OpenPGP Email Forwarding Via Diverted ECDH Key Exchanges

A. Wussler

IETF 114
2022-07-29
Outline

- Automatic e-mail forwarding
- Diverting the secret
- OpenPGP implementation
- Threat model
- Conclusions, Q&A
Automatic E-Mail Forwarding

- Proxy re-encryption: Ciphertext transformation so that it can be decrypted by a different party than was originally intended.
- Transformation is carried out without access to decryption secrets, plaintexts, or interactive communication with secret-key holders[5].
- Used in ElGamal-encrypted mailing list[6], proposed for use in redirection[3, 1, 2].
- We propose it for automatic OpenPGP email forwarding using ECDH.
Diverting the secret (I)

**Alice**
Sends a normal OpenPGP message \((P_B, c)\) to Bob encrypted with his public key \(Q_B\)

**Bob**
Generates the forwarding key \(d_C\) and derive the proxy parameter \(k_{BC}\)

**Proxy**
Transforms the ephemeral key exchange \(P_B\) with the proxy parameter \(K_{BC}\)

**Charles**
Decrypts the message \((P_C, c)\) with the key \(d_C\)
Diverting the secret (II)

**Alice**
- Retrieves $Q_B$ out of band
- Secretly picks $d_A$
- Computes $P_B = d_A G$
- $S = d_A Q_B = d_A d_B G$
- Encrypts $c = \text{Enc}_S(m)$

**Bob**
- Long-term key pair $(Q_B, d_B)$
- Generates $d_C$
- Computes $k_{BC} = \frac{d_B}{d_C} \mod n$

**Proxy**
- Verifies $P_B \in \langle G \rangle$
- Computes $P_C = k_{BC} P_B = \left( \frac{d_A d_B}{d_C} \mod n \right) G$

**Charles**
- Computes $S = d_C P_C = d_A d_B G$
- Decrypts $m = \text{Dec}_S(c)$
Forwarding schema

Alice \rightarrow Proxy \rightarrow Bob

Proxy

k_{BC} \rightarrow k_{BD} \rightarrow k_{DF}

Alice \rightarrow Proxy \rightarrow Charles

Proxy \rightarrow Charles

k_{BC} \rightarrow (P_B, c) \rightarrow (P_B, c)

Bob \rightarrow Charles

k_{BC} \rightarrow (P_C, c)

Proxy \rightarrow Daniel

Proxy \rightarrow Daniel

k_{BD} \rightarrow (P_D, c)

Charles \rightarrow Daniel

k_{BC} \rightarrow (P_C, c)

Proxy \rightarrow Frank

Proxy \rightarrow Frank

k_{BD} \rightarrow (P_D, c)

Charles \rightarrow Frank

 Proxy \rightarrow (P_B, c)

Proxy \rightarrow (P_B, c)

Frank \rightarrow Daniel

Proxy \rightarrow Daniel

k_{DF} \rightarrow (P_F, c)

Proxy \rightarrow (P_B, c)

Frank \rightarrow (P_F, c)
OpenPGP implementation

Alice It is transparent to the sender, that might not support the feature and should not know a forwarding happens.

Bob It requires only a one-time set-up from the original recipient, that does not require to be online.

Server The server, has only one curve multiplication to perform in constant time.

Charles Requires a modifications when deriving the symmetric key from the ephemeral secret[7] using a modified version of the KDF for the forwarded recipient.

It has been implemented in OpenPGP.js and GopenPGP.
OpenPGP changes

In the computation of the shared secret the following KDF is used[7]:

\[
MB = \text{Hash}( 00 || 00 || 00 || 01 || ZB || \text{Param} )
\]

• The recipient key fingerprint is used as channel binding information.
• Channel binding information is recommended[4] for security.
• Charles’s implementation needs to know the original recipient fingerprint when decrypting:
  • Adding the fingerprint to the PKESK. This makes the messages distinguishable.
  • Adding the fingerprint to the forwarder key. This ensures the key can only be used for forwarded messages and is accepted once when the forwarding is set up.
Threat model

We assume the original recipient (Bob) is always honest, since his objective is to protect his key and his e-mails.

- If any set of forwarded parties colludes (and is able to submit messages to the proxy) they are still left with an instance of the ECDH.
- If the proxy and any forwarded party collude it is possible to recover Bob’s private key:

\[ d_i k_i = d_i d_i^{-1} d_B = d_B \mod n. \]

This is partially mitigated from OpenPGP’s key usage flags.

A simulation proof is included in the paper.
Conclusions

- The protocol is compatible on the sending side with all ECC-enabled OpenPGP implementations.
- The protocol is non-interactive, Bob can generate all parameters without any further exchange.
- It is possible to deploy this for a single provider for internal forwarding, with a high practical impact.
- The trust is distributed: the secret key is shared between the server and the forwarded party, both can’t recover the key alone.
- We considered the use of Curve25519 because it is practical to implement in constant time, fast, and well-regarded in the community.
References

Full paper can be accessed at: https://www.wussler.it/ECDHForwarding.pdf


Q & A