Limits of key usage for OSCORE

IETF, CoRE WG interim, February 18th, 2021

Problem Overview

- OSCORE uses AEAD algorithms to provide security properties
  - Confidentiality
  - Integrity

- Forgery attack against AEAD algorithms
  - Adversary may break the security properties of the AEAD algorithm
  - See **draft-wood-cfrg-aead-limits-00**

- Need to describe relevant limits for OSCORE
  - How the forgery attack and the limits affect OSCORE
  - Necessary steps to take during message processing
  - What to do if the limits are exceeded

Limits on key usage

- What you need to count
  - ‘q’: the number of messages protected with a specific key, i.e. the number of times the algorithm has been invoked to encrypt data with that key
  - ‘v’: the number of forgery attempts that have been made against a specific key, i.e. the amount of failed decryptions that has been done with the algorithm for that key

- When a peer uses OSCORE
  - The key used to protect outgoing messages is its Sender Key
  - The key used to decrypt and verify incoming messages is its Recipient Key

- Relevant counters for OSCORE
  - Counting number of times Sender Key has been used for encryption (q value)
  - Counting number of times Recipient Key has been used for decryption (v value)

Limits for ‘q’ and ‘v’

- General limits for AES-CCM-16-64-128 See **draft-wood-cfrg-aead-limits-00**

```
q <= sqrt((p * 2^126) / l^2)

v * 2^64 + (2l * (v + q))^2 <= p * 2^128
```

- Depends on assumptions for the p probability value
  - Considering the values p_q = 2^-60 and p_v = 2^-57
  - Same values used in [I-D.ietf-tls-dtls13]
- Exact limits calculated

```
q <= sqrt(((2^-60) * 2^126) / 1024^2)

q <= 2^23
```

```
v * 2^64 + (2*1024 * (v + 2^23))^2 <= 2^-57 * 2^128

v <= 112
```

Open Points (1/2)

- So far only limits for ‘q’ and ‘v’ for AES-CCM-16-64-128 have been calculated
  - Ideally a table can be created showing limits for all AEAD algorithms used by OSCORE
  - Depending on the algorithm appropriate values needed to calculate this should be found

- Method for constrained devices to efficiently count ‘q’ and ‘v’
  - Need to save these values in the event of reboot
  - They should not have to save ‘q’ and ‘v’ per message as it creates a lot of usage for the nonvolatile memory.
  - This can be done as in OSCORE Appendix B.1. where the values are only periodically stored

Open Points (2/2)

- Consider messages that are replays, do they impact the ‘v’ counter if their decryption would have failed?
  - For instance a message may be detected as replay before decryption while it was actually a forgery attempt
  - Consider that DTLS 1.3 does decryption before the replay detection, in OSCORE replay detection happens before decryption
  - Can we safely not increment v for a replayed message?

Summary and next steps

- Document describing AEAD limits impact on OSCORE
  - Introduce counting of ‘q’ and ‘v’ values for OSCORE
  - What actions to take if limits are reached
  - Including current alternatives for rekeying if the limits are reached
- More work needed on
  - Add limits for further AEAD algorithms
  - Improve solution for constrained devices

Thank you!

Comments/questions?