Internet Architecture Board (IAB)
Request for Comments: 8170
Category: Informational
ISSN: 2070-1721

D. Thaler, Ed.

May 2017

Planning for Protocol Adoption and Subsequent Transitions

Abstract

Over the many years since the introduction of the Internet Protocol, we have seen a number of transitions throughout the protocol stack, such as deploying a new protocol, or updating or replacing an existing protocol. Many protocols and technologies were not designed to enable smooth transition to alternatives or to easily deploy extensions; thus, some transitions, such as the introduction of IPv6, have been difficult. This document attempts to summarize some basic principles to enable future transitions, and it also summarizes what makes for a good transition plan.

Status of This Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

This document is a product of the Internet Architecture Board (IAB) and represents information that the IAB has deemed valuable to provide for permanent record. It represents the consensus of the Internet Architecture Board (IAB). Documents approved for publication by the IAB are not a candidate for any level of Internet Standard; see Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc8170.

Table of Contents

1. Introduction

A "transition" is the process or period of changing from one state or condition to another. There are several types of such transitions, including both technical transitions (e.g., changing protocols or deploying an extension) and organizational transitions (e.g., changing what organization manages a web site). This document focuses solely on technical transitions, although some principles might apply to other types as well.

In this document, we use the term "transition" generically to apply to any of:

- adoption of a new protocol where none existed before,

- deployment of a new protocol that obsoletes a previous protocol,

- deployment of an updated version of an existing protocol, or

- decommissioning of an obsolete protocol.

There have been many IETF and IAB RFCs and IAB statements discussing transitions of various sorts. Most are protocol-specific documents about specific transitions. For example, some relevant ones in which the IAB has been involved include:

- IAB RFC 3424 [RFC3424] recommended that any technology for so-called "UNilateral Self-Address Fixing (UNSAF)" across NATs include an exit strategy to transition away from such a mechanism. Since the IESG, not the IAB, approves IETF documents, the IESG thus became the body to enforce (or not) such a requirement.

- IAB RFC 4690 [RFC4690] gave recommendations around internationalized domain names. It discussed issues around the process of transitioning to new versions of Unicode, and this resulted in the creation of the IETF Precis Working Group (WG) to address this problem.

- The IAB statement on "Follow-up work on NAT-PT" [IabIpv6TransitionStatement] pointed out gaps at the time in transitioning to IPv6, and this resulted in the rechartering of the IETF Behave WG to solve this problem.

More recently, the IAB has done work on more generally applicable principles, including two RFCs.

IAB RFC 5218 [RFC5218] on "What Makes for a Successful Protocol?" studied specifically what factors contribute to, and detract from, the success of a protocol and it made a number of recommendations. It discussed two types of transitions: "initial success" (the transition to the technology) and extensibility (the transition to updated versions of it). The principles and recommendations in that document are generally applicable to all technical transitions. Some important principles included:

1. Incentive: Transition is easiest when the benefits come to those bearing the costs. That is, the benefits should outweigh the costs at *each* entity. Some successful cases did this by providing incentives (e.g., tax breaks), or by reducing costs (e.g., freely available source), or by imposing costs of not transitioning (e.g., regulation), or even by narrowing the scenarios of applicability to just the cases where benefits do outweigh costs at all relevant entities.

2. Incremental Deployability: Backwards compatibility makes transition easier. Furthermore, transition is easiest when changing only one entity still benefits that entity. In the easiest case, the benefit immediately outweighs the cost, so entities are naturally incented to transition. More commonly, the benefits only outweigh the costs once a significant number of other entities also transition. Unfortunately, in such cases, the natural incentive is often to delay transitioning.

3. Total Cost: It is important to consider costs that go beyond the core hardware and software, such as operational tools and processes, personnel training, business model (accounting/ billing) dependencies, and legal (regulation, patents, etc.) costs.

4. Extensibility: Design for extensibility [RFC6709] so that things can be fixed up later.

IAB RFC 7305 [RFC7305] reported on an IAB workshop on Internet Technology Adoption and Transition (ITAT). Like RFC 5218, this workshop also discussed economic aspects of transition, not just technical aspects. Some important observations included:

1. Early-Adopter Incentives: Part of Bitcoin’s strategy was extra incentives for early adopters compared to late adopters. That is, providing a long-term advantage to early adopters can help stimulate transition even when the initial costs outweigh the initial benefit.

2. Policy Partners: Policy-making organizations of various sorts (Regional Internet Registries (RIRs), ICANN, etc.) can be important partners in enabling and facilitating transition.

The remainder of this document continues the discussion started in those two RFCs and provides some additional thoughts on the topic of transition strategies and plans.

2. Extensibility

Many protocols are designed to be extensible, using mechanisms such as options, version negotiation, etc., to ease the transition to new features. However, implementations often succumb to commercial pressures to ignore this flexibility in favor of performance or economy, and as a result such extension mechanisms (e.g., IPv6 Hop-by-Hop Options) often experience problems in practice once they begin to be used. In other cases, a mechanism might be put into a protocol for future use without having an adequate sense of how it will be used, which causes problems later (e.g., SNMP’s original ’security’

field, or the IPv6 Flow Label). Thus, designers need to consider whether it would be easier to transition to a new protocol than it would be to ensure that an extension point is correctly specified and implemented such that it would be available when needed.

A protocol that plans for its own eventual replacement during its design makes later transitions easier. Developing and testing a design for the technical mechanisms needed to signal or negotiate a replacement is essential in such a plan.

When there is interest in translation between a new mechanism and an old one, complexity of such translation must also be considered. The major challenge in translation is for semantic differences. Often, syntactic differences can be translated seamlessly; semantic ones almost never. Hence, when designing for translatability, syntactic and semantic differences should be clearly documented.

See RFC 3692 [RFC3692] and RFC 6709 [RFC6709] for more discussion of design considerations for protocol extensions.

3. Transition vs. Coexistence

There is an important distinction between a strict "flag day" style transition where an old mechanism is immediately replaced with a new mechanism, vs. a looser coexistence-based approach where transition proceeds in stages where a new mechanism is first added alongside an existing one for some overlap period, and then the old mechanism is removed at a later stage.

When a new mechanism is backwards compatible with an existing mechanism, transition is easiest because different parties can transition at different times. However, when no backwards compatibility exists such as in the IPv4 to IPv6 transition, a transition plan must choose either a "flag day" or a period of coexistence. When a large number of entities are involved, a flag day becomes impractical or even impossible. Coexistence, on the other hand, involves additional costs of maintaining two separate mechanisms during the overlap period, which could be quite long. Furthermore, the longer the overlap period, the more the old mechanism might get further deployment and thus increase the overall pain of transition.

Often the decision between a "flag day" and a sustained coexistence period may be complicated when differing incentives are involved (e.g., see the case studies in the Appendix).

Some new protocols or protocol versions are developed with the intent of never retiring the protocol they intend to replace. Such a

protocol might only aim to address a subset of the use cases for which an original is used. For these protocols, coexistence is the end state.

Indefinite coexistence as an approach could be viable if removal of the existing protocol is not an urgent goal. It might also be necessary for "wildly successful" protocols that have more disparate uses than can reasonably be considered during the design of a replacement. For example, HTTP/2 does not aspire to cause the eventual decommissioning of HTTP/1.1 for these reasons.

4. Translation/Adaptation Location

A translation or adaptation mechanism is often required if the old and new mechanisms are not interoperable. Care must be taken when determining whether one will work and where such a translator is best placed.

A translation mechanism may not work for every use case. For example, if translation from one protocol (or protocol version) to another produces indeterminate results, translation will not work reliably. In addition, if translation always produces a downgraded protocol result, the incentive considerations in Section 5.2 will be relevant.

Requiring a translator in the middle of the path can hamper end-to-end security and reliability. For example, see the discussion of network-based filtering in [RFC7754].

On the other hand, requiring a translation layer within an endpoint can be a resource issue in some cases, such as if the endpoint could be a constrained node [RFC7228].

In addition, when a translator is within an endpoint, it can attempt to hide the difference between an older protocol and a newer protocol, either by exposing one of the two sets of behavior to applications and internally mapping it to the other set of behavior, or by exposing a higher level of abstraction that is then alternatively mapped to either one depending on detecting which is needed. In contrast, when a translator is in the middle of the path, typically only the first approach can be done since the middle of the path is typically unable to provide a higher level of abstraction.

Any transition strategy for a non-backward-compatible mechanism should include a discussion of where the incompatible mechanism is placed and a rationale. The transition plan should also consider the transition away from the use of translation and adaptation technologies.

5. Transition Plans

A review of the case studies described in Appendix A suggests that a good transition plan include at least the following components: an understanding of what is already deployed and in use, an explanation of incentives for each entity involved, a description of the phases of the transition along with a proposed criteria for each phase, a method for measuring the transition's success, a contingency plan for failure of the transition, and an effective method for communicating the plan to the entities involved and incorporating their feedback thereon. We recommend that such criteria be considered when evaluating proposals to transition to new or updated protocols. Each of these components is discussed in the subsections below.

5.1. Understanding of Existing Deployment

Often an existing mechanism has variations in implementations and operational deployments. For example, a specification might include optional behaviors that may or may not be implemented or deployed. In addition, there may also be implementations or deployments that deviate from, or include vendor-specific extensions to, various aspects of a specification. It is important when considering a transition to understand what variations one is intending to transition from or coexist with, since the technical and non-technical issues may vary greatly as a result.

5.2. Explanation of Incentives

A transition plan should explain the incentives to each involved entity to support the transition. Note here that many entities other than the endpoint applications and their users may be affected, and the barriers to transition may be non-technical as well as technical. When considering these incentives, also consider network operations tools, practices and processes, personnel training, accounting and billing dependencies, and legal and regulatory incentives.

If there is opposition to a particular new protocol (e.g., from another standards organization, or a government, or some other affected entity), various non-technical issues arise that should be part of what is planned and dealt with. Similarly, if there are significant costs or other disincentives, the plan needs to consider how to overcome them.

It's worth noting that an analysis of incentives can be difficult and at times led astray by wishful thinking, as opposed to adequately considering economic realities. Thus, honestly considering any barriers to transition, and justifying one's conclusions about others' incentives, are key to a successful analysis.

5.3. Description of Phases and Proposed Criteria

Transition phases might include pilot/experimental deployment, coexistence, deprecation, and removal phases for a transition from one technology to another incompatible one.

Timelines are notoriously difficult to predict and impossible to impose on uncoordinated transitions at the scale of the Internet, but rough estimates can sometimes help all involved entities to understand the intended duration of each phase. More often, it is useful to provide criteria that must be met in order to move to the next phase. For example, is removal scheduled for a particular date (e.g., Federal Communications Commission (FCC) regulation to discontinue analog TV broadcasts in the U.S. by June 12, 2009), or is removal to be based on the use of the old mechanism falling below a specified level, or some other criteria?

As one example, RFC 5211 [RFC5211] proposed a transition plan for IPv6 that included a proposed timeline and criteria specific to each phase. While the timeline was not accurately followed, the phases and timeline did serve as inputs to the World IPv6 Day and World IPv6 Launch events.

5.4. Measurement of Success

The degree of deployment of a given protocol or feature at a given phase in its transition can be measured differently, depending on its design. For example, server-side protocols and options that identify themselves through a versioning or negotiation mechanism can be discovered through active Internet measurement studies.

5.5. Contingency Planning

A contingency plan can be as simple as providing for indefinite coexistence between an old and new protocol, or for reverting to the old protocol until an updated version of the new protocol is available. Such a plan is useful in the event that unforeseen problems are discovered during deployment, so that such problems can be quickly mitigated.

For example, World IPv6 Day included a contingency plan that was to revert to the original state at the end of the day. After discovering no issues, some participants found that this contingency plan was unnecessary and kept the new state.

5.6. Communicating the Plan

Many of the entities involved in a protocol transition may not be aware of the IETF or the RFC series, so dissemination through other channels is key for sufficiently broad communication of the transition plan. While flag days are impractical at Internet scale, coordinated "events" such as World IPv6 Launch may improve general awareness of an ongoing transition.

Also, there is often a need for an entity facilitating the transition through advocacy and focus. Such an entity, independent of the IETF, can be key in communicating the plan and its progress.

Some transitions have a risk of breaking backwards compatibility for some fraction of users. In such a case, when a transition affects competing entities facing the risk of losing customers to each other, there is an economic disincentive to transition. Thus, one role for a facilitating entity is to get competitors to transition during the same timeframe, so as to mitigate this fear. For example, the success of World IPv6 Launch was largely due to ISOC playing this role.

6. Security Considerations

This document discusses attributes of protocol transitions. Some types of transition can adversely affect security or privacy. For example, requiring a translator in the middle of the path may hamper end-to-end security and privacy, since it creates an attractive target. For further discussion of some of these issues, see Section 5 of [RFC7754].

In addition, coexistence of two protocols in general increases risk in the sense that it doubles the attack surface. It allows exploiters to choose the weaker of two protocols when both are available, or to force use of the weaker when negotiating between the protocols by claiming not to understand the stronger one.

7. IANA Considerations

This document does not require any IANA actions.

8. Conclusion

This document summarized the set of issues that should be considered by protocol designers and deployers to facilitate transition and provides pointers to previous work (e.g., [RFC3692] and [RFC6709]) that provided detailed design guidelines. This document also covered what makes for a good transition plan and includes several case studies that provide examples. As more experience is gained over time on how to successfully apply these principles and design effective transition plans, we encourage the community to share such learnings with the IETF community and on the architecture-discuss@ietf.org mailing list so that any future document on this topic can leverage such experience.

9. Informative References

[GREASE] Benjamin, D., "Applying GREASE to TLS Extensibility", Work in Progress, draft-ietf-tls-grease-00, January 2017.

[HTTP0.9] Tim Berners-Lee, "The Original HTTP as defined in 1991", 1991, <https://www.w3.org/Protocols/HTTP/AsImplemented.html>.

[IabIpv6TransitionStatement] IAB, "Follow-up work on NAT-PT", October 2007, <https://www.iab.org/documents/correspondence-reports-documents/docs2007/follow-up-work-on-nat-pt/>.

[IPv6Survey2011] Botterman, M., "IPv6 Deployment Survey", 2011, <https://www.nro.net/wp-content/uploads/ipv6_deployment_survey.pdf>.

[IPv6Survey2015] British Telecommunications, "IPv6 Industry Survey Report", August 2015, <http://www.globalservices.bt.com/static/assets/pdf/products/diamond_ip/IPv6-Survey-Report-2015.pdf>.

[PAM2015] Trammell, B., Kuehlewind, M., Boppart, D., Learmonth, I., Fairhurst, G., and R. Scheffenegger, "Enabling Internet-Wide Deployment of Explicit Congestion Notification", Proceedings of PAM 2015, DOI 10.1007/978-3-319-15509-8_15, 2015, <http://ecn.ethz.ch/ecn-pam15.pdf>.

[RFC1883] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 1883, DOI 10.17487/RFC1883, December 1995, <http://www.rfc-editor.org/info/rfc1883>.

[RFC1933] Gilligan, R. and E. Nordmark, "Transition Mechanisms for IPv6 Hosts and Routers", RFC 1933, DOI 10.17487/RFC1933, April 1996, <http://www.rfc-editor.org/info/rfc1933>.

[RFC1945] Berners-Lee, T., Fielding, R., and H. Frystyk, "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, DOI 10.17487/RFC1945, May 1996, <http://www.rfc-editor.org/info/rfc1945>.

[RFC2068] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2068, DOI 10.17487/RFC2068, January 1997, <http://www.rfc-editor.org/info/rfc2068>.

[RFC2145] Mogul, J., Fielding, R., Gettys, J., and H. Frystyk, "Use and Interpretation of HTTP Version Numbers", RFC 2145, DOI 10.17487/RFC2145, May 1997, <http://www.rfc-editor.org/info/rfc2145>.

[RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, DOI 10.17487/RFC3168, September 2001, <http://www.rfc-editor.org/info/rfc3168>.

[RFC3424] Daigle, L., Ed. and IAB, "IAB Considerations for UNilateral Self-Address Fixing (UNSAF) Across Network Address Translation", RFC 3424, DOI 10.17487/RFC3424, November 2002, <http://www.rfc-editor.org/info/rfc3424>.

[RFC3692] Narten, T., "Assigning Experimental and Testing Numbers Considered Useful", BCP 82, RFC 3692, DOI 10.17487/RFC3692, January 2004, <http://www.rfc-editor.org/info/rfc3692>.

[RFC4380] Huitema, C., "Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs)", RFC 4380, DOI 10.17487/RFC4380, February 2006, <http://www.rfc-editor.org/info/rfc4380>.

[RFC4632] Fuller, V. and T. Li, "Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan", BCP 122, RFC 4632, DOI 10.17487/RFC4632, August 2006, <http://www.rfc-editor.org/info/rfc4632>.

[RFC4690] Klensin, J., Faltstrom, P., Karp, C., and IAB, "Review and Recommendations for Internationalized Domain Names (IDNs)", RFC 4690, DOI 10.17487/RFC4690, September 2006, <http://www.rfc-editor.org/info/rfc4690>.

[RFC5211] Curran, J., "An Internet Transition Plan", RFC 5211, DOI 10.17487/RFC5211, July 2008, <http://www.rfc-editor.org/info/rfc5211>.

[RFC5218] Thaler, D. and B. Aboba, "What Makes for a Successful Protocol?", RFC 5218, DOI 10.17487/RFC5218, July 2008, <http://www.rfc-editor.org/info/rfc5218>.

[RFC5894] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Background, Explanation, and Rationale", RFC 5894, DOI 10.17487/RFC5894, August 2010, <http://www.rfc-editor.org/info/rfc5894>.

[RFC5895] Resnick, P. and P. Hoffman, "Mapping Characters for Internationalized Domain Names in Applications (IDNA) 2008", RFC 5895, DOI 10.17487/RFC5895, September 2010, <http://www.rfc-editor.org/info/rfc5895>.

[RFC6055] Thaler, D., Klensin, J., and S. Cheshire, "IAB Thoughts on Encodings for Internationalized Domain Names", RFC 6055, DOI 10.17487/RFC6055, February 2011, <http://www.rfc-editor.org/info/rfc6055>.

[RFC6269] Ford, M., Ed., Boucadair, M., Durand, A., Levis, P., and P. Roberts, "Issues with IP Address Sharing", RFC 6269, DOI 10.17487/RFC6269, June 2011, <http://www.rfc-editor.org/info/rfc6269>.

[RFC6455] Fette, I. and A. Melnikov, "The WebSocket Protocol", RFC 6455, DOI 10.17487/RFC6455, December 2011, <http://www.rfc-editor.org/info/rfc6455>.

[RFC6709] Carpenter, B., Aboba, B., Ed., and S. Cheshire, "Design Considerations for Protocol Extensions", RFC 6709, DOI 10.17487/RFC6709, September 2012, <http://www.rfc-editor.org/info/rfc6709>.

[RFC7021] Donley, C., Ed., Howard, L., Kuarsingh, V., Berg, J., and J. Doshi, "Assessing the Impact of Carrier-Grade NAT on Network Applications", RFC 7021, DOI 10.17487/RFC7021, September 2013, <http://www.rfc-editor.org/info/rfc7021>.

[RFC7228] Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained-Node Networks", RFC 7228, DOI 10.17487/RFC7228, May 2014, <http://www.rfc-editor.org/info/rfc7228>.

[RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <http://www.rfc-editor.org/info/rfc7230>.

[RFC7301] Friedl, S., Popov, A., Langley, A., and E. Stephan, "Transport Layer Security (TLS) Application-Layer Protocol Negotiation Extension", RFC 7301, DOI 10.17487/RFC7301, July 2014, <http://www.rfc-editor.org/info/rfc7301>.

[RFC7305] Lear, E., Ed., "Report from the IAB Workshop on Internet Technology Adoption and Transition (ITAT)", RFC 7305, DOI 10.17487/RFC7305, July 2014, <http://www.rfc-editor.org/info/rfc7305>.

[RFC7540] Belshe, M., Peon, R., and M. Thomson, Ed., "Hypertext Transfer Protocol Version 2 (HTTP/2)", RFC 7540, DOI 10.17487/RFC7540, May 2015, <http://www.rfc-editor.org/info/rfc7540>.

[RFC7541] Peon, R. and H. Ruellan, "HPACK: Header Compression for HTTP/2", RFC 7541, DOI 10.17487/RFC7541, May 2015, <http://www.rfc-editor.org/info/rfc7541>.

[RFC7754] Barnes, R., Cooper, A., Kolkman, O., Thaler, D., and E. Nordmark, "Technical Considerations for Internet Service Blocking and Filtering", RFC 7754, DOI 10.17487/RFC7754, March 2016, <http://www.rfc-editor.org/info/rfc7754>.

[TR46] The Unicode Consortium, "Unicode IDNA Compatibility Processing", Version 9.0.0, June 2016, <http://www.unicode.org/reports/tr46/>.

[TSV2007] Sridharan, M., Bansal, D., and D. Thaler, "Implementation Report on Experiences with Various TCP RFCs", Proceedings of IETF 68, March 2007, <http://www.ietf.org/proceedings/68/slides/tsvarea-3/sld1.htm>.

Appendix A. Case Studies

Appendix A of [RFC5218] describes a number of case studies that are relevant to this document and highlight various transition problems and strategies (see, for instance, the Inter-Domain Multicast case study in Appendix A.4 of [RFC5218]). We now include several additional case studies that focus on transition problems and strategies. Many other equally good case studies could have been included, but, in the interests of brevity, only a sampling is included here that is sufficient to justify the conclusions in the body of this document.

A.1. Explicit Congestion Notification

Explicit Congestion Notification (ECN) is a mechanism to replace loss as the only signal for the detection of congestion. It does this with an explicit signal first sent from a router to a recipient of a packet, which is then reflected back to the sender. It was standardized in 2001 in [RFC3168], and the mechanism consists of two parts: congestion detection in the IP layer, reusing two bits of the old IP Type of Service (TOS) field, and congestion feedback in the transport layer. Feedback in TCP uses two TCP flags, ECN Echo and Congestion Window Reduced. Together with a suitably configured active queue management (AQM), ECN can improve TCP performance on congested links.

The deployment of ECN is a case study in failed transition followed by possible redemption. Initial deployment of ECN in the early and mid 2000s led to severe problems with some network equipment, including home router crashes and reboots when packets with ECN IP or TCP flags were received [TSV2007]. This led to firewalls stripping ECN IP and TCP flags, or even dropping packets with these flags set. This stalled deployment. The need for both endpoints (to negotiate and support ECN) and on-path devices (to mark traffic when congestion occurs) to cooperate in order to see any benefits from ECN deployment was a further issue. The deployment of ECN across the Internet had failed.

In the late 2000s, Linux and Windows servers began defaulting to "passive ECN support", meaning they would negotiate ECN if asked by the client but would not ask to negotiate ECN by default. This decision was regarded as without risk: only if a client was explicitly configured to negotiate ECN would any possible connectivity problems surface. Gradually, this has increased server support in the Internet from near zero in 2008, to 11% of the top million Alexa webservers in 2011, to 30% in 2012, and to 65% in late 2014. In the meantime, the risk to connectivity of ECN negotiation has reduced dramatically [PAM2015], leading to ongoing work to make

Windows, Apple iOS, OSX, and Linux clients negotiate ECN by default. It is hoped that a critical mass of clients and servers negotiating ECN will provide an incentive to mark congestion on ECN-enabled traffic, thus breaking the logjam.

A.2. Internationalized Domain Names

The deployment of Internationalized Domain Names (IDNs) has a long and complicated history. This should not be surprising, since internationalization deals with language and cultural issues regarding differing expectations of users around the world, thus making it inherently difficult to agree on common rules.

Furthermore, because human languages evolve and change over time, even if common rules can be established, there is likely to be a need to review and update them regularly.

There have been multiple technical transitions related to IDNs, including the introduction of non-ASCII in DNS, the transition to each new version of Unicode, and the transition from IDNA 2003 to IDNA 2008. A brief history of the introduction of non-ASCII in DNS and the various complications that arose therein, can be found in Section 3 of [RFC6055]. While IDNA 2003 was limited to Unicode version 3.2 only, one of the IDNA 2008 changes was to decouple its rules from any particular version of Unicode (see [RFC5894], especially Section 1.4, for more discussion of this point, and see [RFC4690] for a list of other issues with IDNA 2003 that motivated IDNA 2008). However, the transition from IDNA 2003 to IDNA 2008 itself presented a problem since IDNA 2008 did not preserve backwards compatibility with IDNA 2003 for a couple of codepoints. Investigations and discussions with affected parties led to the IETF ultimately choosing IDNA 2008 because the overall gain by moving to IDNA 2008 to fix the problems with IDNA 2003 was seen to be much greater than the problems due to the few incompatibilities at the time of the change, as not many IDNs were in use and even fewer that might see incompatibilities.

A couple of browser vendors in particular were concerned about the differences between IDNA 2003 and IDNA 2008, and the fact that if a browser stopped being able to get to some site, or unknowingly sent a user to a different (e.g., phishing) site instead, the browser would be blamed. As such, any user-perceivable change from IDNA 2003 behavior would be painful to the vendor to deal with; hence, they could not depend on solutions that would need action by other entities.

Thus, to deal with issues like such incompatibilities, some applications and client-side frameworks wanted to map one string into another (namely, a string that would give the same result as when IDNA 2003 was used) before invoking DNS.

To provide such mapping (and some other functionality), the Unicode Consortium published [TR46], which continued down the path of IDNA 2003 with a code point by code point selection mechanism. This was implemented by some, but never adopted by the IETF.

Meanwhile, the IETF did not publish any mapping mechanism, but [RFC5895] was published on the Independent Submission stream. In discussions around mapping, one of the key topics was about how long the transition should last. At one end of the duration spectrum is a flag day where some entities would be broken initially but the change would happen before IDN usage became even more ubiquitous. At the other end of the spectrum is the need to maintain mappings indefinitely. Local incentives at each entity who needed to change, however, meant that a short timeframe was impractical.

There are many affected types of entities with very different incentives. For example, the incentives affecting browser vendors, registries, domain name marketers and applicants, app developers, and protocol designers are each quite different, and the various solutions require changes by multiple types of entities, where the benefits do not always align with the costs. If there is some group (or even an individual) that is opposed to a change/transition and able to put significant resources behind their opposition, transitions get a lot harder.

Finally, there are multiple naming contexts, and the protocol behavior (including how internationalized domain names are handled) within each naming context can be different. Hence, applications and frameworks often encounter a variety of behaviors and may or may not be designed to deal with them. See Sections 2 and 3 of [RFC6055] for more discussion.

In summary, all this diversity can cause problems for each affected entity, especially if a competitor does not have such a problem, e.g., for browser vendors if competing browsers do not have the same problems, or for an email server provider if competing server providers do not have the same problems.

A.3. IPv6

Twenty-one years after publication of [RFC1883], the transition to IPv6 is still in progress. The first document to describe a transition plan ([RFC1933]) was published less than a year after the protocol itself. It recommended coexistence (dual-stack or tunneling technology) with the expectation that over time, all hosts would have IPv6, and IPv4 could be quietly retired.

In the early stages, deployment was limited to peer-to-peer uses tunneled over IPv4 networks. For example, Teredo [RFC4380] aligned the cost of fixing the problem with the benefit and allowed for incremental benefits to those who used it.

Operating system vendors had incentives because with such tunneling protocols, they could get peer-to-peer apps working without depending on any infrastructure changes. That resulted in the main apps using IPv6 being in the peer-to-peer category (BitTorrent, Xbox gaming, etc.).

Router vendors had some incentive because IPv6 could be used within an intra-domain network more efficiently than tunneling, once the OS vendors already had IPv6 support and some special-purpose apps existed.

For content providers and ISPs, on the other hand, there was little incentive for deployment: there was no incremental benefit to deploying locally. Since everyone already had IPv4, there was no network effect benefit to deploying IPv6. Even as proponents argued that workarounds to extend the life of IPv4 -- such as Classless Inter-Domain Routing (CIDR) [RFC4632] , NAT, and stingy allocations -- made it more complex, IPv4 continued to work well enough for most applications.

Workarounds to NAT problems documented in [RFC6269] and [RFC7021] included Interactive Connectivity Establishment (ICE), Session Traversal Utilities for NAT (STUN), and Traversal Using Relays around NAT (TURN), technologies that allowed those experiencing the problems to deploy technologies to resolve them. As with end-to-end IPv6 tunneling (e.g., Teredo), the incentives there aligned the cost of fixing the problem with the benefit and allowed for incremental benefits to those who used them. The IAB discussed NAT technology proposals [RFC3424] and recommended that they be considered short-term fixes and said that proposals must include an exit plan, such that they would decline over time. In particular, the IAB warned against generalizing NAT solutions, which would lead to greater

dependence on them. In some ways, these solutions, along with other IPv4 development (e.g., the workarounds above, and retrofitting IPsec into IPv4) continued to reduce the incentive to deploy IPv6.

Some early advocates overstated the benefits of IPv6, suggesting that it had better security (because IPsec was required) or that NAT was worse than it often appeared to be or that IPv4 exhaustion would happen years sooner than it actually did. Some people pushed back on these exaggerations, and decided that the protocol itself somehow lacked credibility.

Not until a few years after IPv4 addresses were exhausted in various RIR regions did IPv6 deployment significantly increase. The RIRs had been advocating in their communities for IPv6 for some time, reducing fees for IPv6, and in some cases providing training; there is little to suggest that these had a significant effect. The RIRs and others conducted surveys of different industries and industry segments to learn why people did not deploy IPv6 [IPv6Survey2011] [IPv6Survey2015], which commonly listed lack of a business case, lack of training, and lack of vendor support as primary hurdles.

Arguably forward-looking companies collaborated, with ISOC, on World IPv6 Day and World IPv6 Launch to jump-start global IPv6 deployment. By including multiple competitors, World IPv6 Day reduced the risk that any of them would lose customers if a user’s IPv6 implementation was broken. World IPv6 Launch then set a goal for content providers to permanently enable IPv6, and for large ISPs to enable IPv6 for at least 1% of end users. These large, visible deployments gave vendors specific features and target dates to support IPv6 well. Key aspects of World IPv6 Day and World IPv6 Launch that contributed to their successes (measured as increased deployment of IPv6) were the communication through ISOC, and that measurement metrics and contingency plans were announced in advance.

Several efforts have been made to mitigate the lack of a business case. Some governments (South Korea and Japan) provided tax incentives to include IPv6. Other governments (Belgium and Singapore) mandated IPv6 support by private companies. Few of these had enough value to drive significant IPv6 deployment.

The concern about lack of training is often a common issue in transitions. Because IPv4 is so ubiquitous, its use is routine and simplified with common tools, and it is taught in network training everywhere. While IPv6 deployment was low, ignorance of it was no obstacle to being hired as a network administrator or developer.

Organizations with the greatest incentives to deploy IPv6 are those that continue to grow quickly, even after IPv4 free-pool exhaustion. Thus, ISPs have had varying levels of commitment, based on the growth of their user base, services being added (especially video over IP), and the number of IPv4 addresses they had available. Cloud-based providers, including Content Delivery Network (CDN) and hosting companies, have been major buyers of IPv4 addresses, and several have been strong deployers and advocates of IPv6.

Different organizations will use different transition models for their networks, based on their needs. Some are electing to use IPv6-only hosts in the network with IPv6-IPv4 translation at the edge. Others are using dual-stack hosts with IPv6-only routers in the core of the network, and IPv4 tunneled or translated through them to dual-stack edge routers. Still others are using native dual-stack throughout the network, but that generally persists as an interim measure: adoption of two technologies is not the same as transitioning from one technology to another. Finally, some walled gardens or isolated networks, such as management networks, use IPv6-only end-to-end.

It is impossible to predict with certainty the path IPv6 deployment will have taken when it is complete. Lessons learned so far include aligning costs and benefits (incentive), and ensuring incremental benefit (network effect or backward compatibility).

A.4. HTTP

HTTP has been through several transitions as a protocol.

The first version [HTTP0.9] was extremely simple, with no headers, status codes, or explicit versioning. HTTP/1.0 [RFC1945] introduced these and a number of other concepts; it succeeded mostly because deployment of HTTP was still relatively new, with a small pool of implementers and (comparatively) small set of deployments and users.

HTTP/1.1 [RFC7230] (first defined in [RFC2068]) was an attempt to make the protocol suitable for the massive scale it was being deployed upon and to introduce some new features.

HTTP/2 [RFC7540] was largely aimed at improving performance. The primary improvement was the introduction of request multiplexing, which is supported by request prioritization and flow control. It also introduced header compression [RFC7541] and binary framing; this made it completely backwards incompatible on the wire, but still semantically compatible with previous versions of the protocol.

A.4.1. Protocol Versioning, Extensions, and 'Grease'

During the development of HTTP/1.1, there was a fair amount of confusion regarding the semantics of HTTP version numbers, resulting in [RFC2145]. Later, it was felt that minor versioning in the protocol caused more confusion than it was worth, so HTTP/2.0 became HTTP/2.

This decision was informed by the observation that many implementations ignored the major version number of the protocol or misinterpreted it. As is the case with many protocol extension points, HTTP versioning had failed to be "greased" by use often enough, and so had become "rusted" so that only a limited range of values could interoperate.

This phenomenon has been observed in other protocols, such as TLS (as exemplified by [GREASE]), and there are active efforts to identify extension points that are in need of such "grease" and making it appear as if they are in use.

Besides the protocol version, HTTP's extension points that are well-greased include header fields, status codes, media types, and cache-control extensions; HTTP methods, content-encodings, and chunk-extensions enjoy less flexibility, and need to be extended more cautiously.

A.4.2. Limits on Changes in Major Versions

Each update to the "major" version of HTTP has been accompanied by changes that weren't compatible with previous versions. This was not uniformly successful given the diversity and scale of deployment and implementations.

HTTP/1.1 introduced pipelining to improve protocol efficiency. Although it did enjoy implementation, interoperability did not follow.

This was partially because many existing implementations had chosen architectures that did not lend themselves to supporting it; pipelining was not uniformly implemented and where it was, support was sometimes incorrect or incomplete. Since support for pipelining was indicated by the protocol version number itself, interop was difficult to achieve, and furthermore its inability to completely address head-of-line blocking issues made pipelining unattractive.

Likewise, HTTP/1.1's Expect/Continue mechanism relied on wide support for the new semantics it introduced and did not have an adequate fallback strategy for previous versions of the protocol. As a

result, interoperability and deployment suffered and is still considered a "problem area" for the protocol.

More recently, the HTTP working group decided that HTTP/2 represented an opportunity to improve security, making the protocol much stricter than previous versions about the use of TLS. To this end, a long list of TLS cipher suites were prohibited, constraints were placed on the key exchange method, and renegotiation was prohibited.

This did cause deployment problems. Though most were minor and transitory, disabling renegotiation caused problems for deployments that relied on the feature to authenticate clients and prompted new work to replace the feature.

A number of other features or characteristics of HTTP were identified as potentially undesirable as part of the HTTP/2 process and considered for removal. This included trailers, the 1xx series of responses, certain modes of request forms, and the unsecured (http://) variant of the protocol.

For each of these, the risk to the successful deployment of the new version was considered to be too great to justify removing the feature. However, deployment of the unsecured variant of HTTP/2 remains extremely limited.

A.4.3. Planning for Replacement

HTTP/1.1 provided the Upgrade header field to enable transitioning a connection to an entirely different protocol. So far, this has been little-used, other than to enable the use of WebSockets [RFC6455].

With performance being a primary motivation for HTTP/2, a new mechanism was needed to avoid spending an additional round trip on protocol negotiation. A new mechanism was added to TLS to permit the negotiation of the new version of HTTP: Application-Layer Protocol Negotiation (ALPN) [RFC7301]. Upgrade was used only for the unsecured variant of the protocol.

ALPN was identified as the primary way in which future protocol versions would be negotiated. The mechanism was well-tested during development of the specification, proving that new versions could be deployed safely and easily. Several draft versions of the protocol were successfully deployed during development, and version negotiation was never shown to be an issue.

Confidence that new versions would be easy to deploy if necessary lead to a particular design stance that might be considered unusual in light of the advice in [RFC5218], though is completely consistent

with [RFC6709]: few extension points were added, unless an immediate need was understood.

This decision was made on the basis that it would be easier to revise the entire protocol than it would be to ensure that an extension point was correctly specified and implemented such that it would be available when needed.

IAB Members at the Time of Approval

Jari Arkko
Ralph Droms
Ted Hardie
Joe Hildebrand
Russ Housley
Lee Howard
Erik Nordmark
Robert Sparks
Andrew Sullivan
Dave Thaler
Martin Thomson
Brian Trammell
Suzanne Woolf

Acknowledgements

This document is a product of the IAB Stack Evolution Program, with input from many others. In particular, Mark Nottingham, Dave Crocker, Eliot Lear, Joe Touch, Cameron Byrne, John Klensin, Patrik Faltstrom, the IETF Applications Area WG, and others provided helpful input on this document.

Author’s Address

Dave Thaler (editor)
One Microsoft Way
Redmond, WA 98052
United States of America

Email: dthaler@microsoft.com