiness of the NIST curves, two of which – P-256 and P-384 – are standardised for use in DNSSEC. E.g., Schneier[9] suggests to *"prefer conventional discrete-log-based systems over elliptic-curve systems; the latter have constants that the NSA influences when they can"*. While it appears likely that a NIST standard for random number generation was influenced, there is no proof that the NIST curves were deliberately weakened. In fact, Bernstein & Lange[10] refute this claim. They do, however, raise other concerns about the NIST curves. In their presentation they provide arguments why NIST curves may be insecure because of implementation errors in the cryptographic operations on these curves. Bernstein & Lange[10] suggest exploring alternatives such as their EdDSA signature scheme Ed25519 [10]. This is also one of the reasons why we include Ed25519 in our scenarios.

3.4.3 *Intellectual property worries*

Another long-running issue with ECC are the intellectual property (IP) rights surrounding it. CertiCom, a Canadian company, holds patents relating to ECC and its implementation. Fear of litigation hampers ECC adoption. The IETF has taken steps around this IP issue by standardising non-patented forms of ECC in RFCs. This, though, sometimes leads to suboptimal implementation choices. As outlined in Sec. 3.2.1, the public key for ECDSA can be stored using so-called 'point compression', which requires less storage space. This optimisation did not make it into the standard due to

[7] To be specific, we used the results from the `h9ivy`, `hydra8`, `titan0` and `hydra9` systems of the eBATS project.

[8] https://ripe70.ripe.net/presentations/85-Alg-13-support.pdf

[9] http://www.theguardian.com/world/2013/sep/05/nsa-how-to-remain-secure-surveillance

[10] http://cr.yp.to/talks/2013.09.16/slides-djb-20130916-a4.pdf

IP concerns. EdDSA does not seem to suffer as much from IP issues. In fact, Bernstein claims[11] that the base curve for EdDSA does not infringe any patents related to ECC.

3.4.4 Support for ECC in DNSSEC software

The final issue is support for ECC in DNSSEC software. In principle, all major DNSSEC software distributions support ECDSA, both for signing as well as validition. Practical measurements by Huston and Michaelson[12], however, show that not all online resolvers that currently perform DNSSEC validation recognise ECDSA signatures, probably because their software needs upgrading. Fortunately, they also observe that the fraction of resolvers that support ECDSA validation is growing (and currently around 80%). We note that for obvious reasons, EdDSA support in DNSSEC software is non-existent. Until EdDSA is standardised for use in DNSSEC, this is unlikely to change. Finally, not unimportantly, the systems used by TLD registries and domain name registrars need to support registration of ECDSA keys for secure delegations. As Guðmundsson[8] recently pointed out, this support is currently still in the early stages.

4. FUTURE RESEARCH

We have shown that ECC signature schemes are attractive for use in DNSSEC. But in Sec. 3.4, we also showed that there are disadvantages. We have initiated research to tackle what we believe to be – despite recent speed advances[8] – the most important problem: validation speed. A large-scale deployment of ECC in DNSSEC may impose an undue burden on DNS resolvers that perform DNSSEC validation. It is currently unknown if this is actually an issue, but we feel strongly that this question must be answered before ECC is deployed on a large scale. Given the buzz around using ECC for DNSSEC in the DNS operator community and the IETF, there is urgency around answering this question. To address this issue, we are working on modelling validating DNS resolvers. The main goal of our model is to be able to calculate the number of signature validations required for certain traffic patterns. This will allow us to extrapolate the impact of different signature schemes on the CPU power required. Our model is in the initial stages. We are currently defining what variables influence the number of validations. In parallel, we are examining live DNS traffic measurements to shape our model and feed its variables. We are actively calling out to network operators to assist us by sharing trace data to be used with our model. We note that another avenue of research in this context is improving ECC performance.

5. CONCLUSIONS

As we showed in earlier work [1, 2], DNSSEC suffers from a number of challenges (fragmentation and amplification) and is complex to deploy. We argued that one of the root causes of these issues is the use of RSA as default signature scheme in DNSSEC. In this paper we build a strong case for using alternative digital signature schemes, based on elliptic curve cryptography. Using measurements, this paper shows how using ECC can effectively prevent fragmentation of DNSSEC responses as well as significantly reduce the amplification attack potential in DNSSEC. Additionally, ECC's cryptographic strength means that DNSSEC key management can be simplified significantly.

While the case for using ECC in DNSSEC is very strong, we believe that one important question needs answering: will the additional computational burden of validating ECC signatures be bearable for operational DNS resolvers? To this end, we have initiated research and are actively working towards quantifying this issue. The outcome of this research will help guide future standardisation.

6. ACKNOWLEDGEMENTS

The authors thank Ólafur Guðmundsson, Benno Overeinder and Wouter Wijngaards, as well as the anonymous reviewers for their valuable feedback.

This work was supported by the EU-FP7 FLAMINGO Network of Excellence Project (318488) and by SURF, the Netherlands collaborative organisation for ICT in higher education and research institutes.

7. REFERENCES

[1] G. van den Broek, R. van Rijswijk, A. Sperotto, and A. Pras. DNSSEC Meets Real World: Dealing with Unreachability Caused by Fragmentation. *IEEE Communications Magazine*, 52(April):154–160, 2014.

[2] R. van Rijswijk-Deij, A. Sperotto, and A. Pras. DNSSEC and its potential for DDoS attacks. In *ACM IMC 2014*, Vancouver, BC, Canada, 2014. ACM Press.

[3] B. Ager, H. Dreger, and A. Feldmann. Predicting the DNSSEC overhead using DNS traces. *Proc. of IEEE CISS 2006*, pages 1484–1489, 2007.

[4] H. Yang, E. Osterweil, D. Massey, S. Lu, and L. Zhang. Deploying cryptography in internet-scale systems: A case study on DNSSEC. *IEEE Trans. on Dependable and Secure Comp.*, 8(5):656–669, 2011.

[5] A. Herzberg and H. Shulman. Cipher-Suite Negotiation for DNSSEC: Hop-by-Hop or End-to-End? *IEEE Internet Computing*, 19:80–84, 2015.

[6] E. Barker, W. Barker, W. Burr, W. Polk, and M. Smid. Recommendation for Key Management - Part 1: General (r. 3). *NIST SP800-57*, 2012.

[7] E. Barker and Q. Dang. Recommendation for Key Management - Part 3: Application-Specific Key Management Guidance (r. 1). *NIST SP 800-57*, 2015.

[8] D. Hankerson, A.J. Menezes, and S. Vanstone. *Guide to Elliptic Curve Cryptography.* Springer, 2004.

[9] NIST. FIPS PUB 186-4 - Digital Signature Standard (DSS). *Processing Standards Publication*, 2009.

[10] D. Bernstein, N. Duif, T. Lange, P. Schwabe, and B.Y. Yang. High-Speed High-Security Signatures. *Journal of Cryptographic Engineering*, 2(2):77–89, 2012.

[11] D. Bernstein, P. Birkner, M. Joye, and T. Lange. Twisted Edwards Curves. In *AFRICACRYPT 2008*, volume 2, pages 389–405. 2008.

[12] S. Josefsson and N. Moeller. EdDSA and Ed25519 (draft-josefsson-eddsa-ed25519-03), 2015.

[13] O. Surý. Ed25519 for DNSSEC (draft-sury-dnskey-ed25519-00), 2015.

[14] N. Smart. ECRYPT II Yearly Report on Algorithms and Keysizes 2011-2012. Technical report, 2012.

[11] http://cr.yp.to/ecdh/patents.html

[12] http://www.iepg.org/2015-03-22-ietf92/