draft-ietf-6tisch-minimal-security

• Authors: Mališa Vučinić (Ed.)
  Jonathan Simon
  Kris Pister
  Michael Richardson
Status

• Published -07 and -08
  • WGLC resolutions, reviews from:
    • Göran Selander
    • Tero Kivinen
    • Xavier Vilajosana
    • Klaus Hartke
    • Jim Schaad
    • Tengfei Chang
    • William Vignat
    • Thomas Watteyne

• Goal of the presentation
  • Discuss WGLC resolutions
• For a full list of WGLC issues resolved in -07 and -08, see:
  https://bitbucket.org/6tisch/draft-ietf-6tisch-minimal-security/issues?content=%7EWGLC
WGLC resolutions 1/8

Stateless-Proxy

https://bitbucket.org/6tisch/draft-ietf-6tisch-minimal-security/issues/29/stateless-proxy

• New work in CoRE: draft-hartke-core-stateless
  • Adopted by CoRE and soon in WGLC. Thanks!
  • Extended the base CoAP token definition to support longer tokens
• Removed the definition of Stateless-Proxy option from minimal-security
• Added Section 8.1 “Statelessness of the JP”
  • RECOMMENDED that the JP operates in a stateless manner
  • Use CoAP token to transport state. If it doesn’t fit => use hartke-core-stateless extended token
  • Informative reference to hartke-core-stateless

New text to accommodate implementations where stateless proxy is not possible:

Note that in some networking stack implementations, a fully (per-pledge) stateless operation of the JP may be challenging from the implementation point of view. In those cases, the JP may operate as a statefull proxy that stores the per-pledge state until the response is received or timed out, but this comes at a price of an additional DoS vector.
WGLC resolutions 2/8

Use of confirmable CoAP message for Join Request

https://bitbucket.org/6tisch/draft-ietf-6tisch-minimal-security/issues/49/use-coap-con-messages-at-pledge

- When proxy makes a request on behalf of a pledge (client), it does not need to use the same message type
- Pledge MUST send a CON Join Request, proxy MUST forward it as NON to avoid maintaining state
- Removes previous minimal-security-specific retransmission mechanism
- Retransmission now handled exclusively by CoAP, same (old) behavior of pledge retransmitting
CoAP already provides a bandwidth cap through its congestion control mechanism

- **NSTART** in number of requests for Confirmable messages
  - (useful for **statefull** JP)
- **PROBING_RATE** in bytes/second for non-confirmable
  - (useful for **stateless** JP)

- No need for *any* new code
- Values in -08 (see next).
WGLC resolutions 3/8
Bandwidth cap at JP in -08 (cont)

Considerations:
- Stateless proxy limited by upstream bandwidth
  - React fast to a potential attack, do not overwhelm the network
- Statefull proxy limited by RAM
  - NSTART=3?
  - 3 pledges concurrently joining

Some numbers:
- Join Request message size ~40 bytes (CoAP + CBOR)
- Bandwidth available upstream of JP: at least ~67 bytes/second
  - ~60 bytes of 802.15.4+6LoWPAN+UDP overhead
  - at minimum 1 cell with preferred parent
  - 101 slots of 10ms in a slotframe

https://bitbucket.org/6tisch/draft-ietf-6tisch-minimal-security/issues/40/specify-the-usage-of-coaps-probing_rate-as
WGLC resolutions 3/8

Bandwidth cap at JP in -08 (cont)

https://bitbucket.org/6tisch/draft-ietf-6tisch-minimal-security/issues/40/specify-the-usage-of-coaps-probing_rate-as

• Considerations:
  • Stateless proxy limited by upstream bandwidth
    • React fast to a potential attack, do not overwhelm the network
  • Statefull proxy limited by RAM
    • NSTART=3 ?
    • 3 pledges concurrently joining

• Some numbers:
  • Join Request message size ~**40 bytes** (CoAP + CBOR)
  • Bandwidth available upstream of JP: at least ~**67 bytes/second**
    • ~60 bytes of 802.15.4+6LoWPAN+UDP overhead
    • at minimum 1 cell with preferred parent
    • 101 slots of 10ms in a slotframe
WGLC resolutions 3/8

Bandwidth cap at JP in -08 (cont)

https://bitbucket.org/6tisch/draft-ietf-6tisch-minimal-security/issues/40/specify-the-usage-of-coaps-probing_rate-as

- Considerations:
  - Stateless proxy limited by upstream bandwidth
    - React fast to a potential attack, do not overwhelm the network
  - Statefull proxy limited by RAM
    - NSTART=3?
    - 3 pledges concurrently joining

- Some numbers:
  - Join Request message size \(~40\) bytes (CoAP + CBOR)
  - Bandwidth available upstream of JP: at least \(~67\) bytes/second
    - \(~60\) bytes of 802.15.4+6LoWPAN+UDP overhead
    - at minimum 1 cell with preferred parent
    - 101 slots of 10ms in a slotframe

17.9% of minimal upstream available bandwidth can and should be higher, but decided by JRC
The WGLC resolutions 4/8 discuss CoJP Error Handling, which is detailed on the IRCS platform at
[https://bitbucket.org/6tisch/draft-ietf-6tisch-minimal-security/issues/41/error-handling-for-cojp-parameters](https://bitbucket.org/6tisch/draft-ietf-6tisch-minimal-security/issues/41/error-handling-for-cojp-parameters). Errors can occur when processing CBOR objects both part of CoAP requests and responses. For requests, transport Error object as part of 4.00 Bad Request response. For responses, add Error object to the next Join_Request:

```plaintext
Join_Request = {
  ? 1 : uint, ; role
  ? 5 : bstr, ; network identifier
  ? 7 : Error, ; response processing error
}
```

New CBOR structure “Error”: transported over OSCORE secure channel

<table>
<thead>
<tr>
<th>Description</th>
<th>Value</th>
<th>Additional info</th>
<th>Additional info type</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invalid Join_Request object</td>
<td>0</td>
<td>None</td>
<td>nil</td>
<td>[[this document]]</td>
</tr>
<tr>
<td>Invalid Configuration object</td>
<td>1</td>
<td>None</td>
<td>nil</td>
<td>[[this document]]</td>
</tr>
<tr>
<td>Invalid parameter</td>
<td>2</td>
<td>Label of the invalid parameter</td>
<td>int</td>
<td>[[this document]]</td>
</tr>
<tr>
<td>Invalid link-layer key</td>
<td>3</td>
<td>Index of the invalid key</td>
<td>uint</td>
<td>[[this document]]</td>
</tr>
<tr>
<td>Significant OSCORE partial IV mismatch</td>
<td>4</td>
<td>Next acceptable OSCORE partial IV</td>
<td>bstr</td>
<td>[[this document]]</td>
</tr>
</tbody>
</table>

Table 4: CoJP error codes.
WGLC resolutions 5/8

AEAD Nonce Reuse on JRC failure

https://bitbucket.org/6tisch/draft-ietf-6tisch-minimal-security/issues/44/nonce-reuse-when-jrc-looses-mutable

- New JRC knows that there has been a failure of the old JRC
- When first Parameter Update is sent by new JRC, AEAD nonce reuse occurs
  - New JRC MUST include dummy payload: at least 40 byte long randomly generated string
- Pledge detects significant mismatch in the received sequence number
  - e.g. Sequence number is 1.
WGLC resolutions 6/8
Add blacklist

https://bitbucket.org/6tisch/draft-ietf-6tisch-minimal-security/issues/39/specify-blacklist-parameter-for-parameter

When the joined node receives this parameter, it MUST silently drop any link-layer frames originating from the indicated pledge identifiers.

- JRC can include a “blacklist” parameter at any time
  - Parameter Update to JP
  - Join Response to pledge
- Contains a list of pledge identifiers
- Useful in case of misconfiguration, can help in case of a DoS attack
WGLC resolutions 7/8

Link_Layer_Key that supports all modes of 802.15.4

https://bitbucket.org/6tisch/draft-ietf-6tisch-minimal-security/issues/43/extend-link_layer_key-cbor-struct-to

New definition:

```plaintext
Link_Layer_Key = {
    key_id : uint,
    ? key_usage : int,
    key_value : bstr,
    ? key_addinfo : bstr,
}
```

- **Key ID Mode 0x00 (Implicit, pairwise)**: key_id parameter MUST be set to 0. key_addinfo parameter MUST be present. key_addinfo parameter MUST be set to the link-layer address(es) of a single peer with whom the key should be used. Depending on the configuration of the network, key_addinfo may carry the peer's long link-layer address (i.e. pledge identifier), short link-layer address, or their concatenation with the long address being encoded first. Which address is carried is determined from the length of the byte string.

- **Key ID Mode 0x01 (Key Index)**: key_id parameter MUST be set to a value different than 0. key_addinfo parameter MUST NOT be present.

- **Key ID Mode 0x02 (4-byte Explicit Key Source)**: key_id parameter MUST be set to a value different than 0. key_addinfo parameter MUST be present. key_addinfo parameter MUST be set to a byte string, exactly 4 bytes long. key_addinfo parameter carries the Key Source parameter used to configure [IEEE802.15.4].

- **Key ID Mode 0x03 (8-byte Explicit Key Source)**: key_id parameter MUST be set to a value different than 0. key_addinfo parameter MUST be present. key_addinfo parameter MUST be set to a byte string, exactly 8 bytes long. key_addinfo parameter carries the Key Source parameter used to configure [IEEE802.15.4].
WGLC resolutions 8/8

Misc

- Update of Security Considerations, uniqueness requirements
- Update of Privacy Considerations
- Encode short address lease time in hours instead of seconds
- PAN ID clarifications
- Mandate unique PSKs
- New appendix on “Lightweight implementation option” that does not require implementations of HKDF and SHA in firmware
- Remove ”network prefix” parameter in order not to go into 6LBR management
- ...

Please see the full list of resolved/open issues:

https://bitbucket.org/6tisch/draft-ietf-6tisch-minimal-security/issues?content=%7EWGLC
Next steps

- Latest IETF103 discussions and proposals for -09.
  - Add "bandwidth cap" parameter that JRC can use to configure JP
    - When not present, stateless JP uses all the available bandwidth
    - In case of an attack, JRC can use the recommended values to limit the affect on the network
    - Can also be useful to switch on/off the join process
  - At IETF102 we agreed on not provisioning the full configuration to 6LBR
    - A remnant in -08: “network identifier” parameter => Remove it.
    - 6LBR joins as any other pledge, but how it gets all other parameters is out of scope
- Publish -09
- Wait one week for additional comments?
  - Ship
- OSCORE is still blocked with one DISCUSS
- EDHOC to key OSCORE for zero-touch?