A Look Back at A Look Back Back

Reviewing Development of the Domain Name System, 1988 Paper by Paul Mockapetris and Kevin Dunlap

Edward Lewis

IDS 2018
13 July 2018
Why (Revisit the Past)?

○ An outcome of the discussion over ONION as a reserved domain name that is not a top-level DNS name
  ○ What is the relation of the DNS protocol and domain names?

○ What in history led to the current state of affairs?

○ Are there lessons from the past, overlooked issues still needing to be solved?

○ It's tempting to make this a history lesson, but the emphasis will be on points made, not the history
Basis of this talk

- Development of the Domain Name System

- Paul V. Mockapetris USC Information Sciences Institute, Marina del Rey, California

- Kevin J. Dunlap Digital Equipment Corp., DECwest Engineering, Washington


- Most of the base slide content after slide 5 is copy-&-pasted from the paper
From Whence We Came (In the 1980's...)

- Numerous naming (spaces) schemes in place
- Numerous naming (servers) systems in place

**Significance**
- There was no name space that the DNS was invented to exclusively implement
- Early goals for DNS included accommodating diverse naming systems, be inclusive via flexibility

**Observation/hypothesis:**
- **Building for inclusivity** rather than domination leads to a more stable system
From Whence We Came (In the 1980's...)

- Numerous naming (spaces) schemes in place
- Numerous naming (servers) systems in place

Significance

- There was no name space that the DNS was invented to exclusively implement
- Early goals for DNS included accommodating diverse naming systems, be inclusive via flexibility

Observation/hypothesis:

- Building for inclusivity rather than domination leads to a more stable system

Reminiscent of Ethernet vs. Asynchronous Transfer Mode
- *NIX/OS vs. others
- Open Systems vs. Proprietary Systems
Basic Assumptions of the DNS Design

- Be a replacement for HOSTS.TXT
- Maintained in a distributed manner
- "Tolerable" performance
- Provide extensible services
- Avoid trying to force a single style
- No obvious size limits
- Interoperate across the DARPA Internet and in as many other environments as possible
Basic Assumptions of the DNS Design

- Be a replacement for HOSTS.TXT
- Maintained in a distributed manner
- "Tolerable" performance
- Provide extensible services
- Avoid trying to force a single style
- No obvious size limits
- Interoperate across the DARPA Internet and in as many other environments as possible

Over time original limits have been "burned in" to DNS software and into surrounding systems
Basic Assumptions of the DNS Design

- Be a replacement for HOSTS.TXT
- Maintained in a distributed manner
- "Tolerable" performance
- Provide extensible services
- Avoid trying to force a single style
- No obvious size limits
- Interoperate across the DARPA Internet and in as many other environments as possible

The Global Public Internet is not the only DNS, but others seem to be forgotten in standards discussions.
Name Space Assumptions

- Size limits ..., limits could be easily changed
- Name space structure mirrors the structure of the organization controlling the domain.
- An administrative decision ... to make the top levels correspond to country codes or broad organization types
- **Case-insensitive manner**
- **Avoid a standard printing rule for names to encourage DNS encoding existing structured names**
  - Separated by dots in configuration files, but applications are free to do otherwise
- **Decouple structure of the tree from implicit semantics**
Name Space Assumptions

- Size limits ..., limits could be easily changed
- Name space structure mirrors the structure of the organization controlling the domain.
- An administrative decision has led to the top levels corresponding to country codes or broad organization types.
- Case-insensitive manner

Avoid a standard printing rule for names to encourage DNS encoding existing structured names
  - Separated by dots in configuration files, but applications are free to do otherwise

Decouple structure of the tree from implicit semantics

In retrospect, this was a bad idea. Should have left case-handling to the edges, consequence has been complicating matching, "IDN"
Name Space Assumptions

- Size limits could be easily changed.
- Name space structure mirrors the structure of the organization controlling the domain.
- An administrative decision to make the top levels in a case-insensitive manner.
- Avoid a standard printing rule for names to encourage DNS encoding existing structured names. Separated by dots in configuration files, but applications are free to do otherwise.
- Decouple structure of the tree from implicit semantics. The "printed" form is believed to be a standard, resulting in bad UI implementations.
- Some ccTLDs have marketed the "dot"
Name Space Assumptions

- Size limits...could be easily changed
- Name space structure mirrors the structure of the organization controlling the domain.
- An administrative decision...to make the top levels correspond to country codes or broad organization types.
- Case-insensitive manner
- Avoid a standard printing rule for names to encourage DNS encoding existing structured names.
  - Separated by dots in configuration files, but applications are free to do otherwise
- Decouple structure of the tree from implicit semantics.

The "underscore" names (started with SRV) are teasing at this assumption.

Other times "don't let the protocol shape the tree" referred to assumptions about where data would be stored.
About "CLASSes", "RRs" and "TTL values"

- The class field is meant to divide the database orthogonally from type and specifies the protocol family or instance.

- The decision to use multiple RRs of a single type rather than including multiple values in a single RR ... was not a clear choice... suited to use in a limited-size datagram environment.

- "The recommended TTL value for host names is two days."
About "CLASSes", "RRs" and "TTL values"

- The class field is meant to divide the database orthogonally from type and specifies the protocol family or instance.

- The decision to use multiple RRs of a single type rather than including multiple values in a single RR was not a clear choice... suited to use in a limited-size datagram environment.

- "The recommended TTL value for host names is two days."

Recommendations regarding timeliness are seemingly never heeded by operators...
Observations related to Root Servers

- Redundant, diverse implementations
- Typical traffic at each on the order of 1 q.p.s
- Queries are four types: all information (25 to 40%), host to address (30-40%), address to host (10 to 15%), and new style mail information called MX (less than 10%)
- The number of clients is falling as more adopt caching
- Static priorities for selecting which root server to use
- Load fluctuations driven by changes in code rather than population
  - 50% of traffic could be eliminated by improvements
  - The root servers refer 10-15% of queries
Observations related to Root Servers

- Redundant, diverse implementations
- Typical traffic at each on the order of 1 q.p.s
- Queries are four types: all information (25 to 40%), host to address (30-40%), address to host (10 to 15%), and new style mail information called MX (less than 10%)
- The number of clients is falling as more adopt caching
- **Static priorities for selecting which root server to use**
- Load fluctuations driven by changes in code rather than population
  - 50% of traffic could be eliminated by improvements
  - The root servers refer 10-15% of queries
Observations related to Root Servers

- Redundant, diverse implementations
- Typical traffic at each on the order of 1 q.p.s
- Queries are four types: all information (25 to 40%), host to address (30-40%), address to host (10 to 15%), and new style mail information called MX (less than 10%)
- The number of clients is falling as more adopt caching
- Static priorities for selecting which root server to use
- Load fluctuations driven by changes in code rather than population
  - 50% of traffic could be eliminated by improvements
  - The root servers refer 10-15% of queries

The idea that code drives load more than user activity...hmm.
Observations related to Root Servers

- Redundant, diverse implementations
- Typical traffic at each on the order of 1 q.p.s
- Queries are four types: all information (25 to 40%), host to address (30 to 40%), address to host (10 to 15%), and new style mail information called MX (less than 10%)
- The number of clients is falling as more adopt caching
- Static priorities for selecting which root server to use
- Load fluctuations driven by changes in code rather than population
  - 50% of traffic could be eliminated by improvements
  - The root servers refer 10-15% of queries

98% [initially] from Duane Wessels Wow, That's a Lot of Packets (2002)
Observations related to Root Servers

- Redundant, diverse implementations
- Typical traffic at each on the order of 1 q.p.s
- Queries are four types: all information (25 to 40%), host to address (30-40%), address to host (10 to 15%), and new style mail information called MX (less than 10%)
- The number of clients is falling as more adopt caching
- Static priorities for selecting which root server to use
- Load fluctuations driven by changes in code rather than population
  - 50% of traffic could be eliminated by improvements
  - The root servers refer 10-15% of queries

Referrals at 34% - Roy Arends (IDS 2017) – including repeated queries
Observations related to Root Servers

- Redundant, diverse implementations
- Typical traffic at each on the order of 1 q.p.s
- Queries are four types: all information (25 to 40%), host to address (30-40%), address to host (10 to 15%), and new style mail information called MX (less than 10%)
- The number of clients is falling as more adopt caching
- Static priorities for selecting which root server to use
- Load fluctuations driven by changes in mode rather than population
  - 50% of traffic could be eliminated by improvements
  - The root servers refer 10-15% of queries

Sebastian Castro's slides from 2010's 8th New Zealand Computer Science Research Student Conference

---

Using 2002 criteria, "legitimate" queries remained a constant small fraction
Section on Surprises

- It was thought that the semantics of the data was clear, it was not

- **Underlying network was much worse** than the original design expected, difficulty in making reasonable measurements of DNS performance

- The prevalence of "no" answers and the need to cache them
  - Initial monitoring of root server activity showed a very high percentage (20 to 60%) of these responses.
  - The search lists produce a steady stream of bad names
Section on Surprises

- It was thought that the semantics of the data was clear, it was not

- Underlying network was much worse than the original design expected, difficulty in making reasonable measurements of DNS performance

- The prevalence of "no" answers and the need to cache them
  - Initial monitoring of root server activity showed a very high percentage (20 to 60%) of these responses.
  - The search lists produce a steady stream of bad names

Notice when DNSSEC enlarged payload, anycast enlarged capacity and DNS became a "utility" for attackers
Section on Surprises

- It was thought that the semantics of the data was clear, it was not.

- Underlying network was much worse than the original design expected, difficulty in making reasonable measurements.

- The prevalence of "no" answers and the need to cache them.
  - Initial monitoring of root server activity showed a very high percentage (20 to 60%) of these responses.
  - The search lists produce a steady stream of bad names.

The reason Negative Caching of DNS Queries (DNS NCACHE) [RFC 2308] is one of the most significant extensions to DNS.
Section on Successes

- Caching – but one administrator reversed the TTL and data values, resulting in the distribution of bad data with a TTL of several years; security of the present system is questionable in an era of local networks and PCs.

- Flexible to accommodate "political" choice; such as to change to the ISO/CCITT directory service

- Datagrams (UDP) much better performance than achieved by TCP

- Variable depth hierarchy; to encapsulate any system; need to organize

- Additional section – to let responder anticipate the next request
Section on Successes

- Caching – but one administrator reversed the TTL and data values, resulting in the distribution of bad data with a TTL of several years; security of the present system is questionable in an era of local networks and PCs.

- Flexible to accommodate “political” choice; such as to change to the ISO/CCITT directory service.

- Datagrams (UDP) much better performance than achieved by TCP.

- Variable depth hierarchy; to encapsulate any system; need to organize.

- Additional section – to let responder anticipate the next request.

The domain name registration market is going away from this – flat is king now in the market.
Section on Successes

- Caching – but one administrator reversed the TTL and data values, resulting in the distribution of bad data with a TTL of several years; security of the present system is questionable.

- Flexible to accommodate “political” choice; such as to change to the ISO/CCITT directory service.

- Datagrams (UDP) much better performance than achieved by TCP.

- Variable depth hierarchy; to encapsulate any system; need to organize.

- Additional section – to let responder anticipate the next request.

Something we seem to have "lost" but should look at with DNSSEC...
Section on Shortcomings

- The type and class data specifiers, which were 8 bits in the draft, should be expanded; A **methodology or guidelines to aid in the design of new** types of information is needed.

- Needs to be **integrated into the operating system** to a much greater degree than providing system call to the resolver; specify search lists and defaults in a manner consistent with other system operations.

- **Demonstrate** operational capability before delegating the domain.

- Documentation should always be written with the assumption that **only the examples are read**.

- Software versions and parameters should be **accessible**.
Section on Shortcomings

- The type and class data specifiers, which were 8 bits in the draft, should be expanded; A **methodology or guidelines to aid in the design of new** types of information is needed.
- Needs to be integrated into the operating system to a much greater degree than providing system call to the resolver; specify search lists and defaults in a manner consistent with other system operations.
- Demonstrate operational capability before delegating the domain.
- Documentation should always be written with the assumption that only the examples are read.
- Software versions and parameters should be accessible.

2004: The TXT vs SPF "incident" Expert Reviews included in 2008 version of *Domain Name System (DNS) IANA Considerations*
Section on Shortcomings

- The type and class data specifiers, which were 8 bits in the draft, should be expanded; A methodology or guidelines to aid in the design of new types of information is needed.

- Needs to be integrated into the operating system to a much greater degree than providing system call to the resolver; specify search lists and defaults in a manner consistent with other system operations.

- Demonstrate operational capability before delegating the domain.

This has happened, and developments like Name Service Switch to integrate DNS with others.

But proper search list processing still plagues SSAC Advisory on DNS “Search List” Processing.
Section on Shortcomings

- The type and class data specifiers, which were 8 bits in the draft, should be expanded to aid in the design of new types of information.
- Needs to be integrated into the operating system to a much greater degree than providing system call to the resolver; specify search lists and defaults in a manner consistent with other system operations.
- **Demonstrate** operational capability before delegating the domain.
- Documentation should always be written with the assumption that **only the examples are read**.
- Software versions and parameters should be **accessible**.

**Of growing significance, lot of legacy resistance; as new operators are added, less "average experience"**
Section on Shortcomings

- The type and class data specifiers, which were 8 bits in the draft, should be expanded; A methodology or guidelines to aid in the design of new types of information is needed.

- Needs to be integrated into the operating system to a much greater degree than providing system call to the resolver; specify search lists and defaults in a manner consistent with other system operations.

- **Demonstrate** operational capability before delegating the domain.

- Documentation should always be written with the assumption that only the examples are read.

- Software versions and parameters should be **accessible**.
Section on Shortcomings

The type and class data specifiers, which were 8 bits in the draft, should be expanded. A methodology or guidelines to aid in the design of new types of information is needed. It needs to be integrated into the operating system to a much greater degree than providing system call to the resolver; specify search lists and defaults in a manner consistent with other system operations.

Demonstrate operational capability before delegating the domain.

Documentation should always be written with the assumption that only the examples are read.

Software versions and parameters should be accessible.
From the Conclusions

- Need to distribute functionality was, we believe, inexorable
- New functionality and opportunities must be key criteria
- Cache negative responses as well
- More difficult to remove functions than get new added
- Variations in the implementation is a great idea; allowing variation in the provided service causes problems.
- Implementors lose interest when system hits initial level
- Distributed software should include a version and table of parameters which can be interrogated
- Systems should include technical means for transferring tuning parameters, or at least defaults, to all installations without requiring the attention of system maintainers
From the Conclusions

- Need to distribute functionality was, we believe, inexorable
- New functionality and opportunities must be key criteria
- Cache negative responses as well
- More difficult to remove functions than get new added
- Variations in the implementation is a great idea; allowing variation in the service causes problems.
- Implementors lose interest when system hits initial level
- Distributed software should include a version and table of parameters which can be interrogated
- Systems should include technical means for transferring tuning parameters, or at least defaults, to all installations without requiring the attention of system maintainers

Concern: Will open source developers be able (financially) to continue to provide "long life" support for code?
Note: my bubble turns from an "interest" guided perspective to "economic"
From the Conclusions

- Need to redistribute functionality was, we believe, inexorable.
- New functionality and opportunities must be key criteria.
- Cache negative responses as well.
- More difficult to remove functions than get new added.
- Variations in the implementation is a great idea; allowing variation in the provided service causes problems.
- Implementors lose interest when system hits initial level.
- Distributed software should include a version and table of parameters which can be interrogated.
- Systems should include technical means for transferring tuning parameters, or at least defaults, to all installations without requiring the attention of system maintainers.

Allowing for benevolent outside intervention or a vulnerability for an active persistent threat to exercise?
Final Thoughts From and On the Paper

- Support for X.500 style addresses for mail, etc
- Tradeoffs between performance, generality, and distribution require at least different styles of use at different levels
- Research in naming systems - technical and/or political solutions to the growing complexity of naming will be a growing need.
- Conspicuously absent: Mention of active and/or persistent threats against the stability of the system
  - Not a surprise
  - But so completely absent, reflective of "the times"
Engage with ICANN

Thank You and Questions
Visit us at icann.org
Email: edward.lewis@icann.org

@icann
facebook.com/icannorg
youtube.com/icannnews
flickr.com/icann

linkedin/company/icann
slideshare/icannpresentations
soundcloud/icann
instagram.com/icannorg