Network Security

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

Week - 08
Lecture - 45

Wireless Cellular Network Security : Part 3

Hello, recall that in the previous lecture, we discussed the security of GSM, and we saw that GSM security has several shortcomings. Today, we'll discuss the security of 3G, in particular UMTS security, and we'll see how it overcomes the shortcomings of GSM security. So, we now discuss features built into UMTS to address the shortcomings in GSM security. So, unlike GSM, signaling messages in UMTS are individually integrity-protected. This helps in detecting any modification to them.

So, signaling messages, such as commands, can have their integrity protected in UMTS. So, one example where this helps is the following. The false base station attack is possible in GSM but not in UMTS. So, an attacker cannot, for example, spoof a cipher mode message instructing the cell phone to suppress encryption. So, we discussed that in the case of GSM, an attacker can spoof a cipher mode message that instructs the cell phone to suppress encryption.

So, the cell phone sends messages in plaintext form, and these can be easily read by the intruder. But because the cipher mode message is protected with message integrity in the case of UMTS, such an attack is not possible in UMTS. The question is: why was integrity protection not provided in GSM? So, in the 1990s, when GSM was designed, false base station attacks were considered too expensive and impractical. But since then, the increased availability of hardware and the falling cost of hardware have made such attacks feasible.

So, hence, it became necessary to defend against cipher mode attacks. So, hence, UMTS incorporated message integrity into commands. So, it can defend against such false base station attacks and similar attacks. Another improvement in UMTS is the following: In GSM, recall that there is no provision for the cell phone to authenticate the network.

So, authentication is one way. The network authenticates the cell phone, but the cell phone does not authenticate the network. In contrast, UMTS supports mutual authentication. That is, the network authenticates the cell phone as before, but the cell phone also authenticates the network. So, as part of the authentication protocol, the SIM card and the network agree on an encryption key and a key for integrity protection of messages.

And the encryption key is used to encrypt the data messages that are sent subsequent to the authentication process, and the integrity key is used for providing message integrity of messages that are sent subsequent to the authentication process. Another improvement in UMTS is that sequence numbers are used and nonces are also used to help prevent replay attacks. So, recall that a replay attack is where an intruder captures an old packet and plays it again to lead the victim into believing that it's a fresh packet. We have seen before how sequence numbers and nonces can help in defending against replay attacks, so in UMTS these mechanisms, sequence numbers and nonces are used to defend against replay attacks. In UMTS, data and signaling messages are encrypted so integrative protection as well as encryption are based on KASUMI which is a 128-bit block cipher Recall that in GSM we discussed that COMP-128 is used.

So, unlike COMP-128 used in GSM, KASUMI has withstood public scrutiny for several years. So, we discussed that COMP-128 was designed in secret by a small group of people, and later on, vulnerabilities were discovered in COMP-128. But in contrast, KASUMI was made available to the public, and it was open for scrutiny. But despite this, no weaknesses have been discovered in KASUMI. So, that makes UMTS security much more secure than GSM security because UMTS security is based on KASUMI, which has withstood public scrutiny for several years.

So, another feature in UMTS is that messages on all wireless links are encrypted, not just the link between the cell phone and the base station. We discussed that in the case of GSM, only the messages between the cell phone and the base station are encrypted, but often the link between the base station and the Base Station Controller (BSC), is also a wireless link, particularly a microwave link. So, messages are sent in the clear, that is, in plain text form, on these links connecting the base station to the BSC. In contrast, in UMTS, messages on all wireless links are encrypted. So, if the link from the base station to the BSC is a wireless link, then messages are encrypted on that link as well.

Also, algorithms for encryption and integrity protection can be negotiated between the SIM and the network. So, if weaknesses are found in one particular encryption algorithm or integrity protection algorithm, then those algorithms can be replaced with stronger ones.

UMTS also addresses network domain security. So, signaling and other data between nodes in the provider domain are protected. So, how are these messages protected?

A variant of IPsec is proposed to secure messages in the wired network, which connects the MSCs, HLRs, etc., to each other. So, we have discussed earlier how IPsec provides security at the network layer. So, a variant of IPsec is used to secure messages in the wired network, which connects the MSCs, HLRs, and other parts of the core network. Thus, we see that the wireless links are protected in UMTS, as well as the wired links. Hence, the security is stronger than in the case of GSM.

Keeping in mind that migration to 3G would be slow and uneven, the UMTS security architecture was carefully designed to maximize compatibility with GSM. So, we'll see that, just like GSM, UMTS also consists of two phases. One is the authentication phase, and the other is the encryption and integrity phase. The authentication handshake is very similar to that in GSM. So, we first discussed the authentication and key agreement phase in UMTS.

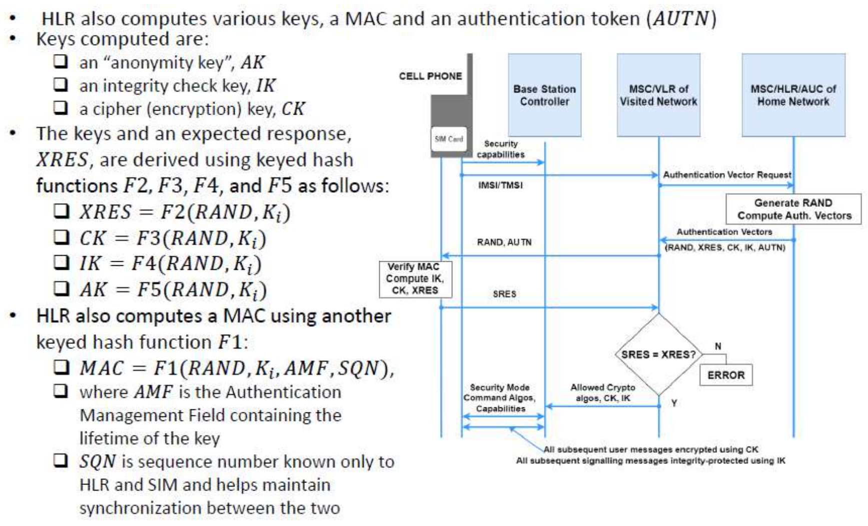
Later on, we'll discuss how encryption and message integrity are done. So, we start with authentication and key agreement. So, this figure shows the process used for authentication and key agreement. This is the cell phone, and it communicates with the base station, which is connected to the BSC or base station controller. And this is the MSC/VLR of the visited network, and this is the MSC/HLR/AUC of the home network.

So, we can see that the process has many similarities with the GSM authentication and key agreement phase. So, let's discuss this in detail. So, step one is the authorization request from the cell phone. That is, these two messages where the cell phone sends a request for being authenticated by the network. So, one observation is that the cell phone sends its security capabilities to the network in this message, and it also sends its IMSI or TMSI to identify itself.

So, we discussed last time how IMSI and TMSI can be used to identify the SIM card. Then, the next step is the creation and transmission of authentication vectors. We discussed that authentication vectors are also used in GSM. So, they are also used in UMTS, but there are some differences between the authentication vectors in UMTS and in GSM. So, let's discuss those differences.

The HLR for the home network generates a random number RAND, which functions as a challenge in a challenge-response authentication protocol. So, we saw that in GSM also, there is a challenge RAND. So, in UMTS also, there is a challenge RAND. The HLR also computes various keys, a message authentication code, and an authentication token called AUTN. So, the keys that are computed by the HLR are as follows:

One is an anonymity key, or AK. We'll see the function of the anonymity key later. Then another is an integrity check key, or IK, that is used for protecting the message integrity of messages that are sent after the authentication process. And the third key is a cipher key or encryption key, CK. It is used for encrypting messages that are sent after the authentication exchange.



- HLR also computes various keys, a MAC and an authentication token ($AUTN$)
- Keys computed are:
  - ❑ an "anonymity key", AK
  - ❑ an integrity check key, IK
  - ❑ a cipher (encryption) key, CK
- The keys and an expected response, $XRES$, are derived using keyed hash functions $F2$, $F3$, $F4$, and $F5$ as follows:
  - ❑ $XRES = F2(RAND, K_i)$
  - ❑ $CK = F3(RAND, K_i)$
  - ❑ $IK = F4(RAND, K_i)$
  - ❑ $AK = F5(RAND, K_i)$
- HLR also computes a MAC using another keyed hash function $F1$:
  - ❑ $MAC = F1(RAND, K_i, AMF, SQN)$,
  - ❑ where AMF is the Authentication Management Field containing the lifetime of the key
  - ❑ SQN is sequence number known only to HLR and SIM and helps maintain synchronization between the two

CELL PHONE
SIM Card
Base Station Controller
MSC/VLR of Visited Network
MSC/HLR/AUC of Home Network
Security capabilities
IMSI/TMSI
Authentication Vector Request
Generate RAND Compute Auth. Vectors
Authentication Vectors (RAND, XRES, CK, IK, AUTN)
RAND, AUTN
Verify MAC Compute IK, CK, XRES
SRES
SRES = XRES?
N
ERROR
Y
Security Mode Command Algos, Capabilities
Allowed Crypto algos, CK, IK
All subsequent user messages encrypted using CK
All subsequent signalling messages integrity-protected using IK

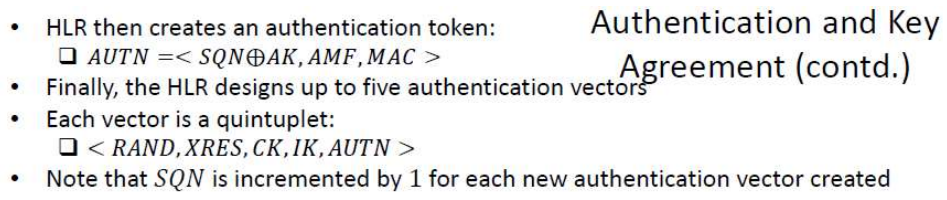
The keys and an expected response, or XRES, are derived using keyed hash functions F2, F3, F4, and F5, which are as follows: XRES = F2(RAND, K_i), where K_i has the same meaning as in GSM. It is a secret which is only stored in the SIM card and in the MSC/HLR. So, XRES is the expected response. XRES is the correct response from the SIM card.

We'll see later that the SIM card responds with SRES which is the assigned response and the authentication of the cell phone to the network is successful only if SRES equals XRES. So, that check is performed in this step. CK = F3(RAND, K_i). So, CK is also a function of RAND and K_i and IK = F4(RAND, K_i) and AK = F5(RAND, K_i) where F2, F3, F4, F5 are keyed hash functions. The HLR also computes a message authentication code using another keyed hash function, F1.

So, the MAC = F1(RAND, K_i, AMF, SQN). The AMF is the Authentication Management Field, which contains the lifetime of the key. So, we will not discuss the AMF in detail. But SQN is used to defend against replay attacks. It is a sequence number.

It is a sequence number known only to the HLR and SIM card, and it helps maintain synchronization between the SIM and the HLR. So, every time authentication takes place, the sequence number is incremented. So, an attacker cannot take a MAC from an old message and replay it in place of the MAC in the current message because the sequence number changes from message to message. So, later on, we will see how the SIM card checks the sequence number in the message. It ensures that the sequence number is not an old sequence number that was used.

Next, the HLR creates an authentication token, or AUTN. So, AUTN= $\langle SQN \oplus AK, AMF, MAC \rangle$. So, this SQN is XORed with the anonymity key, which was a function of RAND and K_i. This XORing is done to protect the SQN from being sniffed; that is, an intruder cannot find out the sequence number because it is XORed with the anonymity key. So, since the anonymity key is a secret, the sequence number is protected against eavesdropping.



Authentication and Key Agreement (contd.)

- HLR then creates an authentication token:
  - ❑ $AUTN = < SQN \oplus AK, AMF, MAC >$
- Finally, the HLR designs up to five authentication vectors
- Each vector is a quintuplet:
  - ❑ $< RAND, XRES, CK, IK, AUTN >$
- Note that SQN is incremented by 1 for each new authentication vector created

Then, the AMF is part of the AUTN and the message authentication code, which was previously computed. So, that is also part of the AUTN. Finally, the HLR generates up to five authentication vectors, just as in GSM. But there are differences between the authentication vectors in GSM and the authentication vectors in UMTS. In the case of UMTS, each vector is a quintuplet.

That is, it consists of five fields. So, the vector is this: <RAND, XRES, CK, IK, AUTN>. Now, we have seen that in UMTS, authentication is mutual. That is, the SIM card authenticates itself to the network, and the network also authenticates itself to the SIM card. This MAC is used to authenticate the network to the SIM card, as we'll see.

So, the SIM card verifies whether the MAC is correct or not. If it is correct, then the network has successfully authenticated itself to the SIM card. Note that the SQN is incremented by one for each new authentication vector that is created. So, this is used to defend against replay attacks. One cannot take an old authentication vector and use it again.

Also, the RAND for each authentication vector is chosen anew. Authentication vectors are forwarded to the MSC/VLR of the visited network. That is shown here. These authentication vectors are sent from the MSC/HLR to the MSC/VLR of the visited network. So, subsequently, this MSC/VLR can use these authentication vectors for five authentication processes between the network and the SIM card.

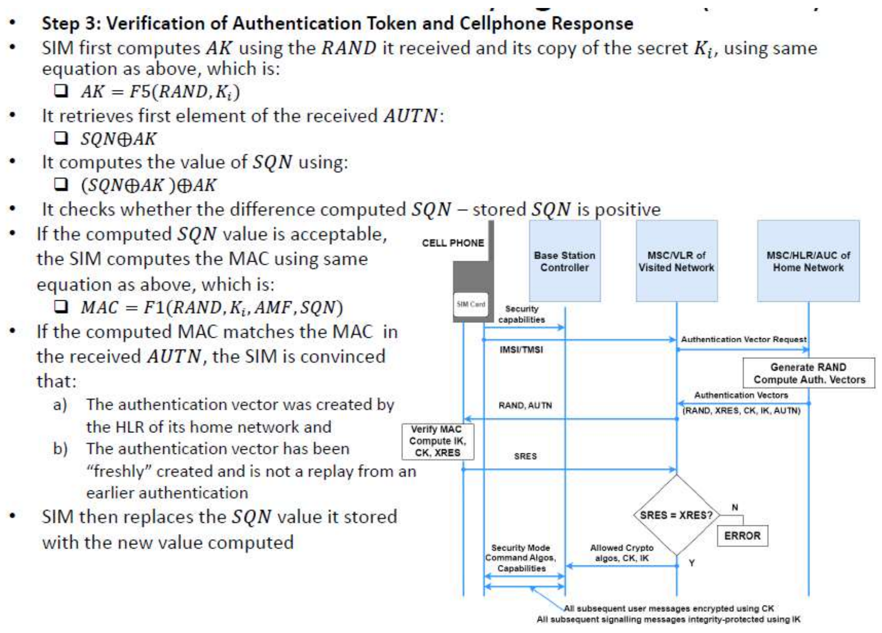
So, again the reason is the same as in GSM. The MSC/HLR sends five authentication vectors to the MSC/VLR so that the MSC/VLR does not have to contact the MSC/HLR again and again. Each time it has to perform an authentication. The MSC/VLR can perform five authentications without having to contact the MSC/HLR again. An authentication vector is used exactly once for a single authentication between the SIM and the MSC/VLR.

In particular, the MAC in the authentication vector is used to authenticate the network to the SIM, and the SIM is supposed to compute SRES, and it should match XRES for the SIM to correctly authenticate itself to the network. The remaining authentication vectors are used by the MSC/VLR in future without needing to involve the home network of the cell phone. So, this helps in reducing the overhead. The MSC/VLR does not have to frequently contact the MSC/HLR to obtain authentication vectors. The MSC/VLR then dispatches RAND and AUTN of the first authentication vector to the BSC which forwards it to the SIM.

So, that is shown by this arrow. The MSC/VLR sends the RAND and AUTN to the BSC, which forwards it to the SIM card. So, at this point, the values RAND and AUTN have reached the cell phone, in particular the SIM. Then, the next step is verification of the authentication token and the cell phone response. So, in this step, the SIM card authenticates the network; that is, the SIM card checks whether the network is legitimate or not.

The SIM first computes the anonymity key AK using the RAND that it received and its copy of the secret K_i, using the same equation as the MSC/HLR had used, and that equation is this. Anonymity key, AK = F5(RAND, K_i). So, we discussed earlier that the same equation is used by the MSC/HLR to compute the anonymity key. Then, the SIM card retrieves the first element of the received AUTN. So, the first element of AUTN, which we discussed earlier, is $SQN \oplus AK$.

Now, from this, the SIM card has to obtain SQN. So, the SIM card computes the value of SQN using this. This first element of the received AUTN is XORed with the anonymity key, which was computed by the SIM in this step. So, $(SQN \oplus AK) \oplus AK = SQN$. So, thus, the SIM card has obtained the value of SQN.



- **Step 3: Verification of Authentication Token and Cellphone Response**
- SIM first computes AK using the $RAND$ it received and its copy of the secret K_i, using same equation as above, which is:
  - ❑ $AK = F5(RAND, K_i)$
- It retrieves first element of the received $AUTN$:
  - ❑ $SQN \oplus AK$
- It computes the value of SQN using:
  - ❑ $(SQN \oplus AK) \oplus AK$
- It checks whether the difference computed SQN − stored SQN is positive
- If the computed SQN value is acceptable, the SIM computes the MAC using same equation as above, which is:
  - ❑ $MAC = F1(RAND, K_i, AMF, SQN)$
- If the computed MAC matches the MAC in the received $AUTN$, the SIM is convinced that:
  - a) The authentication vector was created by the HLR of its home network and
  - b) The authentication vector has been "freshly" created and is not a replay from an earlier authentication
- SIM then replaces the SQN value it stored with the new value computed

CELL PHONE
SIM Card
Base Station Controller
MSC/VLR of Visited Network
MSC/HLR/AUC of Home Network
Security capabilities
IMSI/TMSI
Authentication Vector Request
Generate RAND Compute Auth. Vectors
Authentication Vectors (RAND, XRES, CK, IK, AUTN)
RAND, AUTN
Verify MAC Compute IK, CK, XRES
SRES
SRES = XRES?
N
ERROR
Y
Security Mode Command Algos, Capabilities
Allowed Crypto algos, CK, IK
All subsequent user messages encrypted using CK
All subsequent signalling messages integrity-protected using IK

So, notice that XORing the SQN with the anonymity key helped in defending against eavesdropping. So, hence the SQN was not sent directly, but it was XORed with AK. Next, the SIM card checks whether the difference computed SQN-stored SQN is positive or not. So, stored SQN is the previous sequence number that was used in the previous authentication process. So, each time we have seen that the sequence number increments, hence the computed SQN in this step minus the stored SQN, which is the SQN of the previous step, which is the SQN of the previous authentication process, this should be positive.

So, the SIM card checks whether this difference is positive. If it is negative or it is zero, then the SIM card aborts the authentication process because it means that the network is not legitimate. Some replay attack is going on. If the computed sequence number value is acceptable, then the SIM computes the MAC using the same equation as above, which is this. MAC = F1(RAND, K_i, AMF, SQN).

So, we saw that the MAC is computed by the MSC/HLR using the same equation, and the SIM also computes the MAC using the same equation. If the computed MAC matches the

MAC in the received, then the SIM is convinced that the authentication vector was created by the HLR of its home network. And the authentication vector has been freshly created, and it is not a replay from an earlier authentication. So, the fact that the sequence number is a fresh sequence number, that shows that this is not a replay attack. And also the fact that the MAC computed by the SIM card matches the MAC that is received as part of the authentication token.

So, that convinces the SIM card that the authentication vector was indeed created by the HLR of the home network. In this step, the network has authenticated itself to the SIM card. Now, the SIM card has to prove to the network that it is legitimate. So, the SIM card replaces the SQN value it stored with the new value that it computed. So, since the sequence number has increased, it replaces the stored sequence number with the sequence number used in the current authentication process.

Then, in order to prove to the network that it is indeed a legitimate SIM card, the SIM computes the response SRES, that is signed response to the challenge RAND that is generated by the HLR using the above equation, which is this. SRES = F2(RAND, K_i). This is the same equation used to compute XRES, the expected response. So, this SRES is then sent to the network, as shown in this step. SRES is communicated to the network.

So, SRES is communicated to the MSC/VLR, as shown by this arrow. Then, this decision box shows that the MSC/VLR compares SRES and XRES. If they are not equal, it means the authentication of the SIM card has failed. So, that is shown by this arrow. It's an error, so the MSC/VLR aborts the authentication process.

If SRES = XRES, then the SIM has successfully authenticated itself to the network. So, a match is proof that the SIM has knowledge of the secret K_i, thus completing the authentication of the SIM to the network. To summarize, the MAC in the AUTN was used to authenticate the network to the SIM, and this matching of SRES to XRES that is used to authenticate the SIM to the network. Finally, the SIM computes CK and IK, that is, the ciphering key and the integrity key, and conveys these to the cell phone for providing encryption and integrity checking for all future messages between the cell phone and BSC. So, that is shown here.

The SIM card computes IK and CK, and these are conveyed to the cell phone. So, operations involving the secret key K_i are performed by the SIM, whereas these keys CK and IK are transferred from the SIM to the cell phone, and the ciphering and integrity protection is done by the cell phone. So, this is similar to that in GSM. We have seen the reason for this, that the secret K_i is a secret key and it should not leave the SIM. So, all operations involving the secret K_i are performed in the SIM.

Then the next step is agreement on the encryption and integrity check algorithms. The MSC/VLR sends the list of all permissible MAC encryption algorithms to the BSC. So, that is shown by this arrow. The MSC/VLR sends the allowed cryptographic algorithms to the BSC. The BSC has already received such a list from the cell phone in step 1, that was shown by this arrow.

The security capabilities were sent from the cell phone to the BSC. The BSC decides which of these algorithms it can or will support and sends these to the cell phone. So, that is shown in this step. The BSC sends the algorithms that it wants to support to the cell phone. So, this message, which is shown by this arrow, is integrity-protected to prevent an attacker from creating a spoofed message that contains possibly weaker options.

For example, no encryption. So, an attacker cannot modify this message and replace the strong algorithms with weak algorithms or remove the strong algorithms from the list because it is integrity-protected. And it can be integrity-protected because the SIM card and the network have already agreed on a key that can be used for integrity protection, namely IK. So, using the key IK, this message is integrity-protected. The BSC also receives CK and IK to be used for encryption and integrity protection of all messages between the BSC and the cell phone.

So, that is shown by this arrow. Apart from the allowed cryptographic algorithms, the values of CK and IK are also conveyed by the MSC/VLR to the BSC. Subsequently, all the messages that are sent after this authentication process are encrypted using the key CK and integrity-protected using the key IK. So, this completes the authentication and key agreement process. So, we have seen how authentication and key agreement takes place.

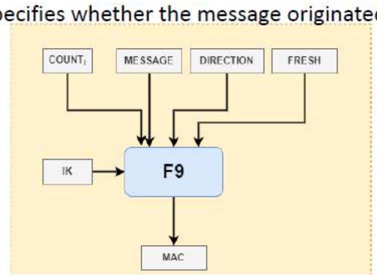
What about the messages exchanged between the cell phone and the BSC after the authentication and key agreement phase? So, we should encrypt them and provide message integrity for them. Now, we will discuss the message integrity and encryption processes. So, we start with message integrity. Message integrity is provided using a message authentication code.

Most signaling messages are MAC-protected. For example, we saw that the false base station attack is defended against using a message authentication code. However, UMTS does not protect the integrity of user messages. So, if the integrity of user messages needs to be protected, it must be done by the application. This shows the procedure for computing a message authentication code.

This F9 is a keyed hash function. The per-message MAC is computed using this process. This is the message whose integrity is to be protected. And there are these other fields. One is the integrity key, which is a secret.

We discussed earlier that for a message m, the message authentication code can be computed as H(m,s), where H is a cryptographic hash function and s is a secret. So, this is a variant of the same principle. But apart from the message and the integrity key, there are some other fields also used to defend against replay attacks and similar attacks. So, the per-message MAC = F9(IK, COUNT_I, FRESH, DIRECTION, message). So, this integrity key IK, which was completed during the authentication and key agreement phase, is used to generate and verify the MAC.

- Per-message MAC is computed using (see fig.):
  - ❑ Per-message MAC = $F9(IK, COUNT_I, FRESH, DIRECTION, message)$
- The integrity key, IK, computed during the authentication and key agreement phase, is used to generate/ verify the MAC
- Two variables, $COUNT_I$ (a sequence number derived from the frame number) and $FRESH$ (a random number) are used to prevent replay attacks
- At connection set-up, $COUNT_I$ is initialized by the cellphone, while $FRESH$ is generated by the BSC
- $DIRECTION$ is a one bit variable, which specifies whether the message originated at the cellphone or the BSC



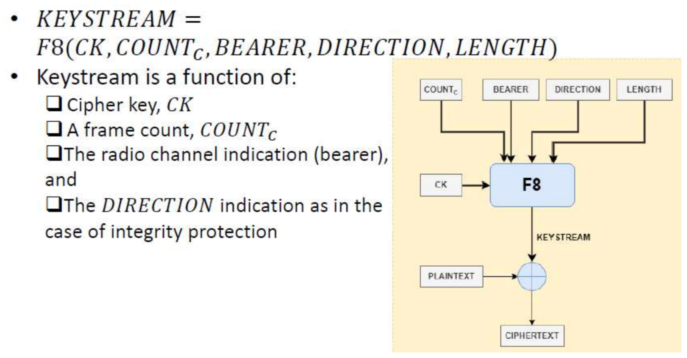
So, clearly, in the message authentication code, there must be some secret which is known only to the two entities sending the message. So, this integrity key is a secret known only to the cell phone and the network. So, two variables, COUNT_I and FRESH, are used to prevent replay attacks. COUNT_I is a sequence number that is derived from the frame number; hence, it changes from message to message. And FRESH is a random number; hence, that also changes from message to message.

So, these are used to prevent replay attacks. At the time of connection setup, COUNT_I is initialized by the cell phone, whereas FRESH is generated by the BSC. So, COUNT_I defends the cell phone against replay attacks, and FRESH defends the BSC against replay attacks. So, these are used to defend against replay attacks. Direction is a one-bit variable that specifies whether the message originated at the cell phone or the BSC.

So, because of the presence of this direction flag, an intruder cannot take a message going in one direction and play it back in the other direction. That’s because the direction flag indicates which direction the message is sent in. So, that concludes our discussion of message integrity. Now we discuss encryption. So, recall that integrity check is performed on only signaling data.

In contrast, encryption is performed on signaling data as well as user data. So, that is, encryption is important because an intruder should not be able to eavesdrop on the information. So, encryption is performed on signaling data as well as user data. So, for encryption, a stream cipher is used. In particular, a key stream is generated, which is a function of the ciphering key CK and some other fields, and this key stream is bitwise XORed with the plaintext to generate the ciphertext.

So, hence it’s a stream cipher. First the key stream is generated, and then it is XORed with the plaintext. And the key stream should be clearly different for different messages. So, this COUNTC parameter, it changes from message to message, and it ensures that the key stream changes from message to message. The key stream is given by this equation.

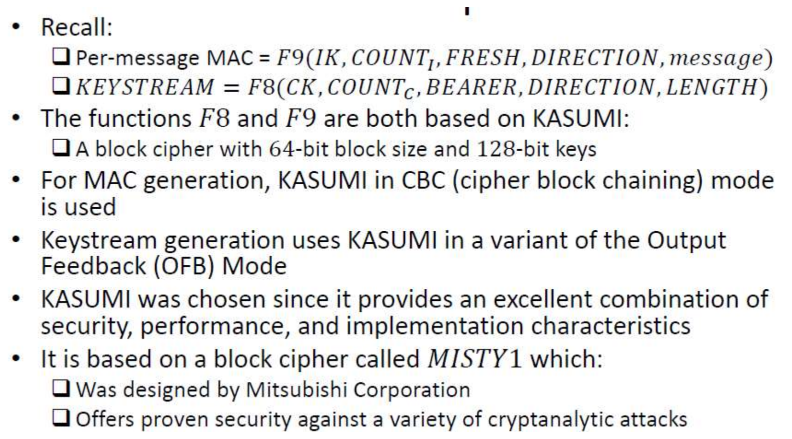


- $KEYSTREAM = F8(CK, COUNT_C, BEARER, DIRECTION, LENGTH)$
- Keystream is a function of:
  - ❑ Cipher key, CK
  - ❑ A frame count, $COUNT_C$
  - ❑ The radio channel indication (bearer), and
  - ❑ The $DIRECTION$ indication as in the case of integrity protection

KEYSTREAM = F8(CK, COUNT$_C$, BEARER, DIRECTION, LENGTH). The keystream is a function of the cipher key CK. So, there must clearly be some secret used to generate the keystream. So, that secret is the ciphering key CK. Then, the keystream is a function of a frame count, that is, COUNT$_C$. It changes from frame to frame.

So, that ensures that the key stream also changes from frame to frame. The key stream is also a function of the radio channel indication BEARER and the DIRECTION indication, as in the case of integrity protection. So, this DIRECTION flag has the same function as in integrity, as we discussed. That is, the intruder should not be able to take a message from one direction and replay it in the other direction. So, that's the purpose of the DIRECTION flag.

So, that concludes our discussion of encryption. Now, all the keyed hash functions that we discussed—F1, F2, F3, and so on—are based on the KASUMI cipher. So, recall that the per-message MAC is F9 of these variables, and the key stream is F8 of these variables. These functions, F8 and F9, are both based on KASUMI. KASUMI is a block cipher with a 64-bit block size and 128-bit keys.



- Recall:
  - ❑ Per-message MAC = $F9(IK, COUNT_I, FRESH, DIRECTION, message)$
  - ❑ $KEYSTREAM = F8(CK, COUNT_C, BEARER, DIRECTION, LENGTH)$
- The functions $F8$ and $F9$ are both based on KASUMI:
  - ❑ A block cipher with 64-bit block size and 128-bit keys
- For MAC generation, KASUMI in CBC (cipher block chaining) mode is used
- Keystream generation uses KASUMI in a variant of the Output Feedback (OFB) Mode
- KASUMI was chosen since it provides an excellent combination of security, performance, and implementation characteristics
- It is based on a block cipher called $MISTY1$ which:
  - ❑ Was designed by Mitsubishi Corporation
  - ❑ Offers proven security against a variety of cryptanalytic attacks

So, for MAC generation, KASUMI in CBC, that is cipher block chaining mode is used. So, in the context of Wi-Fi, we discussed CCMP, and there also cipher block chaining was used to generate the MAC. So, a similar process is used to generate the MAC using KASUMI in the case of UMTS. Keystream generation uses KASUMI in a variant of the output feedback mode. So, during our discussion of encrypting long messages using a block cipher, we have discussed output feedback mode.

And keystream generation uses KASUMI in a variant of the output feedback mode. KASUMI was chosen because it provides an excellent combination of security performance and implementation characteristics. So, KASUMI is based on a block cipher called MISTY1. It was designed by Mitsubishi Corporation. And it offers proven security against a variety of cryptanalytic attacks.

So, KASUMI is more secure than COMP-128 which was used in the case of GSM. KASUMI is also space-efficient. So, the space complexity of the algorithms used is low. A hardware implementation of KASUMI requires less than 1,000 gates. KASUMI can also perform encryption fast.

It can perform encryption at a sustained rate of about 2 Mbps with a clock speed of about 200 MHz. So, because of all these desirable characteristics, KASUMI is used for ciphering and message integrity in UMTS. So, in summary, we discussed UMTS security. It consists of authentication and key agreement, and the authentication is mutual; that is, not only does the SIM authenticate itself to the network, but the network also authenticates itself to the SIM. After authentication and key agreement, subsequent messages are encrypted, and message integrity is provided in them.

So, this is done using the KASUMI cipher. The security of GSM was based on COMP-128, whereas the security of UMTS is based on KASUMI. And UMTS overcomes many of the shortcomings in GSM security. This concludes our discussion on UMTS security. Thank you.