Assignment-4

**Introduction**:

We are going to analyze two different ways of running a protocol, compare them and analyze whether they achieve their security goals. The S/MIME specification allows for encryption before signing or signing and encrypting. Each has different security properties. PGP specifies that one should sign and then encrypt. We will build two different models where we analyze whether PGP way of signing and then encrypting is better than encrypting a message and then signing it.

**Protocol Descriptions**:

For the following descriptions, M is the message, K is a freshly chosen symmetric key, PubK(N) is the public key of N, and PrivK(N) is the private key of N. {}K stands for encryption with key K. Hash(X) is hashing of message X.

1. Sign then encrypt:

A -> B: {M, A, {Hash(M)}PrivK(A)}K, {K}PubK(B)

1. Encrypt then sign:

A -> B: {M}K, A, {Hash({M}K)}Privk(A), {K}PubK(B)

**CPSA Models**:

1. **Sign then encrypt**:

A screenshot of a computer program

Description automatically generated

* For the “Sign then encrypt” model of the protocol, we have defined two roles namely Alice and Bob respectively. Here Alice is sending the message to Bob.
* We have two names, one message, and one symmetric key.
* We are using the Public Key infrastructure of CPSA to make use of public and private keys for encryption.
* We have defined two skeletons, one for Alice’s point of view and one for Bob’s point of view.
* In each skeleton, we assume that Private keys of both A and B are non-originating and kept secret.

**CPSA Model Analysis**:

A screenshot of a computer program

Description automatically generated

* In Alice’s point of view, we have a single node with no arrows, because alice sends a message without receiving a message back from bob.

A screenshot of a computer

Description automatically generated

* In Bob’s point of view, we have two shapes. One shape tells us that it is not possible to complete the communication without alice.
* The second shape is a dashed line going from alice to bob. A dashed line indicates that some of the parameters might have been changed when the message was transmitted from Alice to Bob.

A white text with black letters

Description automatically generated with medium confidence

* If we look at the message sent by Alice and the message received by Bob, we can see that the parameter b was originally b-0 and symmetric key k was originally k-0.
* This means that the original message was meant to be sent to a recipient b-0. The recipient b-0 got the message, decrypted it using his private key and symmetric key k-0, re-encrypted it using public key of b, symmetric key k, and sent it to b.
* When b got the message, he could decrypt the whole message using his private key, and assumes that the message was sent by Alice.
* This model of Sign & Encrypt is vulnerable to surreptitious forwarding where the recipient of the message has no way of knowing that message was sent by the sender or a malicious threat actor.

1. **Encrypt then Sign**:

A screenshot of a computer program

Description automatically generated

* In Encrypt then sign method, we are encrypting the whole message with symmetric key and signing the whole encrypted message with the private key of alice. We also send the name of alice, and encrypted message, along with the key encrypted under the public key of bob.
* We have two roles in which alice sends the message and bob receives the message.
* We are looking at two skeletons i.e. two perspectives, one in alice’s point of view and one in bob’s point of view.
* Since we know that alice does not receive a confirmation back from bob, we won’t be focusing on the point of view of alice.

**CPSA Model Analysis:**

A screenshot of a computer program

Description automatically generated

* In bob’s point of view, we can see that there is a dashed line between alice and bob.
* CPSA determined that one of the parameters was changed when it was being transmitted from alice to bob.
* If we look closely at the messages sent and received, we can clearly see that the original message was meant to be sent to b-0, but then b-0 received the message, decrypted the whole thing with symmetric key k, extracted alice’s signature and repurposed it to send a message to bob (b).
* When bob receives this message, he will assume that the message was sent by alice.
* Encrypt-then-Sign isn't any better than Sign & Encrypt. It's easy for any eavesdropper to replace the sender's signature with his own, so as to claim authorship for the encrypted plaintext.

**Fixing Sign then Encrypt**:

* The problem with “Sign then Encrypt” is that the signature can be used again. That means we need a way for the signature to also verify that the intended receiver is Bob and the symmetric key was never changed during transit.
* For that, we can add the name of intended receiver(In this case, Bob) along with the key.
* That means we calculate the hash of message, key and bob’s name. We encrypt this hash under private key of Alice. Rest of the messages should be same.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

* As we can see, CPSA determined that the message has to be from Alice. The solid line indicates that the message was from Alice to Bob.

**Fixing Encrypt then Sign**:

* In case of “Encrypt then Sign”, we have an issue where there is no way of knowing that the message arrived was intended for Bob. That is why we must add the name of Bob inside the signature. Since the signature cannot be re-encrypted by the intruder, he can’t change the name inside the signature.

A screenshot of a computer code

Description automatically generated

A screenshot of a computer program

Description automatically generated

* The solid line from Alice to Bob indicates that the message has to be sent by Alice to Bob.

**Conclusion**:

We can see from the observations that both the existing ways of sending messages were flawed. In both cases, the signatures did not verify whether the message was intended for the recipient. Therefore, we added the name of the recipient and in case of “Sign the Encrypt”, we had to add the symmetric key as well. This fixes the problem where the signatures and messages were being repurposed and sent to other recipients.