IEEE P802.15

Wireless Personal Area Networks

|  |  |  |
| --- | --- | --- |
| Project | IEEE P802.15 Working Group for Wireless Personal Area Networks (WPANs) | |
| Title | List of issues to be solved by the privacy group | |
| Date Submitted | 14th May 2023 | |
| Source | Tero Kivinen | E-mail: kivinen@iki.fi |
| Re: | May SG Privacy meeting | |
| Abstract | Provide list of issues that SG privacy group needs to address | |
| Purpose | Start working on the 802.15.4 privacy | |
| 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. List of issues
   1. MAC addresses in the frames
      1. Generic

The frames cannot use stable extended address, they need to use extended addresses that are not permanent.

Use AAI-64 as defined in the 802c-2017, setting M/X/Y/Z bits as x/1/0/0 and leaving rest of 60 bits for privacy addressing. Most likely we will need to reserve some bits to identify type of privacy address, i.e., temporary address of device, stable network identifier, stable device identifier, or we might also use M/X/Y/Z bits for those also.

* + 1. Short addresses

There is a need to have shorter privacy addresses than 64-bit, so reusing 16-bit addresses is needed. The mechanisms for changing addresses needs to be different, as the coordinator (or coordinated network function) allocates and assigns the short addresses, so that there are no overlapping addresses.

Even if you have overlapping 16-bit addresses, that does not affect security, but it could cause wrong device receiving message for other device if network wide keys are used.

For 64-bit addresses there is no need for coordination, as if random addresses are used the collision probability is negligible.

* + 1. 32/48-bit addresses?

Do we need something between 16 and 64 bit addresses?

* + 1. PAN ID

If short addresses are used then we also need a way to use temporary PAN ID, as short addresses are associated to it.

* + 1. Only one addresses

Lots of devices only have one configurable extended address or short address recognized by the hardware, thus they can only receive broadcast frames and frames having their extended address as destination address. It is also possible that they do not recognize multicast addresses. On the other hand they can turn radio to promiscuous mode to see all traffic if needed.

For privacy it would be better to have multiple address for each device. Perhaps we need a have capability information telling whether the device supports that.

For example the ranging might use temporary address for each ranging round.

We might also have location specific addresses for example for one shop to get discounts etc.

* + 1. Do not use static addresses

This means devices should not use their static address in clear in any of the messages they are sending out, not even when they are joining a network, resuming from the sleep, or when they are restarted.

* + 1. Do not respond to static address

Devices needs to be able to not respond to messages sent to their stable extended address. Even if someone directly sends message using the stable extended address of the device (i.e., probing attack) device needs to have option not to respond to them.

* 1. Sending updates to the peers when address changes
     1. Generic

We need a method to send a message from on device to another telling that the first device is changing its address, and tell the new address the device is going to be using.

If we use method where we generate addresses using key hierarchy or something then everybody can generate address and find them out even if the node is sleeping or not in range. If we allow sender to generate addresses using their own methods, then we need to send them to recipient and if recipient hasn’t received notification of the addresses, he might not be able to recognize the sender.

Sender could send multiple next addresses, so that even if the device misses some updates it might still be able to sync up by sending request of getting next n addresses.

We also need a method of doing cold boot, i.e., when the crypto context is expired, and device needs to connect to each other without knowing each others privacy addresses, or crypto contexts. We do assume they still have some kind of long lived secret that they can use to authenticate each other.

* + 1. Identification of the address change message

We can use either the stable extended address of the device to identify the device doing the change, or we can use the some kind of authenticated identifier to identify the device. Using the previous temporary address will work in some cases, but if there is device who has missed previous update of the address then they do not know who is sending this message, so some kind of more stable identifier is needed.

* + 1. Update security context

The security context of the 802.15.4 is tied to the extended address of the peer, the when the device receives a message saying one of its peers is changing addresses it needs to update the security context for that peer.

The secDeviceDescriptors are keyed by the extended address, if the extended address changes then all the peers connected to that device needs to know the new temporary address. The information stored in the security context includes the frame counters for replay protection (i.e., what was the last frame counter the other peer used when it sent message to us), thus there is security implications of this change. Using secFrameCounterPerKey feature of the 802.15.4 we will automatically reset the frame counters associated with that device to zero, as there will be new device using the same key, i.e. we will add new entry for the new device, and most likely mark the old entry as something that can be deleted after the address change is finished.

There is not security issues as the nonce used will be different when addresses changes, as the nonce contains the temporary address for the user.

We do not need to reset the frame counter to zero, we could start it from the random value, and we could increment it with random value every time we use.

There is mapping from short to extended address to calculate the nonce. So if there is multiple extended addresses tied to same device using some short address, there needs to be one that is marked as primary and that is used for nonce generation. Some devices can of course loop through all extended addresses in list and try all of them.

* + 1. Send update to peers before going to sleep

We need to send updates to peers before we go to sleep, so when we wake up we will be using new mac address, and there is no way of correlating the old mac address used before going to sleep with new one.

* + 1. Group keys

We need a mechanisms that allows changing address tied to the group keys too, i.e., keys used for beacons or device discovery packets. As some devices might be sleeping when we change addresses, we have to have mechanism where the device needs to find the group key owner again even if the address is changed.

* + 1. KeySource

There is 8-octet KeySource field in the frames which can be tied to the extended address of the owner of the key. We need to use temporary KeySource not tied to the extended address of any device.

* + 1. Changing KeySources

We need a mechanism to update the KeySource too. This needs to be done at the same time when the MAC address of the owner of the key is changed. Note, that when MAC address and KeySource are both changed the attackers listening the traffic at that time, can quite easily correlate the old and new addresses and KeySources.

* 1. Beacons and network discovery

We need a method where new device can find the network to join, even if the device creating the network is using temporary address. Beacons used should not include information that can allow fingerprinting devices (if possible).

In this case we can take the random number + identification of the network we want to join, and send a joining message to broadcast (or multicast) address, and then the network owner, can recognize this and respond to it.

One issue is that 802.15.4 devices do not have hash functions built in to them, thus we most likely would need to use some encryption based systems for this like using the identification of the network as a key, and encrypt random number with it and sending it out along with the random number.

It would be better to allocate special multicast group address for these messages, so other devices in the network do not wake up when those messages are sent.

* 1. Network identification

The device joining network needs a method of identifying the network it is joining to even when the addresses used are random. We might need some form of encrypted network identification where the devices which do not know the network identification key can’t decode which network this.

This could be used to solve issue where there multiple gateways from same vendors and lots of clients are connecting to them and the clients need to know which of the gateways to connect to.

The format could be 64-bit extended address style identification, but we want to make sure nobody uses their actual extended address as network id as the extended addresses are guessable. We might want to make sure those network identifications have a format that does not collide with the actual extended addresses, or with the randomized changing addresses.

We could define use format similar to AAI-64 as defined in the 802c-2017, and setting M/X/Y/Z bits as x/1/1/0 and leaving rest of 60 bits for actual network identification. This network id is never used as source or destination addresses, so it is not really an address, but we want to make sure it cannot be confused as actual address, so we want to use format that is different than any of the actual addresses.

* 1. Sequence numbers

Sequence numbers, DSN, BSN, and EBSN has some privacy concerns, so we will need to reset them to random number after you change addresses. Those could also be suppressed if we are not using Imm-Ack.

* 1. Static identifiers in the KMP

The Key Management Protocol KMP selected should be something that does not use static identifiers sent in clear during the negotiation process (i.e., no static Diffie-Hellman keys).

So when selecting KMP the implementation needs to make sure it selected KMP that has properties that protect the identities as otherwise doing privacy on the 802.15.4 layer is pointless when the KMP leaks identities.

1. Protocols
   1. Protocols for extended addresses
      1. Sending list of addresses

There needs to be a way to send list of addresses to be used to peers.

Device can send list of source addresses it plans to use in the future.

Can they use multiple addresses from the list, or do they only use only one? Most likely just one.

Recipients will store this list and know that if they see any of those addresses, this is same device than who sent this list. Replace the previous list. Can send multiple addresses, limited to max size of the frame, and recipient assumes all of them are something that you can use now or in the future as your address. Always sent encrypted.

Note, that everybody who can see this message will know the addresses, thus can track the device, or claim to be that device.

* + 1. Send multicast message trying to join network

Limit the amount of data we have here, so devices cannot be fingerprinted. Always sent in clear, as there is no crypto context. This can use network id as a key and encrypt random nonce using it and sending both the random nonce, and the encrypted nonce in multicast message, so recipient can verify whether it is intended recipient.

These messages can be replayed, how to solve that issue? One solution is to have network allowing joining to send messages advertising himself, and include random nonce to be used in its message, so when the device joining sends it message it uses that nonce and network id to provide proof that it knows the network id, thus is someone who should be allowed to join.

* + 1. Reply to request to join network

Limit the amount of data we have here, so devices cannot be fingerprinted. We can use similar method than in the request to authenticate the network to the peer asking for it. Always sent in clear as there is no crypto context. After this the peer can start running IEEE Std 802.15.9 over the randomized addresses to create crypto context.

* + 1. Fast reconnect

When device has lost the address of the peer, but still has crypto context, he could send multicast message to the another group encrypted with keys it already has, and using its current random addresses. The remote peer could then try to decrypt this message with all the keys it has and if any of them work, it can then know who the peer is, update the address, and send message that updates the random addresses for him.

Note, this is most likely something that is only implemented in the more powerful device, like your mobile phone.

So when device is trying to reach some peer, but transaction expires, it could send this multicast message to try to see if the peer has changed its address, and this device has just missed that update.

This can only be repeated once because this protected message having frame counter, meaning every attack has to have newer frame counter to get response.

* 1. Protocols for short addresses
     1. Assign list of short addresses to device

For short addresses the coordinator (or the network function) will need a method of assigning new address(es) to the device when it wants to change addresses in the network. Recipients will store this list, and will send separate reply message confirming of the recipient of the list and specify which addresses they are using. The new list replaces the previous list. Recipient device can use any of the addresses in the list (or only the first one if it wants). This also includes the addresses to be used for the coordinator. Always sent encrypted.

Note, that everybody who can see this message will know the addresses, thus can track the device, or claim to be that device.

* + 1. Send list of address to peers

Device can send list of source addresses it plans to use in the future. Recipients will store this list and know that if they see any of those addresses, this is same device than who sent this list. Replace the previous list. Can send multiple addresses, limited to max size of the frame, and recipient assumes all of them are something that you can use now or in the future as your address. Always sent encrypted.

Note, that everybody who can see this message will know the addresses, thus can track the device, or claim to be that device.

* + 1. Changing PAN ID

If network is using short addresses, the coordinator needs a way of changing PAN ID also. This should most likely be incorporated in to where it assigns the short addresses to devices. Most likely in a way, that it can include new PAN ID in that message, and after it has sent all updates to the all devices, the devices will switch to new PAN ID after first time seeing it over the air. They are not allowed to send using the new PAN ID and short address associated to it, before coordinator sends first frame using that PAN ID, during that time they still use the old PAN ID and short address. Always sent encrypted.

* + 1. Orphan scan

If the device is sleeping when coordinator changes addresses, or PAN ID, the device might end up in situation where it does not know the PAN ID, and does not have valid short address. In that case there is needed a method where the device can search for the networks in area, and try to find its own network. This assumes it still has security context with the coordinator, so it can decrypt the frames sent by the coordinator. We might want to do orphan scan using extended addresses.

* + 1. Get list of recipient addresses

When device wakes up, how does he know who has changed addresses, and which of addresses assigned to them they are using. This mean they might also be sleeping so long that none of the addresses are in use anymore. Most likely we need protocol from coordinator to return new list.