# Security Considerations for Session Key Reuse in OpenPGP Crypto-Refresh

Falko Strenzke[MTG]

MTG: MTG AG, Germany

Background: Reply to All with Session Key Reuse

The Session-Key-Reuse Mechanism

Avoiding Pitfalls

Interoperability

Requirements for Secure Use of SKR

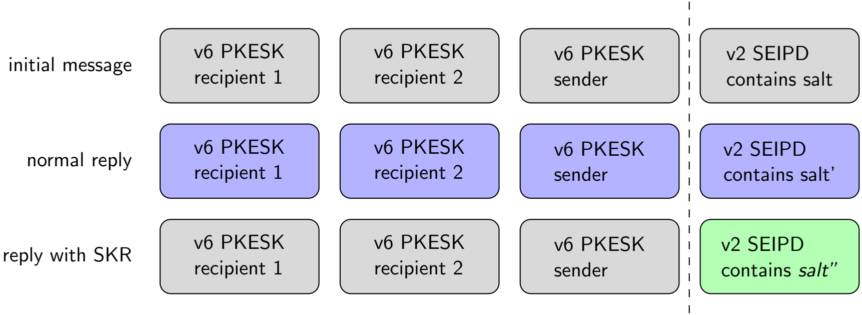
Conclusion

# Background: Reply to All with Session Key Reuse (SKR)

`https://gitlab.com/openpgp-wg/rfc4880bis/-/merge_requests/228`

- Session-Key-Reuse in crypto-refresh
  - previously:
    - new session key for each message encrypted in PKESK
    - encrypt message directly with session key
  - new in v6 PKESK:
    - key derivation of message encryption key from session-key encrypted in v6 PKESK and from per-message salt value
    - key derivation based on HMAC: necessary to avoid CFB downgrade (most likely needed for any of the AE modes!)
    - allows to reuse existing PKESK for reply with different salt value

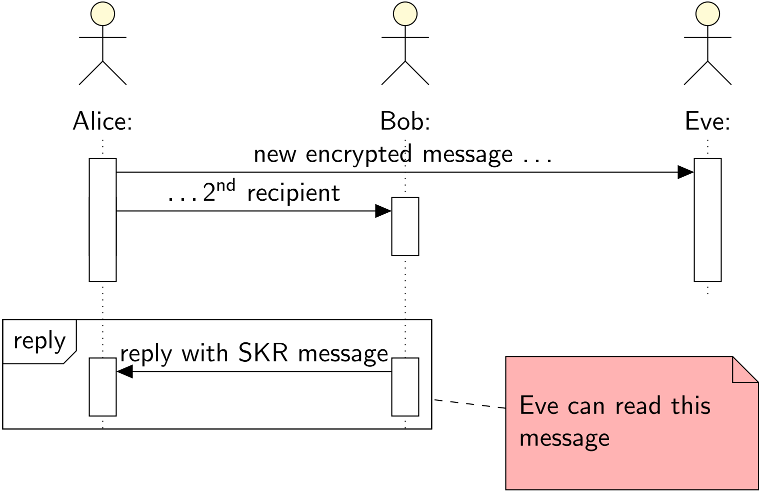
# The Session-Key-Reuse (SKR) Mechanism



| | | | | |
|---|---|---|---|---|
| initial message | v6 PKESK<br>recipient 1 | v6 PKESK<br>recipient 2 | v6 PKESK<br>sender | v2 SEIPD<br>contains salt |
| normal reply | v6 PKESK<br>recipient 1 | v6 PKESK<br>recipient 2 | v6 PKESK<br>sender | v2 SEIPD<br>contains salt’ |
| reply with SKR | v6 PKESK<br>recipient 1 | v6 PKESK<br>recipient 2 | v6 PKESK<br>sender | v2 SEIPD<br>contains *salt*” |

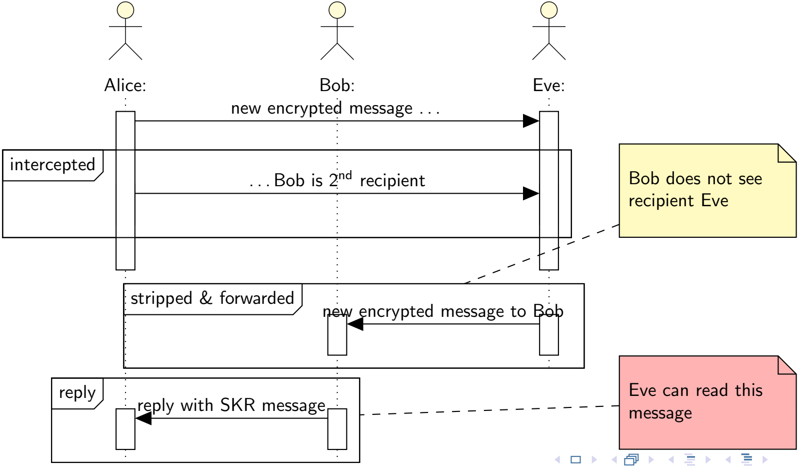
- message-key = HKDF(session-key, salt) // simplified
- new salt for each message

# Pitfall 1: Replying to only a subset of the original recipients



# Pitfall 1a: Attacker removes themselves from recipient list

- like Pitfall 1, but attacker with network / mailbox access removes themselves from recipient list
- → use Intended Recipient Fingerprint subpacket



Alice:

Bob:

Eve:

new encrypted message . . .

intercepted

. . . Bob is 2$^{nd}$ recipient

stripped & forwarded

new encrypted message to Bob

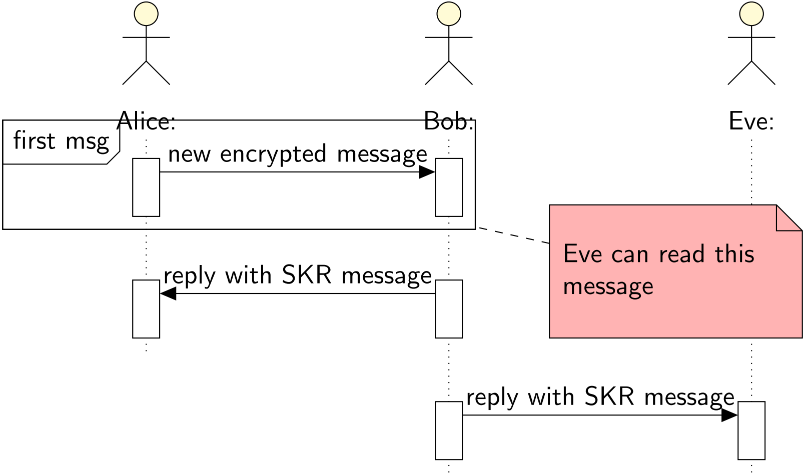
reply

reply with SKR message

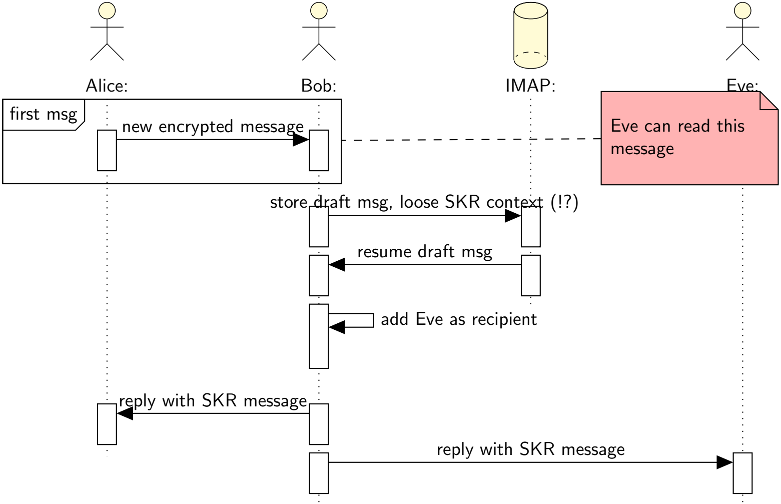
Bob does not see recipient Eve

Eve can read this message

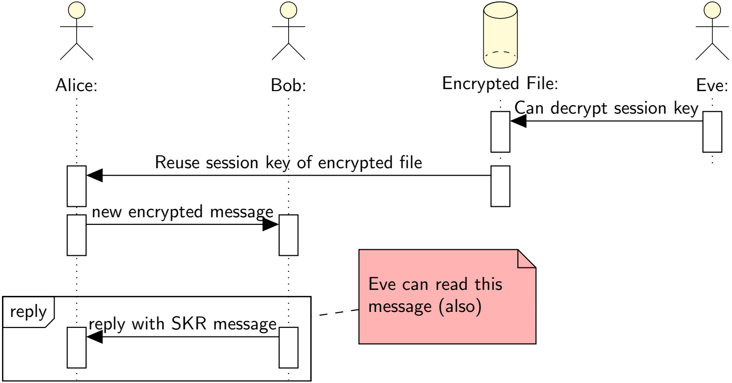
# Pitfall 2: Replying to more than the original recipients



# Pitfall 2a: Save Msg. Then Add more Recipients

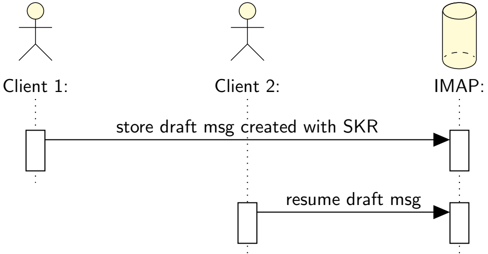


# Pitfall 3: Interfering Session Key Reuse



# Interop: Save Msg. then Open with Other Client

- Possible interoperability problem if user has multiple clients with differing support for SKR
- Non-supporting client sees stored encrypted message to a recipient that it doesn't have public key to. What happens if
  - message is sent unchanged (may work),
  - message is changed (may work),
  - recipient list is changed? (may work, but then Pitfalls 1 & 2 apply![1])



[1] Unsolvable security hole depending on non-supporting client

# Requirements for Secure Use of SKR

Security Considerations:

- signalling of SKR necessary
- user control necessary
- otherwise might be used when user does not expect it:
    - has recipient public key but expires
    - using slightly different e-mail address [2]
- risk of two users being caught in continued session key reuse unknowingly
- in some application context, notion of what is a reply and what a new message might not be clear [2]
- Security considerations strongly suggest to implement SKR only by using application-specific guidance documentation

[2] not explicitly mentioned in security considerations

# Comments?