New IKEv2 Payload Format?

Valery Smyslov
svan@elvis.ru

IETF 115
Existing Format Limitation

• Payload Length field occupies 2 bytes, so payload size is limited to 64 Kbytes
  – might not be enough for some PQ algorithms
  – no problem with Message size, which is limited to 4 Gbytes
Existing Format Redundancy

Many payloads contain substantial redundancy

- Payload Length field occupies 2 bytes, while most payloads are shorter
- most parameters occupy 2 bytes, while less than 256 values are defined
- zero-filled RESERVED fields

Example: SA Payload on the right contains one Proposal with four Transforms:

- ENCR_AES_CBC (128 bits)
- PRF_HMAC_SHA2_256
- AUTH_HMAC_SHA2_256_128
- 2048-bit MODP Group

Payload size is 48 bytes, among which 24 bytes are zeroes.
Lifting 64 Kbytes Size Limit

- Would allow using PQ algorithms with long public keys and signatures
  - Classic McEliece is NIST round 4 candidate, it is also recommended by some national state organizations (e.g. BSI in Germany)
- Would allow transferring large chunks of data (e.g. in CP payload)
Making Payloads Smaller

• Would decrease power and network bandwidth consumption (important for IoT devices)
• Would decrease chances of IP fragmentation in `IKE_SA_INIT` and chances of IKE fragmentation in the following exchanges
  – these chances grow as the number of transforms proposed by initiator increases making SA payload larger, e.g. when draft-ietf-ipsecme-ikev2-multiple-ke is used with full range of PQ algorithms with different parameters
Existing Proposals

- A Larger Internet Key Exchange version 2 (IKEv2) Payload
  [draft-nir-ipsecme-big-payload](draft-nir-ipsecme-big-payload)
- Beyond 64KB Limit of IKEv2 Payloads
  [draft-tjhai-ikev2-beyond-64k-limit](draft-tjhai-ikev2-beyond-64k-limit)
- Compact Format of IKEv2 Payloads
  [draft-smyslov-ipsecme-ikev2-compact](draft-smyslov-ipsecme-ikev2-compact) (expired)
"A Larger Internet Key Exchange version 2 (IKEv2) Payload"

- addresses only 64Kbytes limitation
- generic solution suitable for any payload
  - payloads in new and old formats can be mixed in a message
- explicitly negotiated via exchange of notifies in `IKE_SA_INIT`
  - cannot be used in initial exchange (`IKE_SA_INIT`)
- relatively easy to implement (depending on base IKEv2 code)
  - no implementations exist (?)
“Beyond 64KB Limit of IKEv2 Payloads”

- addresses only 64Kbytes limitation
- suitable only for some payloads (KE, AUTH, CERT)
  - existing payload format is preserved
  - Encrypted Payload is mangled (zero payload length)
- no explicit negotiation, implicitly negotiated in \texttt{IKE\_SA\_INIT} by selecting transforms with large public keys
  - cannot be used in initial exchange (\texttt{IKE\_SA\_INIT})
- relies on mandatory use of IKE fragmentation
- relatively easy to implement
  - implementations exist
“Compact Format of IKEv2 Payloads”

- addresses redundancy of IKE payloads
- suitable for any payload
  - compact and standard payloads can be mixed in a message
- some payloads have special, extremely compact format
- no negotiation, new initial exchange is used
  (ALT_IKE_SA_INIT instead of IKE_SA_INIT)
  - can be used in new initial exchange (ALT_IKE_SA_INIT)
  - initiator can revert to IKE SA_INIT if this extension is not supported by responder (based on receiving of fatal error or on timeout)
- moderately difficult to implement
  - can be implemented as post-/pre- message processing
  - no implementations exist
Questions

• Do we want to revise IKE payload format?
• If yes, then what problems should be addressed:
  – remove 64K limitation?
  – decrease IKEv2 messages redundancy?
  – both?
• Any interest in this work?
Thanks!
Backup Slides

Possible new payload format that would support large payloads and also would make IKE messages smaller by eliminating some redundancy
New Format Overview

• Three formats for new Generic Payload Header
  – for small payloads (up to 64 bytes)
  – for medium size payloads (up to 8 Kbytes)
  – for large payloads (up to 512 Mbytes)
• No RESERVED fields
• Revise existing payloads headers to reduce their size
  – remove unnecessary fields
• Special Format for some payloads (SA, some status notifies)
# New Generic Payload Header

1. **Small payloads (2 bytes, 6 bits for Payload Length)**

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>C</th>
<th>0</th>
<th>Payload Length</th>
</tr>
</thead>
</table>

2. **Medium size payloads (3 bytes, 13 bits for Payload Length)**

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>C</th>
<th>1</th>
<th>0</th>
<th>Payload Length</th>
</tr>
</thead>
</table>

3. **Large payloads (5 bytes, 29 bits for Payload Length)**

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>C</th>
<th>1</th>
<th>1</th>
<th>Payload Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Payload Length (cont)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Revised Existing Payload Headers

The following payload headers can be revised:

• Key Exchange, Identification, Authentication, Configuration
  – remove RESERVED field

• Notify
  – remove SPI Size field (can be deducted from Protocol ID)

• Delete
  – remove SPI Size field (can be deducted from Protocol ID)
  – remove Num of SPIs field (can be deducted from Payload Length)

• Traffic Selector
  – remove RESERVED field
  – remove Number of TSs field (can be deducted from Payload Length)
Special Format

Special format (*) for:

- **SA Payload**
  - SA Payload grows quickly as more and more new transforms are defined and offered by initiators

- **Notify Payload with some Status Type Notification containing no data**
  - Exchange of such payloads is a common way to negotiate support for various protocol extensions, so initial IKEv2 messages grow up as more and more extensions are defined

Both payloads contain a lot of redundancy and can be effectively compacted.

(*) Inspired by draft-smyslov-ipsecme-ikev2-compact
SA Payload

Outline:

• Remove all RESERVED fields
• Remove Length fields in substructures (where they are unnecessary)
• Encode all currently defined transforms w/o attributes using one octet (both Transform Type and Transform ID)
• Encode currently defined Encryption transforms having Key Length attribute using two octets
• Leave possibility to encode arbitrary (even not yet defined) Transform Type and Transform ID, as with regular format

Example: SA Payload with one Proposal and four Transforms:

• ENCR_AES_CBC (128 bits)
• PRF_HMAC_SHA2_256
• AUTH_HMAC_SHA2_256_128
• 2048-bit MODP Group
Notify Payload

Outline: encode notification in one octet (limited to first 256 status notifications) and omit all other fields from Notify Payload

Example: Notify Payload with IKEV2_FRAGMENTATION_SUPPORTED notification.

Original: 8 bytes

Compact: 2 bytes
Negotiation

If new format is used from the very beginning then the following options exist:

- **New major IKE version (v3)**
  - old responders would return `INVALID_MAJOR_VERSION`

- **New type of initial exchange (e.g. `ALT_IKE_SA_INIT`)**
  - old responders would return `INVALID_SYNTAX`

- **New critical payload in the `IKE_SA_INIT`**,** followed by payloads in new format**
  - old responders would return `UNSUPPORTED_CRITICAL_PAYLOAD`
Discussion

• We don’t need to assign new payload types except for special format payloads (SA and empty status Notify), do we? What about revised payloads?

• Transport issues for transferring large payloads are out of scope
  – IKE over TCP combined with IKE fragmentation (to solve limitation on 64 Kbytes on a single IKE message over TCP)
  – mixed mode (draft-tjhai-ikev2-beyond-64k-limit: IKE over TCP combined with plain ESP or ESP over UDP) can be used to avoid ESP performance degradation of TCP encapsulation

• Certificates consume a lot of space, can be compressed
  – RFC 8879 is an example of certificate compression
Thanks again!