Applying GREASE to TLS Extensibility
draft-ietf-tls-grease-03

Abstract

   This document describes GREASE (Generate Random Extensions And
   Sustain Extensibility), a mechanism to prevent extensibility failures
   in the TLS ecosystem. It reserves a set of TLS protocol values that
   may be advertised to ensure peers correctly handle unknown values.

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 January 30, 2020.

Table of Contents

1. Introduction

   The TLS protocol [RFC8446] includes several points of extensibility,
   including the list of cipher suites and several lists of extensions.
   The values transmitted in these lists identify implementation
   capabilities. TLS follows a model where one side, usually the
   client, advertises capabilities and the peer, usually the server,
   selects them. The responding side must ignore unknown values so that
   new capabilities may be introduced to the ecosystem while maintaining
   interoperability.

   However, bugs may cause an implementation to reject unknown values.
   It will interoperate with existing peers, so the mistake may spread
   through the ecosystem unnoticed. Later, when new values are defined,
   updated peers will discover that the metaphorical joint in the
   protocol has rusted shut and that the new values cannot be deployed
   without interoperability failures.

   To avoid this problem, this document reserves some currently unused
   values for TLS implementations to advertise at random. Correctly
   implemented peers will ignore these values and interoperate. Peers
   that do not tolerate unknown values will fail to interoperate,
   revealing the mistake before it is widespread.

   In keeping with the rusted joint metaphor, this technique is named
   GREASE (Generate Random Extensions And Sustain Extensibility).

1.1. Requirements Language

   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 RFC 2119 [RFC2119][RFC2119] RFC 8174 [RFC8174] when, and only
   when, they appear in all capitals, as shown here.

2. GREASE Values

   This document reserves a number of TLS protocol values, referred to
   as GREASE values. These values were allocated sparsely to discourage
   server implementations from conditioning on them. For convenience,
   they were also chosen so all types share a number scheme with a
   consistent pattern while avoiding collisions with any existing
   applicable registries in TLS.

   RFC EDITOR: PLEASE REMOVE THE FOLLOWING PARAGRAPH. The values
   prefaced with {TBD} are suggested values and subject to change prior
   to final allocation by IANA.

   The following values are reserved as GREASE values for cipher suites
   and ALPN [RFC7301] identifiers:

      {TBD} {0x0A,0x0A}
      {TBD} {0x1A,0x1A}
      {TBD} {0x2A,0x2A}
      {TBD} {0x3A,0x3A}
      {TBD} {0x4A,0x4A}
      {TBD} {0x5A,0x5A}
      {TBD} {0x6A,0x6A}
      {TBD} {0x7A,0x7A}
      {TBD} {0x8A,0x8A}
      {TBD} {0x9A,0x9A}
      {TBD} {0xAA,0xAA}
      {TBD} {0xBA,0xBA}
      {TBD} {0xCA,0xCA}
      {TBD} {0xDA,0xDA}
      {TBD} {0xEA,0xEA}
      {TBD} {0xFA,0xFA}

   The following values are reserved as GREASE values for extensions,
   named groups, signature algorithms, and versions:

      {TBD} 0x0A0A
      {TBD} 0x1A1A
      {TBD} 0x2A2A
      {TBD} 0x3A3A

```
{TBD} 0x4A4A
{TBD} 0x5A5A
{TBD} 0x6A6A
{TBD} 0x7A7A
{TBD} 0x8A8A
{TBD} 0x9A9A
{TBD} 0xAAAA
{TBD} 0xBABA
{TBD} 0xCACA
{TBD} 0xDADA
{TBD} 0xEAEA
{TBD} 0xFAFA
```

The values allocated above are thus no longer available for use as
TLS or DTLS [RFC6347] version numbers.

The following values are reserved as GREASE values for
PskKeyExchangeModes.

```
{TBD} 0x0B
{TBD} 0x2A
{TBD} 0x49
{TBD} 0x68
{TBD} 0x87
{TBD} 0xA6
{TBD} 0xC5
{TBD} 0xE4
```

3. Client-Initiated Extension Points

Most extension points in TLS are offered by the client and selected
by the server. This section details client and server behavior
around GREASE values for these.

3.1. Client Behavior

When sending a ClientHello, a client MAY behave as follows:

o A client MAY select one or more GREASE cipher suite values and
   advertise them in the "cipher_suites" field.

o A client MAY select one or more GREASE extension values and
   advertise them as extensions with varying length and contents.

o A client MAY select one or more GREASE named group values and
   advertise them in the "supported_groups" extension, if sent. It
   MAY also send KeyShareEntry values for a subset of those selected

in the "key_share" extension. For each of these, the
"key_exchange" field MAY be any value.

o A client MAY select one or more GREASE signature algorithm values
   and advertise them in the "signature_algorithms" or
   "signature_algorithms_cert" extensions, if sent.

o A client MAY select one or more GREASE version values and
   advertise them in the "supported_versions" extension, if sent.

o A client MAY select one or more GREASE PskKeyExchangeMode values
   and advertise them in the "psk_key_exchange_modes" extension, if
   sent.

o A client MAY select one or more GREASE ALPN identifiers and
   advertise them in the "application_layer_protocol_negotiation"
   extension, if sent.

Clients MUST reject GREASE values when negotiated by the server. In
particular, the client MUST fail the connection if a GREASE value
appears any in the following:

o The "version" value in a ServerHello or HelloRetryRequest

o The "cipher_suite" value in a ServerHello

o Any ServerHello extension

o Any HelloRetryRequest, EncryptedExtensions, or Certificate
   extension in TLS 1.3

o The "namedcurve" value in a ServerKeyExchange for an ECDHE cipher
   in TLS 1.2 or earlier

o The signature algorithm in a ServerKeyExchange signature in TLS
   1.2 or earlier

o The signature algorithm in a server CertificateVerify signature in
   TLS 1.3

Note that this requires no special processing on the client. Clients
are already required to reject unknown values selected by the server.

3.2. Server Behavior

When processing a ClientHello, servers MUST NOT treat GREASE values
differently from any unknown value. Servers MUST NOT negotiate any
GREASE value when offered in a ClientHello. Servers MUST correctly

ignore unknown values in a ClientHello and attempt to negotiate with
one of the remaining parameters. (There may not be any known
parameters remaining, in which case parameter negotiation will fail.)

Note that these requirements are restatements or corollaries of
existing server requirements in TLS.

4. Server-Initiated Extension Points

Some extension points are offered by the server and selected by the
client. This section details client and server behavior around
GREASE values for these.

4.1. Server Behavior

When sending a CertificateRequest in TLS 1.3, a server MAY behave as
follows:

o A server MAY select one or more GREASE extension values and
   advertise them as extensions with varying length and contents.

o A server MAY select one or more GREASE signature algorithm values
   and advertise them in the "signature_algorithms" or
   "signature_algorithms_cert" extensions, if present.

When sending a NewSessionTicket message in TLS 1.3, a server MAY
select one or more GREASE extension values and advertise them as
extensions with varying length and contents.

Servers MUST reject GREASE values when negotiated by the client. In
particular, the server MUST fail the connection if a GREASE value
appears any in the following:

o Any Certificate extension in TLS 1.3

o The signature algorithm in a client CertificateVerify signature

Note that this requires no special processing on the server. Servers
are already required to reject unknown values selected by the client.

4.2. Client Behavior

When processing a CertificateRequest or NewSessionTicket, clients
MUST NOT treat GREASE values differently from any unknown value.
Clients MUST NOT negotiate any GREASE value when offered by the
server. Clients MUST correctly ignore unknown values offered by the
server and attempt to negotiate with one of the remaining parameters.

(There may not be any known parameters remaining, in which case
parameter negotiation will fail.)

Note that these requirements are restatements or corollaries of
existing client requirements in TLS.

5. Sending GREASE Values

Implementations advertising GREASE values SHOULD select them at
random. This is intended to encourage implementations to ignore all
unknown values rather than any individual value. Implementations
MUST honor protocol specifications when sending GREASE values. For
instance, section 4.2 of [RFC8446] forbids duplicate extension types
within a single extension block. Implementations sending multiple
GREASE extensions in a single block thus must ensure the same value
is not selected twice.

Implementations SHOULD balance diversity in GREASE advertisements
with determinism. For example, a client which randomly varies GREASE
value positions for each connection may only fail against a broken
server with some probability. This risks the failure being masked by
automatic retries. A client which positions GREASE values
deterministically over a period of time (such as a single software
release) stresses fewer cases but is more likely to detect bugs from
those cases.

6. IANA Considerations

This document updates the TLS Cipher Suite Registry, available from
<https://www.iana.org/assignments/tls-parameters>:

Value	Description	DTLS-OK	Recommended	Reference
{TBD} {0x0A,0x0A}	Reserved	Y	N	(this document)
{TBD} {0x1A,0x1A}	Reserved	Y	N	(this document)
{TBD} {0x2A,0x2A}	Reserved	Y	N	(this document)
{TBD} {0x3A,0x3A}	Reserved	Y	N	(this document)
{TBD} {0x4A,0x4A}	Reserved	Y	N	(this document)
{TBD} {0x5A,0x5A}	Reserved	Y	N	(this document)
{TBD} {0x6A,0x6A}	Reserved	Y	N	(this document)
{TBD} {0x7A,0x7A}	Reserved	Y	N	(this document)
{TBD} {0x8A,0x8A}	Reserved	Y	N	(this document)
{TBD} {0x9A,0x9A}	Reserved	Y	N	(this document)
{TBD} {0xAA,0xAA}	Reserved	Y	N	(this document)
{TBD} {0xBA,0xBA}	Reserved	Y	N	(this document)
{TBD} {0xCA,0xCA}	Reserved	Y	N	(this document)
{TBD} {0xDA,0xDA}	Reserved	Y	N	(this document)
{TBD} {0xEA,0xEA}	Reserved	Y	N	(this document)
{TBD} {0xFA,0xFA}	Reserved	Y	N	(this document)

Additions to the TLS Cipher Suite Registry

RFC EDITOR: PLEASE REMOVE THE FOLLOWING PARAGRAPH. The cipher suite
numbers listed in the first column are numbers used for
interoperability testing and it's suggested that IANA use these
values for assignment.

This document updates the Supported Groups Registry, available from
<https://www.iana.org/assignments/tls-parameters>:

Value	Description	DTLS-OK	Recommended	Reference
{TBD} 2570	Reserved	Y	N	(this document)
{TBD} 6682	Reserved	Y	N	(this document)
{TBD} 10794	Reserved	Y	N	(this document)
{TBD} 14906	Reserved	Y	N	(this document)
{TBD} 19018	Reserved	Y	N	(this document)
{TBD} 23130	Reserved	Y	N	(this document)
{TBD} 27242	Reserved	Y	N	(this document)
{TBD} 31354	Reserved	Y	N	(this document)
{TBD} 35466	Reserved	Y	N	(this document)
{TBD} 39578	Reserved	Y	N	(this document)
{TBD} 43690	Reserved	Y	N	(this document)
{TBD} 47802	Reserved	Y	N	(this document)
{TBD} 51914	Reserved	Y	N	(this document)
{TBD} 56026	Reserved	Y	N	(this document)
{TBD} 60138	Reserved	Y	N	(this document)
{TBD} 64250	Reserved	Y	N	(this document)

                 Additions to the Supported Groups Registry

   RFC EDITOR: PLEASE REMOVE THE FOLLOWING PARAGRAPH. The named group
   numbers listed in the first column are numbers used for
   interoperability testing and it's suggested that IANA use these
   values for assignment.

   This document updates the ExtensionType Values registry, available
   from <https://www.iana.org/assignments/tls-extensiontype-values>:

Value	Extension name	TLS 1.3	Recommended	Reference
{TBD} 2570	Reserved	CH, CR, NST	N	(this document)
{TBD} 6682	Reserved	CH, CR, NST	N	(this document)
{TBD} 10794	Reserved	CH, CR, NST	N	(this document)
{TBD} 14906	Reserved	CH, CR, NST	N	(this document)
{TBD} 19018	Reserved	CH, CR, NST	N	(this document)
{TBD} 23130	Reserved	CH, CR, NST	N	(this document)
{TBD} 27242	Reserved	CH, CR, NST	N	(this document)
{TBD} 31354	Reserved	CH, CR, NST	N	(this document)
{TBD} 35466	Reserved	CH, CR, NST	N	(this document)
{TBD} 39578	Reserved	CH, CR, NST	N	(this document)
{TBD} 43690	Reserved	CH, CR, NST	N	(this document)
{TBD} 47802	Reserved	CH, CR, NST	N	(this document)
{TBD} 51914	Reserved	CH, CR, NST	N	(this document)
{TBD} 56026	Reserved	CH, CR, NST	N	(this document)
{TBD} 60138	Reserved	CH, CR, NST	N	(this document)
{TBD} 64250	Reserved	CH, CR, NST	N	(this document)

Additions to the ExtensionType Values registry

RFC EDITOR: PLEASE REMOVE THE FOLLOWING PARAGRAPH. The extension numbers listed in the first column are numbers used for interoperability testing and it's suggested that IANA use these values for assignment.

This document updates the TLS Application-Layer Protocol Negotiation (ALPN) Protocol IDs registry, available from

<https://www.iana.org/assignments/tls-extensiontype-values/tls-extensiontype-values>:

```
+----------+-------------------------+----------------+
| Protocol | Identification Sequence |    Reference   |
+----------+-------------------------+----------------+
| Reserved |     {TBD} 0x0A 0x0A     | (this document)|
| Reserved |     {TBD} 0x1A 0x1A     | (this document)|
| Reserved |     {TBD} 0x2A 0x2A     | (this document)|
| Reserved |     {TBD} 0x3A 0x3A     | (this document)|
| Reserved |     {TBD} 0x4A 0x4A     | (this document)|
| Reserved |     {TBD} 0x5A 0x5A     | (this document)|
| Reserved |     {TBD} 0x6A 0x6A     | (this document)|
| Reserved |     {TBD} 0x7A 0x7A     | (this document)|
| Reserved |     {TBD} 0x8A 0x8A     | (this document)|
| Reserved |     {TBD} 0x9A 0x9A     | (this document)|
| Reserved |     {TBD} 0xAA 0xAA     | (this document)|
| Reserved |     {TBD} 0xBA 0xBA     | (this document)|
| Reserved |     {TBD} 0xCA 0xCA     | (this document)|
| Reserved |     {TBD} 0xDA 0xDA     | (this document)|
| Reserved |     {TBD} 0xEA 0xEA     | (this document)|
| Reserved |     {TBD} 0xFA 0xFA     | (this document)|
+----------+-------------------------+----------------+
```

             Additions to the ALPN Protocol IDs registry

7. Security Considerations

   GREASE values may not be negotiated, so they do not directly impact
   the security of TLS connections.

   Historically, when interoperability problems arise in deploying new
   TLS features, implementations have used a fallback retry on error
   with the feature disabled. This allows an active attacker to
   silently disable the new feature. By preventing a class of such
   interoperability problems, GREASE reduces the need for this kind of
   fallback.

   If an implementation does not select GREASE values at random it is
   possible it will allow for fingerprinting of the implementation or
   perhaps even of individual users. This can result in a negative
   impact to a user's privacy.

8. Acknowledgements

   The author would like to thank Adam Langley, Nick Harper, and Steven
   Valdez for their feedback and suggestions. In addition, the rusted
   joint metaphor is originally due to Adam Langley.

[9](). 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/info/rfc2119]()>.

[RFC6347]   Rescorla, E. and N. Modadugu, "Datagram Transport Layer
            Security Version 1.2", [RFC 6347](), DOI 10.17487/RFC6347,
            January 2012, <[https://www.rfc-editor.org/info/rfc6347]()>.

[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, <[https://www.rfc-editor.org/info/rfc7301]()>.

[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/info/rfc8174]()>.

[RFC8446]   Rescorla, E., "The Transport Layer Security (TLS) Protocol
            Version 1.3", [RFC 8446](), DOI 10.17487/RFC8446, August 2018,
            <[https://www.rfc-editor.org/info/rfc8446]()>.

Author's Address

   David Benjamin
   Google LLC

   Email: davidben@google.com