Network Security E4.44/ISE4.45/SO21 Answers to Problem Sheets 1-4

Problem Sheet 1

1. For a particular 64-bit block of plaintext there are 2^{64} possible blocks of ciphertext. There are only 2^{56} possible DES keys. Each key has a one in 2^{64} chance of transforming the chosen plaintext into a particular ciphertext. If all possible keys are tried the overall probability reduces to one in $2^{8.}$

2 If the initial permutation had value than it must be a factor in making DES unbreakable (except by exhaustive key search). If this were the case than removing the permutation should render DES breakable. However, the removal of a known permutation which can easily be reversed cannot enable DES to be broken and therefore it is clear that the permutation has no security value.

3. The 28 left side bits would be the new 28 right side bits and vice versa.

4 The 8 parity bits have no security value.

5. The number of possible payments is 990 There are 10^5 possible home account numbers and an equal number of destination account numbers. The number of possible messages is therefore $9.9\text{x}10^{12}$. The number of possible MACs is just 2^{16} or 65,536. Therefore the number of payment messages with the same MAC will be on average $9.9\text{x}10^{12}/6.5536\text{x}10^4 = 1.51\text{x}10^8$. From the Birthday Paradox the payment clerk will need to test only $2^8 = 256$ messages before he is likely to find two with the same MAC.

6. For a number to have a multiplicative inverse in modular arithmetic it must be relatively prime to the modulus. The only way all numbers from 1 to n-1 are relatively prime to n is for no itself to be prime. $2^{16} + 1$ ia a prime number but 2^{16} is not. Zero does not have a multiplicative inverse and is taken as 2^{16} which does have a multiplicative inverse and incidentally cannot be represented as a 16-bit number.

7. The keys K_b and K_c used in encryption must be converted to their additive inverses for the decryption process. The additive inverses of 9EC3 and AFD4 are 613D and 502C respectively.

8. In the even round keys K_e and K_f remain the same for encryption or decryption.

9. If the computing power for encrypting a single block in DES is C_{DES} and that for encrypting a single block in IDEA is C_{IDEA} then the increase in computing power required for an exhaustive key search on IDEA is $2^{72}\text{x}C_{IDEA}/C_{DES}$.

10. The pseudo random block stream generated in OFB using weak keys will be a stream of alternating blocks.

11. It is assumed that the tenth block of plaintext is known and that it is required to change it to 8 all-zero octets. This may be achieved by adding the original value of the tenth block modulo 2 to the ninth block of ciphertext. This will have the effect of causing an unpredictable change in the ninth block of deciphered plaintext.

12. In CBC a message block is added mod 2 to the previous ciphertext block and then encrypted. If the same key is used to find the CBC residue as MAC as is used for encryption the final message block will be the same as the previous ciphertext block and the result of mod 2 addition will be an all zero block as input to the encryptor. The resulting ciphertext and the all zero input will represent a known plaintext/ciphertext pair.

13. In 8-bit CFB an addition or deletion will cause false output for the next 8 blocks but after that the true input to the encryption process will be restored. In CBC synchronization is never restored following a similar disruption.

Problem Sheet 2

1. A good message digest function has the following properties: if the message digest for a particular message is known, it should not be feasible (from that information alone) to find the message digest for another message; and secondly it should not be feasible to find two messages with the same message digest.
   (i) this fails both the above tests
   (ii) this also fails both tests

2. SHA-1 requires a field which gives the actual length in bits. This will cause the message to require additional padding to satisfy the requirement to be an exact multiple of 512 bits.

3. 1 and 512 bits.

4. Euclid's algorithm may be used to find multiplicative inverses for algorithms such as IDEA and RSA.

   (i) Let the gcd(m,n) = r
   Then as in Euclid's algorithm there must be integers u and v such that um + vn = r and therefore um/r + vn/r = 1 from which it follows that m/r and n/r must be relatively prime

   (ii) If a and b are relatively prime there must be integers u and v such that ua + vb =1 . Therefore uac +vbc = c. But bc = ka and therefore c is seen to be a multiple of a.

5. A prime number since all numbers from 1 to n-1 are relatively prime to n if n is prime
6. A number containing the maximum possible number of prime factors.

7. No because the maximum value for Euler's totient function occurs when n is prime and the value is n – 1.
8. The problem is to find the probability that a message to be encrypted will not be relatively prime to n. The number of elements in Z_n is n = pq where pa and q are prime. The number of elements in Z_n^* is (p – 1)(q – 1) = Φ (n).
   The probability that a number is not relatively prime to n is therefore

   $$(pq - (p-1)(q-1))/pq \quad = \quad (p+q-1)/pq \;=\; 1/p \;+\; 1/q \;-\; 1/pq$$

9. If x has a multiplicative inverse mod n, then there exists a number u such that ux = 1 mod n. Therefore ux + vn =1 mod n from which it follows that x must be relatively prime to n.
10. In a bucket brigade attack an interceptor sets up secure communications using Diffie-Hellman key exchanges with both parties. The attacker can certainly make up a Diffie-Hellman value and send it encrypted with one endpoints public key but to set up a shared secret he must be able to decrypt the Diffie-Hellman value encrypted with the other endpoint's public key. Unless he has stolen the private key he will not be able to achieve this.
11. See the answer to Q. 8 for the probability that a randomly chosen number would not be relatively prime to n.. If such a number were obtained it would lead to a factorization of n from which it would be possible to find Φ (n) and the multiplicative inverse of any public key mod Φ(n) and thus the private key of a key pair.
12. The multiplicative inverse of e mod Φ (n) is unique and therefore there is only one private key d for a given e, p and q (pq = n)

Problem Sheet 3

1. This protocol is designed to provide secure authentication of the workstation A to the server B. It is clear that there is no authentication of the server and indeed all that is required to impersonate the server is to issue a random number R. The real problem is to impersonate A from information obtained by (a) eavesdropping and (b) disclosure of B's database.

(a) Without knowledge of the hash function itself an attacker could build up a table of R and hash(Y,R) and wait for an R to be repeated. The strength of the protocol would then depend upon the size of the field from which R is chosen at random. If the attacker has knowledge of the hash function and how Y is combined with R to calculate hash (Y,R) then an offline dictionary attack could be used to find Y. In this case the strength of the protocol is related to the size of output of the hash function.

(b) If the attacker discovers all that is in the server's database it is clear that he will know Y, the hash function and how Y and R are combined to make hash (Y, R). The attacker has everything he needs to impersonate A or any other authorized user whose details are on B's database.

2. No since B does not send any cryptographic information until A has responded to the initial challenge.
3. No, since an attacker only has to replay an old genuine response of A.
4. .The hash function is a one –way function which does not allow an eavesdropper to find the password from hash^{n-1}(password)
5. Provided the hash function is known an attacker impersonating the server could transmit his own n and perform a dictionary attack on the hash^{n-1}(password) received.
6. There are a number of solutions to this problem. One solution is to have k variants on the hash (password) function perhaps by combining a seed 1, 2, 3… k with the password before hashing. Each variant hashed to n-1 would be stored in the server (a different n for each variant). In sending the challenge the server would identify the variant and its associated n. It is clear that the storage would be k times as large and on the basis that the individual ns would on average be k times smaller, the processing at the client would be reduced k times.
7. If there is no cryptographic integrity check on the KDC database then an attacker could remove an authorized user's entry and substitute his own. Hw would then be in a position to impersonate the other party.
8. A KDC must remember all blacklisted postdated tickets
9. If a CRL were not issue periodically then it would be possible to intercept a CRL without causing any alarm.
10. In order to make the storage of blacklisted certificates manageable.

Problem Sheet 4

1. An attacker should conduct a bucket brigade attack on the Diffie-Hellman exchange. The third message in the protocol allows him to obtain A's identity. However, because he cannot sign his own Diffie-Hellman number (with A's private key) in his exchange with B, B does not respond with his won identity. If it is desired to maintain the client's identity secret at the expense of the server, then the order of the third and fourth messages should be reversed.
   If A knows B's public key then B could sign its Diffie-Hellman number which would prevent a bucket brigade attack on A. If the third and fourth messages are left in their original order this would enable both identities to be kept secret.
2. There are a number of answers to this question depending on the way the context is understood. This is my interpretation:
   Submitting an expense claim (non-r, sa)
   Inviting a friend to lunch (ip)
   Selling illegal merchandise over the network (p, ip, r)
   Sending a purchase order (ip, non-r, sa)
   Sending a message to the tax authorities about a neighbour's alleged tax evasion (p, ip, r)