Home Assignment 2
EITN41

Anon

A-6 *What is the purpose of the random values R_1 in a Mix?*
The purpose of using random values R_1 in a Mix is to hide the correspondence between inputs and outputs to the Mix.

Without R_1 the input to the Mix would be $K_1(K_a(R_0, M), A)$. It would be trivial for an attacker to encrypt the output $K_a(R_0, M), A$ with the Mix's public key K_1 and do a lookup in a list of recorded input messages and find the corresponding input, effectively linking origin and destination.

A-7 *When sending a mail through several Mixes, there are several public keys involved: $K_1, K_2, ..., K_n$ and K_a. What happens if one does not use K_a? Does this risk the anonymity of the sender?*
If K_a is used, the output of the mixes look like this:

$$K_a(R_0, M), A$$

Without using K_a, the output should be:

$$R_0, M, A$$

This means that the randomness R_0 is superfluous and the message M is sent in plaintext. This does not directly mean that the sender's anonymity is immediately revealed , but the identity could be deduced from M.

A-8 *Briefly explain how using several Mixes versus an onion routing circuit differ both in terms of latency and in cryptographic primitives used for encrypting the traffic.*
In an onion routing circuit, asymmetric keys are used to negotiate symmetric keys with routers. These symmetric keys are then used to encrypt messages. With using several Mixes, a sender has to encrypt their message with each Mix's public key. Symmetric cryptography is faster than asymmetric cryptography, meaning that the latency is lower in an onion routing circuit than with using several Mixes.

A-12 *Regarding replay attacks on Mixes, two protections are suggested in the lecture notes. Which? Would you say that any of them is the better choice? Show how the two strategies can be combined and how this can make the protection more efficient.*

1. Calculate hash for each input message and store this. Do a lookup of all subsequent messages, if one matches, throw away.

2. Include a timestamp in each input, to verify that the message is 'fresh'.

One difference to consider is the storage and computational issues with using hashes: computational for calculating hashes and doing later lookups, storage for storing hashes of all messages, whose cardinality can be quite large.

A-13 *It is straightforward to generalize the N - 1 attack to an N - k attack, 0 < k < N. Describe the N - k attack.*

The general idea of a N - 1 attack is to reduce the size of the anonymity set for a user. An attacker controls exactly N - 1 inputs to the Mix (total inputs = N). The attacker knows the recipients to their N - 1 messages and therefore the only message they didn't send is the one sent by the victim. The generalization of this is that an attacker controls N - k inputs. Even if this attack is not as effective as the N - 1 attack, it still reduces the size of the anonymity set of a target user to the size k, which from the attacker's point of view is something positive.

A-15 *In the disclosure attack on mixes, explain m, N, n and why a Mix is insecure if* $m \leq [N/n]$.

- **m**: The number of Alice's communication partners. This is also the result of a successful attack, m sets with exactly one recipient, then these m recipients are Alice's communication partners.
- **N**: The total number of users in the anonymity system.
- **n**: The number of receivers in a batch.

A Mix is insecure if $m \leq [N/n]$ because if $m > [N/n]$ then N cannot be divided into m sets of size n. The implication of this is that m recipients cannot be found.

A-26 *A TCP handshake consists of the client and the server exchanging three messages: SYN, SYN-ACK and ACK. Explain why, in Tor, Alice can connect to a webserver and expect the TCP handshake with the server to be performed with low latency.*

When a client wants to connect, they use a variant of Diffie-Hellman key exchange and the server's (router's) public key to negotiate a symmetric key. This symmetric key is then used to encrypt messages. This works on the basis that symmetric cryptography is faster than asymmetric cryptography. The negotiated symmetric key is then changed fairly often, each time the circuit changes and new nodes are to be connected to symmetric keys have to be negotiated with these. The frequency of changing circuits can be decided by the user.

A-28 *Show that the SSL/TLS handshake, when RSA is used, does not provide perfect forward secrecy.*

When RSA is used in the SSL/TLS handshake, the client generates a session key and encrypts this with the server's public key. The weak point here is that if somebody steals the server's private key, they can decrypt all previous sessions. This situation is precisely what perfect forward secrecy provides protection against. In the draft for TLS 1.3 support for RSA without pfs is dropped.