Large Payloads in IKEv2

`draft-tjhai-ikev2-beyond-64k-limit`

CJ Tjhai (Post-Quantum)
Tobias Heider (genua GmbH)
Valery Smyslov (ELVIS-PLUS)

IETF 109

Motivation

- draft-ietf-ipsecme-ikev2-multiple-ke addresses issues of using large keys for Key Exchange methods (common in PQC) in IKEv2
- This draft still limits the size of any single public key to 64K – the maximum size of IKEv2 payload
  - most NIST Third Round Candidate Algorithms fit into this restriction
- However, some national regulators (e.g. BSI) recommends using Classic McElice PQKE which smallest public-key is 255KB (while more conservative parameter sets are even around 1 MB)
- Using post-quantum signatures and post-quantum certificates (draft-ounsworth-pq-composite-sigs) may lead to the situation when AUTH and CERT payloads also grow beyond 64K

Goals

- The goal of the document is to define a way for using some specific data blobs in IKEv2 if they grow beyond 64K
  - public keys for key exchange methods (KE)
  - signatures (AUTH)
  - certificates (CERT)
- The defined mechanism must be backward compatible
- Reliability of transferring large data in IKEv2 should be addressed
- The defined mechanism must be simple and must introduce minimal changes to IKEv2

Not Goal

- There is no goal to define a generic mechanism for IKEv2 which would allow <u>**any**</u> payload be greater than 64K

Proposed Approach

- If amount of data doesn’t fit into a single payload then split data into chunks less than 64K and put them into a sequence of payloads with the same type; receiving side will concatenate data from a sequence of payloads having the same type
  - this approach works well if only one payload of this type can appear in the message according to IKEv2 (true for KE and AUTH, not true for CERT, but can be worked around)
  - if such sequence of payloads appears inside Encrypted payload (true for AUTH, CERT), then the Length field of the Encrypted payload cannot be used, but this doesn’t matter, since the length of Encrypted payload can always be deduced from the length of IKE message

Example

Initiator | Responder

IKE_SA_INIT
HDR, SAi1, KE1i, Ni

$\longrightarrow$
$\longleftarrow$

IKE_SA_INIT
HDR, SAr1, KE1r, Nr, [CERTREQ,]

IKE_INTERMEDIATE
HDR, SK{**KE2i, KE2i, KE2i**}

$\longrightarrow$
$\longleftarrow$

IKE_INTERMEDIATE
HDR, SK{**KE2r, KE2r**}

IKE_AUTH
HDR, SK{IDi,[**CERT,CERT,CERT**,][CERTREQ,]
[IDr,] **AUTH, AUTH**, SAi2, TSi, TSr}

$\longrightarrow$
$\longleftarrow$

IKE_AUTH
HDR, SK{IDr,[**CERT, CERT**,]
AUTH, AUTH, SAi2, TSi, TSr}

Discussion

- The proposed approach is simple and easy to implement
- It doesn’t touch IKE state machine and doesn’t change sequence of exchanges
- It allows amount of data to transfer to be very different in different directions (very important for some KEMs)
- The proposed approach does require some tweaks (like handling some payloads differently than others)
  - that is that…
- IKE messages will grow in size making it difficult to use UDP to transport them
  - it is anticipated that TCP (or some other reliable transport) will often be used in this case

Thanks

- Comments? Questions?
- Is this problem worth to address?
- Is the suggested approach reasonable?
- WG adoption?