CS 5435 − Computer Security
Homework 3
Contributors: sg2452| yv44

PART-1
Cookie Malleability and Padding Oracles

1.5

Why does using GCM prevent mauling and padding oracle attacks?

AES-CBC is vulnerable to padding oracle attacks, which exploit the tendency of block ciphers to add arbitrary values onto the end of the last block in a sequence in order to meet the specified block size.
GCM combines Galois field multiplication with the counter mode of operation for block ciphers. The counter mode of operation is designed to turn block ciphers into stream ciphers, where each block is encrypted with a pseudorandom value from a “keystream”. This concept achieves this by using successive values of an incrementing “counter” such that every block is encrypted with a unique value that is unlikely to reoccur.

The AES-GCM encryption takes as input a **message** + **encryption key** and produces as output a set of values: { **ciphertext** + **nonce** + **authTag** }.

- The **ciphertext** is the encrypted message.
- The **nonce** is the randomly generated initial vector (IV) for the GCM construction.
- The **authTag** is the message authentication code (MAC) calculated during the encryption.

The chances of an attacker, not knowing the key, being able to change the ciphertext in a consistent way is believed to be difficult, 1 in 2^(key size) chance, so changing the message becomes *less* efficient than simply trying every possible key.

For each attack above, explain whether enabling HTTPS for the entire payment site (as opposed to just the login page) prevents the attack if no other countermeasure is applied.

Enabling HTTPS for the entire payment site would not prevent the attack because HTTPS only secures the connection between server and client but not the actual message in it. So the attacker could still get the HTTP cookies over HTTPS and perform the mauling and padding oracle attack.

PART-2
Denial of Service (DoS Attacks)

2.3
This attack relies on the attacker having knowledge of the SipHash key. Assuming SipHash is a PRF, does sampling a fresh random SipHash key for the hash table every time the web server is started prevent this attack? Why or why not?

Siphash is a PRF which most importantly takes two values: a "string to hash" and a "crypto key": siphash(string, crypto_key) --> number.
The idea is to generate this "crypto_key" randomly on each program execution, to make sure the attacker can't predict it. But in our case since the key is known to the attacker along with the Hash function It is possible to flood the server with a crafted requests that, for example, all headers will end up with precisely the same hash value. However if the attacker is not aware of the key it would prevent the attack because without the key you wouldn't be able to generate enough values that collide.

2.5
One suggested countermeasure to denial-of-service attacks of is proof of work: forcing clients to perform some computational work and including a proof in the request. Is this an effective countermeasure? Why or why not?
Ans: No, this won't be an effective countermeasure since validating the proof would be okay if the number of requests were substantially small. But if an adversary sends a large number of requests at a time, validating every request to check for the proof would take up computational power and would in turn bring down the system.

PART-3
Definitions of Secrecy
(for 3.1 and 3.2 please refer below attached pages)

Answer to Question 3.1

We define perfect secrecy as, regardless of any prior information the attacker has about the plaintext, the ciphertext should leak no additional info about the PT.

To prove: $Pr[M=m \mid C=c] = Pr[M=m]$

Using the one time pad encryption scheme, we try to construct perfect secrecy.

let $M = \{0,1\}^n$ → set of all 'n' bit strings

Gen: chooses a uniform key $K \in \{0,1\}^n$.

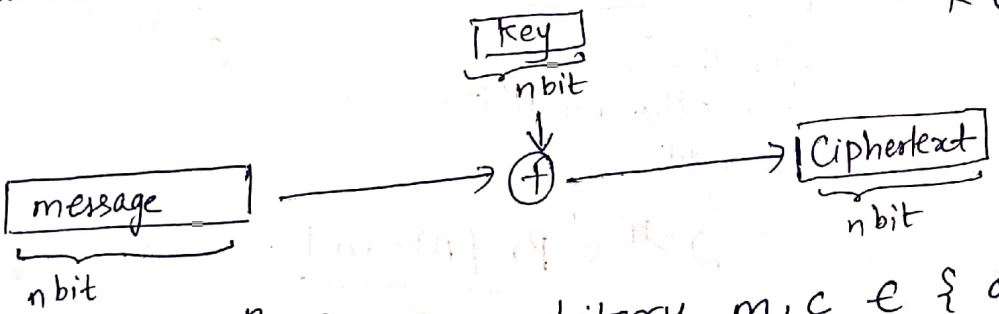
Probability of picking any one key from the key space picked by generator $= \frac{1}{2^n}$

$Enc_K(m) = K \oplus m$

$Dec_K(C) = K \oplus C$

To validate the above, $Dec_K(Enc_K(m)) = K \oplus (K \oplus m)$

$= K \oplus K \oplus m = m$



let $M = \{0,1\}^n$ & any arbitary $m, c \in \{0,1\}^n$, where m, c are random variables which take some value in their respective distribution.

Using Bayes theorem,

$$Pr[M=m \mid C=c] = Pr[C=c \mid M=m] \times \frac{Pr[M=m]}{Pr[C=c]}$$

According to law of total probability,

$$Pr[C=c] = \sum_{m'} Pr[C=c \mid M=m'] \times Pr[M=m']$$

summation over all possible messages m'

$$\sum_{m'} \underbrace{Pr[K = m' \oplus c]} \cdot Pr[M = m']$$

here we rely on an assumption specific to OTP

$$\sum_{m'} \frac{1}{2^n} \times Pr[M = m']$$

$m' \oplus (m' \oplus c)$
$m' \oplus m' \oplus c$
$= c$

$$\sum_{m'} 2^{-n} \times Pr[M = m'] \Longrightarrow (1)$$

$Pr[C = c] = 2^{-n}$

Going back to our equation,

$$Pr[M = m \mid C = c] = Pr[C = c \mid M = m] \times \frac{Pr[M = m]}{Pr[C = c]}$$

$$= \underbrace{Pr[K = m \oplus c]} \times \frac{Pr[M = m]}{2^{-n}}$$

for the 'm' we actually wish to decrypt

$$= 2^{-n} \times \frac{Pr[M = m]}{2^{-n}}$$

$$\boxed{Pr[M = m \mid C = c] = Pr[M = m]}$$

Thus One Time Pad holds perfect secrecy. conditioned on the fact that, observing cipher text $= C$, is exactly equal to the Apriori probability that the message was equal to M.

$|m_1| = |m_2| = n$

Now since OTP holds perfect secrecy we have,

$$Pr[M=m \mid C=c] = Pr[M=m] \longrightarrow (1)$$

Assume $m = m_1$ $Pr[m_1] > 0$,

$$Pr[m=m_1 \mid Enc_k(m)=c] = \frac{Pr[m=m_1]\, Pr[Enc_k(m_1)=c]}{Pr[Enc_k(m)=c]}$$

combining,

$$Pr[M=m_1] = \frac{Pr[m=m_1]\, Pr[Enc_k(m_1)=c]}{Pr[Enc_k(m)=c]}$$

$$Pr[Enc_k(m)=c] = Pr[Enc_k(m_1)=c] \quad \text{for } m_1 \quad \text{↳ (2)}$$

Similarly for m_2

$$Pr[Enc_k(m)=c] = Pr[Enc_k(m_2)=c] \quad \text{↳ (3)}$$

from (2) & (3)

$$Pr[Enc_k(m_2)=c] = Pr[Enc_k(m_1)=c]$$

3.2

Assume shannon secrecy & let m_1 & c be arbitary

to prove:

$$\Pr_{K,m_2}[m_1 = m_2 \mid Enc_K(m_2) = c] = \Pr_{m_2}[m_1 = m_2]$$

$$\Pr[m_1 = m_2 \mid Enc_K(m_2) = c] = \frac{\Pr[m_1 = m_2 \cap Enc_K(m_2) = c]}{\Pr[Enc_K(m_2) = c]}$$

since m_1 is fixed we can substitute m_1 for m_2 in $Enc_K(m_2)$

$$\Pr[m_1 = m_2 \mid Enc_K(m_2) = c] = \frac{\Pr[m_1 = m_2]\Pr[Enc_K(m_1) = c]}{\Pr[Enc_K(m_2) = c]} \hookrightarrow (1)$$

But,

$$\Pr[Enc_K(m_2) = c] = \sum_{m \in M} \Pr[m = m_2] \Pr[Enc_K(m) = c]$$

By assumption of shannon secrecy we have,

$$\Pr[Enc(K, m_1) = c] = \Pr[Enc(K, m_2) = c]$$

$$\Pr[Enc_K(m_2) = c] = \Pr[Enc_K(m_1) = c] \sum_{m \in M} \Pr[m = m_2]$$

$$\Pr[Enc_K(m_2) = c] = \Pr[Enc_K(m_1) = c]$$

Substituting in 1

$$\Pr[m_1 = m_2 \mid Enc_K(m_2) = c] = \Pr[m_1 = m_2]$$