Thomas Hart

CS 465 HW #6

In order to both encrypt and authenticate a message, an encryption method and a message authentication code (MAC) are needed. A message is encrypted, sent through a hash function, and that hash is appended to the message to send. When the ciphertext from the original message is sent through the MAC algorithm on the receiving end, the result should match the appended hash. This ensures confidentiality from encryption and integrity from the MAC. The process of creating the MAC has a few variations, though.

The first option is called “authenticate and encrypt”. This is when the MAC is produced by putting the plaintext through the MAC algorithm. The plaintext is then encrypted, and the MAC is appended to the ciphertext. The second is “authenticate then encrypt”, which is just like the first except the MAC is appended to the plaintext before it is put through the encryption algorithm. The result of the extended message is sent. Lastly, “encrypt then authenticate” is when the plaintext is encrypted first, then the ciphertext is sent through the MAC algorithm, and the result is appended as the certificate. Out of all these methods, the last is the most secure and should be used as a rule of thumb. Let’s go through each example and explain the pros and cons.

Although authenticate and encrypt is probably the simplest method to implement, it is not safe at all. For example, the MAC can reveal clues about the plaintext if some blocks are repeated. The ciphertext also has no integrity since it must be decrypted before sent through the MAC algorithm.

Authenticate then encrypt seems like it would be safe. However, it breaks an important rule: you should never have to do any operations before verifying the MAC when you receive a message. The first method breaks this rule because you first need to decrypt the message before verifying the MAC, and although the second example’s MAC is used in the encryption process, it is still susceptible to a certain danger – Vaudenay’s attack. This attack requires the validating system to notify if there was either a padding error or a MAC error and uses the inevitable padding needed on the end of a message to make it fit a block cipher. The padding is typically made up of enough bytes to fit the block cipher, all of which contain the number of total padding bytes. So, for example, if we had a message with 6 bytes of padding needed, we would append 6 bytes of padding, all with the value 0x06. To validate the padding, someone receiving the message would look at the final byte of padding and work backwards, making sure there are 6 bytes of padding and that each of them contain the value 0x06. An attacker would manipulate the second to last byte of the padding, likely getting back a padding error because it is incorrect. They would try all 0-255 options for an 8-bit value until they no longer received a padding error but a MAC error. This means they’ve finally guessed the correct padding. Whatever they need to XOR the ciphertext with to get that value is the plaintext, and they have discovered a byte of the message. They can do this over and over until they’ve decoded the entire message. Because the MAC authentication is not the first step, this is possible.

The third and final method, encrypt then authenticate, does require the receiver to check the MAC first, since the MAC is produced by putting the ciphertext through the MAC algorithm. Attackers cannot perform attacks until they verify this, which means the previous attack is impossible.

To summarize all the methods, encrypt and MAC does not have integrity of ciphertext since it must be decrypted before verifying. Repeats in the plaintext will also be visible in the MAC and give attackers clues about the nature of that plaintext. Authenticate then encrypt does not reveal things about the plaintext since it is also encrypted. However, the ciphertext has no integrity since this method also requires decrypting before MAC verification. This method also has subtle weaknesses that allow attacks that change the original message and still have a valid MAC, depending on the cipher scheme. Encrypt then authenticate does not have any of these potential vulnerabilities. It has plaintext integrity like the others, ciphertext integrity (assuming the shared secret is not revealed), protection against invalid ciphertexts since the MAC will reveal it first thing, and no discoveries about the plaintext in the MAC since it is also encrypted.

For these reasons, always use encrypt then authenticate. It is the safest method of producing and appending a MAC to a ciphertext.