1. A company wants to get tags for their Data. Their Data are Big but differ in only a few blocks. They have consulted with you to help them with this project. Which Mac algorithm do you suggest to them. Why? Explain in detail. (6 points)

**PMAC (Parallelizable Mac)** seems to be a very good algorithm to use for this project. Other methods of creating tags like the CBC-MAC, NMAC, and the HMAC algorithms, require a computational pipeline that requires computation based on the entirety of the message starting from the beginning. Any additions to a base message have to be recalculated from the beginning, consuming time equal to the computation of the base message added to the computation of the addition. If we have two pieces of data that differ in only a few blocks, we can precompute the blocks of the cascade that the two pieces of data have in common, and simply make modifications where they are different. This gives us a computation time for the additional data as the only computation required, as opposed to the pipelines which suffer a full recompute each time. PMAC, as the name implies **can also be run in parallel** in addition to saving existing known code blocks, further speeding up the process. In addition to all these criteria, it offers a high degree of security, on the same level as the others in it's cohort, making it the clear choice for this use case.

2. Your company wants to encrypt Big Data and for this you need to use a block cipher mode of operation. Two well-known such modes are CBC with random IV and Randomized Counter Mode (rand ctr-mode). Describe these two modes. Which one you choose? Why? Describe in detail all reasons. (6 points)

CBC with random IV is an algorithm that chooses an Initialization Vector (IV) truly at random from the possible encrypted message space. The IV is used with the first piece of the ciphertext to encrypt an entire message, starting at the first block, with a chain of XORs and encryption steps.

CBC with random IV is a mode which does not make use of a ‘Button’ (⟘) to validate a ciphertext produced by the CBC processes. Because a button is never used for cipher validation, any adversary can easily produce a non-negligible advantage through the use of any message.

CBC with a random IV is susceptible to a very simple tampering attack. An adversary can modify an IV in the clear, change the destination port of some information to be encrypted to the port of desire (such as Email, SMTP 25), and have all the unencrypted data sent directly to his/her email address with no issues. If data security is of concern and not simply integrity, this approach is very poor due to how easily it can be exploited.

Due to these two reasons, CBC with random IV does not possess authenticated encryption, and would be a poor choice.

CBC with randomized counter mode is occurs when an Initialization Vector (IV) is fixed truly at random, and then each message block is encrypted using the key and a portion of the IV added to an index (m[0] encrypted with IV + 0, m[1] encrypted with IV + 1), so on and so forth.

The much better choice **is CBC with randomized counter mode** in every way, for the opposite of all the above reasons. CBC with randomized counter mode can make use of a Button in validation, is not susceptible to easy tampering attacks, and is parallelizable, giving it a strong edge in both speed and security. It is the one I would choose in a heartbeat.

3. Describe the Wegman-Carter MAC in detail. Mention one application of this MAC discussed in class. (6 points)

The Wegman-Carter MAC is a MAC creation algorithm that extends the usage of the One-time MAC to allow for the creation of tags for many messages while using the exact same keys. The Wegman-Carter MAC is also quite simple in terms of usage/design. First, the algorithm selects a truly random number r (also called a nonce), and a key string, k1.

k1 and r are inserted into a perfectly security Pseudo-random function (PRF) as arguments.

We use the One-time MAC algorithm with another key string (k2) and our desired message m.

After we do both of these operations, we XOR together the output of our PRF and our One-time MAC to get a value q. Our resultant tag will be represented as (r, q), which is the perfectly random nonce combined with the result of our XOR.

There is a notably difficult problem in Cryptography that relies on the distribution of keys in a secure and discreet manner. Prior to the talk of quantum computers, problems like this were not as big of a deal, but in the modern day, this has become a much hotter topic. There is a technique known as Quantum Key Distribution (QKD), which is a provably secure way to distribute keys in a secure means, independent of the existence of quantum-level power or not.

However, this technique still requires the authentication of the keys being sent in question, to ensure that they have maintained integrity. The notable application that we discussed in class is that this **authentication step is conducted by the Wegman-Carter MAC** more often than not.

4. Is SSL/TLS 1.0 CPA-secure? Explain. (6 points)

Recall that a CPA secure system means that the system is not susceptible to CPA related attacks (Chosen Plaintext Attacks), in which the system does not provide the adversary a significant advantage based on their choice of a plaintext. Additionally, a system in which the attacker can predict the IV is also not CPA-secure.

In SSL/TLS 1.0, there was a bug in which the (N - 1)th record of the IV was equal to that of the Nth record of the IV. That meant if a given adversary encrypted a message m, and sent it to the challenger, the challenger and adversary could pass back and forth until the Nth record of the IV. Upon hitting this block, the adversary could predict the Nth record of the IV, because it would be equal to the (N-1)th record of the IV provided to the adversary a turn before. Thus, by definition of **being able to predict the IV, SSL/TLS 1.0 was not CPA-secure.**

5. You are a developer and would like to secure your data against active attackers, that is, you would like to have both data authenticity (integrity) and confidentiality for your project. You know that you have to find a way for encryption and a way for Macing and then combine them. You know there are various ways and standards for achieving this. However, describe (in detail) an authenticated encryption which avoids using separate encryption and authentication systems. This mode needs only one block cipher operation per block of message, is parallelizable and so much faster than other authenticated encryption standards. (6 points)

In terms of an algorithm that matches this description perfectly, you have the Offset Codeblock (OCB). The OCB combines encryption and authentication together and is perfectly parallelizable, making it the most secure and efficient authenticated encryption method available.

The OCB works as follows:

A given message is broken into several evenly divided blocks (as evenly divided as they can be). For each message block, m[0] through M[L], the message is XORed with a nonce value provided by the key (k) combined with a counter, based upon the length of the message. The result of this XOR is passed through a given encryption block cipher making use of the same key. The result of this encryption is again XOR’ed with the same nonce-key pair to produce ciphertexts c[0] through c[L]. This is how the **encryption** step of the authentication-encryption method takes place.

After this is done, or parallelized at the exact same time, a checksum is provided which is XOR’ed with the nonce-key pair used for the 0th block of the message. The result of this is passed into the same given encryption block cipher with the key, k, and the result of that is XOR’ed with a known authentication bitstring to produce c[L+1]. This is how the **authentication** step of the authentication-encryption method takes place.

With both encryption and authentication accounted for and the whole process possessing the ability to be parallelizable, and possessing the same block cipher operation per block of message, this is easily the superior choice for the project in question.