CS 5435 – Computer Security

Homework 3

Name: David Kogan (dk448), Andrew Palmer (ajp294) *Fall 2019*

Exercise 1.5

1. Galois/Counter Mode (GCM) is an authenticated encryption algorithm, so it can detect improperly-constructed cipher texts and refuse to decrypt them. Thus, by modifying the cipher text and sending it to be decrypted, GCM would detect the unauthentic value. Additionally, GCM does not use padding, which padding oracle attacks depend on.

2. HTTPS was designed to ensure protection of the privacy and integrity of the exchange data[1]. However, enabling HTTPS for the entire payment site is not sufficient in preventing our mauling or padding oracle attacks. These attacks work by abusing drawbacks in Cipher Block Chaining (CBC) mode encryption algorithm used in TLS. Thus, even with HTTPS encrypting the data, if a flawed encryption mode is chosen, the web server will be subject to cookie mauling attacks, such as our maul and padding oracle implementations.

   TLS 1.3 was designed to no longer support problematic encryption modes[2], such as CBC. Thus, a better protection against our attacks is to ensure the web server has modern TLS, instead of just enabling HTTPS across the whole web server.

Exercise 2.3

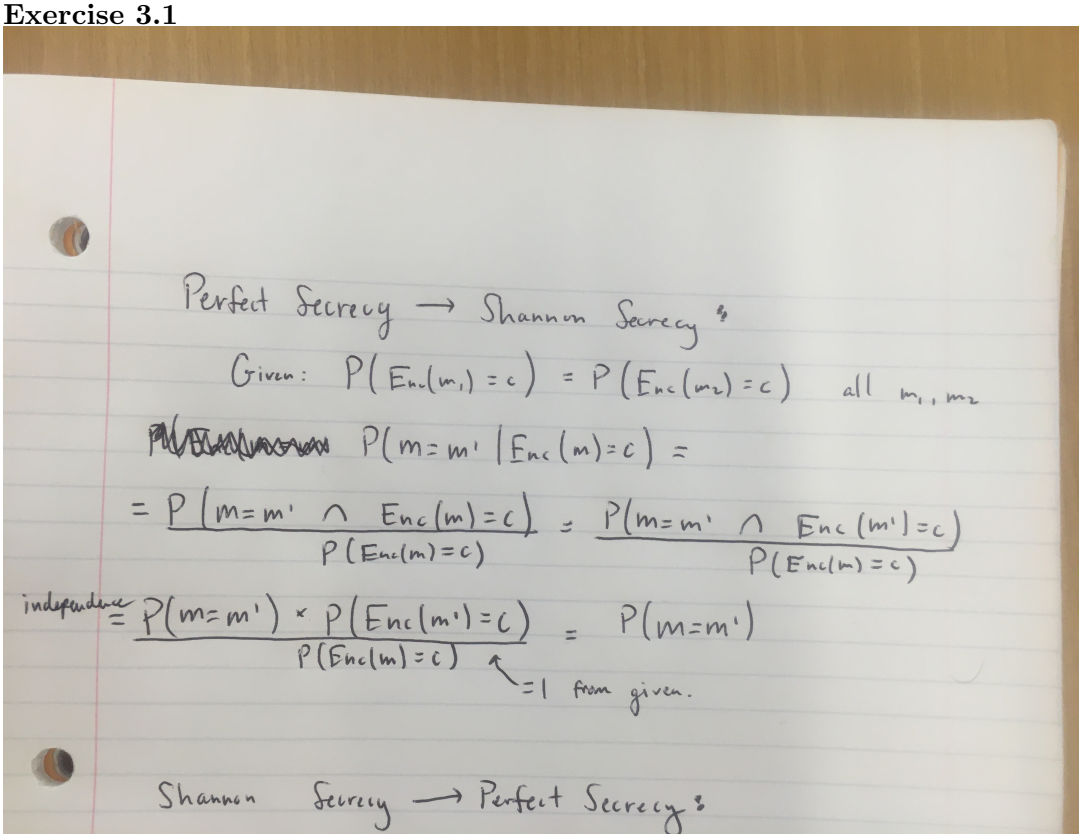
In theory, if the web server samples a fresh random SipHash key at every startup, the web server could prevent the hash DoS attack. If the attacker has no knowledge of the key, then it will be intractable to determine the colliding strings. Being a 16 bit key, and assuming the number of buckets is known, the attacker could attempt to brute force the attack by determining a large number of colliding strings for each key. However, this takes a large number of compute resources, and if the web server key is sampled sufficiently often, the attack can be preventable.
As an example, if determining 1000 collisions per key takes 20 minutes, and assuming 1000 collisions is enough for a DoS attack, the brute force attack would take 65536 * 20 = 1310720 minutes, which is over 900 days of compute time. Assuming the web server resamples a new key more often than 900 days, a randomly sampled SipHash key is an effective countermeasure for this web server.

Exercise 2.5

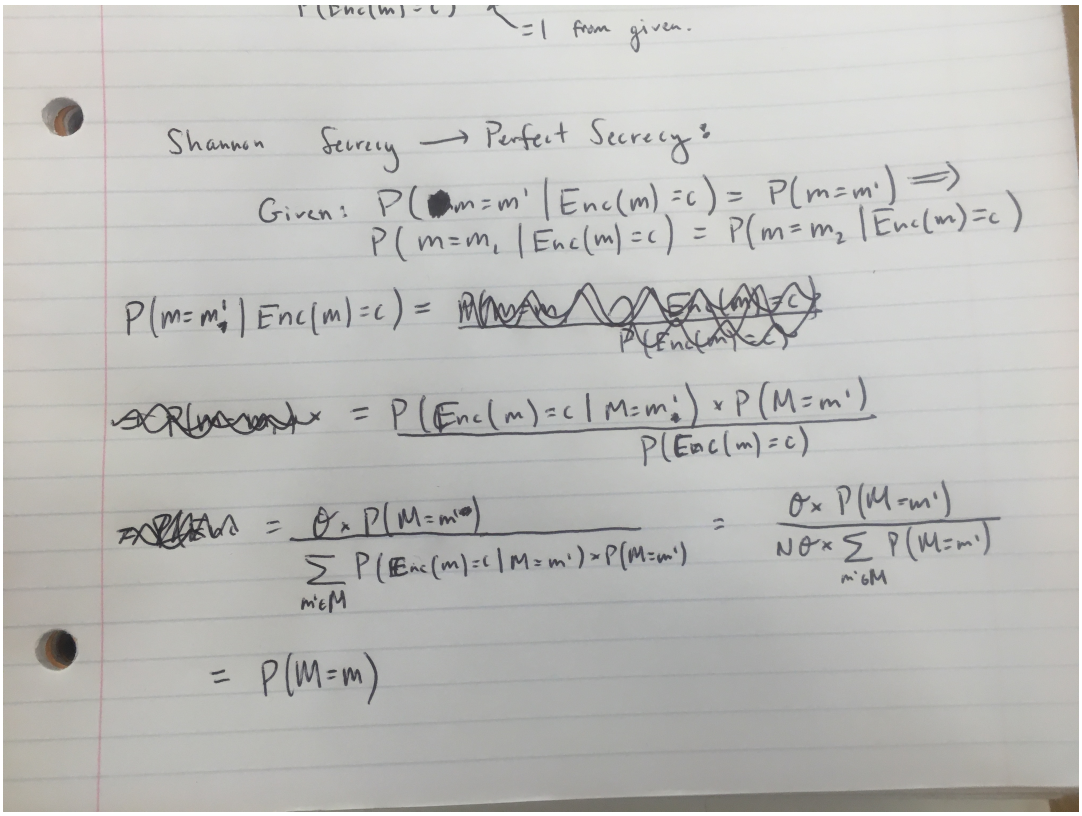
Proof of work is an effective countermeasure for flooding denial-of-service attacks. In order to generate a PoW solution, the attacker will have to use additional computational resources. This will significantly limit the number of requests the attacker can make, in turn making it more difficult to flood the victim's machine.

However, for our hash denial-of-service attacks, proof of work is not effective because the attack requires only one request. We are abusing the architecture of the web server, which allows us to submit all of our hash collision strings in one HTTP request. Therefore, the usual benefits of proof of work are not applicable here and would not be an effective countermeasure. If the web server was restructured to only allow one parameter at a time, then proof of work would provide some potential benefit as a countermeasure.

Exercise 3.1



Exercise 3.2



References

[1] https://en.wikipedia.org/wiki/HTTPS

[2] https://searchsecurity.techtarget.com/answer/How-concerned-should-I-be-about-a-padding-oracle-attack