Network Working Group
Internet-Draft
Intended status: Informational
Expires: 11 January 2024

L. Pardue
Cloudflare

10 July 2023

Maintaining Protocols Using Grease and Variability
draft-edm-protocol-greasing-02

Abstract

Long-term interoperability of protocols is an important goal of the network standards process. Part of realizing long-term protocol deployment success is the ability to support change. Change can require adjustments such as extension to the protocol, modifying usage patterns within the current protocol constraints, or a replacement protocol. This document present considerations for protocol designers and implementers about applying techniques such as greasing or protocol variability as a means to exercise maintenance concepts.

About This Document

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

The latest revision of this draft can be found at https://intarchboard.github.io/draft-protocol-greasing/draft-edm-protocol-greasing.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-edm-protocol-greasing/.

Source for this draft and an issue tracker can be found at https://github.com/intarchboard/draft-protocol-greasing.

Status of This Memo

This Internet-Draft will expire on 11 January 2024.

Table of Contents

1. Introduction

Long-term interoperability of protocols is an important goal of the network standards process [MAINTENANCE]. Part of realizing long-term protocol deployment success [SUCCESS] is the ability to support change. Change can require adjustments such as extension to the protocol, modifying usage patterns within the current protocol constraints, or a replacement protocol.

Greasing is one technique that supports the long term-viability of protocol extension points. It was originally designed for TLS [GREASE] as a later addition to help mitigate observed deployment issues. Subsequently, other protocols such as QUIC [QUIC] and HTTP/3 [HTTP/3] embedded greasing capability into the protocol, along with policies and IANA registries, in order to avoid ossification-related problems emerging in the first place. Greasing is suitable for many protocols but not all. Section 3.3 of [VIABILITY] discusses the

applicability and limitations of greasing. Section 3 presents considerations for applying grease that help to ensure it can most effectively reach its maintenance goals.

Changing user needs [END-USERS] may require that applications modify how they use a protocol without needing to change the protocol itself. For example, a deployment that supported a download-oriented population might wish to enable support for user uploads. This would change interactions and traffic flows but still behave completely within the design constraints of the network protocols. Implementations and deployments might discover ossification affects this form of change because expectations form around patterns of usage. Section 4 presents considerations about how increasing the variability of protocols can mitigate some of these concerns.

Replacing a protocol may be required where the changing needs or environment push protocol usage outside its original design parameters and extensions cannot elegantly fill the gap. Replacing a protocol may also be desirable as a form of baseline, a formal declaration of protocol and extension(s) combination that is common, that may simplify deployment by reducing failures related to combinatorial extensions problems. A replacement protocol version may or may not be compatible with other versions. A protocol may or may not have a mechanism for version selection or agility. Section 5 presents considerations about designing for and/or implementing version negotiation and migration.

2. Conventions and Definitions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Considerations for Applying Greasing

Greasing can take many forms, depending on the protocol and the nature of its extension points.

Where a protocol uses registered values (i.e. codepoints) or numbers in a well defined range, a common approach (see [GREASE], Section 18.1 of [QUIC], or Section 7.2.8 of [HTTP/3]), is to reserve a subset of the range for the purposes of greasing. Ths approach is detailed more thoroughly in Section 3.3 of [VIABILITY]. However, protocol designers or implementers may find it difficult to apply those suggestions in abstract. The likely success or efficacy of this method can be improved by the following suggestions.

It is assumed that endpoint should implement robust and broad extension handling. When acting as a receiver or a parser, the implementation should not treat codepoints reserved for the purposes of greasing as individually special. In other words, rather than implementation looking specifically for reserved values, it is better to have a "catch all" mechanism that can handle receipt of unknown extensions gracefully or without error.

In order to exercise receiver capability, it is advisable that senders send values from the ranges reserved for greasing. However, picking a deterministic value risks a value becoming ossified itself. One outcome of that is receivers being written to handle a single expected value rather than the generic handling described above. One way to help mitigate this is to reserve a sufficiently large range of values for greasing, and ensure that senders chose values from that range with diversity and non-determinism. The specific nature of size and distribution of the grease range needs to accommodate the protocol constraints. For instance, an 8-bit field can only represent a small range of values and it may be too costly to dedicate many of them solely for the purpose of greasing. However, protocols that use 32-bit or 64-bit fields are unlikely to have restrictions.

It is beneficial to have a large set of reserved numbers for the purpose of greasing. However, protocol designers that wish to do so may encounter difficulties in expressing the large range in their protocol documents and/or in an IANA registry. One approach to this problem has been to define the set algorithmically in the protocol definition and request that IANA reserve only a single entry in the respective table that covers the entire range; see for example https://www.iana.org/assignments/http3-parameters/http3-parameters.xhtml#http3-parameters-frame-types. This range should be protected from registering from any other purpose. Deciding an appropriate label for this protected range is important. Labelling it simply "reserved" may be confused with other ranges that are reserved for private or experimental extensions. An implementer that conflates these two meanings may cause a new class of interoperability failure. Therefore a label such as "reserved for greasing" may help to avoid the confusion. If choosing to use an algorithm to define the set, it is encourage to use unique algorithms. This again improves the chances that receivers will build robust extension handling rather that a simple special-case ignore list.

Protocols that do reserve ranges for greasing introduce a new consideration for real extensions. It is common to want to reserve a block of code points for iteration and experimentation. Depending how the algorithm spreads numbers through the full range, any single block of uninterrupted values may be too small to be usable. This could lead to unintentional use of a greased value.

Since it is intended for receivers to ignore values reserved for greasing, designers and implementers need to remain aware that unintentional use of greased values by a sender for a real extension may cause a failure.

Receiver implementations may unintentionally build a reliance on the occurrence of greasing in the temporal or spatial domain. Senders are advised to implementation non-determinism of when they use grease in addition to what values they send.

4. Considerations for Increasing Protocol Variability

While greasing is one method to deal with falsifying active use of a protocols extensions points, it cannot address positive use. A protocol may define a wide-ranging extension capability but if senders do not use it, then interoperability problems may not arise, leading to ossification until a real use case emerges.

Variation of protocol extension points with positive use in mind can help exercise protocols and ensure long-term maintenance and interoperability. This can be thought of, to some extent, as protocol fuzzing. It can be a difficult area to exercise because varying the protocol elements may change the actual outcome of interactions, leading to real errors. However, some elements can be safely changed and the following are some examples.

QUIC packets contain frames. Receivers might build expectations on the longitudinal aspects of packets or frames - size, ordering, frequency, etc. A sender can quite often manipulate these parameters and stay compliant to the requirements of the QUIC protocol. For example, QUIC streams are an ordered reliable byte stream that is serialized as a sequence of STREAM frames with a length and offset. Receivers are expected to reassemble frames into an ordered reliable byte stream such that an application reading from a stream can be abstracted from matters of the transport later. A sender that purposefully reorders STREAM frames will help exercise the reassembly features of the receiver. It is not expected that this would cause a functional failure in the application layer. However, it may introduce delays or stream head-of-line blocking that affect the performance aspects of a transmission, which may not be acceptable for a given use case.

5. Considerations for Protocol Versions

There are intrinsic and well-documented issues related to testing version negotiation of protocols; see [EXTENSIBILITY] and Sections 2.1 and 3.2 of [VIABILITY]. This section will be expanded with advice for protocol designers and implementers about how to approach these problems.

6. Security Considerations

The considerations in [MAINTENANCE], [GREASE], [END-USERS], and [VIABILITY] all apply to the topics discussed in this document.

The use of protocol features, extensions, and versions can already allow fingerprinting. Any techniques that change parameters in any way, including but not limited to those discussed in this document, can affect fingerprinting. A deeper analysis of this topic has been deemed out of scope.

7. IANA Considerations

This document has no IANA actions.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

8.2. Informative References

[END-USERS] Nottingham, M., "The Internet is for End Users", RFC 8890, DOI 10.17487/RFC8890, August 2020, <https://www.rfc-editor.org/rfc/rfc8890>.

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

[GREASE] 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/rfc/rfc8701>.

[HTTP/3] Bishop, M., Ed., "HTTP/3", RFC 9114, DOI 10.17487/RFC9114, June 2022, <https://www.rfc-editor.org/rfc/rfc9114>.

[MAINTENANCE] Thomson, M. and D. Schinazi, "Maintaining Robust Protocols", RFC 9413, DOI 10.17487/RFC9413, June 2023, <https://www.rfc-editor.org/rfc/rfc9413>.

[QUIC] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <https://www.rfc-editor.org/rfc/rfc9000>.

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

[VIABILITY] Thomson, M. and T. Pauly, "Long-Term Viability of Protocol Extension Mechanisms", RFC 9170, DOI 10.17487/RFC9170, December 2021, <https://www.rfc-editor.org/rfc/rfc9170>.

Acknowledgments

This work is a summary of the topics discussed during EDM meetings. The contributors at those meetings are thanked.

Author's Address

Lucas Pardue
Cloudflare
Email: lucaspardue.24.7@gmail.com