openpgp
Internet-Draft
Updates: 9580 (if approved)
Intended status: Standards Track
Expires: 6 December 2026

A. Gallagher, Ed.
PGPKeys.EU

4 June 2026

OpenPGP Message Grammar
draft-gallagher-openpgp-messages-00

Abstract

This document specifies several updates and clarifications to the grammar and semantics of OpenPGP messages.

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://andrewgdotcom.gitlab.io/openpgp-messages. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-gallagher-openpgp-messages/.

Discussion of this document takes place on the OpenPGP Working Group mailing list (mailto:openpgp@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/openpgp/. Subscribe at https://www.ietf.org/mailman/listinfo/openpgp/.

Source for this draft and an issue tracker can be found at https://gitlab.com/andrewgdotcom/openpgp-messages.

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 6 December 2026.

Table of Contents

1. Introduction

OpenPGP messages have a complex grammar and sometimes poorly-understood semantics. This document attempts to address this by:

* Expanding on specifications where [RFC9580] does not fully describe the existing or expected behaviour of deployed implementations.

* Adding clarification where deployed implementations differ in their interpretation of [RFC9580] and its predecessors.

* Deprecating unused or error-prone features.

This document does not specify any new wire formats.

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. Signed Messages

The accepted convention is that a prefixed Signature packet signs over the next literal packet only, skipping any intervening signatures - however this is not explicitly specified in [RFC9580]. Historically, PGP 2.X treated a prefixed Signature packet as applying to the entire following sequence of packets, but this usage is deprecated [FINNEY1998]. See Section 3.4 for an alternative construction.

In addition, One-Pass Signature (OPS) nesting semantics are complex, and under-specified [SCHAUB2022]. Section 5.4 of [RFC9580] defines the nesting octet as:

> A 1-octet number holding a flag showing whether the signature is nested. A zero value indicates that the next packet is another One-Pass Signature packet that describes another signature to be applied to the same message data.

The terminology is imprecise, and non-zero "nesting" flags are completely unspecified. One self-consistent interpretation is as follows:

* A zero nesting octet means that the following OPS and its counterpart signature are not signed over by the current OPS.
  - This process is recursive if multiple sequential OPS packets have a nesting octet of zero.
* To add multiple OPS signatures over the same message data, all OPS constructions except the innermost one have the nesting octet zeroed.
  - It is not clear what happens if the innermost nesting octet is zero but no OPS packet follows.

The above implies that an OPS with a nonzero nesting octet signs over all packets between the OPS packet and its matching signature packet, including any further signatures, however it is not clear whether any current implementation supports this.

This is further expanded in [OPENPGPDEVBOOK].

This still leaves us with an overly complex grammar that resists rigorous formalization. We attempt to improve the formalism below.

3.1. Prefix-Signed Message Constraints

Prefixed signatures are deprecated, and their use is hereby restricted:

* A prefixed Signature packet MUST be followed by one Literal Data packet and no other packets except further prefixed Signature packets, and Marker packets.

* A prefixed Signature packet MUST NOT have a version number greater than 4.

Prefixed signatures SHOULD NOT be generated, but MAY be interpreted. A receiving implementation MAY convert a prefixed signature to the equivalent OPS signature, by moving the signature after the Literal Data packet and prefixing an appropriate OPS packet.

3.2. OPS Message Constraints

We constrain OPS structures to a subset of previously-allowed configurations:

* Prefixed signatures and OPS signatures MUST NOT both be used in the same message.

* When generating an OPS packet that is not followed by another OPS packet, the nesting octet SHOULD be set to 1.

  - Otherwise, the nesting octet SHOULD be set to 0.

* When consuming an OPS packet, the nesting octet MUST be ignored.

This effectively deprecates the nesting octet, while maintaining backwards compatibility with legacy code.

3.3. Subject Normalization

The _subject_ of an OpenPGP signature refers to the packet(s) that are signed over. The _type-specific data_ of an OpenPGP signature refers to the section of the data stream that is passed to the signature's digest function after the optional salt and before the trailer. The type-specific data differs from the subject in that it has been normalized, the details of which are dependent on the signature type.

The subject of a signature in the Literal Data or Timestamping categories ([I-D.gallagher-openpgp-signatures]) is the Literal Data packet that immediately follows one or more prefixed signatures, or is enclosed by one or more OPS constructions. If no Literal Data packet is present, the signature is malformed.

The following normalization steps are applied to the subject of the signature to produce the type-specific data:

* The framing of the Literal Data packet is discarded, and any partial-length packets are concatenated.

* If the Signature Type is 0x01, the Literal Data packet body is converted to Canonical Text, by converting line endings to CRLF and removing any trailing whitespace (Section 3.3.1).

A One-Pass Signature over a Literal Data packet, a prefixed Signature over the same packet, and a detached signature over a file containing the body of the same packet are all calculated the same way. This means that they can be losslessly transformed into each other with the exception of the Literal Data metadata fields, which an application MAY assume contain their recommended default values as per Section 5.9 of [RFC9580].

A signature of Type 0x01 MUST NOT be made over arbitrary binary data, only over UTF-8 text.

3.3.1. Line Ending Normalization

When normalizing line endings, only bare linefeeds (an LF control character that is not preceded by a CR) are normalized to CRLF. In particular, bare carriage returns MUST NOT be converted to CRLF.

3.4. Nested Signatures

To sign over an entire signed message together with its signatures, the wire format of the inner message SHOULD first be encapsulated in a Literal Data packet. A Canonical Text signature MUST NOT be made over such a nested message, and the Cleartext Signature Framework MUST NOT be used.

Beware that the outer signature will thus be sensitive to the inner message's packet framing, i.e. the otherwise inconsequential choice of packet header format and partial body lengths. If the inner message is parsed and re-serialized unmodified, but using a different framing, the outer signature will no longer validate.

4. Formal Grammar

The message grammar in Section 10.3 of [RFC9580] is therefore updated to:

* Literal Message:
  Literal Data Packet.

* Encrypted Session Key:
  Public Key Encrypted Session Key Packet | Symmetric Key Encrypted Session Key Packet.

* Encrypted Data:
  Symmetrically Encrypted Data Packet | Symmetrically Encrypted and Integrity Protected Data Packet.

* Encrypted Message:
  Encrypted Data | Encrypted Session Key, Encrypted Message.

* Prefixed Signed Message: Signature Packet, Prefixed Signed Message | Literal Message.

* Multiply One-Pass Signed Message:
  One-Pass Signature Packet (with nesting octet 0), One-Pass Signed Message, Corresponding Signature Packet.

* Singly One-Pass Signed Message:
  One-Pass Signature Packet (with nesting octet 1), Literal Message, Corresponding Signature Packet.

* One-Pass Signed Message:
  Multiply One-Pass Signed Message | Singly One-Pass Signed Message.

* Signed Message:
  Prefixed Signed Message | One-Pass Signed Message.

* Optionally Signed Message:
  Signed Message | Literal Message.

* Compressed Message:
  Compressed Data Packet.

* Unencrypted Message:
  Compressed Message | Optionally Signed Message.

* Optionally Padded Unencrypted Message:
  Unencrypted Message | Unencrypted Message, Padding Packet.

* OpenPGP Message:
  Encrypted Message | Unencrypted Message.

In addition to these rules, a Marker packet (Section 5.8 of [RFC9580]) can appear anywhere in the sequence.

5. Encrypted and Compressed Messages

[RFC9580] permits an encrypted message to contain another encrypted message, and a compressed message to contain another compressed message, possibly recursively. Such messages require potentially unbounded resources for negligible added utility, and therefore MUST NOT be created.

In addition, encrypt-then-sign messages are not idiomatic OpenPGP, and MUST NOT be generated.

The guidance in Section 10.3.1 of [RFC9580] is therefore updated to:

* Decrypting a version 2 Symmetrically Encrypted and Integrity Protected Data packet MUST yield a valid Optionally Padded Unencrypted Message.

* Decrypting a version 1 Symmetrically Encrypted and Integrity Protected Data packet or -- for historic data -- a Symmetrically Encrypted Data packet MUST yield a valid Unencrypted Message.

* Decompressing a Compressed Data packet MUST yield a valid Optionally Signed Message.

6. Security Considerations

The OPS nesting octet is not signed over and is malleable in principle. An intermediary could swap an outer OPS with its inner OPS by also swapping the nesting octets. The order of OPS nesting therefore MUST NOT be considered meaningful.

In addition, the normalization applied during Literal Data signature calculation may result in semantic collisions. It is possible to construct distinct sequences of packets that map to the same sequence of octets after Literal Data normalization is applied. It is not known whether such a pair of colliding packet sequences might also have different semantics.

7. IANA Considerations

7.1. OpenPGP Packet Types Registry

IANA is requested to update the following existing entry in the registry, to add a reference to this document:

* OPS Packet

8. References

8.1. Normative References

[I-D.gallagher-openpgp-signatures]
Gallagher, A. and D. K. Gillmor, "OpenPGP Signatures and Signed Messages", Work in Progress, Internet-Draft, draft-gallagher-openpgp-signatures-02, 3 November 2025, <https://datatracker.ietf.org/doc/html/draft-gallagher-openpgp-signatures-02>.

[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>.

[RFC9580] Wouters, P., Ed., Huigens, D., Winter, J., and Y. Niibe, "OpenPGP", RFC 9580, DOI 10.17487/RFC9580, July 2024, <https://www.rfc-editor.org/rfc/rfc9580>.

8.2. Informative References

[FINNEY1998]
Finney, H., "Re: More spec-ulations - update", 26 March 1998, <https://mailarchive.ietf.org/arch/msg/openpgp/U4Qg3Z9bj-RDgpwW5nmRNetOZKY/>.

[OPENPGPDEVBOOK]
"OpenPGP for Application Developers", 6 May 2024, <https://openpgp.dev/book/>.

[SCHAUB2022]
Schaub, P., "[openpgp] Proposing a Simplification of Message Syntax", 7 October 2022, <https://mailarchive.ietf.org/arch/msg/openpgp/uepOF6XpSegMO4c59tt9e5H1i4g/>.

Appendix A. Acknowledgments

This document would not have been possible without the extensive work of the authors of [OPENPGPDEVBOOK].

The author would also like to thank Daniel Huigens, Daniel Kahn Gillmor, Heiko Sch辰fer, Neal Walfield, Justus Winter and Paul Schaub for additional discussions and suggestions.

Author’s Address

Andrew Gallagher (editor)
PGPKeys.EU
Email: andrewg@andrewg.com