# **e-Security Test 1 (CSN11117/102)**

There will be four main questions in the exam [[guide](http://asecuritysite.com/csn11117/guide)]:

* Symmetric Key.
* Hashing.
* Public Key.
* Key Exchange.

These are some sample questions which will get you thinking in the right direction.

## 1. Symmetric Key

Key principles: Salting, AES, ECB, CBC, Hashing

Q1 Computing power increases each year. Outline the challenge this gives when protecting encrypted data. [Ref: Symmetric Key]

Moores law states that computing power doubles every 18 months or so. This means that even for single processor machines, encrypted data thought safe (million year scale crackability), would be crackable in well under a year by 2020. Add GPU matrix processors into the mix and the time reduces to hours / minutes / seconds

Q2 What are the possible advantages of using stream ciphers over block ciphers? [Ref: Symmetric Key]

Stream ciphers operate on a stream of bits, so are useful for encrypting a consistent stream of data (such as streaming video). Bock ciphers work on blocks of data, and depending on the mechanism used, use block feedback to increase encryption. This means that though more robust in terms of security, they are somewhat less resilient to errors in the data and may cause the data to be unreadable.

Q3 The AES method is recommended by NIST for symmetric key encryption. What are the main stages involved in the AES process? [Ref: Symmetric Key]

Depending on the size of the key, apply a number of rounds as follows:

Split the data in the 16 byte blocks, then apply a number of rounds to each block (depending on the key size):

A round consists of

1. Add Round Key,
2. Substitute bytes (using an s-box mapping table / reverse s-box for decryption),
3. Shift rows (row 1 <- 1, row 2 <- 2, row 3 <- 3)
4. mix columns (matrix multiplication)

Apply the next round

Q4 Bob encrypts his data using symmetric key encryption and sends it to Alice. Every time he produces the ciphertext it changes, and he is worried that Alice will not be able to decipher the cipher text. He encrypts "Hello" and gets a different cipher stream each time. Why does the cipher text change? [Ref: Symmetric Key]

A mechanism can be employed to change the cipher text each time it’s encrypted. This may be done by setting a initialization vector based on some rule known to Bob and Alice (eg. time-based, or based an index each time data is sent). As long as Bob and Alice’s Init vectors are in sync, then Alice will be able to decode the message.

Q5 Bob is sending encrypted data to Alice, and Eve is listening. After listening for a while, Eve is able to send a valid encrypted message to Alice. By outlining ECB, discuss how this might be possible. [Ref: Symmetric Key]

ECB does not apply a different initialization vector to each block. Since the entire key is applied to each block, then blocks identical blocks will result in identical ciphers for each block.

With non ECB methods (eg cipher block chaining), an initialization vector is applied to the first block, then the output of encrypting a block is used as the initialization vector input to the encryption of the next block.

Q6 Bob is using a password to generate a 128-bit encryption key. Explain why the key space is unlikely to be 2128, and why key entropy could be used to measure the equivalent key size. [Ref: Symmetric Key]

In short, thought he theoretical key size is 2^128 bits, since the key is created using a password, the practical key size is massively reduced because the password will be made up of a sub-set of printable characters only. Since only a sub-set of bit patterns will be used, the practical size of the key-space is reduced.

Key Entropy = log2(Phrases) = log10(Phrases) / log10(2)

Key Entropy = log2(Phrases) = log2(26^8) / log10(2) = 37.6 bits

Q7 Bob uses a six-character password with lower case [a-z]. How many passwords are possible? His password system then tells him he needs to add numeric value [0-9]. If he adds it at the end, how many passwords are possible, and what is the key entropy? [Ref: Symmetric Key]

[a-z] = 26 possible values for each character, so 26^6 possible combinations = 308915776

[a-z] = 26 possible values for each character first 6 characters = 26^6 possible \* 10 for the [0-9]= 3089157760

Q8 Bob says that the number of bytes used for the cipher text will change directly with the number of bytes used in the plain text. Alice disagrees and says that most encryption methods involve having block sizes. Who is correct? Explain why. [Ref: Symmetric Key]

Q9 With block encryption, how do we know where the ciphered data actually ends? Does it just use an end-of-file character or a NULL character? [Ref: Symmetric Key]

It is common for there to be not enough data to fill the last block, so the block is padded. There are a number of mechanism, one option is to pad with a byte value of 0, but this can cause problems (eg. C uses a zero byte value as a string terminator).

Another mechanism, uses a hex value equal to the number of padding bytes (eg “0505050505” a five byte padding, or random value may be used, except for the last byte which specifies the number of (random) padding values.

Q10 Alice says she is confused that Bob is sending her the same message as a cipher, but every time the cipher text changes. Apart from using the shared encryption key, what does Alice use to decipher the cipher text? [Ref: Symmetric Key]

An Initialization vector (IV) can be used to “Salt” the cipher. This salt value is intended to prevent Rainbow table attacks. It is not typically considered secret and is usually sent with the cipher (often the prefix to the cipher will specify both the encryption method and the IV used to encrypt it):

Q11 Why would Eve have an aversion to salt? [Ref: Symmetric Key]

Eve will probably dislike salt as it prevents Rainbow table attacks (ie. A set of precomputed encrypted dictionary values).

This means that she has to use a brute-force dictionary attack which is far more processor intensive.

Q12 Bob tells Alice that she won't be able to view the cipher text, but when she looks at the messages, they seem to be full of printable characters. What format is Bob likely to be using for the encoding of the cipher text, and what would you ask Alice to look for, in order to confirm your guess? [Ref: Symmetric Key]

A common (but not the only) encoding mechanism is the Base64 encoding method. This uses (and she should look for) upper and lower case letters, numbers and the ‘+’ and ‘-‘ (to represent each 6 bit block of data in the data-stream). Additionally, the data is divided into groups of 4 6-bit blocks, with the ‘=’ character used to pad the final set of blocks to 4. One or two “=” character at the end of the encoded text is usually an indication that Base 64 has been used.

Q13 Alice has been reading her crypto books, and she reads that there should be an '=' symbol at the end of the encoding. She observes her encoding of cipher messages to Bob and sees that some do not have an '=' sign at the end. Is there a problem with her encoder? If not, how often, on average, should she see an '=' sign at the end of her ciphered messages? [Ref: Symmetric Key]

The ‘=’ character is used to pad final set of 6-bit blocks to 4. There will be either 0, 1 or 2 ‘=’ characters. On average there is a 66% chance of one or two ‘=’ characters appearing at the end of a block of base 64 encoded text.

## 2. Hashing

Q14 Outline the importance of storing the salt value with the hashed value when storing hashed passwords. [Ref: Hashing]

The salt must be stored with the hashed password because it is essentially a random initialization vector and changes the hashed output. Identical values would normally return an identical hash. Adding the salt changes this, but salt is required in order to recreate a hash that can be used for comparison. The salt is primarily added to prevent pre-computed rainbow table attacks and isn’t regarded as a secret component.

Q15 Eve has captured a hashed password. How might she use the Cloud to be able to crack the hashed password, and what is a likely tool for this? [Ref: Hashing]

She’s would most likely use HashCat or John The Ripper type tool along with GPU prcessors. Typically these consist of multiple arrays of (typically thousands of) highly specialized matrix processors. HashCat has optimizations that will divide the key-space amongst the processors.

Q16 Bob is an administrator for a network, and he tells his management team that user passwords are now salted, and they are thus completely secure against attacks. Is he correct? Explain your viewpoint. [Ref: Hashing]

They will be secured against precomputed Rainbow Table attacks, but not against Brute-Force dictionary based attacks. They cannot therefore be regarded as totally secure.

Q17 Bob looks at the **passwd** file on his server and wants to know the type of salting that is used. How would he do this? [Ref: Hashing]

Typically, the salt is stored along with the password has (and encryption type) as a prefix. Examples below:

$1${salt}${hash} – MD5

$2a${salt}${hash} – Blowfish

$2y${salt}${hash} – Eksblowfish

$5${salt}${hash} – SHA-256

$6${salt}${hash} – SHA-512

Q18 Bob is looking for a new hashing method for storing passwords and thinks that he will pick the fastest one. Is this a good approach? Explain your answer. [Ref: Hashing]

Not really, it depends on the security requirements of the application.

The problem with this somewhat broad-brush approach is that hash speed may come at the expense of reduced security (ie. The faster the hash, the easier to crack). For an IoT device a fast low-resource intensive hash mechanism may be desirable, but if the device is in a DMZ then it is a much higher risk of attack.

Q19 What are the typical tools that are used to crack hashed passwords, and what are the methods they will use to crack them? [Ref: Hashing]

HashCat and Joh The Ripper are examples of tools used for password cracking. Methods employed include (but are not limited to):

* A dictionary of common words (with may be ordered by their frequency of use in passwords)
* Pattern matching (eg [a-z]:[0-9]:… The password length and password complexity increases the key size exponentially.
* Parallel-processing – dividing the key-space amongst processors

Q20 If we have a 16-bit key, but only use 200 phrases. What is the key entropy? [Ref: Hashing]

The Key Entropy will be log2(200) =

math.log(200,2)

7.643856189774724

Q21 If it takes 10ns to test an encryption key. How long will it take to crack a 20-bit key? [Ref: Hashing]

=2^20 / 1000(seconds) == 1048576 /1000 = 1048.576 seconds / 60 = 17.46 minutes

Q22 It was stated in the recent Yahoo hack that:

"We have confirmed, based on a recent investigation, that a copy of certain user account information was stolen from our networks in late 2014 by what we believe is a state-sponsored actor," Lord wrote. "The account information may have included names, e-mail addresses, telephone numbers, dates of birth, hashed passwords (the vast majority with Bcrypt), and, in some cases, encrypted or unencrypted security questions and answers."

Do you think the vast majority of the hashed passwords will be cracked? Do you think they had good practice in place for hashed passwords? [Ref: Hashing]

The majority of hashshed passwords could be cracked mainly due to the potentially reduced requirements in relation password complexity compared to today. It is not known whether the passwords were salted, so may be susceptible to a Rainbow table attack.

However, assume salt was used, BCrypt is a relatively slow and resource intensive hashing mechanism, so they are potentially more secure from the increased overhead in brute-force attack.

Having said that, it may be possible to use the unencrypted security questions and answers to reset the password hashses anyway. User accounts are most definitely at risk and should all details (not just the password) should be changed.

Q23 You are working with a security consultant, and he says that you don't need to check the hashing of passwords, as it should work without testing. You disagree with him and decide to test your hashing method. Initially you must find test vectors for MD5, SHA-1 and SHA-256. Can you find three test vectors, and test them against an on-line calculator? [Ref: Hashing]

(come back to this one)

Q24 At a security presentation a researcher gives a demonstration of Scrypt. In the presentation he shows a demonstration with a password of "password" and fixed salt of "NaCl". For each run he runs the hashing function, the hashed value changes, but, each time, the computation took longer. Which parameter is the researcher likely to be changing, and why does that parameter exist? Can the researcher select any value for the parameter? [[Example](https://asecuritysite.com/encryption/scrypt)] [Ref: Hashing]

(come back to this one)

Q25 There has been a major data breach within your company, and you are to appear on Sky News to report it. Your company has used PBKDF2 to hash its passwords. How do you explain to your customers that their passwords are unlikely to be breached? [Ref: Hashing]

## 3. Public Key

Key topics: RSA, Elliptic Curve

Q26 Explain how public key provides both privacy and identity verification. [Ref: Public key]

Take Bob, Alice and Eve;

1. Alice wants to send a secure message to Bob such that Eve cannot read it.

She encrypts the message using Bob’s public key and sends the message to Bob.

Now, the message can only be decrypted using Bob’s private key, so Bob is able to decrypt the message, but Eve isn’t.

1. Bob wants to send a message back to Alice, but wants to ensure that Alice can verify that the message came from him (rather than Eve pretending to be him), and that the message hasn’t been intercepted and changed by Eve. Bob creates a hash of the message, and encrypts the hash with his private key. The message is then sent with the encrypted hash to Alice. Alice decrypts the hash with Bob’s public key. She then hashes the message. If her hash, and Bob’s decrypted hash match, then she can be sure that Bob sent the message, and that the message hasn’t been changed.

Q27 Explain how the *e* and *d* values are determined within the RSA method. What are the values that are distributed and which are kept secret? [Ref: Public key]

A value for e must be selected such that it shares no common factors with PHI (ie. GCD(e,PHI) = 1).

A value for d is selected such that it is the Inverse\_Mod(e,PHI) => de Mod PHI = 1

Q28 Bob has just produced a key pair, in a Base-64 format, and now wants to send this to Alice. What advice would you give him on sending the key pair to Alice? [Ref: Public key]

Don’t send the key pair. Just send public key. His private key should remain private.

Q29 Bob has two numbers which give a GCD of 1. Trent says that this happens because the numbers are prime. Is Trent correct? Explain your answer. [Ref: Public key]

The answer is “maybe”.

Both numbers may be prime, but they don’t have to be. However they must be relatively prime (ie. They share no common factors other than 1)

Q30 With RSA, Bob selects two prime numbers of: p=3, q=5. What are the encryption and decryption keys? For a message of 4, prove that the decrypted value is the same of the message. [Ref: Public key]:

Selecting value of e = 91:

Encryption key = 91

Decryption key = 3

>>> from eSecurity import CryptoMath

>>> CryptoMath.RSA()

Primes 1 to 100

[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97]

Select Prime p=3

Select Prime q=5

N=p\*q=15

PHI=(p-1)(q-1)=8

Select a value for e from the list

[1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47, 49, 51, 53, 55, 57, 59, 61, 63, 65, 67, 69, 71, 73, 75, 77, 79, 81, 83, 85, 87, 89, 91, 93, 95, 97, 99]

Select e=91

finding d=inverse\_mod(e,PHI)=3

M=4

Cipher C=4

Decipher D=4

>>>

Q31 Bob selects a *p* value of 7 and a *q* value of 9, but he cannot get his RSA encryption to work. What is the problem? [Ref: Public key]

9 is not a prime number so the maths doesn’t work

Q32 Bob has selected a p value of 11 and a q value of 7. Which of the following are possible encryption keys: (5,77), (3,77), (9,77), (11,77), and (24,77). [Ref: Public key]

>>> CryptoMath.RSA()

Primes 1 to 100

[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97]

Select Prime p=11

Select Prime q=7

N=p\*q=77

PHI=(p-1)(q-1)=60

Select a value for e from the list

[1, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 49, 53, 59, 61, 67, 71, 73, 77, 79, 83, 89, 91, 97]

Select e=11

finding d=inverse\_mod(e,PHI)=11

M=4

Cipher C=37

Decipher D=4

Q33 Bob and Alice decide to use RSA encryption to send secure email, where Bob uses Alice's public key to encrypt, and she uses her private key to decrypt. What is the main problem caused with this, as apposed to using symmetric encryption? [Ref: Public key]

One problem is speed and resource requirements. RSA is relatively slow process.

However, RSA is not Forward Secure, meaning that if the private key is compromised, then so are all of the messages sent using the public key.

A better mechanism would be to generate a shared (Symetric) Session Key to encrypt the message, and exchange this using a use a forward-secure Public Key exchange mechanism.

Q34 Bob tells Alice that she should send her private key in order that he should encrypt something for her. Outline the main problem caused by this. [Ref: Public key]

Alice should not share her private key with anyone, since any messages encrypted with her public key would be vulnerable to decryption (ie. Bob could read messages from anyone to Alice). Additionally, he could pretend to be Alice by signing messages (HMAC) using her private key.

Q35 Security professionals say that RSA keys of over 1,024 bits are secure. What is the core protection against the RSA method being cracked for keys of 1,024 bits and more. [Ref: Public key]

The main protection here is that given the public key, it’s difficult to factor the Prime numbers that are used to calculate the Public / Private key pair.

Q36 Bob and Alice get into a debate about the size of the d and e values in the RSA encryption key. Bob says that, in real-life keys, the length of the e value in (e,n) is normally about the same size as the d value (d,n). Alice disagrees. Who is correct? [Ref: Public key]

They can be similar, but e is often smaller. However it shouldn’t be too small, since it would be vulnerable to a brute-force (using arrays of maxtrix processors)

Q37 Bob says that Elliptic Curve Cryptography (ECC) is an easy method to crack. Explain to Bob how ECC operates, and why it can be a secure method. [Ref: Public key]

Elliptic Curve Cryptography works by selecting points on an mathematical curve described by a particular type of function:

Y^2 = x^3 – 3x + 5

Elliptic Curves are special in that a non-vertical or horizontal line may intersect the curve at 3 points (point G, n and P)

Both the Generator (G) and the Curve function (shown above) are well known.

The Public key P is essentially an X,Y coordinate where line projected from G through n intersects the curve. This is calculated as nG

Where n is a gradient, and is a point that intersects the curve between P and G

The private key n is selected, and from this the public key P is derived.

Given the curve function, Generator G, and Public key P, it is mathematically very difficult to compute the value of n in a reasonable amount of time.

## 4. Key Exchange

Key topics: Diffie-Hellman, ECDH, Using Public Key to Exchange Key

Q39 For Diffie-Hellman: G=2,351; N=5,683; x=7 and y=14. What is the shared key? [Ref: Key Exchange]

Bob:-

Selects random value x=7

Calculates A=(g\*\*x)%p=(2351\*\*7) % 5683 = 4612

then sends A to Alice

Alice:-

enter (Random) value for y:14

Selects random value y=14

Calculates B=(g\*\*y)%p=(2351\*\*14) % 5683 = 4758

then sends B to Bob

Alice Calculates:

keyA = (A\*\*y) % p = (4612\*\*14 % 5683) = 4614

KeyA = 0b9d9315522cf43c3b956a67f2c95679967aefdd126cefa7c525e99bfc343764

Bob Calculates:

keyB = (B\*\*x) % p = (4758\*\*7 % 5683) = 4614

KeyB = 0b9d9315522cf43c3b956a67f2c95679967aefdd126cefa7c525e99bfc343764

Q40 With Diffie-Hellman, G is 1579, and N is 7561. Bob selects 13 and Alice selects 14. Prove that the shared key is 868. [Ref: Key Exchange]

Bob:-

Selects random value x=13

Calculates A=(g\*\*x)%p=(1579\*\*13) % 7561 = 37

then sends A to Alice

Alice:-

enter (Random) value for y:14

Selects random value y=14

Calculates B=(g\*\*y)%p=(1579\*\*14) % 7561 = 5496

then sends B to Bob

Alice Calculates:

keyA = (A\*\*y) % p = (37\*\*14 % 7561) = 868

KeyA = d2e655334ee2e4841be477484381df1617a8b891adc04cbc536cc1bed229d713

Bob Calculates:

keyB = (B\*\*x) % p = (5496\*\*13 % 7561) = 868

KeyB = d2e655334ee2e4841be477484381df1617a8b891adc04cbc536cc1bed229d713

Q41 Eve says that she sees the values passed within ECDH by Bob, and that she can crack the key. By explaining the ECHD key exchange method, outline how it would likely to be difficult for Eve to determine the shared key. [Ref: Key Exchange]

ECDH applies Elliptic curve methods to DH Key exchange.

Both A & Gagree on Generator G and value b

A selects random point a & calculates Qa = da \* G : Qa 🡪 B

B select random point b & calculates Qb = db \* G : Qb 🡪 A

Shared Key A = da \* Qb == da \* db \* G

Shared Key B = db \* Qa == db \*da \* G

Though Qa & Qb may be known, G, d are agreed in advance (and are secret), also random points a & b should be kept secret.

Not capturable by Eve

## Exam paper from 2016/2017

This exam paper was a closed book test.

**Question 1**

Bob and Co is an ISP, and they have recently been hacked, and their passwords released to the Internet. Their lead Information Officer defines that the passwords use eight-character passwords and were salted with a three-character hex value. The regular expression to filter the passwords defines the range of [a-z0-9] with a letter of the alphabet in the first character.

1. What advice would you give to the company on their current policy on hashing their passwords? [5]

Stronger hashing method, increased length of password, remove restriction of letter for first char, larger salt value, increase entropy by using non-alpha characters.

NB: Can’t be using PBKDF2 as salt is too small, so not

(b) In the investigation, a hash cracker of 1 Tera hashes per second has been used. Can you estimate how long it would take to crack all the passwords in the data? Give the working-out. [5]

Salt = 3 character Hex value = (2^(3\*4)) == (2^12)   
Alpha character in first position (26)  
7 characters [a-z0-9] = (36^7) :

>>> ((2\*\*12) \* 26 \* (36\*\*7))/1000000000000/60

139.0911669927936

NB: Each hex value has 4 bits

**Question 2**

1. Calculate, for Diffie-Hellman, the shared key, if the agreed values are G=201, N=31, and Bob selects 15 and Alice selects 3. Give the working-out. [Marks: 3].

Bob:-

Selects random value x=15

Calculates A=(g\*\*x)%p=(201\*\*15) % 31 = 30

then sends A to Alice

Alice:-

enter (Random) value for y:3

Selects random value y=3

Calculates B=(g\*\*y)%p=(201\*\*3) % 31 = 27

then sends B to Bob

Alice Calculates:

keyA = (A\*\*y) % p = (30\*\*3 % 31) = 30

KeyA = 624b60c58c9d8bfb6ff1886c2fd605d2adeb6ea4da576068201b6c6958ce93f4

Bob Calculates:

keyB = (B\*\*x) % p = (27\*\*15 % 31) = 30

KeyB = 624b60c58c9d8bfb6ff1886c2fd605d2adeb6ea4da576068201b6c6958ce93f4

1. In RSA, Bob generates two prime numbers: 13 and 11. From this create the encryption and decryption key. Give the working-out. [Marks: 3]

Select Prime p=13

Select Prime q=11

N=p\*q=143

PHI=(p-1)(q-1)=120

Select a value for e from the list

[1, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 49, 53, 59, 61, 67, 71, 73, 77, 79, 83, 89, 91, 97]

Select e=19

finding d=inverse\_mod(e,PHI)=19

M=19

Cipher C=7

Decipher D=19

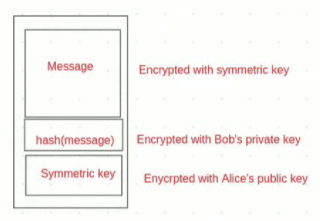
(c) Mallory and Eve are being watched by law enforcement agencies. The law enforcement agency decides that they want to decrypt the messages sent by Mallory to Eve, and thus sends Mallory a digital certificate related Eve, with a fake public key (for which they have the private key). Outline the problems that could be caused by this method, and how might the law enforcement agency overcome them? [Marks: 4]

**Question 3**

1. PKI uses key pairs for encryption and digital certificates to prove identity. Explain how PKI can be used to keep messages between Bob and Alice secret, and also how we can prove Bob's identity and the integrity of the message. How might an intruder manage to pretend to be Bob? [Marks: 5]

Bob signs the hash of the message with his private key, then his identity is proved.

The entire message + hash



(b) PGP provides a method of securing email. Outline how PGP uses asymmetric and symmetric encryption in order to secure emails, while proving identities. [Marks: 5]

**Question 4**

(a) Outline the main weaknesses of passwords which use hashed values, and how salting overcomes these problems. How might a hacking tool overcome the usage of salt? [Marks: 5]

(b) For the following salted password (128-bits of salt) outline the process that would be involved when the user logs-in, and how the password would be checked. [Marks: 5]:

$2a$06$NkYh0RCM8pNWPaYvRLgN9.LbJw4gcnWCOQYIom0P08UEZRQQjbfpy

* CSN11117/CSN11102 fake exam questions (2018). Any questions? Ask Bill (Slack: esecurityworkspace.slack.com).