DBOUND
Internet-Draft
Intended status: Standards Track
Expires: August 21, 2016

A. Sullivan
Dyn, Inc.
J. Hodges
PayPal
J. Levine
Taughannock Networks
February 18, 2016

DBOUND: DNS Administrative Boundaries Problem Statement
draft-sullivan-dbound-problem-statement-02

Abstract

Some Internet client entities on the Internet make inferences about the administrative relationships among services on the Internet based on the domain names at which they are offered. At present, it is not possible to ascertain organizational administrative boundaries in the DNS, therefore such inferences can be erroneous in various ways. Mitigation strategies deployed so far will not scale. This memo outlines what issues are to be addressed.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 21, 2016.

Copyright Notice

Table of Contents

1. Prerequisites, Terminology, and Organization of this Memo . . 2
2. Introduction and Motivation 2
3. For the Use Case, Must an Ancestor Impose Policy? 5
4. Use Cases 6
5. Security Considerations 8
6. IANA Considerations 8
7. Acknowledgements 8
8. Informative References 8
Appendix A. Discussion Venue 10
Appendix B. Change History 10
Authors' Addresses 11

1. Prerequisites, Terminology, and Organization of this Memo

The reader is assumed to be familiar with the DNS ([RFC1034] [RFC1035]) and the Domain Name System Security Extensions (DNSSEC) ([RFC4033] [RFC4034] [RFC4035] [RFC5155]).

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

To begin, Section 2 describes introduces the problem space and motivations for this work. Then, Section 4 discusses the cases where a there are needs for discerning administrative boundaries in the DNS domain name space.

2. Introduction and Motivation

Many Internet resources and services, especially at the application layer, are identified primarily by DNS domain names [RFC1034]. As a result, domain names have become fundamental elements in building security policies and also in affecting user agent behaviour.

For example, domain names are used for defining the scope of HTTP state management "cookies" [RFC6265]. In addition there is a user interface convention that purports to display an "actual domain name" differently from other parts of a fully-qualified domain name, in an effort to decrease the success of phishing attacks. In this strategy, for instance, a domain name like

"www.bank.example.com.attackersite.tld" is formatted to highlight that the actual domain name ends in "attackersite.tld", in the hope of reducing user's potential impression of visiting "www.bank.example.com".

Issuers of X.509 certificates make judgements about administrative boundaries around domains when issuing the certificates. For some discussion of the relationship between domain names and X.509 certificates, see [RFC6125].

We can call the interpretation of domain names for these security policies a domain-use rule. The simplest rule, and the one most likely to work, is to treat each different domain name distinctly. Under this approach, foo.example.org, bar.example.org, and baz.example.org are all just different domains. Unfortunately, this approach is too naive to be useful. Often, the real control over domain names is the same in several names (in this example, example.org and its children). Therefore, clients have attempted to make more sophisticated rules around some idea of such shared control. We call such an area of shared control a "policy realm", and the control held by the administrator of policy realm the "policy authority".

Historically, rules were sometimes based on the DNS tree. Early rules made a firm distinction between top-level domains and everything else; but this was also too naive, and later attempts were based on inferences from the domain names themselves. That did not work well, because there is no way in the DNS to discover the boundaries of policy realms.

Some have attempted to use the boundary of zone cuts (i.e. the location of the zone's apex, which is at the SOA record; see [RFC1034] and [RFC1035]). That boundary is neither necessary nor sufficient for these purposes: it is possible for a large site to have many, administratively distinct subdomain-named sites without inserting an SOA record, and it is also possible that an administrative entity (like a company) might divide its domain up into different zones for administrative reasons unrelated to the names in that domain. It was also, prior to the advent of DNSSEC, difficult to find zone cuts. Regardless, the location of a zone cut is an administrative matter to do with the operation of the DNS itself, and not useful for determining relationships among policy realms.

The different uses of domain names and their related issues often appear to be different kinds of problems. The issue of whether two names may set cookies for one another appears to be a different matter from whether two names get the same highlighting in a

browser's address bar, or whether a particular name "owns" all the names underneath it. But the problems all boil down to the same fundamental problem, which is that of determining whether two different names in the DNS are under the control of the same entity and ought to be treated as having an important administrative relationship to one another.

What appears to be needed is a mechanism to determine policy realm boundaries in the DNS. That is, given two domain names, one needs to be able to answer whether the first and the second are either within the same policy realm or have policy realms that are related in some way. We may suppose that, if this information were to be available, it would be possible to make useful decisions based on the information.

A particularly important distinction for security purposes has been the one between names that are mostly used to contain other domains, as compared to those that are mostly used to operate services. The former are often "delegation-centric" domains, delegating parts of their name space to others, and are frequently called "public suffix" domains or "effective TLDs". The term "public suffix" comes from a site, [publicsuffix.org], that publishes a list of domains -- which is also known as the "effective TLD (eTLD) list", and henceforth in this memo as the "public suffix list" -- that are used to contain other domains. Not all, but most, delegation-centric domains are public suffix domains; and not all public suffix domains need to do DNS delegation, although most of them do. The reason for the public suffix list is to make the distinction between names that must never be treated as being in the same policy realm as another, and those that might be so treated. For instance, if "com" is on the public suffix list, that means that "example.com" lies in a policy realm distinct from that of com.

Unfortunately, the public suffix list has several inherent limitations. To begin with, it is a list that is separately maintained from the list of DNS delegations. As a result, the data in the public suffix list can diverge from the actual use of the DNS. Second, because its semantics are not the same as those of the DNS, it does not capture unusual features of the DNS that are a consequence of its structure (see [RFC1034] for background on that structure). Third, as the size of the root zone grows, keeping the list both accurate and synchronized with the expanding services will become difficult and unreliable. Perhaps most importantly, it puts the power of assertion about the operational policies of a domain outside the control of the operators of that domain, and in the control of a third party possibly unrelated to those operators.

There have been suggestions for improvements of the public suffix list, most notably in [I-D.pettersen-subtld-structure]. It is unclear the extent to which those improvements would help, because they represent improvements on the fundamental mechanism of keeping metadata about the DNS tree apart from the DNS tree itself.

Moreover, it is not entirely plain that the public/private distinction is really the best framework with which to understand the problem. It is plain that any solution that emerges will need, to be useful, to provide a way of making the public/private distinction, since so much deployed software relies on that distinction. It seems possible, however, that greater nuance would provide distinctions that are currently desired but cannot be supported using the public suffix list. The best way to figure this out is to enumerate known problems and see whether there is something common underlying them all, or whether the different problems might at least be grouped into a few common cases.

3. For the Use Case, Must an Ancestor Impose Policy?

It is possible to identify two common policy patterns into which practical uses fall. One is a positive policy that will necessarily be imposed by an ancestor in case a policy for the owner name itself is not available. The other is a policy that need not get inherited from an ancestor. Negative assertions by an ancestor (i.e. that a descendent does not share a policy realm) fall into this category, because the descendent does not have a positive policy imposed.

The first pattern we may call the inheritance type. In this use pattern, a client attempting to identify a policy that applies at a given name will use a policy found at a name closer to the root of the DNS, if need be. This approach is useful when a client must have some kind of policy in order to continue processing. Because the DNS is a hierarchical name system, it is always possible for a subordinate name to be permitted only in case the superordinate policies are followed.

The second pattern we may call the orphan type. In this use pattern, if a policy at a name is not specifically offered then it is better to assume there is a null policy than to infer some inherited policy. Note that orphan names might be related to other names (which makes the term somewhat unfortunate). Rather, in these cases policy is assumed to be unshared unless there is evidence otherwise. [[CREF1: Probably something better than "orphan" would be good, but I can't think of a better name. --ajs@anvilwalrusden.com]]

The choice of which pattern is preferable depends largely on what the use of a policy seeks to achieve. Some uses of policy require

determination of commonality among domains; in such cases, the inheritance pattern may be needed. Other uses are attempts to identify differences between domains; in such cases, the orphan pattern is useful.

The public suffix list provides a starting point for both patterns, but is neither necessary nor sufficient for either case. Where the inheritance pattern is used, the public suffix list provides a minimal starting point whence inheritance can start. Where the orphan pattern is used, the public suffix list provides the exclusion needed, but cannot provide either evidence that the list is up to date nor evidence that two owner names reside in the same policy realm.

4. Use Cases

This section outlines some questions and identifies some known use cases of the public suffix list.

HTTP state management cookies The mechanism can be used to determine the scope for data sharing of HTTP state management cookies [RFC6265]. Using this mechanism, it is possible to determine whether a service at one name may be permitted to set a cookie for a service at a different name. (Other protocols use cookies, too, and those approaches could benefit similarly.) An application has to answer in this case the question, "Should I accept a cookie for domain X from the domain Y I am currently visiting?"

User interface indicators User interfaces sometimes attempt to indicate the "real" domain name in a given domain name. A common use is to highlight the portion of the domain name believed to be the "real" name -- usually the rightmost three or four labels in a domain name string. An application has to answer in this case the question, "What domain name is relevant to show the user in this case?" The answer to this must be some portion of the domain name being displayed, but it is user- and context-sensitive.

Setting the document.domain property The DOM same-origin policy might be helped by being able to identify a common policy realm. An application has to answer in this case the question, "Is domain X under the same control as domain Y?" It's worth noting that, in this case, neither X nor Y need be actually visible to a user.

Email authentication mechanisms Mail authentication mechanisms such as DMARC [I-D.kucherawy-dmarc-base] need to be able to find policy documents for a given domain name given a subdomain. An application performing DMARC processing must answer the question, "Given the domain X currently being evaluated, where in the DNS is

the DMARC record?" DMARC depends on the DNS hierarchical relationship, and unlike some other cases wants to find the DMARC record that is closest to the root zone.

SSL and TLS certificates Certificate authorities need to be able to discover delegation-centric domains in order to avoid issuance of certificates at or above those domains. There are two cases:

* A certificate authority must answer the question, "Should I sign a certificate at this domain name given the request before me?"

* A certificate authority must answer the question, "Should I sign a certificate for a wildcard at this domain name?"

[[CREF2: There is another case here, noted by Jeffrey Walton, about "verifying the end-entity certificate issued by an organizational subordinate CA *without* constraints." I didn't understand the issue well enough to write the text here. --ajs@anvilwalrusden.com]]

HSTS and Public Key Pinning with includeSubDomains flag set
: Clients that are using HSTS and public key pinning using includeSubDomains need to be able to determine whether a subdomain is properly within the policy realm of the parent. An application performing this operation must answer the question, "Should I accept the rules for using X as valid for Y.X?"

Linking domains together for merging operations
: It is frequently the case that domain names are aliases for one another. Sometimes this is because of an ongoing merger (as when one company takes over another and merges operations). A client encountering such a site needs to answer the question, "Is domain X just another name for domain Y?"

Linking domains together for reporting purposes
: An application that wants to categorize domains for the purposes of reporting must answer the question, "Are these two names versions of each other for the purposes of reporting statistics?"

DMARC science fiction use case DMARC's current use of the PSL is to determine the 'Organizational Domain'.. for use when discovering DMARC policy records. PSL works well enough for production environments in today's world. However, after hearing about cross-domain requirements of cookies and cross-domain security use

cases in the browser, it strikes me that any functionality (policy authority?) that allows domains to be linked would be incredibly useful in the DMARC world, too. DMARC?s requirement for Identifier Alignment between SPF-authenticated domain, DKIM d=domain, and a message?s From: domain could be relaxed to include domains that were somehow associated via a policy authority. This capability would be *very* nice to have at hand.

5. Security Considerations

A mechanism that satisfied the needs outline above would enable publication of assertions about administrative relationships of different DNS-named systems on the Internet. If such assertions were to be accepted without checking that both sides agree to the assertion, it would be possible for one site to become an illegitimate source for data to be consumed in some other site. In general, positive assertions about another name should never be accepted without querying the other name for agreement.

Undertaking any of the inferences suggested in this draft without the use of the DNS Security Extensions exposes the user to the possibility of forged DNS responses.

This memo does not actually specify any mechanisms, so it raises no security considerations itself.

6. IANA Considerations

This memo makes no requests of IANA.

7. Acknowledgements

The authors thank Adam Barth, Dave Crocker, Casey Deccio, Brian Dickson, Jothan Frakes, Daniel Kahn Gillmor, Phillip Hallam-Baker, John Klensin, Murray Kucherawy, Gervase Markham, Patrick McManus, Henrik Nordstrom, Yngve N. Pettersen, Eric Rescorla, Thomas Roessler, Peter Saint-Andre, Maciej Stachowiak, and Jeffrey Walton for helpful comments or suggestions.

8. Informative References

[I-D.kucherawy-dmarc-base]
Kucherawy, M., "Domain-based Message Authentication, Reporting and Conformance (DMARC)", draft-kucherawy-dmarc-base-00 (work in progress), March 2013.

[I-D.pettersen-subtld-structure]
Pettersen, Y., "The Public Suffix Structure file format and its use for Cookie domain validation", draft-pettersen-subtld-structure-09 (work in progress), March 2012.

[publicsuffix.org]
Mozilla Foundation, "Public Suffix List", also known as: Effective TLD (eTLD) List.

https://publicsuffix.org/

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, March 2005.

[RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, March 2005.

[RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, March 2005.

[RFC5155] Laurie, B., Sisson, G., Arends, R., and D. Blacka, "DNS Security (DNSSEC) Hashed Authenticated Denial of Existence", RFC 5155, March 2008.

[RFC6125] Saint-Andre, P. and J. Hodges, "Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)", RFC 6125, March 2011.

[RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, April 2011.

Appendix A. Discussion Venue

This Internet-Draft is discussed on the applications area working group mailing list: dbound@ietf.org.

Appendix B. Change History

[this section to be removed by RFC-Editor prior to publication as an RFC]

Version 01 Add questions from John Levine posting to mailing list.

Version 00 Initial version.

This is a -00 Internet-draft, but borrows from various prior draft works, listed below, as well as from discussions on the mailing list.

Andrew Sullivan, Jeff Hodges: Asserting DNS Administrative Boundaries Within DNS Zones

http://tools.ietf.org/html/draft-sullivan-domain-policy-authority-01

https://github.com/equalsJeffH/dbound/blob/master/draft-sullivan-dbound-problem-statement-00.xml

John Levine: Publishing Organization Boundaries in the DNS

https://tools.ietf.org/html/draft-levine-orgboundary-02

https://github.com/equalsJeffH/dbound/blob/master/draft-levine-orgboundary-02.txt

Casey Deccio, John Levine: Defining and Signaling Relationships Between Domains

http://www.ietf.org/mail-archive/web/dbound/current/pdfwad2AxxkYo.pdf

http://www.ietf.org/mail-archive/web/dbound/current/msg00141.html

https://github.com/equalsJeffH/dbound/blob/master/deccio-dbound-problem_statement-v3.pdf?raw=true

https://github.com/equalsJeffH/dbound/blob/master/deccio-dbound-problem_statement-v3.txt

Authors' Addresses

Andrew Sullivan
Dyn, Inc.
150 Dow St
Manchester, NH 03101
U.S.A.

Email: asullivan@dyn.com

Jeff Hodges
PayPal
2211 North First Street
San Jose, California 95131
US

Email: Jeff.Hodges@KingsMountain.com

John Levine
Taughannock Networks
PO Box 727
Trumansburg, NY 14886

Phone: +1 831 480 2300
Email: standards@taugh.com
URI: http://jl.ly