Workgroup:
Open Specification for Pretty Good Privacy

Internet-Draft:
draft-wussler-openpgp-forwarding-00

Published: 10 July 2023
Intended Status: Informational
Expires: 11 January 2024
Authors: A. Wussler
Proton AG

Automatic Forwarding for ECDH Curve25519 OpenPGP messages

Abstract

An OpenPGP user may want to request their email provider to automatically forward some or all of the messages they receive to a third party. Given that messages are encrypted, this requires transforming them into ciphertexts decryptable by the intended forwarded parties, while maintaining confidentiality and authentication. This can be achieved using Proxy transformations on the Curve25519 elliptic curve field with minimal changes to the OpenPGP protocol, in particular no change is required on the sender side. In this document we implement the forwarding scheme described in [FORWARDING].

About This Document

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

Status information for this document may be found at https://datatracker.ietf.org/doc/draft-wussler-openpgp-forwarding/.

Discussion of this document takes place on the Open Specification for Pretty Good Privacy 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://github.com/wussler/draft-forwarding.

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/.
Table of Contents

1. Introduction
2. Conventions and Definitions
3. Terminology
4. Description of the protocol
   4.1. Key Flag 0x40
5. Setting up a forwarding instance
   5.1. Generating the forwardee key
   5.2. Computing the proxy parameter
6. Forwarding messages
7. Decrypting forwarded messages
8. Security Considerations
   8.1. Collusion between Proxy and Forwardee
   8.2. Key Flags
   8.3. Proxy transformation factors management
   8.4. Proxy transformation
   8.5. Message forwarding selection
9. IANA Considerations
10. References
    10.1. Normative References
    10.2. Informative References
Appendix A. Test vectors
   A.1. Proxy parameter
   A.2. Message transformation
   A.3. End-to-end tests
Contributors
1. Introduction

An OpenPGP user might be interested in forwarding their email to another user without delegating decryption or interacting beyond protocol setup. In this document we outline the changes necessary to the OpenPGP protocol to safely allow:

*Recipients to delegate trust to third parties to read their messages;

*MTAs to act as cryptographic Proxies and transform select messages;

*Forwardees to read the transformed email.

This is achieved by diverting the ECDH key exchange of messages encrypted using Curve25519, as described in [FORWARDING]. It requires a proxy to multiply the ephemeral ECDH value by a known factor on the elliptic curve field, and the forwardee to alter the Key Derivation Function (KDF) when computing the Key Encryption Key (KEK) in a Public Key Encrypted Session Key Packet (PKESK).

Security is provided as long as there is no collusion involving the Proxy, i.e. we consider that the MTA that takes care of the forwarding is a semi-trusted proxy that is not able to decrypt.

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. Terminology

Sender: The person who originally sends the email. They are no active part in this protocol as this forwarding scheme is transparent to them and they are unaware such transformation is being done.

Forwarder: The intended recipient of the email, as specified by the sender. They delegate the trust by setting up the protocol.

Forwardee: The person who receives the forwarded email.
**Forwardee subkey**: An OpenPGP encryption subkey generated by the forwarder for the forwardee that allows them to read the transformed messages.

**Proxy**: An OpenPGP-aware Mail Transfer Agent (MTA) with the task of forwarding (some or all) emails intended for the forwarder to the forwardee.

**Proxy parameter**: A value that enables the proxy to transform a message from being decryptable with one key, to being decryptable with another key.

4. **Description of the protocol**

In this section we'll provide an illustration of the overall protocol.

**NON-NORMATIVE EXPLANATION**

The scenario we address is the following: Bob (the recipient) wants to allow Charles (the forwardee) to decrypt email that was originally encrypted to Bob's public key without having access to Bob’s private key or any online interaction. Naturally, MTAs (the Proxies) should not have the ability to read the contents of such messages. To achieve this, the protocol requires to be set up:

First, Bob generates two secret elements, a regular secret key, and a proxy factor K; second, Bob securely transfers the key to Charles and the proxy factor to the trusted MTA. With the proxy factor, the MTA gains the ability to transform any PGP message encrypted to Bob’s public key into another PGP message that can be decrypted with the newly generated private key, which is now held by Charles. At the same time, the MTA cannot decrypt the message, nor transform it to another public key. Upon participating in ECDH key exchanges, proxies need to store one random field element and two OpenPGP Key IDs per forwarding, and compute a single scalar multiplication on the elliptic curve per forwarded ciphertext. In the following illustration, we show an example with a sender (Alice), a recipient (Bob), multiple direct forwardees (Charles and Daniel), and one indirect forwardee (Frank). The proxy transformations are done by the two MTAs using the proxy transformation parameters K_BC, K_BD, and K_DF. This transforms the Public Key Encrypted Session Key Packet P_B into P_C, P_D, and P_F, while the Symmetrically Encrypted Data c is not transformed.
In this document we define the protocol for a single instance, but the same procedure can be applied to multiple recipients independently. Each instance **MUST** have an independent instantiation, generating fresh keys and computing separate proxy transformation parameters.

### 4.1. Key Flag 0x40

The key flag 0x40 is added to the first octet of the key flags (Table 9 of [I-D.ietf-openpgp-crypto-refresh]). It indicates that the key may be used to decrypt forwarded communications.
This is intended to prevent implementations unaware of forwarding keys from using this key for direct encryption, and thus generating unreadable messages.

An implementation **SHOULD NOT** export public subkeys with key flag 0x40. A public key directory **SHOULD NOT** accept subkeys with key flag 0x40.

Keys with this flag **MUST** have the forwarding KDF parameters version 0xFF defined in Section 5.1.

Subkeys flagged as 0x40 **MUST NOT** be unflagged or reused as the private key material is generated from a third party and therefore is not secret.

5. Setting up a forwarding instance

This section describes how to compute a proxy transformation parameter and a forwardee subkey for a v4 OpenPGP certificate with a Curve25519 encryption-only subkey.

The subkeys used for forwarding **MUST** be ECDH keys (algorithm ID 18, as defined in Section 9.1 of [I-D.ietf-openpgp-crypto-refresh]) with only the 0x04 (encrypt communications) and/or 0x08 (encrypt storage) key flags set (as defined in Section 5.2.3.29 of [I-D.ietf-openpgp-crypto-refresh]). The original key **MUST** contain at least one subkey suitable for forwarding. An implementation **SHOULD** generate a proxy parameter for all the valid subkeys suitable for forwarding.

5.1. Generating the forwardee key

The implementation **MUST** generate a fresh OpenPGP certificate with only Curve25519 encryption subkeys. There **MUST** be the same amount of subkeys as the number of forwarder subkeys being transformed. This key **SHOULD** have the identity of the forwardee in the user ID.

The forwardee subkeys **MUST** have the following Key Flags, defined in [I-D.ietf-openpgp-crypto-refresh] Section 5.2.3.29, in the subkey binding signature:

* **0x10** - The private component of this key may have been split by a secret-sharing mechanism.

* **0x40** - This key may be used for forwarded communications.

Furthermore the flag **0x10 MAY** be added to the existing recipient encryption subkey, if the implementation desires to make the forwarding known to other parties.
The forwardee encryption subkey **MUST** contain the following variable-length field containing KDF parameters, which is formatted as follows, differing from [I-D.ietf-openpgp-crypto-refresh], Section 11.5:

- A one-octet size of the following fields; values 0 and 0xFF are reserved for future extensions,
- A one-octet value 0xFF, indicating a fingerprint replacement.
- A one-octet hash function ID used with a KDF.
- A one-octet algorithm ID for the symmetric algorithm used to wrap the symmetric key used for the message encryption; see [I-D.ietf-openpgp-crypto-refresh] Section 11.5 for details.
- A 20-octet version 4 key fingerprint to be used in the KDF.

The forwardee key **MUST** be communicated securely to the forwardee.

### 5.2. Computing the proxy parameter

Given the recipient and forwardee encryption subkeys, the recipient’s implementation **MUST** compute the proxy transformation parameter as specified.

```c
// Implements ComputeProxyParameter( dB, dC );
// Input:
// dB - the recipient's private key integer
// dC - the forwardee's private key integer
// n - the size of the field of Curve25519

k = dB/dC mod n
return k
```

The value \( n \) is defined in [RFC7748] as:

\[
2^{252} + 0x14def9dea2f79cd65812631a5cf5d3ed
\]

Converted to hex:

```
10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
14 de f9 de a2 f7 9c d6 58 12 63 1a 5c f5 d3 ed
```

The value \( k \) is then encoded as little-endian in a 32-byte octet string, and referred as proxy transformation parameter.

The proxy transformation parameter **MUST** be communicated securely to the MTA acting as proxy. The proxy **MUST** safely store it in a way that
is not accessible by other parties. The proxy **MUST** delete the parameter when the forwarding is revoked.

6. **Forwarding messages**

When forwarding a message, the proxy **MUST** parse the PKESK and check whether the key ID embedded in the PKESK, as specified in [I-D.ietf-openpgp-crypto-refresh] Section 5.1.1, matches the recipient's subkey key ID designated for forwarding. If the value differs, the proxy **SHOULD NOT** transform the message. If the key ID is set to version 0 for "anonymous recipient", see [I-D.ietf-openpgp-crypto-refresh] Section 5.1.8, the proxy **MAY** transform all PKESKs in a message that it is supposed to forward. In this case it **SHOULD** leave all key IDs unaltered to 0.

The proxy **MUST** then check that the ephemeral public key does not belong to a small subgroup of the curve. This is done by parsing the MPI of an EC point as specified in [I-D.ietf-openpgp-crypto-refresh] Section 5.1.5, multiplying by the integer 0x08. If this multiplication returns 0 the proxy **MUST** abort the forwarding and it **MAY** notify the sender, for instance by bouncing the message. If this multiplication returns any non-zero value the proxy can proceed with the transformation.

```c
// Implements TransformMessage( eB, k );
// Input:
// eB - the ECDH ephemeral public key decoded from the PKESK
// k - the proxy transformation parameter retrieved from storage

if 0x08 * eB == 0 then abort

eC = k * eB
return eC
```

The proxy **MUST** change the value of a non-null fingerprint in the PKESK to the forwardee’s key fingerprint. The proxy **MUST** change the value of the EC ephemeral public key in the algorithm specific data of the PKESK to the the encoding of eC, using the encoding described in [I-D.ietf-openpgp-crypto-refresh], Section 9.2.1.

7. **Decrypting forwarded messages**

Upon receiving the forwardee key, the forwardee **MAY** re-generate a fresh primary key and attach the received forwardee subkey. This enhances security by preventing the forwardee from storing a signature-capable key of which the forwarder knows the secret material. An implementation **SHOULD** keep the forwardee key separate from the generic keyring, and associated to a specific forwarding instance instead.
Upon receiving a message encrypted to a subkey flagged as 0x40, the implementation **MUST** replace the fingerprint in the ECDH KDF with the fingerprint specified in the subkey KDF parameters.

The implementation **SHOULD** inform the user that the message was originally sent to a different recipient and forwarded to them. If the implementation does so it **MAY** ignore the intended recipient fingerprint signature subpacket, as described in [I-D.ietf-openpgp-crypto-refresh], Section 5.2.3.36.

### 8. Security Considerations

#### 8.1. Collusion between Proxy and Forwardee

It is important to note that any forwardee that colludes with the proxy can recover the forwarder's encryption subkey's secret key material. This allows the colluding parties to decrypt all messages encrypted using that subkey, even ones that weren't forwarded (for example because they were encrypted and received before the forwarding started, or because only a subset of received messages were forwarded).

To minimize this risk, the forwarder may want to generate a key specifically for the duration of the forwarding.

Given that the signing-capable primary key is independently generated, forging signatures is out of scope of this attack.

A complete security analysis can be found in [FORWARDING], Section 4 and a simulation-based security proof in appendix A.

#### 8.2. Key Flags

Suitable subkeys for proxy forwarding are limited to flags 0x04 (encrypt communications) and 0x08 (encrypt storage) as defined in [I-D.ietf-openpgp-crypto-refresh] Section 5.2.3.29 to limit the scope of the attack in case of compromise.

Forwardee encryption subkeys have flags 0x40 and 0x10 only, in order to prevent forwarding-capable implementation from exporting the public key and stop other implementations from encrypting messages directly to this key.

#### 8.3. Proxy transformation factors management

When a forwarding is stopped or revoked, by deleting the stored proxy factor, the proxy ensures that a future compromise does not retroactively endanger older messages.
8.4. Proxy transformation

By checking that $8P$ is not $\emptyset$ and aborting otherwise, where $P$ is the ephemeral public key included in the PKESK before performing the transformation, the proxy ensures no information about the proxy parameter is leaked to an adversary that is able to submit messages and observe the applied transformation.

A proxy SHOULD also perform the multiplication on the elliptic curve with the proxy parameter in constant time. This prevents an adversary from timing the transformation and derive information about the proxy parameter. Alternatively, a proxy MAY decide to pad all the forwarded messages to a constant delay, thus preventing such an attack from an external submitter.

8.5. Message forwarding selection

The criteria to choose which message to forward the messages is left up to the implementation, and may be based on reception time, sender, or any policy that can be determined from the message metadata. Filtering message has a security implication in case of compromise: the messages that were not forwarded may be decrypted by an adversary that can compute the recipient's key.

9. IANA Considerations

The 0x40 value is to be added to the OpenPGP IANA Key Flags Extensions registry, representing "This key may be used for forwarded communication". The flag is defined in Section 4.1.

A new registry "ECDH KDF type" is to be created the OpenPGP IANA registry:

* 0x01: "Native fingerprint KDF"

* 0xFF: "Replaced fingerprint KDF"

10. References

10.1. Normative References


[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/


10.2. Informative References


Appendix A. Test vectors

The following test vectors are independent instances, and do not share the key material.

A.1. Proxy parameter

Recipient secret integer, clamped and big endian, OpenPGP wire format

59 89 21 63 65 05 3d cf 9e 35 a0 4b 2a 1f c1 9b
83 32 84 26 be 6b b7 d0 a2 ae 78 10 5e 2e 31 88

Forwardere secret integer, clamped and big endian, OpenPGP wire format

68 4d a6 22 5b cd 44 d8 80 16 8f c5 be c7 d2 f7
46 21 7f 01 4c 80 19 00 5f 14 4c c1 48 f1 6a 00

Derived proxy parameter, little-endian

e8 97 86 98 7c 3a 3e c7 61 a6 79 bc 37 2c d1 1a
42 5e da 72 bd 52 65 d7 8a d0 f5 f3 2e e6 4f 02

A.2. Message transformation

Proxy parameter, little-endian

83 c5 7c be 64 5a 13 24 77 af 55 d5 02 02 81 30
58 60 20 16 08 e8 1a 1d e4 3f f8 3f 24 5f b3 02
Ephemeral point $P$, 0x40 prefixed, OpenPGP wire format

40 aa ea 7b 3b b9 2f 5f 54 5d 02 3c cb 15 b5 0f 84
ba 1b dd 53 be 7f 5c fa dc fb 01 06 85 9b f7 7e

Transformed point $kP$, 0x40 prefixed, OpenPGP wire format

40 ec 31 bb 93 7d 7e f0 8c 45 1d 51 6b e1 d7 97 61
79 aa 71 71 ee a5 98 37 06 61 d1 15 2b 85 00 5a

A point of order 4 on the twist of Curve25519 to test small subgroup point detection, 0x40 prefixed, OpenPGP wire format

40 ec ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff

A.3. End-to-end tests

Armored recipient key

-----BEGIN PGP PRIVATE KEY BLOCK-----
xVgEZAdtGBYJKwYBBAdHw88AQdAGzrOivCFCxQ6hmpP52fBtbYmqkPM+TF9oB ei
x9QWcnEAAQQAa54PERHLVdQIMo0f03+mJXMTR3Dwq+qi5LTaflQFDGxEdzRNib2Ig
PGJvYkBwcm90b24ubWU+wooEExYIADwFAMQHbRgJkCLL+xMJ+Hy4FiEEe7z6Zb syLV1z0yIsv7Ewn4fLgMChEcGgECQCcCcCFQgCFgACIgEAAAoFAwPoXgScgPr
KQFzui1ltPuHodEaDTtb+/wQ1oAbuSdGQD7B82NJgyEZInC/4Bwuc+yFsGaswW2WgtupW5vZm44FAzHXQRkB20YeqorBcEAEhvAQUBAOdAeU0hI02RUGH67127u
a82Mnv62/GKZMbpNFqgqAcDAqoJAAD/Sd14Xkjfy1l8r0vQ5rM+jBG4Exh2G8XCPZgMz5RLa6gQ4MJ4BBgWCAaqBQJKB20YcZAIy/sTCfh8uYhBJu+81emW7MiSMZ
siLL+xMJ+Hy4AhsmAAAKaqE4Knbj6S6nG24nuXfqkkyTPlFTHwzurjv3+qQxwW6L3RgA/RvY/NcpCizSL3tLLznwSag7/m6Jvy9g6unU2mZ5QoI
=un50
-----END PGP PRIVATE KEY BLOCK-----

Armored forwardee key
 Proxy parameter K

04 b6 57 04 5f c9 c0 75 9c 5f d1 1d 8c a7 5a 2b
1a a1 01 c9 c8 96 49 0b ce c1 00 f9 41 e9 7e 0e

Plaintext

Message for Bob

Encrypted message

Transformed message

Contributors

Daniel Huigens (Proton AG)
Acknowledgments

A heartfelt thank you to Francisco Vial-Prado for the work on designing and proving the forwarding scheme. We also thank Lara Bruseghini, Ilya Chesnokov, and Eduardo Conde Pena for their collaboration and help in applying the scheme to OpenPGP.

Author's Address

Aron Wussler
Proton AG
Switzerland

Email: aron@wussler.it