IEEE P802.15

Wireless Personal Area Networks

|  |  |
| --- | --- |
| Project | IEEE P802.15 Working Group for Wireless Personal Area Networks (WPANs) |
| Title | Privacy primitives |
| Date Submitted | 12th March 2024 |
| Source | Tero Kivinen | E-mail: kivinen@iki.fi |
| Re: | TG4ac draft |
| Abstract | Primitives needed to provide privacy on 802.15.4 |
| Purpose | Create TG4ac draft |
| Notice | This document has been prepared to assist the IEEE P802.15. It is offered as a basis for discussion and is not binding on the contributing individual(s) or organization(s). The material in this document is subject to change in form and content after further study. The contributor(s) reserve(s) the right to add, amend, or withdraw material contained herein. |
| Release | The contributor acknowledges and accepts that this contribution becomes the property of IEEE and may be made publicly available by P802.15. |

1. Identifiers
	1. Network identifier

64-bit random identifier used to identify network. This is always sent encrypted, and it will never change during the lifetime of the network. Different networks will use different network identifiers, and there is one “owner” of the network identifier, i.e., the one who announces the network using that network identifier.

* 1. Network key

128-bit key used to for network announcements. If not specified then network identifier is zero padded to 128-bits and that is used as network key. Do we need crypto agility?

* 1. Device identifier

64-bit random identifier used to identify device. This is always sent encrypted and it will never change during the lifetime of the network. Device can use different device identifiers in different networks. This is never used as source or destination address of the frame, and it is never used when generating nonce for the frame.

1. Primitives
	1. Sending list of addresses (AddressListMessage)

This primitive is used to send list of addresses used by sender of this frame. This can be sent to either unicast or multicast address. The source address of this can be either short private address, or extended private address. If this is sent to multicast address there will not be confirmation of receipt of this message.

Data in the frame is:

* 8-bit Flags (sender identification present, sequence number present, short address list present, extended address list present, confirmation required)
* 64-bit identification of the sender (can be omitted), this might be needed if the network allows devices to get out of sync, and then out of sync devices, can try to decrypt incoming frames to see if they can decrypt them using any key they have, and if successful, then they will be able to see the sender from here.
* 8-bit? Sequence number of the address list (can be omitted). This is used to prevent replays, i.e., if the sequence number is older than latest one seen, the list is ignored. Attackers can’t replay address lists to try to get device using old addresses again.
* 16-bit PAN Id used for short addresses. This field is omitted if short address list is not present.
* 8-bit Number of short addresses. This field is omitted if short address list is not present.
* List of 16-bit short private addresses. This field is omitted if short address list is not present, or if number of short addresses is zero.
* 8-bit Number of extended addresses. This field is omitted if extended address list is not present.
* List of 64-bit extended private addresses.

New list always replaces old one if the field is present. If the short address list is not present, the previous short address list is used. If the short address list is present, but number of addresses is zero, then device is no longer using short addresses. Same extended addresses, meaning you can update short address and extended address list separately.

* 1. Confirmation of receipt of address list (AddressListConfirmMessage)

This frame is always sent in unicast frame to the sender of the AddressListMessage. This shall not be sent if the destination address of the address list was not unicast address.

This is used to confirm the reception of the AddressListMessage. Ack frame cannot be used for replacement of this message.

Data in the frame is:

* 8-bit Flags (sequence number present)
* 8-bit? Sequence number of the address list (can be omitted). Identify the address list message that is confirmed.
	1. Request to get list of addresses (RequestAddressesMessage)

This frame is sent to unicast or multicast address. This message is used when the device does not know the currently used private address for remote device, or where it thinks the list might be out of sync. Can be sent to last known unicast address, or to the multicast address. The source address is typically extended private address.

Data in the frame is:

* 8-bit Flags (sender identification present, recipient identification present, sequence number present)
* 64-bit identification of the sender (can be omitted), this identifies the actual sender.
* 64-bit identification of the receipient (can be omitted), this identifies the actual receiver, i.e., whose addresses the device wants to learn. This shall be present if the destination address is multicast address.
* 8-bit? Sequence number of the address list (can be omitted). Sequence number of the last address list the device has seen.

This message could be sent after or during oprhan scan, i.e., where the device things remote peer has changed address, and device do now know currently used addresses. The recipient of this will reply to that with AddressListMessage.

* 1. Assignment of addresses to remote peer (AssignAddressesMessage)

This message is sent by the owner of the network to assign short addresses to devices. It is usually sent to the unicast address of the intended recipient, but if the network owner thinks remote peer might be out of sync it can also send this to multicast address.

Data in the frame is:

* 8-bit Flags (sender identification present, recipient identification present, sequence number present, confirmation required)
* 64-bit identification of the sender (can be omitted), this identifies the actual sender.
* 64-bit identification of the receipient (can be omitted), this identifies the actual receiver, i.e., to whom the addresses are assigned to. This shall be present if the destination address is multicast address.
* 8-bit? Sequence number of the address list (can be omitted). Sequence number of the address assignment.
* 16-bit PAN Id used for short addresses.
* 8-bit Number of short addresses.
* List of 16-bit short private addresses. This field is omitted if number of short addresses is zero.

If device is assigned zero addresses, then it cannot use any short addresses anymore.

* 1. Confirmation of address assignment (AssignAddressesConfirmMessage)

This frame is always sent in unicast frame to the sender of the AssignAddressesMessage.

This is used to confirm the reception of the AssignAddressesMessage. Ack frame cannot be used for replacement of this message.

Data in the frame is:

* 8-bit Flags (sequence number present)
* 8-bit? Sequence number of the address list (can be omitted). Identify the assigned addresses message that is confirmed.

This sent as an reply to the AssignAddressesMessage to confirm that address assignment was successful.

* 1. Network announcement (AnnouncementMessage)

This frame is sent to the multicast address, can use beacon frames. This is sent in clear, as this is used to find existing networks, and devices wanting to join might not have security context. Source address is extended private address

Data in the frame is:

* 8-bit Flags (3-bit security level of the verifier, do we need crypto agility)
* 64-bit random Announcement Nonce to make message unique.
* Encrypted verified generated by encrypting 64-bit Announcement Nonce || 32-bit sequence number with AES-CCM using the network key and the security level given in flags. The nonce for the encryption is generated using extended private address || Announcement Nonce

Recipent of this message who know the network key (network key can either be random 128-bit key distributed during the network joining, or it can also be just 64-bit network identifier padded with zeros to make it 128-bits), can decrypt and verify the AES-CCM field inside the data, and it can use it to verify that the Announcement Nonce inside matches that of outside, and that sequence number is not old.

If the device does not have security context with the network, it will start IEEE Std 802.15.9 KMP with the sender of this message to create security context, and join the network. This method requires that devices wanting to join the network needs to be configured with the 64-bit network identifier, and the 128-bit network key (if network key is used, if not only the network identifier is needed).

Devices who already have security context with the network, can use this message to find that network is available, and send RequestAddressesMessage to sender in case the source address used in this message was not already known to them.

* 1. Network request (AnnouncementRequestMessage)

This frame is sent to the multicast address to see if there is known network nearby. This is usually sent in clear, as this is used to find existing networks, and device sending this might not have addresses that are recognized by the network anymore. Can also be sent encrypted in case device assumes the network owner recognizes source address, and can find security context based on that. Source address is extended private address

Data in the frame is:

* 8-bit Flags (3-bit security level of the verifier, do we need crypto agility)
* 64-bit random Announcement Nonce to make message unique.
* Encrypted verified generated by encrypting 64-bit Announcement Nonce || 32-bit sequence number with AES-CCM using the network key and the security level given in flags. The nonce for the encryption is generated using extended private address || Announcement Nonce

Processing is same as in the AnnouncementMessage meaning if the recipient can verify the verifier, it can send AddressListMessage to the sender of this message to update the addresses.

* 1. Updating key source (KeySourceUpdateMessage)

This can be sent as unicast or multicast message. If sent as multicast message there shall not be confirmations. If the keysource is using extended addresses or similar, then those need to be updated when the addresses are updated too, thus this message is sent to update the key source.

Data in the frame is:

* 8-bit Flags (sender identification present, old key source present, key id mode, require confirmation)
* 64-bit identification of the sender (can be omitted), this identifies the key owner, i.e., the sender.
* 8/40/72-bit Old key source to be changed. If omitted then the key source from the frame is used. Length depends on the key id mode.
* 8/40/72-bit New key source value for the key source. The key id mode shall be same.

When this frame is received the recipient will update the key source to the new value defined, but will keep the old value also there. When the new key value is first time used, the old value is removed.

* 1. Confirmation of updating key source (KeySourceUpdateConfirmMessage)

This frame is always sent in unicast frame to the sender of the KeySourceUpdateMessage.

This is used to confirm the reception of the KeySourceUpdateMessage. Ack frame cannot be used for replacement of this message.

Data in the frame is:

* 8-bit Flags ()
* 8/40/72-bit Old key source to be changed.

This sent as an reply to the KeySourceUpdateMessage to confirm that key source update was successful. It is using the same keysource than the KeySourceUpdateMessage. The key source to updated will be identified inside the message, even the outer keysource might be different.