Network Security

Professor Gaurav S. Kasbekar
Department of Electrical Engineering
Indian Institute of Technology, Bombay

Week - 08

Lecture - 49

Wireless Cellular Network Security : Part 7

Hello, recall that we are discussing the security of LTE. In the previous lectures, we discussed the authentication and key agreement phase, which is performed between the MME and the UE. Next, we discussed the key hierarchy. There is an elaborate key hierarchy in LTE. There is a master key, K_{ASME}, which we discussed in the previous lecture.

From this key, several keys used for encryption and message integrity are derived. So, we'll discuss this key hierarchy in this lecture. All cryptographic keys required for various security mechanisms are derived from the intermediate key, K_{ASME}. We discussed that K_{ASME} is derived as part of the authentication and key agreement phase. So, K_{ASME} is computed by the Home Subscriber Server (HSS), and it is also computed independently by the cell phone.

- All cryptographic keys that are needed for various security mechanisms are derived from the intermediate key K_{ASME}
  - can be viewed as a 'local master key' for the subscriber, in contrast to the permanent master key K

This K_{ASME} is then transferred from the HSS to the MME on the network side. So, at the end of the authentication and key agreement phase, the MME and the cell phone have this shared key K_{ASME} between them. This key can be viewed as a local master key for the subscriber, in contrast to the permanent master key, K. So, K is only present in the SIM card and in the authentication center, and it's a permanent key. In contrast, K_{ASME} is derived every time the authentication and key agreement process takes place.

On the network side, the intermediate local master key K_{ASME} is stored in the MME, that is, Mobility Management Entity, and the permanent master key K is stored in the authentication center. So, what are the advantages of using an intermediate key, that is, K_{ASME}? It enables cryptographic key separation and this implies that each key is usable in one specific situation or context only. For example, there is a separate key for encryption

of the data messages that are exchanged between the UE and the eNodeB. There is a separate key for integrity protection of the data messages between the UE and the eNodeB.

There is a separate key for encryption of the messages that are sent between the UE and the MME and so on and so forth. So, for every context, there is a separate key So, this intermediate key enables cryptographic key separation, which implies that each key is usable in only one specific situation or context, such as encryption or message integrity of a particular flow. Knowing a key that is used in one context does not help in trying to guess what key is being used in another context. For example, if we know the key that is used for encryption of the messages between the MME and the UE, then that does not help us in deducing the key that is used for integrity protection of the messages between the UE and the eNodeB, which is another different context.

An intermediate key also improves the system in terms of providing key freshness. That is, it is possible to more often renew the keys using security mechanisms, for example, in ciphering. So, we have discussed that it is good security practice to renew the keys frequently so that if an attacker is trying to guess the key by collecting a lot of ciphertext, then they have to start all over again when the keys are renewed. So, we do not have to run the EPS authentication and key agreement process. Each time we want to renew the keys used for protecting the radio interface.

So, we do not have to involve the home network to obtain fresh keys. Recall that when the authentication and key agreement process is run, then the MME contacts the home network to obtain the authentication vectors. But because of the intermediate key, these fresh keys can be just derived from the intermediate key. We don't have to run the authentication and key agreement process all over again. But there is also a disadvantage of using intermediate keys.

So, the disadvantage is there is added complexity. There are more types of keys in the system and all these different types of keys need to be computed, stored, protected, kept in sync, and so on. So, this intermediate key increases the complexity of the system in this respect. So, we have a security versus complexity trade-off. Use of an intermediate key results in higher security, but also it increases the complexity.

For EPS, the security benefits of using an intermediate key outweigh the added complexity. Hence, it was decided to use an intermediate key in the case of EPS, whereas the reverse was true at the design phase of 3G security, where the knowledge of hardware and software, etc., was not as advanced as during the development of EPS. Hence, in the case of EPS,

the decision was to use an intermediate key, whereas in the case of 3G security, no intermediate key was used. And the advantage was lower complexity for 3G security. After the idea of using the intermediate key that is K_{ASME} was introduced in the design of EPS security, it was natural to take a further step.

- After the idea of using the intermediate key K_{ASME} was introduced in the design of EPS security, it was natural to take a further step:
  - another intermediate key K_{eNB} was added that is stored in the eNB

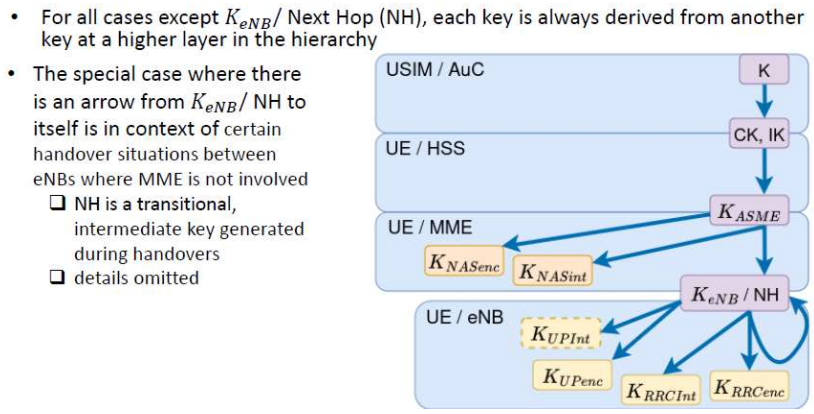
So, another intermediate key K_{eNB} was added, which is stored at the eNodeB. And from this intermediate key K_{eNB}, the keys that are used for encryption and message integrity of the messages exchanged between the UE and the eNodeB are derived. So, there is one intermediate key K_{ASME}, and from that, another intermediate key K_{eNB} is derived. We'll see that from K_{eNB}, further keys are derived, which are used for ciphering and message integrity. The addition of K_{eNB} makes it possible to renew keys for the protection of radio access without involving the MME. So, the radio access is between the UE and the eNodeB.

From the K_{eNB}, keys can be derived and they can be used for protection of radio access, that is encryption and message integrity. So, one does not need to involve the MME in this process. This figure shows the entire key hierarchy of EPS. This is the permanent key, that is, K, which is only present in the AUC and in the SIM card. From this K, the ciphering key (CK) and the integrity key (IK) are derived.

So, this is just as in the UMTS authentication vector generation. We discussed this in the previous lecture. So, CK and IK are derived, and after that, CK and IK along with some other parameters are used to derive the intermediate key K_{ASME}. And from K_{ASME}, another intermediate key is derived, that is K_{eNB}. We'll discuss NH later. So, from K_{ASME}, which is present in the MME and in the UE, two further keys are derived: K_{NASenc} and K_{NASint}.

This K_{NASenc} is used for encryption of the data that is exchanged between the UE and the MME. This K_{NASint} is used for integrity protection of the messages exchanged between the UE and the MME. From K_{eNB}, other keys are derived. These are K_{UPInt}, K_{UPenc}, K_{RRCInt}, and K_{RRCenc}. K_{RRCInt} and K_{RRCenc} are used for message integrity and encryption of the control messages or signaling messages that are exchanged between the UE and the eNodeB.

And K_{UPInt} is used for integrity protection of the data messages that are exchanged between the UE and the eNodeB and K_{UPenc} is used for encryption of the data messages that are exchanged between the UE and the eNodeB. Let's discuss this key hierarchy in detail. So, in this key hierarchy, the UMTS key hierarchy is a small subset of this and it just consists of the keys K, CK and IK which are shown at the top of the figure. An arrow between two keys means one key is derived from the other, specifically the key to which the arrow points is derived from the other key. For example, K_{ASME} is derived from CK and IK because the arrow points from CK-IK to K_{ASME}, and so on.



In all cases, additional input parameters are required for the derivation. None of the additional parameters is assumed to be secret information. For example, in the derivation of K_{eNB}, another parameter which is a counter parameter that is used for deriving K_{eNB}. And we discussed that SN id, that is serving network id, is one of the other inputs used in deriving the K_{ASME}. For all cases except K_{eNB} /Next Hop (NH), each key is derived from another key at a higher layer in the hierarchy.

So, what is this next hop key? In the case of next hop, we see that there is an arrow from next hop to itself. So, this special case where there is an arrow from K_{eNB} / next hop (NH) to itself is in the context of certain handover situations between eNodeBs where the MME is not involved. When the UE moves from the range of one eNodeB to the range of another eNodeB and both the eNodeBs are being controlled by the same MME, in that case, a handover is performed from the first eNodeB to the second eNodeB and the K_{eNB} / NH is used in this context. NH is a transitional intermediate key generated during handovers and we omit the details for brevity.

So, except for this case, we see that there is always an arrow from a higher level in the hierarchy to a lower level in the hierarchy. An important property of key derivation is that

starting from keys in the lower layers of the key hierarchy, it is impossible in practice to compute keys in the higher layers. So, for example, starting from K_{NASenc}, it is not computationally feasible to compute K_{ASME}, and from K_{eNB} it is not computationally feasible to compute K_{ASME}, and so on. So, these functions used to generate a key from the key above it in the hierarchy, that is like a cryptographic hash function. So, it has a one-way property, we cannot compute the key higher up in the hierarchy from a key that is lower in the hierarchy.

The topmost key derivation from K to CK and IK happens on the user side inside the USIM and on the network side inside the authentication center as we have discussed during our discussion of authentication and key agreement. K_{ASME} is independently derived in the UE and the MME from CK, IK, and other inputs as we discussed earlier. K_{eNB} is derived from K_{ASME} and an additional input which is a counter parameter. So, each time K_{eNB} is derived, the counter parameter increments so a different K_{eNB} is derived each time from the K_{ASME}. This additional parameter is required to ensure that each new K_{eNB} derived from K_{ASME} differs from the ones derived earlier.

The purpose of K_{eNB} is to be a local master key in an eNodeB. So, an eNodeB can securely communicate with the UEs without having to involve the MME using the K_{eNB}, which is the local master key at the eNodeB. K_{NASenc} is a key used to encrypt NAS signaling traffic, and K_{NASint} is a key used to protect the integrity of NAS signaling traffic. So, these keys are shown over here: this one and this one. So, these keys are derived from K_{ASME} and two additional parameters.

K_{RRCenc} is a key used to encrypt the RRC signaling traffic, that is the control traffic that flows between the UE and the eNodeB. Similarly, K_{RRCInt} is a key used to protect the message integrity of RRC signaling traffic. These keys are derived from the K_{eNB} and two additional parameters. This key, K_{UPenc}, is a key used to encrypt user plane traffic, which is exchanged between the UE and the eNodeB. It is derived from K_{eNB} and two additional parameters.

So, these additional parameters are not secret. K_{UPenc}, that is this key—in fact, this is K_{UPInt}, that is this key. K_{UPInt} is a key used to protect the integrity of a certain type of user plane traffic. So, we will not get into the details of this for brevity. So, one purpose of the complex key hierarchy is to provide key separation.

That is each key is used in a single unique context for cryptographic protection of either user traffic or signaling traffic. Also notice that all the keys used for such protection are

leaves in the hierarchy. So, it is infeasible to derive a key used in one protection context from another key or set of keys used in other contexts. So, if we go back to this picture, then we see that all the keys used for encryption and message integrity are leaves in this tree. So, this key hierarchy is a tree and these are all leaves in the tree.

K_{NASenc}, K_{NASint}, K_{UPInt}, K_{UPenc}, and all these keys are leaves. We already seen that from a particular key, it is not computationally feasible to compute a key at a higher level in the hierarchy. So, from all these leaves, we cannot compute another key. So, the intention is that attackers cannot find out any keys used in one context from keys used in any other context. However, note that cryptographic key separation does not help if there is a leakage of some higher layer keys.

For example, because someone has been able to get access to the keys stored in MME. So, for example, if someone steals the key K_{ASME}, then they are able to derive all the keys derived from it, for example, K_{eNB} and K_{NASint}, so the one key is for encryption and the other is used for message integrity. So, K_{NASenc} and K_{NASint}. An intruder will be able to derive these keys if they steal K_{ASME} and they'll also be able to derive K_{eNB} and the keys derived from K_{eNB} if they steal K_{ASME}. Hence, cryptographic key separation does not help if there is a leakage of higher layer keys. For example, if someone has been able to get access to keys stored in the MME.

Another benefit of the complex key hierarchy is that keys can be renewed without affecting all other keys. When one key is changed, only the keys that are dependent on the changed key have to be changed and the other keys can remain the same. For example, K_{eNB} can be re-derived without changing K_{ASME} in the process. K_{eNB} is below K_{ASME} in the tree hierarchy. So, if K_{eNB} is changed, there is no need to change K_{ASME} as well.

If K_{eNB} is changed, then all the keys derived from K_{eNB}, for example K_{RRCenc} and K_{RRCInt}, these will have to be changed as well. So, whenever a key is changed, then all the keys below it in the hierarchy, they have to be changed. So, what is the reason renewing keys is useful? Recall that it is good cryptographic practice to periodically change keys. An attacker who is trying to guess the key needs to start all over again when the key is changed.

So, an attacker may be collecting a lot of ciphertext in order to guess the keys. So, if the keys are changed, then the intruder has to start all over again collecting ciphertext. So, periodically changing the keys helps in defending against such attacks. Another reason follows from a generic security principle. We should minimize the need to distribute the same secret to many entities.

For example, in the case of K_{eNB}, it is renewed whenever it is derived for a new eNodeB. So, this prevents two eNodeBs from using the same key. So, earlier, suppose a particular K_{eNB} is being used with one base station, then a handover happens, and K_{eNB} needs to be derived for another base station. In that case, a fresh K_{eNB} is derived, so this prevents two eNodeBs from using the same key. Next, we discuss protection for signaling and user data.

We have discussed the key hierarchy in the case of EPS. Now, we discuss how signaling and user data are protected using these keys. One process that needs to be completed is security algorithm negotiation. That is, which algorithms will be used for encryption and which algorithms will be used for message integrity. Before the communication can be protected, the UE and the network need to agree on what security algorithms to use.

EPS supports multiple algorithms and includes two mandatory sets of security algorithms. So, 128-EEA1 and 128-EIA1 are based on the stream cipher SNOW 3G. And 128-EEA2 and 128-EIA2 are based on the Advanced Encryption Standard (AES). All implementations of UEs, eNodeBs, and MMEs need to support these two sets of ciphers. A third set of security algorithms is optional for implementation.

- ❑ 128-EEA1 and 128-EIA1 based on the stream cipher SNOW 3G, and 128-EEA2 and 128-EIA2 based on Advanced Encryption Standard (AES), which all implementations of UEs, eNBs and MMEs need to support
- A third set of security algorithms is optional for implementation
  - ❑ 128-EEA3 and 128-EIA3 based on the stream cipher ZUC

128-EEA3 and 128-EIA3 are based on the stream cipher ZUC, and these are optional for implementation. EPS can be extended to support more algorithms in the future. So, the advantage of this flexibility, as we have discussed before, is that if some of the algorithms are compromised, they can be replaced with other algorithms that are secure. Hence, EPS, like many other security systems, allows a choice of many algorithms to be used for integrity protection and encryption. Algorithms are negotiated separately between the UE and eNodeBs.

So, this is the AS level or access stratum level. Algorithms are also negotiated separately between the UE and the core network, that is, the MME. This is the non-access stratum or NAS level. The network selects the algorithms based on the UE security capabilities and the configured list of allowed security algorithms for the network entities, specifically eNodeBs and MMEs. The UE provides its security capabilities to the network during the attachment procedure.

Then, the network repeats these security capabilities in a message sent to the UE. This message is integrity-protected. So, the security capabilities that the UE provided to the network are repeated in an integrity-protected response message from the network. What is the purpose of this? This is to detect any tampering in the security capabilities that the UE provides to the network.

So, this protects against bidding down attacks where the attacker modifies the message carrying the UE security capabilities from the UE to the network. For example, the attacker may remove the strong algorithms from these security capabilities and just retain the weak algorithms. Because of this, the network may be led into selecting a weak algorithm. If the UE detects a mismatch between the security capabilities it sent to the network and once that is received from the network, then the UE cancels the attached procedure. So in this way, one can ensure that there is no tampering of the security capabilities that are sent from the UE to the network.

The MME is responsible for selecting the NAS level algorithms and the eNodeB is responsible for selecting the AS level algorithms including the user plane algorithm. Recall that NAS is the traffic exchange between the UE and the MME and AS is the traffic exchange between the UE and the eNodeB. Hence, naturally, MME is responsible for selecting the NAS level algorithms and the eNodeB is responsible for selecting the AS level algorithms. So, in each case, the network entity at one end of the connection that is responsible for selecting the algorithms. The operator configures the MMEs with a list of allowed algorithms for NAS signaling in priority order.

So, there is one list for the integrity algorithms and another list for the ciphering algorithms. During the security setup, the MME chooses one NAS ciphering and one NAS integrity algorithm based on the configured lists and signals the decision to the UE. There are pre-configured lists for integrity algorithms and for ciphering algorithms. There is one list for integrity algorithms and one for ciphering algorithms and during the security setup the MME chooses one NAS ciphering and one NS integrity algorithm from these lists. Similar to the MME configuration, each eNodeB is also configured with a list of allowed algorithms in priority order.

Again, there is one list for integrity protection algorithms and another list for ciphering algorithms. Thus, the eNodeB decides which algorithms are used with the UE for AS signaling protection and for AS user plane data protection. At the eNodeB, there is one list for integrity protection and one list for ciphering, and the UE sends security capabilities to

the eNodeB, which has a list of the algorithms it supports. So, the eNodeB selects one algorithm for integrity protection and one for ciphering. So, these algorithms are selected from the intersection of the security capabilities that the UE provided to the eNodeB and the lists that are present in the eNodeB.

So, generally, the most secure algorithm for integrity protection and the most secure algorithm for ciphering are used from these choices. The MME sends K_{eNB} key to eNodeB from which the actual protection keys are derived. So, we have seen that K_{eNB} is another intermediate key, which is derived from the K_{ASME}, and this key is sent to the eNodeB, and from this key, actual protection keys are derived. We now discuss NAS signaling, AS signaling, and user data protection. NAS and AS signaling messages are encrypted and provided with message integrity and replay protection.

Recall that NAS messages are the messages exchanged between the UE and the MME, and AS messages are exchanged between the UE and the eNodeB. So, these are encrypted and provided with message integrity and replay protection. User plane data is encrypted, but no message integrity or replay protection is provided. So, if message integrity and replay protection are required, then they need to be provided by the application. The relevant keys from the key hierarchy discussed earlier are used for these purposes.

For brevity, we'll omit the details. They are similar to our discussions of previous security systems. So, this concludes our discussion of EPS security. We discussed the EPS authentication and key agreement process. Then, we discussed the key hierarchy in the case of EPS.

We discussed that there is an elaborate key hierarchy with K_{ASME} as an intermediate key and K_{eNB}, which is derived from K_{ASME}. And several keys are derived from K_{ASME} and K_{eNB} for integrity protection and encryption of traffic that flows in the AS as well as in the NAS. So, we then discussed how security negotiation is done and how algorithms are selected for encryption and message integrity. So, this concludes our discussion of 4G security. In the next lecture, we'll discuss 5G security.

Thank you.