Sample GCHQ Mathematics Applications Test

The applications paper contains questions designed to see how you can apply mathematical ideas that may perhaps be less familiar to you to more practical problems. Therefore, in some cases, answers will be more discursive than directly mathematical. As with the aptitude test, we are most interested in finding out what your particular areas of expertise are, so you are advised to read the whole paper before starting and focus on those questions that you can approach confidently. More credit is given for detailed answers to a smaller number of questions than for a larger number of partial attempts. Results from this paper will be combined with those from the accompanying aptitude test.

Question 1

A hash is a one-way (hard to invert) function H which can take an input of any size and produce a fixed-size output. Typical requirements for good hash functions are:

- Pre-image resistance: given $H(A)$, it is very difficult to find A.
- Second pre-image resistance: given A, it is very difficult to find $B \neq A$ such that $H(A) = H(B)$.
- Collision resistance: it is very difficult to find A, B, with $A \neq B$, such that $H(A) = H(B)$.

Alice wishes to vote in an upcoming election for which there are just two voting options. She can securely share a verification value with the voting authority in advance but will be voting over an untrustworthy channel.

Consider the following voting scheme based on a hash function H.

- Alice chooses two secret keys, s_1 and s_2.
- Alice generates the verification value $H(H(s_1)||H(s_2))$. This value is shared with the voting authority in advance of the election.
- If Alice wishes to vote 'yes' she sends $(s_1, H(s_2))$; if she wishes to vote 'no' she sends $(H(s_1), s_2)$.

Note that the symbol $||$ is used to denote the concatenation of two values.

a. i) How does the authority determine how Alice voted?
   ii) Why are the secret keys used in this voting scheme one-time use only?
   iii) Justify why an adversary cannot change Alice's vote.

b. Without increasing the number of secret keys nor increasing the size of the verification value, how can you generalise this scheme to voting for one of 8 options? Justify why an adversary still cannot change Alice's vote in your proposed scheme.

c. Discuss whether the property of second pre-image resistance is required in the choice of H for this voting scheme.

Question 2

Alice and Bob are good friends who lead busy lives and rarely get a chance to catch up in person. They wish to communicate over an insecure channel by securing their messages using a symmetric encryption algorithm. Eve intends to cause as much disruption as possible to communications between Alice and Bob, and in particular wishes to read the messages sent.

To start communications in an encrypted manner Alice sends Bob an unencrypted message over the channel. The message contains instructions on how they can encrypt their messages as follows:

- Generate a key, K_1, by cryptographically hashing the date.
- Use this key to encrypt a message before sending, or decrypt any messages received that day.

a. Explain any problems with this system. How may Eve read any encrypted messages?

Alice realises the error she made. She randomly creates a new symmetric key, K_2, for communications and meets with Bob to exchange the key. Both subsequently store the key on their devices which connect to the insecure channel.

b. Eve sees messages being exchanged between Alice and Bob. What can she do without knowledge of the new key, K_2? How could Eve obtain the new key?

Alice receives the same message from Bob multiple times. She suspects Eve is repeatedly sending an old message from Bob. She devises a replay protection mechanism. Within each encrypted message Alice and Bob send they include the time. When a message is received, the time is checked and if it falls within the last 30 seconds they trust the message is new.

c. What problems may occur in this new scheme? How may Eve disrupt things further? Suggest an improvement to this scheme, and explain how it mitigates these flaws.

Alice suspects that Eve may have discovered the encryption key, K_2. She is getting fed up of having to meet with Bob to exchange keys, and researches Public Key Cryptography.

During one of their infrequent meetings, Alice explains to Bob how they can use the RSA cryptography system to communicate securely. Alice generates integers N, e and d such that for any integer M, we have

$$M^{ed} \equiv M \bmod N$$

(Assume that all messages are represented as positive integers less than the RSA modulus, N.)

She publishes N and e, but keeps d private. If someone wishes to send a message to Alice, they compute and send $Z \equiv M^e \bmod N$. Alice can recover M by computing $M \equiv Z^d \bmod N$.

d. How can Alice prove to Bob that a message could only have been written by her? This process is known as signing the message.

Alice gives Bob her public key, N and e, on a USB flash drive. Unfortunately, Bob loses the flash drive on the train.

e. Does Alice need to generate a new key pair? Justify your answer.

After hearing of Bob's carelessness, Alice sends her public key to Bob in an unencrypted email.

f. Explain how Eve, with the ability to intercept and modify messages, could eavesdrop on their conversations without either of them knowing. Why would this not be a problem if Bob hadn't lost the flash drive?

Alice is pleased with her secure instant messaging system. She decides to develop it further and releases Alice Messenger System (AMS) to the other residents of her town. Word travels fast, and soon all the townsfolk are aware of Alice's system, and know her public key.

Alice's friend Charlie wants to use AMS. He generates a key pair and hands Alice his public key on a USB flash drive he found on the train. Alice signs the message "Charlie's public key is: [Public Key]". This message, along with its signature, is called a certificate, which she gives to Charlie.

g. Given that everyone trusts Alice, how can Charlie use his certificate to convince any resident that a message was written by him?

Alice would like to sign certificates for everyone in the town. She announces to everyone that she would like them to email her with their name and public key, and she will respond with a certificate.

h. Comment on the suitability of this method of certification. Suggest any faults and improvements.

Alice designed AMS so that when a character is typed, it is interpreted as an integer between 0 and 255, then RSA encrypted and sent to the recipient where it is immediately decrypted and displayed on their screen.

i. Suggest any faults and improvements for this scheme. How does this implementation affect an attacker's ability to recover a private key?

Question 3

Shannon Entropy is a measure of randomness of a discrete random variable X which can be defined as

$$H(X) = -\sum_{i=0}^{n} P(x_i)\log(P(x_i))$$

where each x_i is a possible value of X.

For the purposes of cryptographic investigation we consider X to be the output of a random number generator, with x_i denoting all the possible values it could take. By using logarithms in base 2, we can think of entropy in terms of bits. Some key properties of entropy include:

- Data can have less entropy than the total number of bits, but not more.
- Different length blocks of data can contain the same amount of entropy.
- Identical length blocks of data can contain different amounts of entropy.

This is a very important concept in cryptography, particularly when considering cryptographic keys and passwords. For instance, an 8 byte key, where the value of each byte is independent and uniformly distributed, can have up to 64 bits of entropy. We can see this by applying the above formula to each byte to obtain

$$-\sum_{i=0}^{255} P(i)\cdot\log(P(i)) = -256\cdot(\frac{1}{256}\cdot\log_2 2^{-8}) = 8,$$

and then summing this over all the bytes.

However, if those bytes can only take the values 1 or 0, then we see that the entropy in the key cannot exceed 8 bits, since $-(\frac{1}{2}\cdot\log_2 2^{-1} + \frac{1}{2}\cdot\log_2 2^{-1}) + 0 + 0 + \ldots) = 1$

a. (i) If each byte in this 64 bit key has exactly 1 non-zero bit, and this bit can take any position in the byte with equal probability, how much entropy is there in the entire key?

(ii) The entropy in a stream is preserved under a 1-to-1 map. State a 1-to-1 map from the 64 bit space in (i) such that the resultant stream will have maximal entropy.

If we take 2 independent streams A and B of 64 bits and concatenate them, then $A||B$ has the sum of the entropy in A and B.

b. (i) If A and B are dependent, in that $B = F(A)$, where F is an 1-to-1 map, how much entropy does $A||B$ have?

(ii) If F is not invertible (say a cryptographic hash) how would that affect your answer? Justify your reasoning briefly.

In a cryptographic system, in order for a key to have full entropy, random data is taken from a random source and pooled into a much larger store than the size of the key to be generated. When a key is required, the large quantity of data is processed and compacted to produce the key using a non invertible function. This ensures that if there is relatively little entropy in the collected data, the resultant key will have higher entropy.

Consider the following example system. We have a timer which starts at zero, and has accuracy to 1 microsecond (10^{-6} seconds). We have a perfect random source which will reset the timer to 0 at most 1 second after the previous reset. The value of the timer at reset is read off as a 64 bit value and added to our entropy pool. When a 64 bit key is needed, all the 64 bit timer values in the entropy pool are bitwise XORed.

Bitwise XORing 2 data streams will result in a stream with greater than or equal entropy to both the input streams.

c. i) How much entropy is in each 64 bit timer value (use the fact that $2^{10} \approx 1000$ to help with your estimate).

ii) How many of these values (assuming independence) would need to be concatenated to produce a value with ≥ 64 bits of entropy.

iii) What is wrong with the suggested XOR combining function? Suggest a better function to produce a high entropy 64 bit output.

Question 4

SecretStore is an online cloud computing service which is designed to allow users to store their files, encrypted, on its servers, without the server administrators being able to decrypt them.

To set up SecretStore, the user must provide a username and password of their choice on the login page. The SecretStore software computes a hash of their password, then sends the result and the username to the SecretStore server which saves them. When the user needs to prove who they are to the server, they only send this hash; this way, SecretStore never needs to know their actual password.

If a user supplies the wrong password, the server sends a message back telling the user they have entered the incorrect password. The SecretStore software keeps track of how many times this has happened that day in a file stored on the user's computer; once a person has failed to log in three times, it refuses to send further attempts to the server until the next day (at which point it deletes the log file).

a. How might an attacker gain access to one of the users' accounts, if they can get to the login page but have no direct access to the server, any users' machine or the communications between the two?

b. What measures could SecretStore employ to protect against such attempts?

Once a user has logged in, she can download any of her encrypted files. The SecretStore software on her computer uses the hash of her password as a key to encrypt or decrypt files. This way, the server never sees the unencrypted files, and they are never sent over the internet.

c. How could an attacker gain or deny access to or modify the user's files if:

   i) He is able to intercept messages between the user and the server?

   ii) He had access to the user's computer?

d. What measures might the user or the server employ to prevent or mitigate against these attacks?

To let users share files with each other, SecretStore decide to implement a new system using public key cryptography. When setting up SecretStore, each user generates a public and a private key. Anyone can encrypt a file with the public key, but only the person who knows the private key can then decrypt it.

To share files with others, one user generates a secret File Key (FK) by hashing the number of milliseconds since midnight. The user then encrypts a copy of the FK with each sharer's public key, and sends the results to the server. When one sharer wants to view or upload a file, they download the version of the FK which was encrypted with their public key, decrypt it using their private key, then use that to encrypt or decrypt the files themselves. The FK replaces the password hash for encrypting these files.

e. Suggest a weakness in the encryption used in the sharing group which might allow an attacker to decrypt an encrypted file. How might you improve the system to remove this weakness?

f. If an attacker had the ability to interfere with the set-up of the sharing group, could he use this capability to ensure that he could later read the files which the users encrypt?

g. Are there any further measures SecretStore could take to improve their security?

Question 5

Suppose you are trying to detect illicit activity on a system. You are able to observe vast amounts of data – more than you could possibly store and process offline. For simplicity, assume that the data come from some univariate process, and that illicit behaviour is represented as a sudden and persistent (at least in the short-term) change in this process. You are responsible for developing analytics / statistical tests to identify such changes.

a. Suggest what properties these analytics need to have to be of practical use.

One commonly used changepoint detection test is the CUSUM algorithm. It performs optimally when the pre-change mean and variance of the process are known, μ_1 and σ_1 is required. Then the statistic S_j is defined as:

$$\begin{aligned} S_0 &= \mu_1 \\ S_j &= \max 0, S_{j-1} + x_j - k\mu_1, \end{aligned}$$

where x_j is the j^{th} observation of the process, k is a control parameter, tuned depending on the magnitude of the change one is trying to detect. If S_j exceeds some threshold $h\sigma_1$ (where h is another control parameter), then a change is declared.

b. In a streaming environment, where we only store the most recent few observations and the underlying process may drift over time, suggest reasons why the CUSUM algorithm as described above may not be suitable.

The most recent observations in a data stream tend to give us the most accurate view of its current behaviour. So, we want to use adaptive estimation techniques. Consider a set of N observations, $x_1, x_2, \ldots, x_N$. Recall that the usual arithmetic mean is given by

$$\bar{x}_N = \frac{1}{N} \sum_{k=1}^{N} x_k.$$

Adaptive estimation introduces an exponential *forgetting factor*, $\lambda \in [0, 1]$. The **forgetting factor mean** is defined as:

$$\bar{x}_{N,\lambda} = \frac{1}{w_{N,\lambda}} \sum_{k=1}^{N} \lambda^{N-k} x_k,$$

where $w_{N,\lambda} = \sum_{k=1}^{N} \lambda^{N-k}$.

c. Describe how $\bar{x}_\lambda$ behaves for each of the 3 cases $\lambda = 0, \lambda = 1, 0 < \lambda < 1$.

d. i) Assume that $x_1, x_2, \ldots, x_N$ are sampled from random variables $X_1, X_2 \ldots, X_N$, all of which have expected value μ. Calculate the expected value, $E[\bar{X}_{N,\lambda}]$, and variance $Var[\bar{X}_{N,\lambda}]$, of the forgetting factor mean.

ii) Assuming that the X_i are normally distributed with mean μ and variance σ^2. Describe how you would construct a test to identify a change in the mean of the process.

In the streaming data context, we need to be able to update the value of $\bar{x}_{N,\lambda}$ efficiently when a new data point arrives.

e. Given $\lambda, w_{N,\lambda}, \bar{x}_{N,\lambda}$, suggest an efficient sequential updating scheme for calculating $\bar{x}_{N+1,\lambda}, \bar{x}_{N+2,\lambda}, \ldots$ as observations $x_{N+1}, x_{N+2}, \ldots$ arrive.