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. Definitions, acronyms, and abbreviations
	1. Definitions

**device identifier:** Random 64-bit identifier for the device.

**extended privacy address:** Random 64-bit address in form of AAI-64.

**network identifier:** Random 64-bit identifier to identify the network.

**network key:** Key shared between all devices in the network.

* 1. Acronyms and abbreviations

DI device identifier

1. Optional features

10.9a Privacy enhancements

1. General

This optional feature tries to provide protection against tracking of the devices, i.e., tracking the location of the device as it moves around and using that to find out the movements of the person carrying that device. Because different devices in the same network might have different privacy requirements, and it is up to the device needing strict privacy protection to drive the process to provide or require it. In common use case this is for example the mobile phone carried by the person, thus the device needs protection against tracking, and requires use of addresses that cannot be tracked, i.e., it needs change addresses frequently enough so that tracking of the user is not feasible.

Complete protection against tracking of individual user is impossible, but the enhancements here provide methods to block mass surveillance and tracking of users. Attacker could still for example store the radio fingerprint of the device to be tracked, and use special radios to keep track of users location, but this is not something that can be scaled to keep track of all users.

Fixed installations like light switches and fixtures do not have privacy requirements for themselves, but they need to cope with the device that have, thus they need to implement this enhancement enough to be able to allow other devices connecting to them to have privacy addresses, and they need to be able to understand that device connecting them do not want to keep stable static address for them.

Privacy enhancements require the use of encryption and authentication as defined in section 9 to protect the message exchange between devices. IEEE Std 802.15.9 may be used to generate keys for the security layer. All MAC commands described in this section shall be sent in encrypted and authenticated frames. The IEs described in this section contain separate protection inside and may be also sent in frames which are not encrypted and/or authenticated.

1. Identifiers
	1. Generic

All IDs and addresses used in the privacy enhancement feature use address formats defined in the IEEE Std 802c-2017 section 8.4.4.3, but use the transmission order defined in the section 4.5.1. This enhancement adds two more bits (S, and T) to the format defined in the IEEE 802c.

Figure 1 shows the location of the S, T, X, Y, Z and M bits in the transmission of the address.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| RMO→LMO | Octet 8RMO | Octet 7 | Octet 6 | Octet 5 | Octet 4 | Octet 3 | Octet 2 | Octet 1LMO |
|  | 01 | 89 | 67 | 45 | 23 | 48 | DE | 82 |
| LSB→MSB | 10000000 | 10010001 | 11100110 | 10100010 | 11000100 | 00010010 | 01111011 | 01000001MXYZST |
| Transmission order |

Figure 1—Location of the M/X/Y/Z/S/T fields

M bit has its normal meaning, i.e., it defines whether the address is unicast or multicast address. The X, Y, and Z bits shall be set to one, zero, zero, respectively to indicate an Administratively Assigned Identifier (AAI-64) addresses for all extended privacy addresses. The next two bits S and T identify the type of the address to be used. For the extended address used over the air the S and T bit shall be set to zero which makes sure it does not collide with multicast addresses defined by IETF RFC 2464. For other uses see Table 1.

Table 1—Usage of S and T bits

|  |  |  |  |
| --- | --- | --- | --- |
| **S** | **T** | **Usage** | **Description** |
| 0 | 0 | Extended privacy address | Extended address used in the frames. This is only value used over the air in clear. |
| 0 | 1 | Device identifier (DI) | DI is used to identify the device and is always sent in frames that are encrypted, thus it will not appear over the air in clear. |
| 1 | 0 | Network ID | Network ID is only used internally to identify the network, and it can be used to generate the network key, but is not otherwise sent over the air. |
| 1 | 1 | Reserved | This value may overlap with the IETF RFC 2464 which uses 33-33 for the first two octets. |

* 1. Network ID

Network ID is a random 64-bit ID used to identify network. Privacy enhancements feature described in this section never sends this inside any frames, but next higher layer may need to send this over air while provisioning devices or when device first time joins the network. In such cases when network ID is included in any frame, the frame shall be encrypted. Network ID does not change during the lifetime of the network. Different networks should use different network IDs, and there is one “owner” of the network ID, i.e., the one who announces the network using the network ID.

When the device creates a network, it will generate random network ID and other devices joining the network will need to get this out of band or using next higher layer protocol that is outside the scope of this standard.

In some networks this is not needed and the network key is used to as a substitute for identifying the network.

Network ID is an AAI-64 address as defined in section 10.9a.2.1.

* 1. Network key

Network key is the 128-bit key used to for network announcements. This can either be negotiated using IEEE Std 802.15.9 or distributed out of band, or it can also be 64-bit network ID (as a string) concatenated with 64-bits of zeros to make it 128-bit, in which case anybody who knows the network ID knows the network key, and the security is reduced to 58-bits.

NOTE—The network ID when generating network key uses the representation format described in section 9.3.1.

The format of generating network key from network ID is show in Figure 2.

|  |  |
| --- | --- |
| **Network IDOctets: 8** | **PaddingOctets: 8** |
| LMOOctet 1MSB→LSBxxTSZYXM | 2 | 3 | 4 | 5 | 6 | 7 | RMOOctet 8 | 0x00 00 00 00 00 00 00 00 |

Figure 2—Generation of the network key from network ID

This key is used to authenticate the network announcements. This does not provide source authentication, thus anybody who knows the network key can claim to be the owner of the network.

* 1. Device identifier (DI)

Device identifier (DI) is a random 64-bit ID used to identify device. When this is sent in any frames the frames shall be encrypted. This does not change during the lifetime of the network for the device. Device may use different DIs in different networks. This is never used in source or destination address fields of the MAC header of the frame, and it is never used when generating security nonce for the frame.

DI is an AAI-64 address as defined in section 10.9a.2.1.

* 1. Extended privacy address

To provide privacy of the device the device shall not use its stable extended address, but instead device will generate random extended privacy addresses as needed. The frequency of generation of addresses and how many addresses are used is left to the next higher layer, as different uses cases have different requirements. For example the light switch which is in static location does not really benefit from generating privacy addresses. On the other hand light fixtures might need to use different addresses, not to protect their own privacy but to protect the privacy of the device connecting to them, i.e., when mobile phone connects to light fixture, it wants to use its own randomly generated extended privacy address as a source address, and it wants the light fixture to also have extended privacy address that is not static all the time.

The extended privacy addresses are generated by the device and device can have multiple extended privacy addresses for the same network. A device should use a different set of extended privacy addresses for each network it is connected to. It may use a different set of extended privacy addresses when communicating with different devices in the same network. When it has used one extended privacy address with one peer and then does not use it with that peer anymore i.e., it shall remove it from the list of extended privacy addresses sent to the remote peer, it shall not use that address anymore with that peer.

The probability of generating twice same random extended privacy address is so small, that it can be ignored when generating addresses, so even when devices not allowed to reuse same extended privacy addresses they have once stopped using there is no requirement for remembering all extended privacy addresses used before.

Extended privacy address is an AAI-64 address as defined in section 10.9a.2.1.

* 1. Short addresses

Short address are already locally allocated and are not stable, so using them will provide limited privacy. The privacy enhancements option adds option to have multiple short addresses to identify same device, and to be able to assign set of addresses to devices. The number of addresses assigned and the frequency of their reassignment is left to the next higher layer of the device assigning those addresses.

Each short address is associated with a PAN ID, and this section provides a way to use different PAN IDs associated with different set of addresses as needed. Next higher layer assigning addresses may use only PAN ID at time, and assign addresses from there to each device in the network, and then change this PAN ID by assigning new set of addresses to each device from new PAN ID. Or it may assign different PAN IDs and short address list to different devices. Between pair of devices only one PAN ID is used at same time.

To be able to use short addresses for encrypted frames the receiving device needs to know the extended address of the sender to be used for nonce generation as described in section 9.3.2.1. This means that each device assigning short addresses shall announce at least one extended privacy address. The first extended privacy address in the announcement list is used for nonce generation for all frames using short address as a source address.

* 1. Static extended address

The privacy enhancements option requires that device do not respond to their own static extended address after initial deployment. During the initial deployment the static extended address of the device can be used, but after that the device should not respond to any frames sent that address, also the device should not use its own static extended address as source address.

In some use cases there might be uses where device still wants to use the static addresses, for example the light fixture might respond to its own static extended address to allow new devices to join the network. This does not cause privacy issues as the light fixture in fixed location does not really have privacy requirements for itself, it only needs to do privacy enhancements to be able to cope with other devices which do have privacy requirements.

* 1. Address list sequence number

Address list sent by the device may be associated with the address list sequence number. This number is needed in case the devices use multiple extended privacy addresses at the same time, and it is used to provide order for the address lists received in the Address List command. If only one extended privacy address is used, or if short addresses are used, then the replay prevention mechanism provided by the security layer prevents replay of old Address List command.

In case the device has multiple extended privacy addresses and those are used to send Address List commands, the attacker could take one using extended privacy address 1 and make sure it never reaches the recipient, and then when device moves to use extended privacy address 2 later, but still keeps extended privacy address 1 in use, the attacker can replay the old frame stored before to the device, and as it is not replay (device never received it) the security layer does not drop it, and it is passed to the next higher layer. Next higher layer can then detect this situation by checking the address list sequence number in Address List commands. If the Address List Sequence Number field of the received Address List command is smaller (using sequence number arithmetic as defined in RFC1982) then the Address List command is old, and shall be dropped.

* 1. Data structures
		1. Remote DI table

List of addresses associated with each remote DI is stored in a table containing entries as shown in Table 2. This table is indexed by the network ID and the remote DI, and contains entry for each peer of the device.

Table 2—Elements of the remote DI table

|  |  |  |  |
| --- | --- | --- | --- |
| **Element** | **Type** | **Range** | **Description** |
| Network | IEEE address | An extended IEEE address | Network ID of the network that remote device is part of. This can also be empty, in which case the same DI is assumed to work on all networks. |
| RemoteDevice | IEEE address | An extended IEEE address | DI of the remote device. |
| ExtendedPrivacyAddresses | List of extended privacy addresses | — | List of extended privacy addresses used by that device. The first extended privacy address in this list is used as primary extended privacy address when short addresses are used. |
| PanId | Device PAN ID | 0x0000-0xffff | PAN ID associated with the short addresses. If ShortAddresses element is empty then this is ignored. |
| ShortAddresses | List of short addresses | — | List of short addresses used by the device. |

If short addresses are used then table entry also contains the PAN id and list of short addresses associated with that PAN ID. The short addresses are always associated with one primary extended privacy address which is used when generating nonce for those short addresses.

When new update for the extended privacy addresses or short addresses is received then the table for that peer (identified by the DI and the network ID) is updated. In addition that, the *secDeviceDescriptor* table described in 9.5.8 is updated so that any entry matching the extended privacy addresses or short addresses that were removed from the DI table are also removed there, and all new extended privacy addresses or short addresses are added.

The *secKeyDeviceFrameCounterDescriptor* table may be searched for any matching extended privacy addresses, and those addresses may be removed.

* + 1. Local addresses table

List of addresses used the be local device for each network and for each remote DI is stored in the table containing entries as shown in Table 3. This table is indexed by the Network ID and optionally remote DI.

Table 3—Elements of the local addresses table

|  |  |  |  |
| --- | --- | --- | --- |
| **Element** | **Type** | **Range** | **Description** |
| Network | IEEE address | An extended IEEE address | Network ID of the network that local device is part of. This can also be empty in case the device uses same DI for all networks. |
| DeviceIdentifier | IEEE address | An extended IEEE address | DI of the local device in this network |
| RemoteDevice | IEEE address | An extended IEEE address | DI of the remote device. This can be empty in case the device uses same DI for all remote devices in network. |
| ExtendedPrivacyAddresses | List of extended privacy addresses | — | List of extended privacy addresses used by local device for this network and remote device (unless they are empty). The first extended privacy address in this list is used as primary extended privacy address when short addresses are used. |
| PanId | Device PAN ID | 0x0000-0xffff | PAN ID associated with the short addresses. If ShortAddresses element is empty then this is ignored. |
| ShortAddresses | List of short addresses | — | List of short addresses used by the device. |

* + 1. Networks table

List of networks known by the device is stored in the table containing entries as shown in Table 4. This table is indexed by the Network ID, but when trying to verify encrypted verifier of the Net Announcement or Net Request IEs all entries in this table are considered.

Table 4—Elements of the network table

|  |  |  |  |
| --- | --- | --- | --- |
| **Element** | **Type** | **Range** | **Description** |
| Network | IEEE address | An extended IEEE address | Network ID of the network. |
| NetworkKey | Set of octets | — | Key used by this network. |
| SequenceNumber | Integer | 0x00000000-0xffffffff | Latest sequence number from the announcement packet received. |

1. Primitives
	1. Sending list of addresses
		1. General

Sending list of address used by this device is done using Address List command.

This can be sent to either unicast or multicast address. The source address of this can either be short address, or extended privacy address.

The Address List command contains list of short address or list of extended privacy addresses or both. The recipient will replace the address list it has with the list received in the frame. If frame only contains one type of address list then only that list shall be replaced, and other list shall be left intact. To clear list of addresses the number of addresses is sent as zero.

There are three main uses for this command as described in following sections.

* + 1. Sending address list with confirmation

When the device wants to send updated list of addresses to the another device it can send Address List command by issuing MLME-PRIV-ADDR-LIST primitive. The MLME-PRIV-ADDR-LIST.request sends the list to the other device, and returns a MLME-PRIV-ADDR-LIST.confirm immediately after the Address List command has been transmitted.

When the device receives the Address List command it will issue MLME-PRIV-ADDR-LIST.indication primitive to the next higher layer. The next higher layer may then send a confirmation message back by issuing MLME-PRIV-ADDR-LIST.response primitive.

When the MLME-PRIV-ADDR-LIST.response is issued the MAC shall send Address List Confirm command. When the device receives that frame it will issue MLME-PRIV-ADDR-LIST-CONFIRM.indication to indicate that the confirmation has been received.

In this use case the source and destination address of the frames shall be unicast addresses of the devices. Sender ID is normally not needed as the recipient should know the sender from the source address, but it can be included in case the sender is not sure that the recipient knows its current privacy address.

Message sequence chart to send address list with confirmation is shown in Figure 3.

In case the recipient does not recognize the sender from the source address and the confirmation is required it shall send confirmation with error code Unknown Source Address, and then sending device should resend the Address List command, but include the Sender ID field to indicate sender.

Figure 3—Sending address list with confirmation

Message sequence chart of the error case where the recipient does not recognize the senders address is shown in Figure 4.

Figure 4—Sending address list and recovery after other end returns error

* + 1. Sending address list without confirmation

When device does not require confirmation from the other peer it can send the address list by using MLME-PRIV-ADDR-LIST.request and set the ConfirmationRequired parameter to FALSE.

Message sequence chart to send address list without confirmation is shown in Figure 1.

Figure 5—Sending address list without confirmation

* + 1. Sending address list to multicast address

If the device wants to announce its new address to group of devices in the network it can send the Address List command to the multicast address. In that case the ConfirmationRequired shall be set to FALSE.

Message sequence chart to send address list to multicast address is shown in Figure 6.

Figure 6—Sending address list to multicast address

* 1. Confirmation of receipt of address list

When the Address List command is received and MLME-PRIV-ADDR-LIST.indication primitive is called with ConfirmationRequested set to TRUE, the next higher layer may call MLME-PRIV-ADDR-LIST.response to request the MAC to send Address List Confirm command. This frame is always sent in unicast frame to the sender of the Address List command. 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 Address List command.

1. Request to get list of addresses

If the device does not know or suspects it is out of sync with the device it has communicated before it may request a remote device to send its list of addresses by sending Request Addresses command. This command is sent to unicast or multicast address. If the device knows some address used by the remote device, it may send the command to that addresses, or if it does not know any addresses it may send command to multicast address, in which case it shall include remote peers DI in Recipient ID field. The source address is typically extended privacy address.

Typical exchange is shown in Figure 7.

Figure 7—Requesting address list from remote peer

This command may be sent after or during orphan 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 Address List command.

* 1. Assignment of addresses to remote peer

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 recipient (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 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

This frame is always sent in unicast frame to the sender of the Assign Addresses command.

This is used to confirm the reception of the Assign Addresses command. 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 Assign Addresses command to confirm that address assignment was successful.

* 1. Network announcement (Net Announcement IE)

This IE is sent to the multicast address, can use beacon frames. Frame 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 privacy address, but this IE contains encrypted and authenticated verifier.

Data in the frame is:

* 8-bit Flags (3-bit security level of the verifier, algorithm)
* 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 privacy address || Announcement Nonce

Recipient 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 ID 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 ID, and the 128-bit network key (if network key is used, if not only the network ID is needed).

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

* 1. Network request (Net Request IE)

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 privacy address

Data in the frame is:

* 8-bit Flags (3-bit security level of the verifier, algorithm)
* 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 privacy address || Announcement Nonce

Processing is same as in the Net Announcement IE meaning if the recipient can verify the verifier, it can send Address List command to the sender of this message to update the addresses.

* 1. Updating key source

This can be sent as unicast or multicast message. If sent as multicast message there shall not be confirmations. If the key source 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.

XXX when the key source is changed, privacy addresses must be changed immediately after, i.e., update key source, update address list, then take new address list AND new key source in use, in which case old key source is forgotten. This is required as the extended privacy address used to generate nonce + frame counter (random) needs to be unique.

* 1. Confirmation of updating key source

This frame is always sent in unicast frame to the sender of the Key Id Update command.

This is used to confirm the reception of the Key Id Update command. 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 Key Id Update command to confirm that key source update was successful. It is using the same key source than the Key Id Update command. The key source to updated will be identified inside the message, even the outer key source might be different.