COMP90043: Cryptography and security

Week 7: Hash Functions

1. **What are the advantages using Hash functions in digital signatures?**

   Some uses are:

   - You can sign arbitrary long messages.
   - You can use to build authentication mechanisms (see text/slides).
   - Assuring integrity of messages.

2. **Explain how you can use RSA encryption function to construct a digital signature scheme.**

   Public Key Parameters: (n, e)

   Private Key parameter: (n, d)

   Hash Function: (H)

   Compute Signature (S) = $(H(M)^d)$ mod n; [M, s] form message signature pair.

   Verification Algorithm: If H(M) == $((s^d)$ mod n) then accept the signature else declare verification failure.

3. **What characteristics are needed in a secure hash function?**

   1. H can be applied to a block of data of any size.
   2. H produces a fixed-length output.
   3. H(x) is relatively easy to compute for any given x, making both hardware and software implementations practical.
   4. For any given value h, it is computationally infeasible to find x such that H(x) = h. This is sometimes referred to in the literature as the one-way property.
   5. For any given block x, it is computationally infeasible to find y ≠ x with H(y) = H(x).
   6. It is computationally infeasible to find any pair (x, y) such that H(x) = H(y).

4. **What is the difference between weak and strong collision resistance?**

   Weak collision resistance: Property 5 of answer in previous question;

   Strong collision resistance: Property 6 of answer in previous question.

5. **Is it possible to use a hash function to construct a DES like block cipher? How is it possible?**

   If you examine the structure of a single round of DES, you see that the round includes a one-way function, f, and an XOR: $R_i = L_i–1 \ f(R_i–1, K_i)$

   For DES, the function f maps a 32-bit R and a 48-bit K into a 32-bit output (see the slides/text for details). That is, it maps an 80-bit input into a 32-bit output. This is clearly a one-way function. Any hash function that produces a 32-bit output could be used for f. The demonstration in the text that decryption works is still valid for any one-way function f.

6. **Explain the birthday paradox in simple words? What is the main implication of this for hash function?**

   The problem for adversary in collision resistant attack is to produce to messages x and y which give rise to same hash value, i.e. H(x) = H(y).

   Birthday paradox is not a paradox! If we choose random variables from a uniform distribution in the range 0 to N-1, then the probability that a repeated element is encountered exceeds 0.5 after about $\sqrt{N}$ (square root N) choices have been made.

   You can see Appendix 1A of Chapter 11 of the text for precise derivation of this result.

   Hence for m bit hash, if we pick messages at random, we can expect to find two messages with the same hash value with about $\sqrt{2^m} = 2^{m/2}$ attempts. Hence two break the collision resistant property of an m bit hash function; the effort for adversary is upper bounded by a quantity approximately $2^{m/2}$.

   This implies that m should be chosen reasonably high.

7. **Name three important hash functions used in practice.**

   MD4, MD5, SHA-1/2/3.

8. **Discuss how the security of the hash functions depends on the length of the hash.**

   (Discuss Brute force attack and birthday attacks and show how they influence the decision on length of the hash.) 32 bit hash is trivial to break. The length of the hash should be at least 160 bits or more.

9. **Why CRC checksum cannot be used as a secure hash function?**

   Collisions are easy to find and does not satisfy one way property requirement of hash functions.

10. **Consider the following questions in regards to Timing Attacks:**

    a. **What is a Timing Attack?**

       This is a very dangerous attack side channel attack, as it factors in the time complexity involved in deciphering a message by the intended receiver in order to be able to recover the private key.

    b. **How can Timing Attacks be prevented?**

       The following are examples of approaches which can be used to prevent timing attacks:

       - **Constant Exponentiation Time:** Requires the modifications of algorithms so as to ensure that all exponentiation operations take a fixed amount of time to execute before producing a result. This can be detrimental on performance however.

       - **Random Delay:** Furthers on the exponentiation time problem by introducing random delay periods during each exponentiation operation to throw off an attacker from knowing how long the operation actually took. Offers better performance than constant exponentiation time operations.

- **Blinding:** Involves the multiplication of the plaintext by a random integer prior to performing any exponentiation operations on it.