The Internet Corporation for Assigned Names and Numbers

Identifier Technology Innovation Panel - Draft Report

Feb 21, 2014

Table of Contents

1. Introduction

The Identifier Technology Innovation (ITI) panel was chartered by the Internet Corporation for Assigned Names and Numbers (ICANN) with the following objectives:

1. Develop a technology roadmap for the Domain Name System (DNS) and other identifiers

2. Develop best practice recommendations and reference systems

3. Provide technology guidance to ICANN operations, security, policy, and technical functions

4. Engage with ICANN community and public on technology matters

The panel was selected during September and October 2013, with Paul Mockapetris as chair. All members served as individuals, with their affiliations for identification purposes only:

- Jari Arkko — Chair, Internet Engineering Task Force (IETF)
- Rick Boivie — IBM Thomas J. Watson Research Center
- Anne-Marie Eklund-Löwinder — Security Manager, The Internet Infrastructure Foundation
- Geoff Huston — Chief Scientist, Asia-Pacific Network Information Center
- James Seng — CEO, Zodiac Holdings
- Paul Vixie — CEO, Farsight Security
- Lixia Zhang — Postel Chair of Computer Science, University of California Los Angeles

In-person meetings were held at the IETF Vancouver (November 2013), ICANN Buenos Aires (November 2013), and at ICANN offices in Los Angeles (January 2014). The Buenos Aires Meeting was open to the public, and a summary of the panel's activities was also presented via two webinars in January 2014. Electronic discussions via email, et al supplemented these. The report will be available for public comment in February 2014, and finalized following the March IETF in London.

The chair wishes to thank the panel for all of their insights and ideas, and ICANN for supporting the panel. Thanks also go to Elise Gerich and Alice Jansen of ICANN who contributed ideas and support for all of the panel's work.

2. Panel Strategy

The title of the panel is no accident. The scope was extended beyond that of DNS per se in recognition of the growing importance of identifiers of all sorts to the Internet, as well as ICANN's role in management of other identifiers. A partial list of ICANN's current portfolio includes:

- Domain names
- Autonomous system numbers
- IPv4 internet addresses
- IPv6 internet addresses
- Multicast addresses
- Port numbers
- Protocol numbers
- Uniform Resource Identifier (URI) registry
- Management Information Base (MIB)
- Time zone database

However, in parallel with this expansion, the panel's timeframe was compressed from the original one-year timeframe to approximately six months. This had the effect of a more DNS-oriented focus than hoped for.

To compensate, the panel adopted the following principles:

- Try to document all ideas considered, but focus on a few
- Look for particular forcing trends (e.g. Internet expansion, trends in processor architecture)
- Look for "burning" needs
- Avoid focusing on "well-ploughed fields" (e.g. DNSSEC deployment, existing strategies for collisions) and look for novel ideas

The central purpose of the panel is to inform ICANN's strategic planning process. Although the panel did consider ideas that were close to the operational needs of ICANN, it did not restrict itself to ideas that would be implemented by ICANN per se. Implementation of many of the ideas discussed herein would most naturally fall in the IETF or elsewhere. A few of the ideas raise policy questions which we did not address other than to point them out.

Lastly, given the immense amount of activity in the identifier space, the panel merely sampled the space. The reader should not assume that we knew all ongoing activities, or that ideas not addressed herein are less important.

3. Roadmap

Identifiers are a hot area in the Internet community. In the short term new Top Level Domains (TLDs) will come online. Your Facebook account is looking to become your single sign-on credential for the Internet - as is your Google account. Over the long term, the research community has a lot of different projects including Content Centric Networking (CCN), Information Centric Networking (ICN), Named Data Networking (NDN), and many other variants. While they can't agree on a name for the field, they do all agree that content should be identified by name, not location, and that caching should be opportunistic. Other proposals have insisted that flat names are the wave of the future, and self-certifying names should be the basis of any new system.

Identifiers are central to any network in terms of uniquely identifying components of the network to all other components of the network. In addition, modern networks are not a single homogenous domain, but are constructed as an amalgam of a number of technologies, and there is a requirement to map between identity realms. This mapping function is performed in a number of ways. In the context of the Internet, one of the most visible identity realms is the domain name realm, which is a hierarchically structured name space. Associated with this name space is a mapping function that can map from domain names to other identities (such as IP addresses, for example). When we look at a road map for identifiers we need to be aware of the distinction between the identifier realm and the mapping function, and look at the roadmap for each.

In the current Internet, the panel identified several factors that will tend to expand the use of the DNS, as well as several that will act to contract it. These factors are not all technical, and the struggle seemed more Darwinian than based on elegance or some other virtue.

Current Expansion Factors

- The DNS enjoys a legacy advantage in that it is implemented in every device that touches the Internet. Simple growth in the existing base will expand its use. For example, an application that wants to pass through firewalls and be cached throughout the Internet finds the DNS as an existing base.
- The new TLDs will attempt to monetize their brands. While there's a lot of skepticism in the technical community, over a thousand new brands will be struggling to prosper, and there's likely to be innovation and several surprises.
- Emerging new capabilities, such as the security capabilities of Domain Name System Security Extensions (DDNSSEC) or DNS-based Authentication of Named Entities (DANE), may motivate additional use.
- New data in the DNS could expand its use, especially when combined with DNSSEC to guarantee authenticity. One panelist advocated publishing the "birthday" and "activity" of domains as basic reputation information. Other proposals have used the DNS as a registry of address blocks, etc. ICANN has restricted the use of some labels in domain names, and a real time

registry of such might be appropriate, particularly when paper specifications come in multiple alphabets.

Current Contraction Factors

- The DNS is the legacy standard, but that's also a handicap in that DNS logic embedded in WIFI access points, cable and Digital Subscriber Line (DSL) modems, firewalls, routers, and the software base of the Internet often limits the scope of use and restricts innovation. Implementations are often less than complete, current or compliant with the standards. These issues have hindered the implementation of DNSSEC and make implementation of any new DNS data types or features problematic. This leads to design practices such as limiting all use to address and text (TXT) records. This ossification is not unique to the DNS.
- There's a commercial interest in being in control ("owning") the search window and/or identifier space. The interest here is on seeing the user's intent in the free form and keeping it hidden from the open Internet. We noted the trend to devices that are hard coded to a specific DNS service, as well as proprietary extensions, as a path to Balkanization.
- Users favor a more powerful interface. Rather than entering DNS names, users and applications often employ search and other mechanisms to reach particular information. The Uniform Resource Locator (URL) bar in browsers is largely a search tool today, for instance. Today's user interface is the mobile device, which doesn't favor typing. Voice recognition and other types of Artificial Intelligence (AI) in the browser bar lead to incompatibilities between different vendors. As an example, an experiment by Geoff Huston (see contribution) observed the search triggered by "Geoff.Huston" in multiple browsers and observed virtually NO consistency between vendors. This lack of consistency may be tolerable in a browser search where the user is expected to vet the results, but can be dangerous in configuration files in systems – one of the concerns re collisions.

The sense of the panel was that while the DNS use may fade from the user interface, it's like to remain an infrastructural tool. One analogy was that the DNS is not paper facing the attack of eBooks, but rather a computer instruction set that is accessed through higher level languages.

Opinions differed about whether it was possible or advisable to seek a DNS renaissance or restructuring. The technology is discussed in the "DNS Fundamentals" section of this report. There's the policy question of whether ICANN should try and preserve and extend the DNS system. If so, how does one get a consistent architecture based on the various views of the ICANN constituency, the IETF (where the work would presumably be done) and other parties in the Internet?

The longer term

One set of ideas on the long term is the Named Data Networking model. Its key ideas are content access by name, digital authentication everywhere, opportunistic caching, and a flow scheme in which content requests and responses follow the same path. The model for routing queries is sometimes expressed as just using a name hierarchy for longest prefix match routing decisions which skeptics find unscalable. In any case software, hardware, and several network testbeds are implemented. The most obvious applications are content distribution, but advocates claim the model is good for process control, automotive networks, etc.

In a sense, the DNS was the first of the dirty slate alternatives to pure ICN, just like more current approaches [Fayazbakhsh 2013] that try and preserve just the most important parts of the ICN model. Importance here is in the eye of the beholder.

DNS retrieves data by name. It doesn't attempt to route by name and instead uses the Internet's addressing layer; this scheme fixes what some regard as the central scaling problem for ICN. DNS has been somewhat infamously known as a vehicle for tunneling video [Kaminsky 2004] and illicit tunneling of access through DNS queries that are performed before authentication by some WIFI access points. (Google "DNS tunneling" returns about 1,620,000 hits.)

ICN has longest prefix match and selectors which allow for media transport, facilities that were anticipated in the query section of the original DNS protocol spec, but never developed.

In any case, assuming that one could make DNS packets larger and add some additional query fields, the content services could be replicated in DNS. ICN's matching of authenticated requests and responses may be the best way to avoid DNS amplification attacks.

In conclusion, one could imagine an NDN scheme replacing the DNS, most likely starting as a superset of the DNS facilities in a transition that would take years or decades to complete. Any attempts to enhance the DNS architecture should feel free to borrow from NDN.

ICN is by no means the only model for the future, just one of the most developed. We believe that it's always useful to try and abstract out the basic principles and then study the composition. [Ghodsi2011] is a good example in the way it relates the trinity of name, real world ID, and Public Key Infrastructure (PKI).

More recently, an emphasis on distribution of control [Newyorker 2014] and privacy has surfaced, with the Namecoin system being the most well known example. The PKI that exists represents a resource to large scale surveillance and hence a problem for privacy. A mixture of self-certifying objects and an opt-in PKI or perhaps parallel PKIs and Peer to Peer (P2P) systems may be the answer.

4. Operations Issues

Several issues arise out of the day to day ICANN operations. These mostly revolve around the root.

4.1.Hardening the root

Given the central importance of the root infrastructure, there were several external suggestions that the panel look at trusted computing technology. The panel thought that there might be merit for this sort of technology in the systems used to edit and sign the root, but thought that looking at improving the distribution of signed data over commodity hardware was a better priority for the panel. The Snowden revelations raise some hardware security concerns which may not have been considered in the design of current systems, such as BIOS infections, hard drive spyware, et al [Spiegel 2014].

4.2.Replication

The DNS has always had two complementary mechanisms for distributing data: preplanned replication of zones, and on demand queries. From the point of view of an individual piece of DNS data, a resource record (RR), it starts out at its ultimate source as part of a zone, travels with that zone in one or more zone transfers, and then completes its journey to its ultimate destination when pulled via a query.

For example, the root zone is generated by ICANN in partnership with Verisign and the US Department of Commerce, and then distributed to all of the root servers via zone transfers. Conceptually, that distribution, like the distribution of any other zone in the DNS, can be done via any mechanism: magnetic tapes and Federal Express (FEDEX) deliveries, file transfers via the File Transfer Protocol (FTP) or Rsynch, or more optimally by incremental zone transfer which sends changes from a previous version rather than the whole zone. Copies can be either pushed via DNS notify or pulled via a polling strategy that looks for changes. Security for zone transfers can be done via DNS Transaction Signature (TSIG) and/or by any number of transport protocols, e.g. Internet Protocol Security (IPSEC), Hypertext Transfer Protocol Secure (HTTPS), etc. There are hundreds of instances of root servers with copies of the root zone.

When users want to access data in the root zone, they send queries to the root. The queries are routed by two mechanisms: first the destination IP address in the query identifies a set of root servers that share a common anycast address, and second the routing system decides which server in the anycast set will actually get the query. This scheme is the result of an evolution that started with 3 root servers with unicast addresses, then expanded to 13 root server organizations with load shared clusters, then the present scheme (with many smaller steps in between). In simplified terms, “13 root servers” are really “13 root server organizations” that eventually deliver the zone to hundreds or thousands of individual servers[1]. The reason we have only 13 root server organizations, and use anycast, is that it was far easier to do that than relax the size limitation of DNS User Datagram Protocol (UDP) packets. There are also other size problems related to adding IPv6 addresses. On the path from the root server to the user, security can optionally be provided by DNSSEC.

Over the years, the root servers have been subjected to attacks, mostly of the Distributed Denial Of Service (DDOS) variety. For such an attack to be successful against a particular user, it must disrupt queries to all of the anycast addresses of the 13 different root server organizations. Disrupting a subset will slow performance while the requester learns which root servers to avoid. The disruption can be either taking out the server or the network path to the server, typically with overload. So, for example, in one such attack users in California thought the root server in Stockholm was down, and in Stockholm

[1] Today two of the root server organizations are operated by the same entity, Verisign

users observed just the reverse. The response of the root server organizations to a recent threat by the *anonymous* hacker organization was to deploy more bandwidth, servers and fanfare.

Of course, the attack need not be directed against the root server constellation, it can be directed against the user's connection(s) to the Internet. While more limited in damage, the correlation of forces between an attacking botnet and a single enterprise is typically much more in the attacker's favor even for larger enterprises.

It has been the practice of some of the panelists to recommend to enterprises that they internally distribute copies of the root, **and any other critical zones**, so that during an attack, normal operation can continue, at least for DNS. ICANN makes it simple for any organization to get a copy of the root zone, and with a further bit of work to become a root server instance in ICANN's root server organization. It's also a good idea for an enterprise to be internally self sufficient with regard to DNS, and not threatened by lack of access to outside servers, or actions by one's registry, registrar, root server operators, etc, whether by accident or intent.

Given DNSSEC, we have a way to distribute a zone which can be verified using embedded digital signatures. We believe that the principle can be further extended, for example by protecting the delegation and glue data. It may also be possible to eliminate or reduce the root server organization and address data. One scheme, described in detail in Paul Vixie's contribution, is included in the Contributions section of this report.

There are also significant political aspects. There are 13 root server organizations, and several countries feel that they are left out, even if they can have as many ICANN root server instances in their country as they care to install. (Not to mention that several of the other root server organizations are willing to have their anycast constellations extended.) So let's just make the issue go away.

It should be noted that there's no technical need to replace the existing root server system for those that prefer it; let's merely make replication easier for the root, and also set an example for other zones.

4.3.Shared Zone Control

In the previous section, we discussed the political feelings that make countries want to own a root server organization. These concerns may or may not be well founded, but there is no question that the current root operation is based in the US and subject to US jurisdiction.

In simple outline, the root is updated in a sequence:

- ICANN receives update requests from TLDs, and vets them for errors
- ICANN submits the changes to the Department of Commerce
- ICANN sends approved changes to Verisign
- Verisign generates a signed root and distributes it

Is there a technical way to think about sharing control over the root? Some theories have been advanced. One school of thought is that data should have N multiple signatures. And then M/N, signatures are required to authenticate the data. Of course there are arguments about M and N, and whether different crypto is needed/desirable.

It's not our intention to argue for a specific system here, but we do feel that a good design could allow the political process of deciding how control should be shared to start. Our vision is the creation of a toolbox for shared zone control, not only for the root, but also for other zone coordination problems. We note that the DNS Operations (DNSOPS) working group in the IETF has two proposals for coordinating DNSSEC signing information, but wonder if it might be better to create a general facility rather than a solution to this point problem. Coordination of forward and reverse addresses might be another application.

So what's required? We speculate that the right model is one in which all of the parties sharing control have a set of capabilities:

- A system for initiating a shared zone consisting of the zone itself, rules, and individual journals for each of the participants to post their requests and actions
- Each type of request is visible to all of the other participants who can approve, disapprove, or timeout
- Rules define what happens to a request
    - One type of a rule is a vote which defines the conditions for a request to succeed. This might include a delay for all parties to have time to consider the request.
        - For ccTLDs the WSIS rules would dictate 1 of N, so each Country Code Top Level Domain (ccTLD) could unilaterally change its own data.
        - Other domains might use a simple majority
    - Specified delays could be important so that others might be able to point out operational issues and let the requesters reconsider
    - Different conditions might apply for different operations, such as creating a new vs. editing, etc.

The participants could then each do a standard algorithm to generate consistent state. This might seem like a fantasy, but Byzantine algorithms like Bitcoin [Andreesen 2014] and Namecoin show that such systems are possible today.

(Note that we aren't proposing the rules, just a distributed system for implementing whatever rules the community wants.)

4.4.Registry/Registrar Operations

Some panelists argued that ICANN operations should provide service level guarantees, but the panel didn't feel this was an issue it could progress.

4.5.What Data Should ICANN publish?

4.5.1. ICANN Parameters

ICANN has many sets of parameters it manages as part of the Internet Assigned Numbers Authority (IANA) functions as well as the new TLD process and elsewhere, for example reserved labels in multiple languages. All of these should be made available online, perhaps in the DNS, and certainly in secure form, so they can be directly used by anyone in the Internet community.

4.5.2. Domain Birthdays, Activities, and Bailiwicks

DNS reputation is a valuable security tool. The date of creation of a domain is perhaps the single most indicative piece of information. Another is a domain's update rate for server names and addresses. New domains and high update activity are suspicious. It would be desirable for this information to be available in real time.

Bailiwick information was discussed in a similar way, but will be taken up by the IETF in their next meeting in London March 2014.

4.5.3. The LISP example

Early on the panel was asked to consider having ICANN support a Super-root service for the Locator/Identifier Separation Protocol (LISP) [RFC 6830]. As explained to us by Dino Farinacci et al, ICANN would run LISP servers as an experimental service to refer requests to existing LISP servers that do not currently offer universal connectivity. We located resources for four servers, but the project never started due to some unresolved issues:

- What would be the scope (duration, etc) of the experiment? What were the criteria for success?
- What software would be used and who would support it? Two proprietary alternatives were available.
- Who would have policy and operational control?
- Should ICANN be doing such a thing or the Regional Internet Registries (RIRs)?
- Would the answer change if IP addresses were not involved?

The LISP materials are attached as an appendix. No action was taken on this experiment.

Some of the panel felt that "LISP is just one instance of a more generic class of transport tunneling technologies, and as such did not present any novel identifier management tasks that fell outside of current operational identifier management practices, and therefore the case that this particular form of tunneling required particular attention and support from ICANN was not clearly substantiated."

ICANN should anticipate the policy and technical questions around new identifiers will arise again, and plan accordingly.

4.6. Collisions

Many of the panelists were familiar with the DNS collision issue, and while there was a lot of discussion about the issue, no substantial new directions arose. The panel did feel that real world prototyping of the system described in [ICANN 2013] is highly recommended.

5. DNS Protocol Fundamentals

Can we imagine a fundamental revision, upgrading or renaissance in the DNS? Many, including some panel members believe that the installed base is too resistant, or that the process is broken, or starting over is the right idea.

Surprisingly, the panel was unanimous is thinking that an effort to characterize the issues, and look for solutions was worthwhile; perhaps if only to put the issue to rest. In this section we outline some of the issues that would have to be studied if a broader effort was to be undertaken.

The history of innovation in the DNS has had its successes and failures. One of the central lessons is that technology only gets widely adopted if it provides a specific benefit. Administrators are careful to keep their zones connected to the global DNS and their A and MX records up to date otherwise they get no mail or web traffic. But of the 60 or so record types that have been defined, fewer than 10 see wide use.

Efforts to create application have faced similar difficulties.

The first set of DNS RFCs suggested a method for routing mail to specific mailboxes, but was never implemented. A second scheme, the MX RR, solved the problem of providing redundant mail servers as well as providing mail routing through organizational boundaries – it is the basis of mail routing today. Anti-spam databases were widely adopted without standardization. Competing standards effort for mail authentication led to two implementation using TXT RRs, and a debate about whether standardizing new types would ever be useful.

The E.164 NUMber mapping (ENUM) effort to standardize phone and other media routing using the DNS also had very limited success. Even though the Name Authority Pointer (NAPTR) technology is seen as a real innovation, the ENUM designers ignored the need to route on information other than destination phone number, and the equipment manufacturers preferred to keep the value in their proprietary systems.

5.1.Overall principles

Any new design should:

- Remove size limitations – the 576 byte Maximum Transmission Unit (MTU) has probably done more to retard the DNS than any other single factor; DNSSEC doesn't fit and Extension Mechanism for DNS (EDNS0) notwithstanding, a lot of hardware and software won't pass large packets.
- Preserve connectivity
- Try to foster consistent implementations – If different implementers don't follow the specs, then the user gets restricted to whatever common overlap exists
- Allow for future expansion
- Provide incentives for adoption

5.2.Data Model

The early DNS RFCs imagined parallel name spaces for different "classes" of information, and new data types built out of simple components. The class notion was never explored. New data types were defined, but more recently many have argued for using the generic TXT record meant for arbitrary text strings to carry the data, together with another level of label as a surrogate for the RR type.

We would argue that either the DNS should define its own RR types and formats in metadata carried in the DNS, or we should formalize child labels as the last data type and extend querying to allow for more flexible matching.

Lastly we need to explore self-signed data objects that can exist independent of domain name.

5.3. Distribution

The zone structure of data and caching by the resource record is implemented with somewhat uneven "improvements" to the Time To Live (TTL) standard, and prefetching of expiring information. It may be worthwhile to consider new ways to group data with serial numbers that could refresh groups of cached data without actually transferring the data.

We also think security could be improved by more frequent replication of (possibly smaller) zones. This data does not need to be secured by DNSSEC, and hence can improve security in places where DNSSEC isn't implemented.

5.4. Application Program Interface (API)

The DNS API comes in two forms: a user interface and names at the API level. In both cases we would benefit from a standard syntax that allows an explicit Fully Qualified Domain Name (FQDN). The user community would be better served by a consistent set of search policies across UIs, but it isn't clear there's any way to get vendors to do this.

The programming API has gone through several attempted revisions, mostly failures. Recently, we saw a presentation by Paul Hoffman about a new design featuring asynchronous interfaces and DNSSEC support. See appendix. We understand the work is now ongoing at Verisign Labs and NLnet, but haven't been able to obtain further information, though a release is said to be imminent.

But regardless of the API, there is a related question of where DNSSEC validation and DNS filtering (if any) should be performed. The panel was unanimous that technically DNSSEC termination should be allowed in the end system (which could be a virtual machine, a laptop, a server in the user's environment, etc depending on the user's preference) despite the fact that this might be impossible due to router, firewall or other legacy restrictions. Similarly, while DNS filtering is not everyone's choice, it should be under user control.

None of this should mean that the user is forbidden from outsourcing these tasks to and ISP or other service.

Policy and legal constraints may say otherwise.

Query Protocol

5.5. Query Protocol

The DNS query protocol has two types of Issues: those relating to the transport of queries/responses from a requester to a server, and second enlarging the power of the query.

The original UDP transport issues start with the traditional 576 byte MTU limitation. The original fix was to fall back to TCP for larger transfers. The size of the root's data was perhaps the first place where MTU limitations had a very widespread impact leading to the 13 root server limit; later the addition of DNSSEC signatures substantially expanded reply packets. EDNS0 was conceived to solve this problem, among others, with some success. But there are other limits such as the 1582 Ethernet frame size, or IPv6's 1280, etc, which fundamentally limit UDP.

Also EDNS0 can't solve the problem of access points, routers, firewalls, and other hardware that block access to TCP port 53, or limit packet size, or even intercept DNS requests in transparent proxies, often to the detriment of the service. Similar problems can exist in caching name servers which don't support large packets, all DNS data types, EDNS0, etc. Some problems can be quite subtle. In one example, DNSSEC packets would normally pass but not during DNSSEC key rollover, a normal maintenance process, when packets are slightly larger.

A related problem is DNS DDOS attacks, particularly using reflection and amplification. In these cases, you want some way to identify legitimate traffic from attack traffic. Source address validation would solve a significant part of the problem, both for DNS and many other protocols. The panel supports it, but isn't widely deployed. Rate shaping and various heuristics can help, but are hardly a definitive solution. Various lightweight authentication mechanisms have been and remain as candidates.

One school of thought about solving the transport problem is to put all DNS traffic in https:. The logic is that everyone has a vested interest in seeing secure web traffic flow, and hence it is a guaranteed path (some say the ONLY guaranteed path). The price is connection state and the related overhead. The alternatives involve some new transaction protocol or way of using UDP, both of which may not work in parts of the installed base. In either case there is the issue of whether the DNS transactions use a traditional or new format.

Regardless of the transport, the DNS query protocol should be expanded to allow for more flexible queries. These could include some sort of access control to successor labels in lieu of NSEC.

The research world protocols such as CCN learned from the DNS and incorporate all of these features. The problem is more one of figuring out how to motivate an upgrade of the existing infrastructure with some backward compatibility, rather than some new breakthrough in protocol science.

6. Observations and Recommendations

- DNS use in the infrastructure will continue to grow; DNS use in the User Interface (UI) is challenged by search-based alternatives, mobile interfaces, etc.
- ICANN should publish more DNSSEC signed data for reserved labels, etc.
- In cooperation with IETF et al, do a study to define an architectural vision for DNS in 2020.
- Design & prototype open root publication.
- Design a shared zone control system for the root.
- Perform collision exercises to test the ease of implementing [ICANN 2013].

7. References

[Andreesen 2014] Andreesen, “Why Bitcoin Matters”, http://dealbook.nytimes.com/2014/01/21/why-bitcoin-matters

[DNS/TCP] https://lists.dns-oarc.net/mailman/listinfo/tcp-testing

[Fayazbakhsh 2013] Fayazbakhsh et al, “Less Pain, Most of the Gain: Incrementally Deployable ICN”, Sigcomm 2013

[Ghodsi 2011] Ghodsi et al, “Naming in Content-Oriented Architecture”, Sigcomm 2011

[Huston 2013] DNS-over-TCP-only study.

http://www.circleid.com/posts/20130820_a_question_of_dns_protocols/
and the ensuing thread on dns-operations

[ICANN 2013] “Guide to Name Collision Identification and Mitigation for IT Professionals”, https://www.icann.org/en/about/staff/security/ssr/name-collisionq-mitigation-05dec13-en.pdf

[Kaminsky 2004] D. Kaminsky, “Tunneling Audio, Video, and SSH over DNS”, BlackHat 2004

[Merit] Sections about domains and DNS

http://www.afnic.fr/en/about-afnic/news/general-news/6391/show/the-internet-in-10-years-professionals-answer-the-afnic-survey.html

[Mockapetris 88] P. Mockapetris and K. Dunlap, “Development of the Domain Name System”, SIGCOMM 88

[Newyorker 2013]

http://www.newyorker.com/online/blogs/elements/2013/12/the-mission-to-decentralize-the-internet.html?goback=%2Egde_1430_member_5817512945197801473#%21

[RFC 881] J. Postel, “The Domain Names Plan and Schedule”, November 1983

[RFC 882] P. Mockapetris, “Domain Names – Concepts and Facilities”, November 1983

[RFC 883] P. Mockapetris, “Domain Names – Implementation and Specification”, November 1983

[RFC 1034] P. Mockapetris, “Domain Names – Concepts and Facilities”, November 1987

[RFC 1035] P. Mockapetris, “Domain Names – Implementation and Specification”, November 1987

[Spiegel 2014] http://www.spiegel.de/international/world/nsa-secret-toolbox-ant-unit-offers-spy-gadgets-for-every-need-a-941006.html

8. Glossary

AI	Artificial Intelligence
API	Application Program Interface
CCN	Content Centric Networking
ccTLD	Country Code Top Level Domain – a TLD assigned to a particular country
DANE	DNS-based Authentication of Named Entities
DDOS	Distributed Denial Of Service
DNS	Domain Name System – The Internet’s naming system
DNSOPS	DNS Operations – an IETF working group concerned with DNS Operations issues et al
DNSSEC	Domain Name System Security Extensions
DSL	Digital Subscriber Line
E.164	an ITU-T recommendation, entitled *The international public telecommunication numbering plan*, that defines a numbering plan for the world-wide public switched telephone network (PSTN) and some other data networks
EDNS0	Extension Mechanism for DNS [RFC 2671] – A standard for extending the size and fields of the original DNS specifications
ENUM	E.164 NUMber mapping - a system for unifying the international telephone number system of the public switched telephone network with the Internet addressing and identification name spaces, for example to route a phone call
FEDEX	Federal Express
FQDN	Fully Qualified Domain Name
FTP	File Transfer Protocol
gTLD	Generic Top Level Domain – a TLD which does not correspond to a country code
HTTPS	Hypertext Transfer Protocol Secure
IANA	Internet Assigned Numbers Authority
ICANN	Internet Corporation for Assigned Names and Numbers
ICN	Information Centric Networking

IEEE Institute of Electrical and Electronics Engineers

IETF Internet Engineering Task Force

IP Internet Protocol

IPSEC Internet Protocol Security

IPv4 Internet Protocol version 4

IPv6 Internet Protocol version 6

ITI Identifier Technology Innovation – An ICANN strategic panel

LISP Locator/Identifier Separation Protocol [RFC 6830]

MIB Management Information Base

MTU Maximum Transmission Unit – The size of the maximum data unit that can pass, or pass without fragmentation.

MX Mail eXchange – A DNS data type that specifies the mail exchange that handles mail for a
specific
domain

NAPTR Name Authority PoinTeR – A DNS data type most commonly used in Internet telephony

NDN Named Data Networking

P2P Peer to Peer

PKI Public Key Infrastructure

RFC Request For Comments – Memos which document technical and operational Internet issues

RIR Regional Internet Registry – One of the organizations that manage the allocation and registration of Internet number resources within a particular region of the world. For example, ARIN, the American Registry for Internet Numbers handles Canada, the United States, and many Caribbean and North Atlantic islands.

Rsynch Remote synchronization protocol – synchronizes files and directories while minimizing data transfer by using delta encoding.

RR Resource Record – the atomic unit of information in the DNS

TSIG Transaction Signature

TTL Time To Live

TXT	The text RR type in the DNS which allows free format text fields
UDP	User Datagram Protocol – the Internet’s connectionless datagram protocol
UI	User Interface
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
WIFI	Wireless Fidelity – the wireless network standards defined by the IEEE 802.11 standards family

9. Contributions by Panel Members

Note that all contributions are verbatim as submitted by the individual.

9.1.Contribution from James Seng

Technical Architecture

The hacker inside me likes decentralization architecture. It could be argue that much of the "political problems" we have today derives from the centralized nature of the DNS with the root.

So technology like namecoins or other decentralized identifier system intricates me.

However, there is no decentralized-but-coordinated identifier system that I know that actually been widely used. So like it or not, DNS system is still one of the deployed identifier system we have. As we do in IETF, it is the "running codes" that wins, not necessary that the best designed.

I do not believe in multi-root, or alternative root. As I said in Buenos Aires, I stand behind RFC 2826. Multi-root, alternative root and all related proposal only moves the political problem to another layer, but does not solve the fundamental political problem. Note I said, political problem because I do not think multi-root solve any technical problem at all; If anything else, it only increase the technical complexity

ICANN

The DNS and its centralized nature of the root resulted in part of the original simple IANA function operation to become the huge organisation called ICANN today.

I have participated in ICANN since the first meeting in 1999 and have almost attended everything single one. Over these years, there are things which I wish ICANN could have done differently, i.e., our position is not always aligned.

However, ICANN is the "running code" of the coordination of the DNS identifiers. Perhaps there are other better design, maybe more simple and elegant (as many in the IETF community wish we could go back to the days of Jon Postel), but it is what it is today, and most importantly, altho it could be better, it works. The alternative proposed (ITU) that we know have other problems or worst.Â

So I support ICANN because it is just the best working system we have for coordination of DNS identifiers and the root.

Extension of DNS and its system to other areas

Hence, I have little interest to redesign DNS or alternative proposals to naming identifiers. Eventually, someone, some organization has to exists to do the coordination and we will face with the same political problems all over.

I support and I like to see the DNS ecosystem (DNS standards, root operation, ICANN, ...) we have that has originally designed for DNS and evolved to extend to other areas (e.g. RFID), so more community can be fold. The work we did on IDN, in some sense, is folding a group of community of users who needs to use their own native language into the DNS ecosystem; instead of letting them build out their own.

While some have argue with me that if we have done IDN outside of DNS ecosystem, the deployment could have been much faster (e.g. see Native Language Keywords), I say IDN are also better because it is part of the DNS ecosystem, where there are well-defined open standards, open implementations, companies that builds on the legitimacy of DNS, and similar the protection of the IDN registrants and end-users.

As such, I have no qualms and I support to explore how we can extend DNS into identifiers that it wasn't original intended to designed for. Engineers designing identifiers are often naive about the politics that comes with identifiers, esp. if such identifiers are expose to end-users. They could learn a thing or two from the history of DNS identifiers and ICANN.Â

Politics of Root

The politics of ICANN, and how many views ICANN as part of the "Internet Governance" comes from ICANN role in the coordination of the root servers.

To make it worst, 11 out of the 13 roots servers are based in US, due to historical accident, but nevertheless makes the perception of ICANN been in the control of US even worst, esp. in these days of post-Snowden.

Whenever someone comes about and talks about such and such country should have a root server, we deflect it using historical or technical reasons that there is no way to extend beyond 13 roots.

History, I can accept as a reason.

Technically reasons, I cannot. It is more of an excuse because I have not been aware that of any effort IETF seriously looking how to extend beyond 13 roots. This is why I said during Buenos Aires that I can think of a couple of technical solutions, at least suffice as an I-D. We cannot let ICANN continue to use IETF / technical reasons as an excuse for the political problems they face. We should be able to tell ICANN, yes it can be done, but the politics to do it or not is for you to decide.

Beside, more importantly, root servers operation isn't hype it to be.

Having a root does not means so-and-so immediately have control over the Internet. In fact, it is as boring as an Anycast Root. Altho if the root operator does not follows some of the Best Practices of Root Server Operation (e.g. RFC 2010 and RFC 2870), then it can cause a lot of harm to the Internet.

Most engineers probably understood what I said above but most ICANNer don't.

So there are considerations when selecting a root server operator, because it is pinnacle to the stability of the Internet identifiers, and much of its are based on Trust. But Trust, like it or not, is not an engineering problem.

-James Seng

http://chineseseoshifu.com/blog/dnspod-in-china.html

Why DNSPod is useful in China, despite the way it "broke" the DNS.

9.2.DNS Resolution and Search List Application Behaviour - Geoff Huston

none – does NOT undertake any DNS lookup

never – looks up the base name, but does not apply the search list

pre – applies the search list, and if this returns NXDOMAIN then lookup the base name

post – lookup the base name, and if this returns NXDOMAIN then apply the search list

always – does NOT lookup the base name – only apply the search list

Base Operating system DNS resolver library behaviour

System	**Absolute** *server.*	**Relative Single Label** *server*	**Relative Multi-Label** *www.server*
MAC OSX 10.9	never	always	never
Windows XP	never	always	post
Windows Vista	never	always	never
Windows 7	never	always	never
Windows 8	never	always	never
FreeBSD 9.1	never	pre	post
Ubuntu 13.04	never	pre	post

Browser behavior on MAC and Windows platforms

MAC OSX 10.9

	server.	*server*	*www.server*
Chrome (31.0.1650.39 beta)	never	always	pre
Opera (12.16)	never	always	never
Firefox (25.0)	post*	always	post*
Safari (7.0 9537.71)	none**	none**	none**

* Added prefix of “www.”, then tried prefixing the “www.” and also appending the search list

** Safari seems to be aware of TLDs and does not perform DNS lookups when the name is not a TLD

Windows 8.1

	server.	*server*	*www.server*
Explorer (11.0.900.16384)	none	none	never
Firefox (25.0)	never*	always	never
Opera (17.0)	none	none	none**
Safari (5.1.7 7534.57.2)	never*	always***	never

* added a prefix of “www”

** OPERA is aware of delegated tlds, and only asks when the last label is a TLD

*** added a prefix of “www” and a suffix of “.com”

9.3. Observations on Consistency and Drift Contribution - Geoff Huston

If one were to look back to the origins of the Domain Name System, one finds the so-called “hosts file” as an early attempt to bring human use names into the context of computer networks. The ARPANET used a network node naming model where each connected node had a local configuration file, the hosts file, that contained the names of all other ARPANET nodes, and the protocol addresses of each node. There was no enforced consistency across these multiple instances of this hosts file across the set of ARPANET-connected nodes, nor, at the time, was there any method to distribute a copy of the hosts file across the network. The utility of this hosts file was to provide human-friendly names in place of the more obtuse protocol level addresses. Users were able to identify network nodes by their symbolic name, which was then translated into a protocol-specific binary address through a lookup into the hosts file. As the ARPANET grew, so did the size and update rate of the hosts file and the overhead of maintaining an accurate local hosts also grew. The hosts file format was standardised (RFC952) and a central hosts file service was defined (RFC953) which could take the place many local copies of the hosts file.

This was then replaced by the Domain Name System (DNS), originally specified in 1983 in RFC 882 and RFC 883. The mechanism of translation of a name, specified as a human friendly string, to a protocol-specific service address was maintained by the transition from the hosts file to the DNS.

This identifier space has a number of properties, including the observation that the DNS spans a name space that is suitable for use in human discourse, while at the same time admitting sufficient formal structure to allow names to be manipulated by computer applications in a deterministic fashion. The DNS name space is a hierarchically structure space, allowing for the name space to be efficiently searched for exact matches, and at the same time allowing for a framework of distributed management of name space. As long as label collisions are avoided within any individual zone of the DNS name hierarchy, name collisions can be avoided within the overall DNS name space, allowing name uniqueness to be readily managed within the context of the DNS. The DNS is flexible in terms of its mapping function, and can be used to map from a structured name space to any other form of named resource our service point. The DNS is intended to be consistent, in that, given a consistent name entry in the DNS, queries of that name should provide the same answer across varying locations of the querier and varying times of the query. This allows for referential consistency, in that a DNS name can be passed between parties and refer to consistent resource of service location. The DNS is not intended to replace a directory system or a search system. If there is an exact match of the name being queried in the DNS, the DNS query will return the mapped value as the result of the query, otherwise the query will return a match failure.

This model of the DNS name space as the identifier name space used to support a human interface with the network has since undergone a number of changes, principally in response to the mode of human use of identifiers in discourse. We tend to use identifiers in ways that are less precise, and in ways that include elements of local context, that use local languages and scripts, and over time the role of the DNS as a form of human interface to the network's resources and services has been subsumed by efforts to support interfaces that act in a manner that is more "natural" to human use.

RFC1034 proposed the use of a form of shorthand in the specification of DNS names, where names that did not end with a trailing '.' were termed "relative names", and, as noted in RFC1034, "relative names appear mostly at the user interface, where their interpretation varies from implementation to implementation." Typically, such local interpretation involved the placation of a local search list of label suffices, allowing the user to specify the initial part of a domain name, and relay on the local application or name resolution software routines to add a locally defined suffix to form a complete DNS name.

This form of selective occlusion of the DNS identifier space through the use of name suffixes was taken a further step in the user interface provided by web browsers, where common practice with web browsers was to take the DNS identifier component of a URL and apply a name transform of prepending the string "www." and adding a locally defined suffix (typically ".com."). In this way the identifier that the user specified, and the identifier name used in the subsequent DNS query were related, but not necessarily the same.

This use of local name transforms was further extended in the manner in which identifiers formed from language scripts other than US ASCII were mapped into the DNS (IDNs: RFC5891). Here was an explicitly defined process where the identifier entered by the user is transformed into an encoded label string that forms the DNS query. In this case the transform is precisely defined, so that multiple implementations of the IDN standard are intended to support a consistent view of the mapping of an identifier in a given script to an encoded DNS name form.

A further evolution of the refinement of the model of human interaction was the unification of search terms and URLs as input to browsers. In this case if the user has not used the complete specification of a URL to the browser, the browser will attempt to date.

9.4.Contribution from Paul Vixie

Universal Anycast for the Root Zone

Overview

We propose that IANA produce several additional forms of the DNS root zone, to allow universal anycast and operational research. "Universal anycast" in this context means a root zone whose apex NS records list only two name servers, whose associated "well known" addresses (as given by A and AAAA records) can be hosted by anyone. "Operational research" in this context includes wide scale public testing of IPv6-only root name service and wide scale public testing of "new gTLD" collision effects. This approach treats root name service as an unmanaged utility rather than as a managed utility.

Background

Universal anycast for the root zone could not be safely and responsibly deployed before the advent of DNSSEC, since without DNSSEC, any responding server could be configured with arbitrary DNS root data including new TLD's or re-delegated existing TLD's. With DNSSEC, it is now possible for recursive name server operators to configure DNSSEC validation, such that any gTLD information heard from a universal anycast root name server must be IANA-approved as indicated by DNSSEC signatures made with IANA's root zone signing key (ZSK).

Criticisms of the current and historical Root Name Server System include lack of resistance to DDoS attack, noting that even with the current wide scale anycasting by every Root Name Server Operator, there are still only a few hundred name servers in the world who can answer authoritatively for the DNS root zone. We are also concerned that reachability of the Root Name Server System is required even for purely local communication, since otherwise local clients have no way to discover local services. In a world sized distributed system like the Internet, critical services ought to be extremely well distributed.

Details

There are several useful variations to be constructed. First, basic universal anycast will allow any name server operator to capture traffic headed toward the root name server system and respond to it locally. IANA would generate and digitally sign (with DNSSEC) an additional version of the root zone that has a different set of NS records at its apex. These NS records will denote name servers whose addresses are not assigned to any particular Root Name Server Operator (RNSO) but are instead held in trust by IANA for use by any or all interested parties. IANA would request infrastructure micro-allocations from an RIR (such as ARIN or APNIC), as several IPv4 24-bit prefixes and several IPv6 48-bit prefixes, for use in universal anycasting of the root zone.

A second variation on the current root zone would provide universal anycast as above, but would denote name servers that had only IPv6 connectivity (indicated by the presence of AAAA records) and no IPv4 connectivity (as indicated by the absence of A records). This variation would facilitate operational research into IPv6-only networking.

A third variation on the current root zone would provide universal anycast as above, but would include delegations for all known new gTLDs including those not otherwise ready for delegation (such as .CORP and .HOME). These new gTLDs would be delegated to a name server operated by IANA itself, for measurement purposes. Each new gTLD will be assigned wildcard A and AAAA records, whose addresses will reach web servers operated by IANA for measurement purposes.

Impact

Given the hierarchical nature of Internet routing, anycast address blocks can be advertised at multiple levels. A virtual machine (VM) running on a laptop computer might have its own name server process which listens on the appropriate well-known addresses, in which case no root name service queries will leave that VM. The laptop computer itself might also capture outbound traffic aimed at these well-known addresses, which would serve other VM's or other processes running on that laptop computer. The wireless router upstream of this laptop might have servers listening on these addresses, in which case no root name server queries would leave that wireless LAN. The ISP might operate servers who listen on these well-known addresses, to serve any and all customers who do not operate their own servers. Finally, the global Internet is expected to have many operators who advertise routes to these well-known address blocks, not the least of whom would be the twelve existing root name server operators.

The positive impact of this would be greater potential resiliency, and reduced root name service latency. The negative impact of this would be reduced diagnostic capability, and the increased vulnerability to "route poisoning" or "hijacking" of root name service traffic. It is in any case vital that DNSSEC validation become common in order to reduce the payback for this kind of hijacking. We want the payoff for an attacker to be "victim loses root name service" rather than "victim sees a different DNS name space".

Examples

The following examples show the apex NS record set for each root zone variant, including address glue. This data would be included in a variant root zone before DNSSEC signing, and would also be published as a "root hints" file. The data shown for iana-servers.net would also be present in the real iana-servers.net zone. These examples would require four IPv4 micro-allocations and six IPv6 micro-allocations.

Variant 1: universal anycast

```
. IN NS anycast-1.iana-servers.net.

. IN NS anycast-2.iana-servers.net.

$ORIGIN iana-servers.net.

anycast-1 IN AAAA 2001:?:1::1

anycast-1 IN A ?.?.1.1

anycast-2 IN AAAA 2001:?:2::2

anycast-2 IN A ?.?.2.2
```

Variant 2: universal IPv6-only anycast

```
. IN NS v6only-1.iana-servers.net.

. IN NS v6only-2.iana-servers.net.

$ORIGIN iana-servers.net.

v6only-1 IN AAAA 2001:?:3::1

v6only-2 IN AAAA 2001:?:4::2
```

Variant 3: gTLD collision study anycast

```
. IN NS gtldstudy-1.iana-servers.net.

. IN NS gtldstudy-2.iana-servers.net.

$ORIGIN iana-servers.net.

gtldstudy-1 IN AAAA 2001:?:5::1

gtldstudy-1 IN A ?.?.5.1

gtldstudy-2 IN AAAA 2001:?:6::2

gtldstudy-2 IN A ?.?.6.2
```

10. Appendicies

10.1. LISP Materials

Introducing LISP to ICANN

Dino Farinacci
Elise Gerich
Terry Manderson
March 2013

Agenda

- What if ICANN (roughly by itself) could get IPv6 ubiquity?
- What if ICANN could create Internet goodwill by deploying a Mapping Database System?

Proposal

- Developing country wants addresses - there are no more IPv4 addresses

- They **can** get IPv6 addresses and connect to either dual-stack or IPv4-only data centers

- Facebook did this 2 years ago, their proven claim:

  "We could get IPv6-only clients anywhere in the world to connect to our IPv4-only servers. It took us 15 minutes to deploy this."

- How did Facebook do this?

LiSP is the Technology

World without LISP

- Is this John at the location ‘cisco’?
- Is this John at location ‘home’?
- Is this John at ‘Starbucks’?

Does my connection to John break because he changed locations?

Answer: Yes

chambers.cisco.com

↑

Application ID

198.133.219.25

↓ ↓

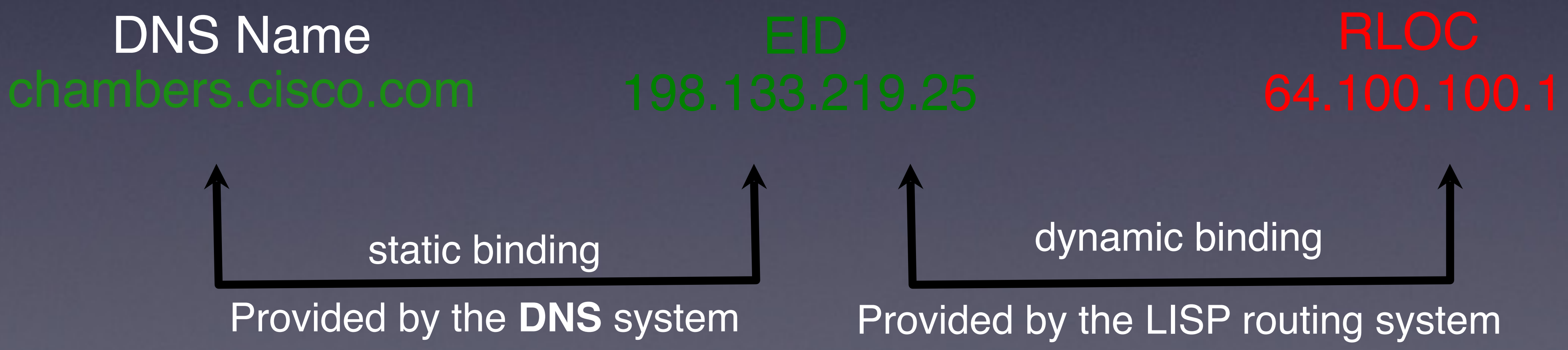
Network ID Network Location

Semantic Overload!

World with LISP

The service binds to an Application Name (**DNS**)
The Application Name binds to a Endpoint ID (**EID**)
The EID binds dynamically to a Routing Locator (RLOC)

1. Users use DNS names (the human "who")
2. Applications use EIDs (the network "who")
3. Routing uses RLOCs (the network "where")



DNS Name
chambers.cisco.com

EID
198.133.219.25

RLOC
64.100.100.1

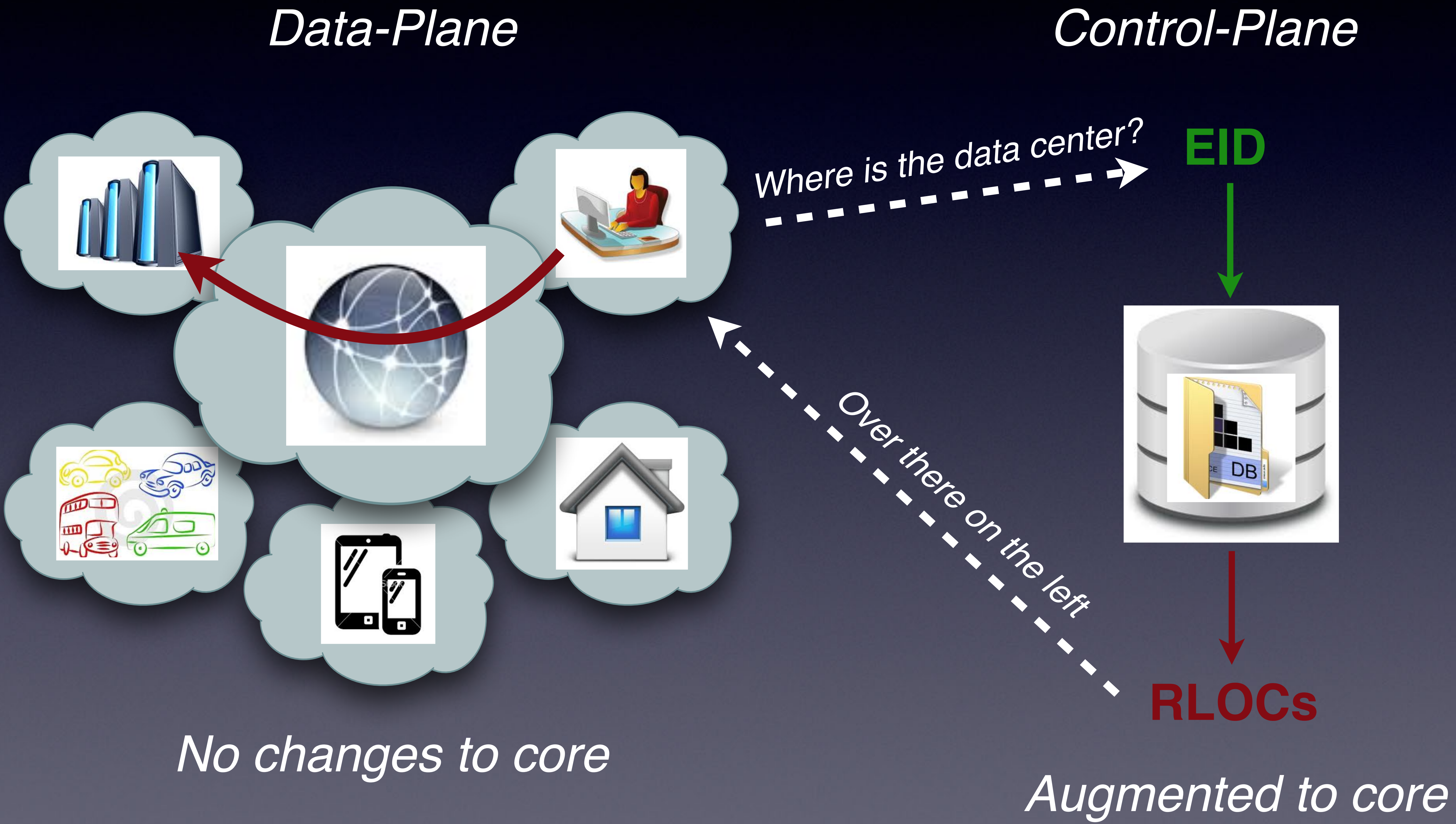
static binding

Provided by the **DNS** system

dynamic binding

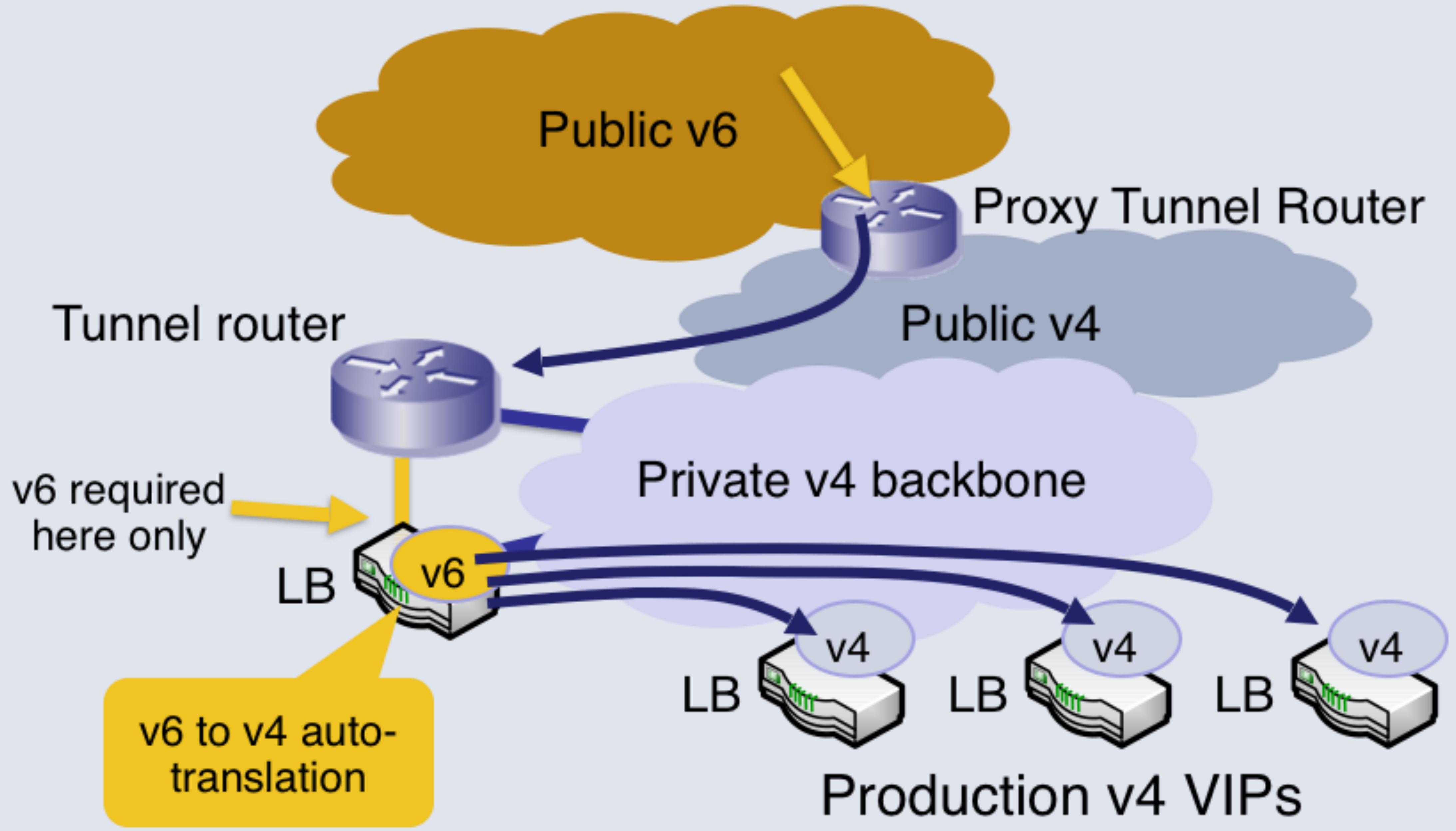
Provided by the LISP routing system

LISP


Data-Plane
Control-Plane
Where is the data center?
EID
DB
Over there on the left
RLOCs
No changes to core
Augmented to core

How facebook deployed LiSP

LISP for v6


Public v6
Proxy Tunnel Router
Public v4
Tunnel router
Private v4 backbone
v6 required here only
LB
v6
v6 to v4 auto-translation
LB
v4
LB
v4
LB
v4
Production v4 VIPs

www.lisp6.facebook.com results

Zero cost

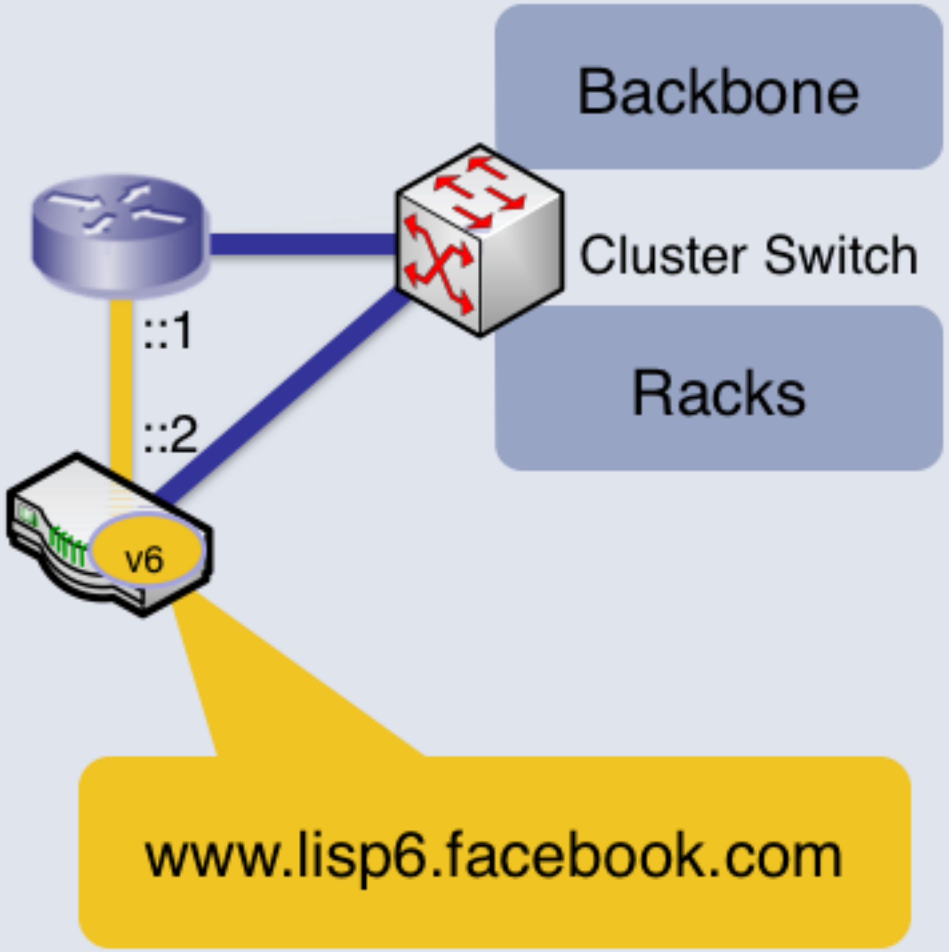
No v6 routing needed

Three v6 addresses configured

No other v6 deployed

Global v6 reachability

Same Facebook experience



What the Future Could Bring

- There are dozens of use-cases
  - Mobility, Incremental Security/Policy, Auto Industry
- Central to all of them is the Mapping Database System
- ICANN runs the global Mapping Database System that doesn't conflict with RIR policy

10.2. Hoffman API materials

Towards a new DNS API: Development Status of getdns

Paul Hoffman

June 2013

Overview

- Original design principles
- Development of the API in a vacuum
- Implementation status as of June 2013 (hint: Verisign)
- What's next
- Full spec: http://www.vpnc.org/getdns-api/

getdns API design principles

- Design by application developers, not DNS weenies
- Asynchronous / event-driven
- Mirror gethostbyname() where appropriate
- Multiple levels of DNSSEC support
- As future-proof as possible
- Assume a higher-level programmer than earlier APIs, but definitely not just experts

Developing an API without coding it at the same time

- The problem with having a parallel implementation process is that some suggestions get tossed early for being too hard
- We made some radical changes a few times during the design process; some of them would have been cut short
- Comes with a bunch of examples

What it looks like at the top level

- Async functions:
  - getdns_address() – close to getaddrinfo()
  - getdns_hostname() – close to gethostinfo()
  - getdns_service() – for SRV lookups
  - getdns_general() – can be all of the above
- There are also synchronous functions for systems that don’t have an async interface (or programmers who don’t care)

Async callbacks

- Original call has two arguments, one that the callback will not touch (for keeping local data) and a pointer that becomes a transaction ID
- Responses are allocated and come with a custom deallocator
- Biggest problem: they are specific to event libraries

Extensions to calls

- Extensions for common things that a developer might want to have in a response but might cause problems for other developers
  - DNSSEC (next slide)
  - Returning IPv4 and IPv6 in one call, which would slow down some applications
  - Getting OPT RRs in additional data
  - Non-IN classes
  - Others

Three extensions for DNSSEC

- Return the DNSSEC status of each record
  - Needed for DANE / TLSA which differentiates between different status type
- Only return secure results
  - Simpler for programs that are not going to report problems
- Return supporting responses
  - For programs that have internal validators or need to log all security status

Typical program flow

Includes for one or more regular C libraries

An include for the getdns library specific to the event library you use

Definition of your callback function

- Get the DNS data from the allocated pointer
- Process that data
- Check for errors

Definition of main()

- Create context
- *Set up your event base*
- *Point the context to your event base*
- Set up the getdns call arguments
- Make the getdns call
- Check if the getdns return is good
- Destroy the context
- Exit

Current implementation status

- Verisign is actively developing an implementation
- Unbound’s ldns is a major component
- Aiming at multiple platforms (*ix, Linux, Windows, ...)
- Open-source releases under a FreeBSD-like license

What’s next

- Maybe other implementations
- Once the Verisign implementation is available, there will likely be another round of design, hopefully for details
- Distribution for OSs, maybe built-in or added-on
- Then it could get interesting

Wild extrapolation

- Local DNSSEC validation
- Expansion of SRV uses
- DANE
- And...?