PRINCIPLES OF SYSTEM SECURITY

IS 336: SSIGNMENT 1&2

2017-04-07311

WESTON, BARAKA

Questions:

1. By Fermat’s theorem, what is the multiplicative inverse of 2 in the field of integers mod 11?

work:

From p=11, a=2

x=

x=

x=512 mod 11

x=6

1. With a public key encryption, suppose A wants to send a message to B. Let Apub and Apriv be A’s public key and private key respectively, similarly for B. Suppose C knows both public keys but neither private key. If A sends a message to B,

* What encryption should A use so that only B can decrypt the message? (secrecy property)

Ans:

‘A’ should you use Asymmetric key encryption because A and B have private Keys, meaning that the public message will be transmitted over the network and available for C, but can not decrypt the message without knowing B’s private key.

* Can A encrypt a message so that anyone receiving the message will be assured the message came only from A? (authenticity property)

Ans:

Yes, by using digital signature A will be able to do this. Thus, A will use the private key to encrypt the message where by anyone receiving message will use A’s public key combined with cryptographic hash function to decrypt and ensure that A was the sender of the message.

1. Can A achieve both secrecy and authenticity for one message?

Ans:

By using public key cryptography, A can achieve both Authenticity and Secrecy by encrypting the message with public key and private key respectively

1. Find keys d and e for the RSA cryptography where p = 7 and q = 11

work :

First, we will define p and q

ɸ(n) = p\*q

such that if n is prime

ɸ(n) = n -1

and if n is not prime

ɸ(n) = (p-1) (q-1) such that q and p are prime factors

Therefore; ɸ(n) = 7 \* 11,

Since p = 7 and q = 11, given

ɸ(n) = (11-1) (7-1)

= 10\*6

=60

ɸ(n) = 60

But e is such that 1< e < n and e should have no common factors as that of ɸ(n)

Factors of ɸ(n) are: {1,2,3,4,5,6,10,12,15,20,30}

Therefore, we can take any factor for the value of e which we can use to find d

Taking e = 11

(de)mod ɸ(n) = 1

Thus: 11e mod ɸ(n) = 1

11e mod 60 = 1

With modulus of 1 by 60 the number to be a multiple of 60, the number is 121 giving us d = 11.

Therefore, the values of d = 11 and e = 11

1. Is the DES an onto function, that is, is every 64-bit binary string the result of encrypting some string? Justify your answer

Ans:

No. DES is not an onto function because it has an effective key length of 56 bits and not 64 bits because 8 bits of the 64 bits are not used by the encryption algorithm since they function as check bits only.

1. Could the full 64 bits of DES key be used, thereby giving it a strength of 264 instead of 256­­­? Justify your answer

Ans:

NO because 8 of the 64 bits are used for parity cheque and only 56 of the 64 bits are used.

* 1. Assume each S-box substitution takes 8 units of time (because of the eight 6-bit substitutions), each P-box permutation takes 4 units of time (counting 1 unit per byte), each expansion permutation takes 8 units of time (because of the eight 4-bit expansions and permutations) each initial and final permutation takes 8 units. Compute the number of units of time for an entire 16-round cycle of the DES.

Data: work:

S-box (SB) = 8 units the sum of all products

P-box (PB) = 4 units (SB\*16 + SP\*16 + E \* 2 + IP + FP)units

Expansion (E) = 8 units 8\*16 + 4\*16 + 8\*2 +8 + 8

Initial Permutation (IP) = 8 units 128 + 64 + 16 + 8 + 8

Final Permutation (FP) = 8 units 224 units

* 1. Now suppose DES were redesigned to work with a 112-bit key and a cycle on 128 bits of input, by increasing the number of S- and P- boxes. You do not have to define the details of this design. Using similar timing assumptions as in the first part of this question. Compute the number of units of time for an entire 16-round cycle of 112-bit DES.

Data: work:

S-box (SB) = 16 units the sum of all products

P-box (PB) = 8 units (SB\*16 + SP\*16 + E\*2 + IP + FP)units

Expansion (E) = 16 units 16\*16 + 8\*16 + 16\*2 +8 + 8

Initial Permutation (IP) = 8 units 256 + 128 + 32 + 8 + 8

Final Permutation (FP) = 8 units 432 units

* 1. Perform a similar estimate for the timing of triple DES using E (k1, D (k2, E (k1, m)))

Work:

Encryption

m-E-k1-D-k2-E-k1

1. Write a computer program that implements fast exponentiation (successive squaring) module

#author: 2017-04-07311

#computer program to compute exponential value under

#modulo using binary exponentiation.

# prime modulo value (N)

N = 1000000007;

# Function part of code

def exponentiation (base, expo):

if (expo == 0):

return 1;

if (expo == 1):

return base % N;

t = exponentiation (base, int (expo / 2));

t = (t \* t) % N;

# if exponent is even value

if (expo % 2 == 0):

return t;

# if exponent is odd value

else:

return ((base % N) \* t) % N;

# Sample code driver

base = 5;

expo = 4;

modulo = exponentiation (base, expo);

print(modulo);