Looking at TLD DNSSEC Practices

Developers vs. Operators

Edward Lewis

Neustar

At RIPE 64

Tuesday, April 17, 2012

DNSSEC

- DNSSEC is a set of extensions to the DNS protocol
  - Considerations for incremental deployment as well as for future adjustments were important in its development
  - Resulting in many aspects being left to the operator to decide
- DNSSEC has begun to be deployed
  - Interesting to look at what the early adopting operators have decided and compare this to the expectations of the protocol developers

Why the Study Began

- Two concrete reasons prompted this work
  - We (=my employer) operate a few TLD registries
  - We also offer Managed DNS services
- We needed to pick our DNSSEC parameters
  - Besides reading, experimenting...
  - A good way to do this is to review what others are doing
    - The TLD operator club serves as a good example
- Outcomes of the study
  - We've picked and adjusted our parameters
  - Compare expectations of developers to actions of operators

Characterizing DNSSEC

- Just to introduce some terminology
- Roles of keys
  - Single key pair or "KSK and ZSK"
- Key Management Parameters
  - Cryptographic parameters (algorithm, hash, key length)
  - Operations cycles (durations of use, schedule of changes)
- Negative Answer Style
  - NSEC or NSEC3
  - Parameters of NSEC3

What I Do

- Hourly, grab a copy of the DNS root zone
- Query the TLDs for the records at the top of their zones
- Smooth the data daily (no intermittent drops wanted)
- Boil the raw data, seeking a more useful form
- Simmer the data, looking for trends ("streaks")
- Cool down the data, making it easier to "eyeball"
- And then - more analysis and inspection of interesting bits

- No cool "visuals" - the numbers are small and don't change a lot. ;)

Summary of TLD DNSSEC

- Root plus TLDs minus experimental TLDs
  - 303 zones
- Number of signed TLDs
  - 82 (or 27% of 303), counting the root zone
  - Since June, 2011, 19 started and 1 stopped
- Of the 82 signed TLDs
  - 100% use the KSK/ZSK roles
  - Over 90% use one of two cryptographic algorithms
  - Over 90% use the same set of sizes for their keys
  - Over 90% are linked to the distributed root key

Expectations About Crypto

- The expectations of protocol developers
  - Operators would use more than one cryptographic algorithm to reach the broadest base of clients
  - Two kinds of keys would be used because of the difficulty of exchanging with the DNS "parent" of the operator
  - Parameters like length of the keys would be determined by the operators, optimizing for needed protection
  - Operators would change the keys in use according to the strength of the keys

Cryptographic Algorithms

- Timeline of algorithm definitions in DNSSEC
  - Originally DSA and RSA-SHA1, in 2009 added RSA-SHA256
  - In 2012 another algorithm is being introduced
- Of the currently signed zones
  - 50% are signed with RSA-SHA1 (increased by 3 since June)
  - 45% are signed with RSA-SHA256 (increased by 14)
  - Last summer the balance was 60%/36%
- No operator uses multiple algorithms
- One TLD has changed algorithms
  - Proving it can be done, but *only* one has changed

Why That Is Interesting

- Defining new cryptographic elements impacts choices made on new deployments and one can see the inertia of an existing deployment
  - Operators pick "the best one (available)"
- It is rare that a protocol extension is adopted with crucial elements designed to change
  - Tech-refresh is tough even when it is just software updates
  - "Liberal in what you accept, conservative in what you send" doesn't help here
- Capability of "the other side" is something one can't control

Key Lengths

- RFC 4641 cites the choice of two lengths
  - 1024 bits for a ZSK
  - 2048 bits for a KSK
- 90% of signed zones follow these numbers exactly
- 96% use 1024 bit ZSK (with any size KSK)
- 93% use 2048 bit KSK (with any size ZSK)
- 1% uses neither of these choices...(that's one zone)
- There has been no empirical evidence that the suggested sizes are sufficient, and only some scientific evidence
  - Just the power of suggestion...

Changing Keys

- This was anticipated to be the biggest burden of DNSSEC
- There are three factors to consider
  - Frequency
  - Duration
  - Style (mechanics)
- The expectations of protocol developers
  - Key changes would be needed due to the lifetime of keys
  - Key changes would try to minimize excessively large messages and/or be shortened as much as possible

Frequency

- Frequency was anticipated to be annual for KSK and monthly for ZSK (RFC 4641)
- Once deployment happened, some crypto-engineers said there should not be any changes until needed
- Operators change to establish a pattern of actions
  - Practice in case of emergency
- Observed is that operators, for the ZSK role
  - 35% change monthly, 10% bi-monthly, 18% quarterly
  - Rest have either never or haven't established a pattern
- While cryptography tends to randomness, operators tend to like the predictable

Duration

- Because DNS employs caches, data can't be simply swapped, timing of actions is important
  - The key set's TTL is important when introducing a key
  - The signature duration is important when retiring
- For a while the protocol engineers were writing a very detailed document on the timing of changes
  - Very interesting work, but as an operator hopelessly complex
- Looking at the zones
  - In general, TLDs introduce keys well before they have to
  - For retirement, keys generally hang around longer than needed

Style

- There are two approaches to changing keys
  - Old + new key plus a signature (or double key approach)
  - Old + new signatures plus a key (or double signature)
- The preferred approach differs between ZSK and KSK
- For ZSK, 72% one signature, 2.5% one key, 26% can't tell
  - Preferring to minimize signatures because there are more of them and signatures are bigger than keys
- For KSK, not enough data yet
  - But it looks like "one key" (in this case "one DS") is the leader

Why Keys Appear "Early"

- The real question is - how many keys are published?
- Minimizing size, a TLD would have one KSK and one ZSK
- But some TLDs publish two of each, for on-line backup
  - A key appears as a backup, later promoted to active
- Counting for ZSK
  - 47% have one ZSK, 44% have two ZSKs, rest around 3
- Counting for KSK (but this is premature, lack of data)
  - 60% averaged one, others averaged 2, and one averaged 3
- "Average" because during key changes, keys are added

DS Hashes

- The DS record contains the information the root publishes about the security of the TLD (and so on down the tree)
- In the root zone there are currently DS records for 75 TLDs (less than the 82 signed)
- The DS can have an SHA1 or SHA256 hash, and RFC 4509 recommends publishing both for the time being
  - 47% have both, 47% have SHA-256 only, 7% SHA-1 only
- The protocol engineers anticipated having multiple hashes, operators split between that and doing what the root does
  - Latter rationale - software availability

NSEC3 Salt

- Most TLDs use NSEC3, 78%
- There's a recommendation in RFC 5155 to change the salt with every signing
  - 4% of TLDs change it daily, 75% haven't changed it since June
  - Others change it regularly, such as monthly
- The impact of a salt change can be significant
  - Changing salt changes all NSEC3 records and their signatures
  - With a high DNSSEC adoption rate (or a non TLD-type zone) that is a lot of data to move between servers
- An observation: specs assume "batch" operations which is no longer the preferred way to work

What Emerges From This

- In operations "optimization" of the protocol is backed off
  - Simplicity in operations
  - Optimizing for other features, resiliency and staff turnover
  - Configuring so that "the normal state" can be easily observed
- Other factors
  - Availability of software tools and the limitations of the tools
    - Operators are generally not software developers
  - Engineering the adoption of DNSSEC takes one to two years
    - Changes to specifications take a while to be seen in operations
  - "What the root does"

The Gap (For TLDs)

- Between protocol engineering and operations there is a gap
  - Protocol engineers "optimize" (to their criteria)
  - Operators do what it takes to make it work
- What the operators need still
  - More guidance on cryptography
    - A means to determine when to switch algorithms
    - Guidance on how parameters impact performance
  - A better means to track the capabilities of clients
    - When is new parameter understood by "enough" clients
    - How to trigger tech-refresh at the client end
  - A definitive BCP document!

Compliance

- Increasingly procurements want compliance with standards
  - This is why a definitive BCP is needed
- RFCs are written as guidance - e.g., RFC 4641's discussion contradicts itself because it isn't a "BCP"
  - From looking at the survey and asking, would the TLDs "conform" to various RFC documents? The answer in some cases is no
  - There's no deficiency, it's that some documents are not meant to specify operational behavior
- A set of clear requirements is beneficial to operators

Questions on Performance

- Quite a few choices made are without full understanding
  - Choices sometimes forced by limited tool selection
- There have been some significant bugs in cryptographic libraries that have caused some suboptimal choices
  - These have warped "conventional wisdom"
- Guidance on things like NSEC3 iterations, key exponents vs. bit lengths is needed
  - Recent blog entries and other discussions have started thoughts that perhaps operational parameters need to be adjusted, such as, how beneficial is larger and larger key exponent?

Tracking Client Capability

- There is some work in the IETF to do a form of this
  - Limited to cryptographic algorithm capabilities
- There are more elements that would be interesting
  - Such as the DS record hash algorithms
- The idea for this is just beginning to form
  - Can be it expanded to allow a client to reveal - not it's implementation - but what functionality it is built with?
  - Perhaps a list of RFC documents used in design and implementation?

Summary

- Protocol engineers and operators have different roles to play
  - Should add that the operators considered here are only domain name registry operators, there are other perspectives
- There's naturally a gap between the two functions
  - And because engineering takes a long time, the era of development is different from the era of deployment
- Neither "side" has a better viewpoint
  - The gap though exists and seeing it closed would be good
  - Retrospectives are not meant to find fault but to identify places for improvement

Related presentations

- APRICOT 2012
  - A description of the study, in greater detail
- ICANN 43
  - A summarized fashion, what a "middle of the road TLD" does
- IEPG (before IETF 84)
  - Compared the observations to RFC recommendations

- If you want pointers to these ask, otherwise these should be apparent from archives of the conferences

Questions?

- That's all I prepared...but there is a lot more of detail available

- Post-presentation comments to ed.lewis@neustar.biz