Network Security

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

Week - 07
Lecture - 40
Securing Wireless LANs : Part 6

Hello, recall that in the previous lecture, we discussed achieving encryption, message integrity, and replay attack prevention using TKIP. In this lecture, we will discuss how to achieve similar objectives, namely encryption, message integrity, and defense against replay attacks using CCMP, which is a protocol used in 802.11i. CCMP stands for Counter Mode with CBC-MAC Protocol. CBC is Cipher Block Chaining, and MAC is Message Authentication Code. We have discussed counter mode earlier in our discussion of using block ciphers to encrypt long messages, and we also discussed CBC, cipher block chaining, at that time.

In this protocol, CBC is used to generate the message integrity check, and counter mode is used for encryption of the packet. So, we'll discuss how these are done. CCMP uses AES for both encryption and message integrity. AES is a secure protocol. Hence, it is used for encryption as well as message integrity.

Unlike TKIP, the same key, which is the 128-bit Temporal Key (TK), is used for encryption and MIC computation. We have discussed earlier that typically different keys are used for encryption and MIC computation, but CCMP uses the same key for encryption and MIC computation. So, despite this, it is secure. We'll discuss this in more detail later. Recall that a fresh encryption key is computed for each frame in WEP and TKIP. In contrast, the same encryption key is used for each frame in CCMP.

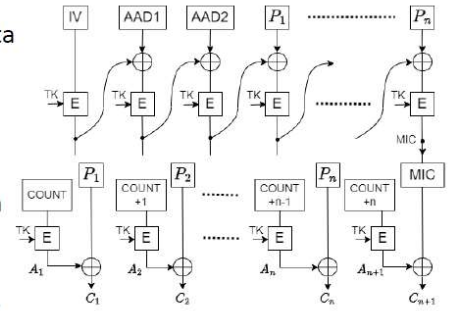
So, what's the reason for this? The reason is that CCMP uses AES, which is a block cipher, in contrast to WEP and TKIP, which use RC4, a stream cipher. So, RC4 generates a key stream, which is just XORed with the plaintext. So, the key stream has to be different for different frames. If the key stream is the same in two different frames, then we can just get the XOR of the plaintext in the two frames by XORing the ciphertext.

So, for this reason, different encryption keys have to be used for different frames in RC4. But in contrast, CCMP uses a block cipher, so there is no need to use different encryption keys for different frames. A 48-bit Packet Number (PN) is initialized to 0 when a session is started between a sender and a receiver, and it increments by 1 for every packet that is sent. It is analogous to the frame sequence counter that is used in TKIP. So, one of the functions of the packet number is to defend against replay attacks.

And it is also used to derive the IV, that is, the initialization vector. Recall that CBC uses an initialization vector, and counter mode also uses an initialization vector. So, these initialization vectors are derived using the packet number. So, hence, the IV is different for different frames. Now, first, we discuss message integrity in CCMP.

Subsequently, we will discuss encryption. The upper part of the figure shows the procedure used to generate the MIC. So, at this point, the MIC is generated. This is the MIC that is generated. So, there is an IV or initialization vector, and this is a temporal key.

- MIC is computed for each packet using AES in Cipher Block Chaining (CBC) mode with block size 128 bits as shown in fig.
  - Output of the last block is 128 bits in length; its lower 64 bits are discarded to get a 64-bit MIC
- MIC computed over frame data ($P_1, \ldots, P_n$ in fig.) and some fields in MAC header (e.g., source and destination MAC addresses)
- Key for performing encryption in each stage is TK
- IV for MIC computation is a nonce, which includes the 48-bit PN



The IV is encrypted to get this text, which is then XORed with AAD1, which is additional authentication data. It includes certain immutable fields of the MAC header, fields which should not be modified. Similarly, AAD2 also includes additional authentication data. Notice that this process here is just cipher block chaining; that is, the ciphertext generated from one block is XORed with the next plaintext block, and then the result is encrypted using the secret key, which is the temporary key in this case. Then, the ciphertext is XORed with the next plaintext block, and the XOR output is encrypted using the same key, TK, and so on and so forth.

So, this is just cipher block chaining, and this is used to generate the MIC. So, the MIC is computed for each packet using AES in cipher block chaining mode with a block size of 128 bits, as shown in this figure. So, each of these blocks is 128 bits in length. The output of the last block is hence 128 bits in length. So, the output of this block is 128 bits in length.

And its lower 64 bits are discarded to get a 64-bit MIC. Since these lower 64 bits are discarded, using the MIC, we cannot reverse this process and get the inputs. So, because these lower 64 bits are discarded, this makes it impossible to reverse this process even if the temporal key is known. So, now the MIC is computed over the frame data, that is, P_1 to P_n in the figure, and some fields in the MAC header, for example, source and destination MAC addresses, to detect any tampering in these fields in the MAC header. The key for performing encryption in each stage is the temporal key that is negotiated during the authentication process of 802.11i.

The IV for MIC computation, that is this one, as we said earlier, it is a nonce which includes the 48-bit packet number. Hence, the IV is different for different frames because it's a function of the 48-bit packet number. Recall that earlier we have discussed that for a message m, one way to generate the message authentication code or the MIC is to generate this H(m,s), where s is the secret key. So, this can serve as the message integrity check. So, in this case, an alternative process is used where cipher block chaining is used to generate a MIC instead of just using a cryptographic hash function and finding H(m,s). So, this is an alternative method to generate a message authentication code or a MIC.

So, again we see that the MIC in this case is a complicated function of all the inputs along with some secret key that is the temporal key. So, in spirit, it is similar to the generation of H(m,s). So, this is an alternative process. Now, this method for computing the MIC is called Cipher Block Chaining Message Authentication Code or CBC-MAC. Note that the MIC generated using this method is a function of all the bits of the message over which it is computed.

That is AAD1, AAD2, and P_1 to P_n. If one or more bits of the message change, then it is very likely that the MIC will change. Hence, if an intruder tries to modify some of the bits of the plaintext, AAD1, or AAD2, then this modification will be detected because MIC verification will fail. CBC-MAC is a MIC generation technique that has been used for many years and has been standardized internationally. So, if you're interested in studying its security analysis, then you can refer to this reference, which studies the security of this method, CBC-MAC, for generating a message integrity check.

So, we have discussed message integrity in CCMP. Now, we will discuss encryption in CCMP. So, this lower part of the figure shows the scheme for encryption. So, the plaintext bytes P_1, P_2 up to P_n, these have to be encrypted, and the MIC, which is generated as discussed previously. So, this MIC also has to be encrypted. That is encrypted using the usual counter mode.

So, we have discussed the counter mode for encrypting long messages using the block cipher. So, this counter mode is used to encrypt the plaintext and the MIC. So, in particular, there is an IV that is, 'COUNT'. This is encrypted using the temporal key to get one ciphertext, which is denoted here by A_1, and then that A_1 is just XORed with P_1 to get C_1, which is the first block of the ciphertext. Then, this IV, 'COUNT', is incremented, and then the incremented value is encrypted to get A_2, which is then XORed with P_2, and so on and so forth.

So, at every step, we increment the value of 'COUNT', and we increment the previous value 'COUNT+1', 'COUNT+2', 'COUNT+3', and so on. We keep on incrementing it each time and then encrypt it to get a value A_n, which is a ciphertext. A_n is the value obtained by encrypting this number, and that is XORed with the plaintext to get the ciphertext. And then, finally, this value is encrypted using the temporal key, and we get A_{n+1}, which is XORed with the MIC to get this ciphertext corresponding to the MIC. So, the frame data and the MIC are concatenated and then encrypted using AES in counter mode, as shown in this figure.

So, this 'COUNT' is a function of the P_n, and hence the 'COUNT' is different for different frames. The key for performing encryption of each block is the temporal key, which was derived as part of the 802.11i authentication process. And the receiver performs decryption followed by MIC verification. So, the receiver reverses this process to get the values P_1, P_2 up to P_n and the MIC, and then verifies whether this MIC is correct. So, the receiver generates the same MIC using the same process as the transmitter used and then checks whether that is equal to the MIC that is in the received frame.

Now, notice that the same key is used for MIC generation and encryption in CCMP. So, the same key, in particular, the Temporal Key (TK), is used for encryption as well as MIC generation. We can see that for MIC generation, the temporal key is used in this, and again for encryption, the same key, that is, the temporal key, is used here as well. So, as discussed earlier, it is, in general, not good practice to use the same key for two separate

cryptographic functions. So, we typically use a different key for encryption and a different key for message integrity.

So, this rule is broken here. But although the same key is used, it is in each case used in conjunction with an IV. So, that's denoted by this IV here and COUNT here. COUNT is the IV in this encryption process. So, these are different.

IV is different from COUNT. So, the construction of the IV is different for the counter mode and CBC-MAC portions. So, this value IV is different from this COUNT. So, using this fact, this protocol has been shown to be secure by cryptographers. So, if you are interested in the security analysis, you can refer to this reference.

So, despite the fact that the same key is used for generation and encryption, through a security analysis, it is shown that this protocol is still secure. Now, we discuss replay attack prevention in CCMP. Recall that a 48-bit Packet Number (PN) is initialized to 0 when a session is started between a sender and receiver, and it increments by 1 for every packet sent. The packet number is included in the header field in a CCMP frame. And the sender and receiver keep track of the PN of the last frame sent and received.

And this PN is used for replay attack prevention. The process is similar to that in TKIP. Upon receipt of a frame, the receiver compares the value of PN in the frame to the PN of the last frame that was received. And if the former is less than the latter, then the received frame is discarded. Only if the PN in the received frame is greater than the PN of the last frame that was received, it is accepted.

Hence, if an old frame is replayed by an intruder, it is discarded. So, the PN in the old frame will be less than or equal to the PN of the last frame that was received. Hence, the received frame will be discarded by the receiver. Now, suppose an intruder creates a new frame with a higher PN than the PN of the last frame sent by the legitimate sender and sends it to the receiver. So, will the receiver accept the frame?

So, again the receiver will not accept the frame because the IV that is used for MIC computation includes the packet number. So, the MIC verification at the receiver will fail. So, the MIC is a function of the packet number, and hence, the intruder doesn't know the temporal key used to generate the MIC; hence, the intruder cannot generate the correct value of the MIC. So, the IV used for MIC computation includes the packet number, and hence, although the intruder has added a higher packet number than the previous packet number sent by a legitimate sender, the intruder is not able to compute the correct MIC,

and for this reason, the MIC verification at the receiver will fail. So, in this way, CCMP defends against replay attacks.

So, in summary, we discussed CCMP, which is used in 802.11i. We discussed how message integrity is achieved in CCMP, and then we discussed encryption. And we also discussed defense against replay attacks. So, TKIP was an intermediate solution, and it was later found to have security vulnerabilities, but CCMP is a much more secure solution than TKIP, and CCMP is currently used in 802.11 products for security. So, CCMP is a secure protocol based on AES, which is currently used, and we will discuss how it operates.

Thank you.