Network Security

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

Week - 07

Lecture - 39

Securing Wireless LANs : Part 5

Hello, in the previous lecture, we discussed how the access point and station can authenticate each other in 802.11i. After authentication, they must exchange data messages with each other, and these data messages must be encrypted, and message integrity must be provided in them. So, we will now discuss the encryption and message integrity processes in 802.11i. So, we will discuss confidentiality and message integrity in 802.11i as well as WPA, which, as we discussed, was an intermediate measure until the deployment of 802.11i. So, in WPA, a protocol known as TKIP is used, and in 802.11i, CCMP is used.

What are these? Recall that RC4, as used in WEP, has several weaknesses. So, this prompted the 802.11i standards committee to seek a replacement. So, AES was just standardized at that time. So, this standard for symmetric key cryptography was an obvious choice to replace RC4.

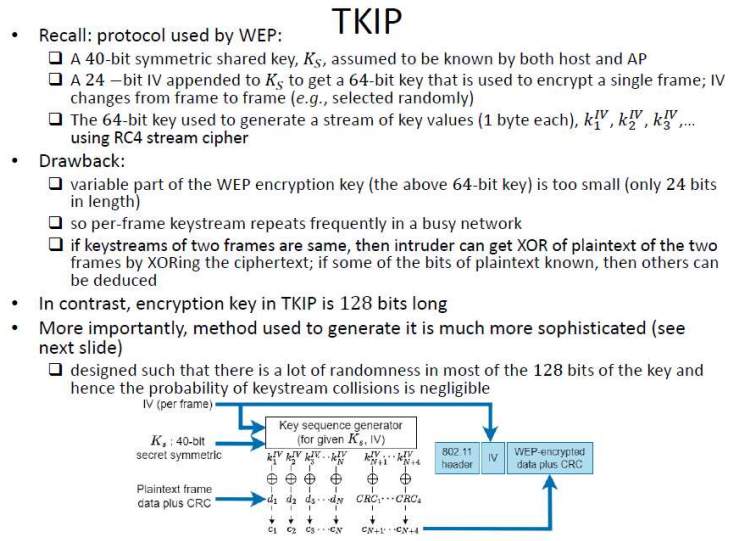
But the problem was that there was a lot of existing hardware that already deployed RC4, and AES could not be implemented in that hardware. The use of AES required the use of new hardware. So, when weaknesses were found in WEP, there were already millions of installed Wi-Fi systems, and these were rendered insecure. So, instead of replacing all this hardware with new hardware that could run AES, which would be very expensive, it was better to deploy some intermediate solution that would allow these already deployed Wi-Fi systems to be upgraded and become secure again. So, an intermediate solution was required to run on already installed hardware.

And that intermediate solution was TKIP. So, this intermediate solution that was developed was a firmware upgrade that retained the existing 802.11 hardware, including RC4. But there were several changes in the overall design that eliminated the vulnerabilities in WEP. So, we discussed several vulnerabilities in WEP. Those were because of the way in which the RC4 cipher was used as part of WEP.

But in TKIP, several changes in the overall design were made which eliminated these vulnerabilities. And using the same hardware but with upgraded firmware, these vulnerabilities were eliminated and a secure solution was obtained. That was TKIP. So, this intermediate solution was known as Temporal Key Integrity Protocol or TKIP, and it was used for encryption and message integrity in WPA. Recall that WPA is Wi-Fi Protected Access and 802.11i is WPA2.

So, this TKIP was an intermediate solution, but the final solution, which used new hardware, in 802.11i, it used new hardware to implement AES. This final solution is referred to as counter mode with CBC-MAC protocol, or CCMP. So, CBC stands for cipher block chaining, which we discussed earlier, and counter mode, also we had discussed earlier during our description of sending long messages using symmetric key ciphers. So, this final solution uses counter mode as well as CBC. It's known as CCMP, and it is used for encryption and message integrity in WPA2.

So, this was the final solution. We'll first discuss TKIP in detail, and then we'll discuss CCMP. So, we start with TKIP. So, first, let's recall how WEP operated. So, in the protocol used by WEP, there was a 40-bit symmetric shared key, which we denote by K_S.



It was assumed to be known by the host as well as the access point. So, that is this key, K_S. It's a 40-bit secret symmetric key. And a 24-bit initialization vector was appended to this key K_S. That resulted in a 64-bit key, and this 64-bit key shown by these two arrows here, this is used to encrypt a single frame.

So, in particular, this 64-bit key was used to generate a key stream, which is denoted by k_1^{IV}, k_2^{IV}, and so on up to k_{N+4}^{IV}, and this key stream is used to encrypt a single frame, and the IV changes from frame to frame. For example, it may be selected randomly. So, this 64-bit key, which is a combination of K_S and the IV, it is used to generate a stream of key values, each of one byte, shown here by k_1^{IV}, k_2^{IV}, up to k_{N+4}^{IV}. And the method to generate this key stream was the RC4 stream cipher. So, the RC4 stream cipher was used to generate this key stream using the 64-bit key.

And then this key stream was bitwise XORed with the data and cyclic redundancy check to get the ciphertext, which was put in the frame. And then the IV was also included in plaintext form. And there was a header, and this was the WEP packet. But this had drawbacks, so one drawback was that the variable part of the WEP encryption key, that is, the above 64-bit key, was the IV, which is only 24 bits in length. So, since this was only 24 bits in length, there were only 2^{24} distinct values of the IV.

So, in a busy network where a lot of packets were sent, the per-frame key stream repeats frequently. So, whenever the IV is the same in two different frames, in that case, the key stream is also the same in the two frames. So, if the key streams of the two frames are the same, then the intruder can get the XOR of the plaintext of the two frames by just XORing the ciphertext. So, this is because let the common key stream in the two frames be K. So, this is a vector, which is a key stream, and the plaintext in the first of those frames is P_1. So, $P_1 \oplus K = C_1$, which is the ciphertext in the first frame.

And the plaintext in the second frame is P_2, and that is XORed with the same key stream, that is K, to get the ciphertext in the second frame, that is C2. So, we can clearly see that if we XOR C1 and C2, then that is just $P_1 \oplus P_2$. So, one can get the XOR of the plaintext by just XORing the ciphertext. And if some of the bits of the plaintext are known, then the other bits can be deduced. For example, if we know some of the bits in P_1 somehow, then we can deduce the corresponding bits of P_2 because we know the XOR of P_1 and P_2.

So, this was a drawback in the way in which this RC4 was used in WEP. Now, in contrast, the encryption key in TKIP is 128 bits long, so it is double the length of the key in WEP, 128 instead of 64. And more importantly, the method used to generate the encryption key in TKIP is much more sophisticated. We'll discuss it on the next slide. It is designed such that there is a lot of randomness in most of the 128 bits of the key, and hence the probability of key stream collisions is negligible.

So, there are 2^{128} possible values of the encryption key. And this encryption key is generated randomly, so because of this randomness, it is highly unlikely that the encryption key will be the same in two different frames. We now discuss the method for the generation of this 128-bit encryption key in TKIP. So, this figure shows the process used to generate the encryption key in TKIP. So, this TK is the temporal key, and the temporal key along with the sender's MAC address and the four most significant bytes of the sequence counter, which is a kind of sequence number.

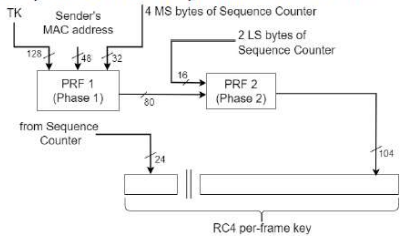
So, these are combined into a pseudo-random function known as PRF1, and the output of PRF1 goes to another pseudo-random function, PRF2, and the two least significant bytes of the sequence counter are fed to the PRF2 function, and its output is 104 bits, and these 24 bits are from the sequence counter, which changes from frame to frame. And then the combination of these 24 bits and the 104 bits is the RC4 per-frame key, which is 128 bits in length. So, the inputs to this process are the 128-bit Temporal Key (TK), the sender's MAC address, and a 48-bit frame sequence counter. So, we discussed how the temporal key is derived as part of the authentication process in 802.11i and WPA. So, that's the temporal key, which is known to the access point and the station.

The sequence counter is incremented for every frame sent. It is carried in the header of each frame. So, the sequence counter is a sequence number. One of its purposes is to defend against replay attacks. So, the use of this sequence counter ensures that the per-frame key is different for different frames with a higher probability.

So, two pseudorandom functions, PRF1 and PRF2, are used in these two phases. So, this is PRF1 and this is PRF2. So, one important observation is that the least significant 16 bits of the sequence counter are inputs to PRF2, as shown here by this arrow. So, the output of PRF2 changes for every frame sent. Note that the least significant bit will toggle every frame.

So, hence the output of PRF2 will change for every frame sent because these two least significant bytes of the sequence counter change from frame to frame. But in contrast, the 32 most significant bits of the sequence counter are input to PRF1, and this input changes once every 2^{16} frames sent. So, that's because this sequence counter is updated from frame to frame. So, it's represented in binary using its binary representation, and it increments every frame. So, only after 2^{16} frames are sent will one of the bits in these four most significant bytes change.

- Two pseudo-random functions (PRF1 and PRF2) are used in the two phases
  - ❑ The least significant 16 bits of the sequence counter are inputs to PRF2
  - ❑ So output of PRF2 changes for every frame sent
  - ❑ The 32 most significant bits of the sequence counter are input to PRF1
  - ❑ This input changes after every $2^{16} = 65{,}536$ frames sent
  - ❑ Hence, PRF1 is executed very rarely and overall computation time is saved
- The randomizing properties of the key mixing function and the large size (128 bits) of the key space ensure that keystream collisions occur very rarely



So, hence the input to PRF1 changes after every 2^{16} frames sent. Hence, PRF1 is executed very rarely, and overall computation time is saved. So, every frame only PRF2 has to be executed, and PRF1 is executed once every 2^{16} frames. So, that saves on computation cost. And the randomizing properties of the key mixing function and the large size, which is 128 bits, of the key space ensure that key stream collisions occur very rarely.

So, these PRF1 and PRF2 produce outputs that look very random. And also, the length of the key is 128 bits, which is double that in WEP. So, because of these reasons, key stream collisions occur very rarely. And once we have generated this 128-bit key, how do we do the encryption? The encryption in TKIP is done just as in WEP.

- Encryption in TKIP is done as in WEP, with the change that instead of the 64-bit encryption key (K_S+IV), the 128-bit output of the above two-phase key mixing function is used as the encryption key for RC4 keystream generation

That is, this 128-bit key is used to generate a key stream, which is XORed bitwise with the plaintext and CRC. Plaintext and another input, which we'll discuss later. So, the plaintext is XORed with the key stream as in WEP, but the change is that instead of the 64-bit encryption key, which is (K_S+IV), the 128-bit output of this two-phase key mixing function

is used as the encryption key for the RC4 key stream generation. So, the RC4 key stream is generated using this 128-bit key as its input, and then the key stream is XORed with the plaintext. So, now recall the reason for the weakness of WEP.

It was shown that when WEP is used, an attacker can find the secret key used for encryption in a small amount of time after eavesdropping on the network. And in a busy network, a successful key recovery may take as little as one minute. So, in a network that is not busy, it may take longer. But in any case, it is likely that key recovery can be done. And automated tools are available on the internet that implement this attack.

But in contrast, it can be shown that this two-phase key mixing function used in TKIP, which is shown over here, eliminates the WEP key recovery attacks. We'll omit the details for brevity, but this was the most serious weakness of WEP, that is, the key could be recovered, the encryption key could be recovered in a small amount of time. So, that weakness is overcome by this two-phase key mixing function used in TKIP. So, because of this, the most serious weakness of WEP was overcome using TKIP. So, we have discussed how encryption is done in TKIP.

Now we discuss message integrity. So recall that the method used to compute the 4-byte CRC in WEP is such that it is possible to predict which bits in the CRC will change if we change a single bit in the data message. So, we discussed earlier how using this property, the intruder can break the message integrity of WEP. So, this property arises because of the fact that the CRC in WEP is a linear function, that is, $CRC(m_1 \oplus m_2)$ is the same as $CRC(m_1) \oplus CRC(m_2)$. So, this property is known as linearity, and CRC used in WEP is a linear function, and it is because of this property that one can predict which bits in the CRC will change if we change a certain number of bits in the plaintext.

- Above property arises due to the fact that the *CRC in WEP is a linear function,* i.e., $CRC(m_1 \oplus m_2) = CRC(m_1) \oplus CRC(m_2)$
- 64-bit message integrity check in TKIP, called MIC or Michael, is a significant improvement over the CRC in WEP
- Unlike the CRC, MIC is non-linear, i.e.:
  - $MIC(m_1 \oplus m_2) \neq MIC(m_1) \oplus MIC(m_2)$

If we change a certain set of bits in the plaintext, which bits in the CRC will change can be predicted because of this linearity of the CRC. Now, the 64-bit message-integrity check in TKIP, called MIC or Michael, is a significant improvement over the CRC in WEP. So, unlike the CRC, the MIC is nonlinear; that is, the $MIC(m_1 \oplus m_2) \neq MIC(m_1 \oplus m_2)$. So, because of this, it is no longer possible to predict which bits of the MIC will change when

a certain set of bits of the plaintext message are changed. And, in addition, the MIC is computed as a function of the data in the frame and some fields in the header, such as the source and destination MAC addresses.

So, any alteration in these fields in the header can be detected because of the computation of the MIC over these fields as well. So, clearly, the MIC must use some secret as an input. So, the MIC also uses as input a key derived from the pairwise transient key, which was computed during the four-way handshake that we discussed in the previous lecture. So, this MIC is computed as a function of the data, some fields in the header, such as source and destination MAC addresses, as well as a secret key that is derived from the pairwise transient key. So, hence, because of the presence of the secret key among the inputs of the MIC, an intruder cannot calculate the correct MIC for a particular packet.

Now, due to limitations of WEP hardware, the MIC only uses simple logical functions, shift, and add. No multiplications are used because they are computationally expensive. So, there were these limitations because of which the best possible MIC could not be implemented, but this was a compromise. The MIC only uses some simple logical functions, shift and add, and no multiplications are used. Hence, because of these compromises, the MIC is not as secure as a keyed cryptographic hash function.

So, we discussed that one way to generate a MAC is H(m,s) for a message m. If H is a cryptographic hash function, then H(m,s) can be used as a secure message authentication code or a MIC. So, this MIC used in TKIP is not as secure as a keyed cryptographic hash. On the other hand, it is much stronger than the CRC checksum used in WEP. So, hence, it was used as an intermediate measure until the deployment of 802.11i, which was based on AES. Next, we discuss replay attack prevention in TKIP.

Recall that WEP provides no protection against replay attacks, so we discussed how replay attacks are possible in WEP. Replay attack prevention is provided in TKIP as follows. A 48-bit frame sequence counter is carried in the header of each frame. So, we discussed that this sequence counter is one of the inputs to the key generation functions, PRF1 and PRF2. This sequence counter starts from 0 and increments by 1 for every packet sent.

It is unlikely to wrap around since it is 48 bits long. If it gets close to wrapping around, then the client renegotiates a new PTK, or Pairwise Transient Key. This sequence counter is extracted by the receiver and is used to compute the RC4 key for decryption. So, recall that the RC4 key for encryption is generated using the sequence counter, and the same

process is used to compute the RC4 key for decryption at the receiver side. And the sender and receiver keep track of the sequence number of the last frame that was sent or received.

Now, the receiver accepts a frame only if its sequence number is greater than that of the previous correctly received frame. So, if the intruder just takes an old frame and replaces it, then its sequence number will be less than or equal to the sequence number of the last frame that was received correctly. So, hence, it will be rejected by the receiver. So, because of this check, if an intruder replaces an old frame, then it is rejected by the receiver. But now suppose an intruder creates a new frame which has a higher sequence number than the sequence number of the last frame sent and sends it to the receiver.

Then, will the receiver accept the frame? So, from this process as well as the process that we discussed to generate the MIC, we can see that the receiver will not accept the packet because the RC4 key for the generation of the key stream is a function of the sequence number and temporal key, which is unknown to the intruder. So, the intruder cannot calculate the correct RC4 key. So, after the receiver computes the RC4 key and decrypts the frame, the MIC verification will fail. So, hence, the receiver will reject the frame.

So, hence, TKIP provides protection against replay attacks. So, this is the procedure used for defending against replay attacks in TKIP. So, now we have discussed encryption, message integrity, and defense against replay attacks in TKIP. What about its security? How secure is it?

So, as we have discussed already, TKIP is much more secure than WEP. So, that allowed TKIP to be used as an intermediate solution until the deployment of 802.11i based on AES. So, it incorporates, TKIP incorporates mechanisms to defend against several attacks to which WEP is vulnerable. For example, key recovery attacks, attacks on the message integrity of packets, and replay attacks. However, it has been shown that TKIP is vulnerable to some new attacks, and we do not discuss the details of those attacks for brevity.

So, because of these attacks, TKIP is no longer considered secure, and it was deprecated in the 2012 revision of the 802.11 standard. But TKIP served its purpose, that is, it was meant as an intermediate solution until 802.11i could be used. So, it served as a temporary solution, and subsequently 802.11i was deployed, and hence TKIP was deprecated. So, in summary, we are discussing the encryption and message integrity processes used in WPA and 802.11i. We started with TKIP and we discussed the encryption, message integrity, and replay attack prevention processes used in TKIP.

In the next lecture, we'll discuss encryption, message integrity, and replay attack prevention in 802.11i using the protocol CCMP. Thank you.