**HW 5**

1. **Given:**

Table w/ 256 connection requests

Retry SYN-ACK packet 5x when no ACK in response  
every 30 seconds before purging

ie: initial send – wait 30 sec -send-wait 30 sec- send- wait 30- sec etc

6 SYN-ACK packets

TCP Syn packet = 40 bytes in size

**Solution to keep table full:** We can calculate the number of requests the attacker needs to send in order to keep the table full by using the following:

# of requests per second = num of requests in table/ attack rate

attack rate = # SYN-ACK packets \* Time it takes per packet

ie: # or requests per second = 256/(30\*6) = 1.42 to translate to per minute 1.42x60 = 85.33

**86 requests per minute or 1.42 per second to keep the table full**

Solution for Required Bandwidth:

Bandwidth = # requests per second \* TCP Syn packet size \* bits per byte

Bandwidth = 1.42 \* 40 \* \* = **454.4 bits per second**

2) a) Let us assume that n will only be one 0 bit. this would mean that every 1/10 numbers will be a hit for the correct answer.

With this in mind we can calculate the probability of this happening, let us use 20 tries as an example.

P = 1- (1-(1/10))^20 = 0.878 or around 88%

b) Here n will affect security in a drastic way, let us use 2 zeroes instead of one like we did last time to calculate probability with the same number of tries

P = 1- (1-(1/100))^20 = 0.1821 or around 18%

with 3 zeros

P = 1- (1-(1/1000))^20 = 0.0198 or around 2%

**As the difficulty of the challenge increases so does the security.**

c) The number of calculations is solely dependent on the number of zeros the n is I set to. The probability will never hit 100% but we can get closer and closer to it. With the case of n= 10 we can calculate that 99% probability of finding the number happens after around 50 iterations

P = 1- (1-(1/10))^50 = 0.995

But with n = 1000 that number increases dramatically, jumping to around 5,300 iterations

P = 1- (1-(1/1000))^5300 = 0.995

d) I believe the question to this answer depends entirely on the capabilities of the system as well as the amount of traffic the system gets. But ideally yes k should be random for each user.

Since it is used to generate the hash for each client, randomizing it will allow for no collisions to occur if the call for it is used a large amount of times. However, this might put some strain on a low budget system that receives little traffic and has less concerns of collisions.

3) One of the more high-profile cases to have occurred ion the recent years was the DNS poisoning attack on a Amazon’s DNS servers and pretended to be a crypto currency wallet, MyEatherWallet.com. This resulted in about $150,000 of stolen cryptocurrency assets from unwitting consumers. So what could have been done to prevent this?

First up as a consumer it is possible to guard yourself from going on redirected websites or at the very least stop yourself from falling prey to the attacker. This can be done by (a) keeping your anti-virus up to date as well as pay attention to any and all browser warnings that might pop up. If we go back to the MyEatherWallet.com situation, according to official reports those who had their assets stolen had to have clicked through the warning their browser gave in order to access the phishing website. Furthermore, it would be a good idea to set your browser to ask permission before downloading any files, it can be tedious, but it might save you in the long run.

But what could have been done to prevent such an attack server side? Let us take a step back and look at what exactly makes the DNS poisoning so dangerous, in short terms: its ability to rapidly spread over peered networks with the issues being resolved only after ALL poisoned DNS catches have been cleared. This leads to the idea of having your DNS server have a very small amount of communication with other DNS servers as a solution to the problem of rapid spread. Next up it is also important to make sure that the only thing being run is the request currently being processed, if additional processes are being run it would open up the door for more malicious background meddling. Last but not least make sure to keep the DNS server up to date. As time goes on new vulnerabilities get inevitably discovered and tern subsequently patched just to have the cycle repeat indefinitely.

\

4) a) First of all, when a DoS attack is detected it is necessary to establish what kind of attack it is. After the analysis is complete and the type of attack has been determined it is time to apply appropriate filters onto the ISP or their router to block out the attack packets. Last step is optional but highly recommended, in it you would attempt to trace the packets in an attempt to identify their source to hopefully stop potential future attacks.

b) One of the ways to track the DoS attack is by contacting the ISP to trace where these packets have come from. This gets exponentially difficult if the attacker is using a spoofed address. Unfortunately, no packets are easy to trace, all of them require significant manpower to trace.

c) Yes, DoS attacks will still be a large concern and further countermeasures will have to be developed. No matter how advanced the technology gets the system will still be prone to be overloaded by mass connection of “real” connections, which may or may not be utilized by the attacker. To combat such events providers can limit the bandwidth usage to a fraction of what is available. While this approach will detract from the experience of the end user, it can prevent a complete shutdown of the system.