To DNSSEC

Geoff Huston,
APNIC

To DNSSEC or Not to DNSSEC

Geoff Huston,
APNIC

Why?

- Isn’t www.google.com DNSSEC-signed?
- Or www.amazon.com, www.facebook.com, or www.microsoft.com?
- Or many other major Internet service names?

- Yet there are other places that mandate the use of DNSSEC, including gTLDs in the root zone, the USG OMB directive?

- Why the mixed signals about DNSSEC?

What's going on?

- Some parts of the DNS world see net benefit in having users assure themselves that the answer provided to a DNS query is genuine, by using DNSSEC and performing validation of the DNS answer
- Other parts of the DNS world see the sum of risks and costs of of adding DNSSEC to their part of the DNS as being greater than the benefit
- Lets look at this topic, and also look at the way we provide the secure foundations for today’s Internet

What is DNSSEC?

(This answer could be really long or very short – I'll go for the ultra short version here)

- A DNS zone administration generates a public/private key pair and generates a digital signature for every authoritative record in a zone. These signatures are placed into the zone as RRSIG records. DNSSEC also signs across the "spans" between each pair of adjacent names in the zone.
- A hash of the zone's public key is passed to the zone's parent, which is placed into the signed parent zone as an authoritative (signed) record
- Clients can authenticate a DNS record by validating it against the associated signature record and assembling a validation chain from child to parent up to the root zone to validate the sequence of interlocking zone signing keys

DNSSEC - The Case for NO

The Zone Admin's perspective:

- Managing keys can be tricky and many zones have got themselves tied up with misconfigured keys (https://ianix.com/pub/dnssec-outages.html)
  - The operational preference to use keys for shorter periods and regularly roll across the new keys appears to generate its own failures
  - Performing key rolls some years apart tend to resist treating key rolls as part of Standard Operational Practice

DNSSEC - The Case for NO

The Zone Admin's perspective:

- The procedures for passing the public key of the zone entry point to the parent are inconsistent and error prone
  - Can this be automated?
  - How does automation of the DS record sit with the current paraphernalia of semi-automation of delegation information through EPP and related processes?

Publishing Signed Zones

Things get warped and twisted…

- Key Management
  - ZSK/KSK issues
  - Key Revocation
  - Key Rollover
- Zone Signing
  - Signature record lifetimes
  - Whole-of-zone signing or Front-End signers
  - Incremental signers and assembling negative spans (Compact answers)
  - Handling Large Zones
- TTL settings
- Multiple DNS operators
  - Single shared key and issues of coordination and key risks
  - Multiple keys and issues of DNS response bloat with multiple keys
- NSEC3 issues

Incremental Signing

- NSEC and NSEC3 are both based on the assumption that online signers are not viable
  - Private keys should not be actively used in online environments
  - They also assume that the entirety of a zone can be assembled for signature
- As incremental signers proliferate we are seeing a defacto redefinition of NODATA and NXDOMAIN responses where the signed payload is a minimally spanning response to the query name and query type instead of a pre-computed maximal span
- This compact-form denial response can confuse some resolvers

Validation

- Key and Signature Size and Cryptographic strength
  - Careful choice of crypto needed
- Elapsed time for users to incrementally create the validation path
  - The incremental query/response approach to path creation creates an unacceptable time penalty
  - And large DNSKEY responses have an additional switch-to-TCP time penalty
- Time to repair DNSSEC config errors is exacerbated by TTL choice
- Stub resolvers generally do NOT validate – the recursive to stub path is still susceptible to various forms of manipulation

Who Validates?

- Validation is slow, error prone and stresses out the DNS over UDP model
- Most end users do not even use DNSSEC-validating functions in their local stub resolver
- They generally rely on the recursive resolver to perform validation – and the recursive-to-stub hop is still vulnerable to tampering

What is DNSSEC protecting you against?

- What's the threat issue going on here?
  - Kaminsky styled attack of off-path cache poisoning?
    - Between port and case randomisation there is probably adequate randomisation to protect the client from an off-path guessing attack
  - On path direct attack of response substitution?

- Even then - so what?
  - If we are looking at poisoning the name-to-address relationship and misdirecting the user then this is much the same as a routing attack -- TLS is going to help here by **authenticating the named identity** of the remote service – its IP address is irrelevant to this authentication!
  - If the service does not use some form of authentication then the client is very exposed in any case and DNSSEC is not going to mitigate all risks here!

Why bother with DNSSEC at all?

- It slows down name resolution
- Its more to go wrong
- We don’t really know what we are protecting against

- That’s a lot of complexity, fragility and cost without a clear incremental benefit
- And TLS is doing a Fine Job!

DNSSEC - The Case for YES

It’s not about DNSSEC per se

But if we could place trust in DNS responses that they are authentic and current then we could use the DNS for roles where it would highly unwise to do so otherwise

Is TLS really doing a Fine Job?

Are Domain Name certificates robust?

Are they achieving what we need from them?

- A domain owner demonstrates to a third party certificate issuer that it has control over a domain
  - There are more involved certificates that include additional validation of this “control”, but they all present to the client in the same format, so the client is unaware of the strength of the validation steps that were taken to issue the certificate in the first place
- On this basis the certificate issuer issues a certificate for the domain owner that attest that the domain is owned by this party
- The certificate is used without any further ongoing reference to the certificate issuer, nor to the parent domain, nor with any check over the certificate holder’s continuing control over the domain

Domain Name Certificate Issuers

- The Certificate and Browser Forum (CAB Forum) is a group of ~1000 entities who issue domain name certificates and the browser platforms who use them
  - The certificate issues undertake to apply a set of tests to determine that the applicant “controls” a given domain name
  - And they undertake to never deviate from these tests (*)
  - And they undertake never to lie in the certificates they issue (**)
  - And they undertake never to have their certificate issuance systems compromised by hostile attacks (***)

* except when they do
** except when they lie
*** except when their systems are compromised

Domain Name Certificates

- Certificate Issuance Abuse
  - Pinning - Any CA can issue a domain name certificate for any domain name. How can a client know which is the “right” CA for a domain name certificate?
  - Transparency – All CAs should stash a copy of all issued certificates in a transparency log, so that all issuance transactions are recorded

Certificate Revocation

This is challenging to scale

- Sending a full revocation list to check the status of a single certificate is hopelessly inefficient
- An online certificate status check represents a potential privacy leak as well as a scaling issue for the CA
- OSCP stapling is somewhat pointless
  - OCSP data is a commentary about the pasts state of a certificate, not the current state
  - And why pass the client a certificate AND an OCSP revocation notice?

DANE and DNSSEC

- If the entire aim of the domain name certificate framework is to securely associate a key pair with a domain name then why not just stash the public key in the DNS (DANE)?
- And sign it with DNSSEC

- You get currency and authenticity and eliminate a bunch of third party intermediaries

Comparison

- The model of DANE allows the key to be bound to the domain name, and DNSSEC is used to support authentication of this key value
- The model of Domain Name Certificates is a detached third party commentary about the control of a domain name

But DNSSEC over DNS over UDP still sucks!

But DNSSEC over DNS over UDP still sucks!

- So don’t use it!
  - Why not turn off the use of DNSSEC OK in DNS over UDP queries by default

But DNSSEC over DNS over UDP still sucks!

- So don’t use it!
  - Why not turn off the use of DNSSEC OK in DNS over UDP queries by default

- But validation is still a pain
- But if we are using TCP to transport credentials then we can also look at pushing the validation workload from the client to the server
  - RFC7901 – Chain Query Requests
    - Task the authoritative server for a zone to maintain the current validation path for the zone
    - And package the path in TCP responses when the query has the chain extension

Then we can use this in TLS..

- RFC 9102 - TLS DNSSEC Chain Extension
  - ”staple” the DNSSEC chain extension to the TLS data alongside the DANE TLSA record
  - The server maintains a current copy of the DANE TLSA record and the DNSSEC validation path to construct the chain extension
  - The client can validate the DANE date based on a local copy of the DNS root zone KSK without any direct query to the DNS

- Which integrates query-less DANE and DNSSEC into TLS

DNSSEC

- If we are serious about DNSSEC, then we really need to fix the validation time and stub resolver issues
- We can do both of that with DNSSEC Chain Extensions but to get there then we need to push DNS + DNSSEC queries to TCP (or perhaps DoT / DoH)
- We can do that if both DNS clients and servers are willing to tolerate the incrementally higher overheads of DNS over reliable (and possibly secure) transport
- Which might be achievable if anyone actually pays for the DNS!
  - But they don’t
  - So DNS over UDP wins because its cheaper!

DNSSEC

- Were we trying to be too clever in trying to cram DNSSEC into the UDP query/response mechanism?
- What if we re-thought DNSSEC as a server-side function?
  - And use TCP transport to push validation keychains to validating clients

Chickens and Eggs

- Why bother?

  There is no compelling use case than makes DNSSEC essential

- No point!
  - There is no point in creating widespread critical dependences on DNSSEC while there is only piecemeal adoption of DNSSEC

DNSSEC: Yes? Or No?

- Widely distributed diverse systems have a strong preference for stasis
  - Change is challenging to orchestrate
  - Costs and benefit are often misaligned
  - Very large scale systems are averse to complexity and fragility

- If all that we are doing here is making a case for replacing X.509 domain name certificates with DNSSEC-signed DANE records then this is going to be a difficult sell job!
  - The benefits are marginal, but the transitional costs are high

Musing on Infrastructure Security Mechanisms

Do we get the level of security we **need**?

Or the level of security that we are **prepared to pay for**?

Thanks

Questions?