Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: 4 September 2025

S. Huque
Salesforce
M. Andrews
Internet Systems Consortium

3 March 2025

Greasing Protocol Extension Points in the DNS
draft-ietf-dnsop-grease-01

Abstract

Long term evolvability of the Domain Name System (DNS) protocol requires the ability to support change. Greasing is one technique that exercises the regular use of unallocated protocol extension points to prevent ossification of their current usage patterns by middleboxes or DNS implementations. This document describes considerations and proposals for applying grease to the DNS protocol.

Discussion Venues

This note is to be removed before publishing as an RFC.

Source for this draft and an issue tracker can be found at https://github.com/ietf-wg-dnsop/draft-ietf-dnsop-grease.

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 https://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 4 September 2025.

Copyright Notice

Table of Contents

1. Introduction

Long term evolvability of the Domain Name System (DNS) protocol requires the ability to support change. Greasing [GREASING] is one technique that exercises the regular use of unallocated protocol extension points to prevent ossification of their current usage patterns by middleboxes or DNS implementations.

Greasing was originally developed for the TLS protocol [RFC8701]. Ongoing discussion of improving the technique as well as applying it more generally to other protocols continues in the IETF.

This document outlines considerations and proposals for applying grease to the [RFC1034][RFC1035] Domain Name System (DNS).

2. Background

Historically, DNS protocol evolution has encountered some significant barriers. There are various reasons, including outdated systems, inertia, faulty implementations of DNS servers, middleboxes that have actively blocked the deployment of new protocol features, etc. Over time as some of these impediments have been uncovered and repaired, progress has been possible. However, a more systematic approach to ensure future progress is desirable.

Remarkably, the Extension Mechanisms for DNS (EDNS) specification, originally published in 1999 [RFC2671], is still not universally deployed, and often deployed incorrectly or incompletely. This eventually resulted in the exercise of a DNS Flag Day effort [FLAGDAY] to identify and eradicate implementations and network paths not compliant with the specification. Tools like ednscomp.isc.org [EDNSCOMP] have been testing this and other protocol defects in deployed infrastructure for many years. Even then, some level of incorrect behavior remains prevalent, necessitating probing and pre-arrangement of the use of some extension features like [RFC7871]EDNS Client Subnet and [RFC7873]DNS Cookies.

3. Greasing Opportunities

The DNS has a number of protocol elements where the greasing of unallocated code points could be employed. Some of them are listed in the table below.

Protocol Element	Size	Number of Values
DNS Header Flags	7-bits	7
Resource Record Type	16-bits	65,536
Opcode	4-bits	16
EDNS Version	8-bits	256
EDNS Header Flags	16-bits	16
EDNS Opt Code	16-bits	65,536

Table 1

The goals of periodically exercising the use of unallocated code points are (1) to discourage and prevent middleboxes and DNS implementations from hardcoding notions of what are the only allowable protocol parameter values, and (2) to prevent future extensibility failures by not causing them to malfunction in the presence of new values when they are defined.

If and when encrypted transports are common on the DNS resolver to authoritative server path, middleboxes will have less capability to interfere with DNS traffic. Greasing would still however continue to be useful for identifying deficient DNS proxies, load balancers, authoritative servers, etc.

Note that EDNS [RFC6891] options have a more complex structure involving both a code point and data. So greasing EDNS options would also require generating some random option data.

Correctly implemented DNS servers will ignore these values and interoperate. Servers that do not tolerate unknown values will fail to interoperate and return an error (or may fail to respond). These failures could be logged and be used to identify broken implementations in the field that could be targeted for repair. DNS resolvers should generally retry such failed queries without the unallocated extension, except for greasing operations where new queries are constructed (for example, greasing new resource record types).

4. Randomized or Reserved code points

DNS resolver implementations are proposed to periodically advertise unallocated code points at random in requests that they send out.

Resolvers could select randomly from the unallocated range, but then would have to consider what happens when such code points are allocated in the future and whether or not that will give rise to a different class of interoperability failure. One possible way to deal with this is for software to have pre-configured or configurable end-of-test dates.

Alternatively, a subset of the currently unused values could be reserved for DNS resolver implementations to advertise at random. The expectation is that receivers will have uniform handling of unknown code points whether they fall in a reserved range or not. But there is a slight risk that the reserved values could then become ossified in implementations. Furthermore, Some DNS protocol elements have only a small range of supported values, and it may not be practical to reserve a a subset of such ranges. Some larger ranges also have a sub structure, such as the data vs meta vs q-type classification of the RR-type space, where multiple greasing ranges would need to be reserved.

5. Reserved Code Point Values

[Propose reserved ranges for some DNS protocol elements]

6. Sampled Selection of Traffic

To avoid the overhead of needing to retry many queries in the event of large scale intolerance of unallocated code points, only a sampled fraction of DNS requests emitted by a resolver should advertise unallocated code points. Many DNS resolvers are very high transaction rate systems, so only a small sample size of such DNS requests is sufficient to get a rough picture of non-compliant servers, perhaps 1 in 1000 requests? Furthermore, a community effort of aggregating and analysing the results of failed queries from many DNS resolver operators can provide an even more comprehensive view of the ecosystem.

7. Telemetry and Results Evaluation

DNS resolvers are expected to record the results of failures from the use of unallocated code points. This could be in a traditional log file, or a more complex centralized telemetry system.

Additionally, the DNS Error Reporting [ERROR-RPT] mechanism could be employed to proactively notify operators that their authoritative DNS servers are deficient.

8. Detailed Behavior

Work in-progress section .. Some topics to expand on:

* Detailed expected behavior of DNS resolvers/clients.

* Detailed expected behavior of DNS servers.

* Detection of errors.

* Fallback behavior (or not).

* Settings: nameservers should be shipped with a default end-of-test date to prevent tests from interfering with future code point assignments.

* Testing should be multi-factored.

* Tests should be able to be individually disabled.

* Sharing telemetry.

9. Greasing Initiated By DNS Responders

This document largely focusses on greasing initiated by the DNS resolver or querier. Greasing operations could also be initiated by an authoritative server or DNS responder. This seems potentially a bit more fraught, since the responder cannot necessarily know what the result of the greasing action was, e.g., did the querier accept the answer with no problems, did it not accept it and retry other servers, did it not not accept it and just fail, causing a denial of service to the downstream application, etc. However, greasing in this direction can be very helpful in targeted experiments. For example, some early measurements for DELEG proposals utilized server side greasing to insert unexpected record types in referral responses to test the behavior of DNS resolvers. Operators of DNS zones could perform targeted active measurements or rely on reports by users to determine if server side greasing works well.

Some things a responder could do include: test setting the final DNS header flag (Z), send back unknown EDNS header flags, options. higher EDNS versions etc. All of these should in theory be ignored on reception.

Greasing by DNS responders should be disabled by default.

10. Security Considerations

If an implementation does not select GREASE values at random, it may allow others to fingerprint specific resolvers or resolver implementations.

Some DNS resolver implementations have traditionally resorted to falling back to retrying queries with various extension options disabled in the face of interoperability problems. Depending on the specific extension affected, this may allow an adversary to silently disable a security feature. Greasing of unallocated code points aims to identify such interoperability problems and help DNS resolver

operators and implementations to decide when it is ok to disable fallback behavior for future extensions. Hence, this mechanism is expected to generally reduce the need for resolver fallback behavior, and improve security over time.

11. IANA Considerations

[If reserved code point ranges are decided, IANA will need to formally reserve them in the relevant protocol parameter registries. The annotation "Reserved for Greasing" should be employed to clearly distinguish such ranges from other ranges that might be reserved for private use or other purposes.]

12. References

12.1. Normative References

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <https://www.rfc-editor.org/info/rfc1034>.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <https://www.rfc-editor.org/info/rfc1035>.

[RFC6891] Damas, J., Graff, M., and P. Vixie, "Extension Mechanisms for DNS (EDNS(0))", STD 75, RFC 6891, DOI 10.17487/RFC6891, April 2013, <https://www.rfc-editor.org/info/rfc6891>.

12.2. Informative References

[EDNSCOMP] Consortium, I. S., "EDNS Compliance Tester", <https://ednscomp.isc.org/>.

[ERROR-RPT] Arends, R. and M. Larson, "DNS Error Reporting", <https://datatracker.ietf.org/doc/html/draft-ietf-dnsop-dns-error-reporting>.

[FLAGDAY] Consortium, I. S., "DNS Flag Day 2019", <https://www.isc.org/blogs/dns-flag-day/>.

[GREASING] Pardue, L., "Maintaining Protocols Using Grease and Variability", <https://datatracker.ietf.org/doc/html/draft-edm-protocol-greasing>.

[RFC2671] Vixie, P., "Extension Mechanisms for DNS (EDNS0)", RFC 2671, DOI 10.17487/RFC2671, August 1999, <https://www.rfc-editor.org/info/rfc2671>.

[RFC7871] Contavalli, C., van der Gaast, W., Lawrence, D., and W. Kumari, "Client Subnet in DNS Queries", RFC 7871, DOI 10.17487/RFC7871, May 2016, <https://www.rfc-editor.org/info/rfc7871>.

[RFC7873] Eastlake 3rd, D. and M. Andrews, "Domain Name System (DNS) Cookies", RFC 7873, DOI 10.17487/RFC7873, May 2016, <https://www.rfc-editor.org/info/rfc7873>.

[RFC8701] Benjamin, D., "Applying Generate Random Extensions And Sustain Extensibility (GREASE) to TLS Extensibility", RFC 8701, DOI 10.17487/RFC8701, January 2020, <https://www.rfc-editor.org/info/rfc8701>.

Authors' Addresses

Shumon Huque
Salesforce
Email: shuque@gmail.com

Mark Andrews
Internet Systems Consortium
Email: marka@isc.org