Final Exam
Gunnar Yonker

1.

The TCP 3-way handshake works as the following: Client -> Server, Client <- ACK/SYN – Server, Client -ACK-> Server. The main reason for having a 3-way handshake is that it allows for both parties to send data over the connection. TCP is a bi-directional communication and for the data to travel between the two parties in a 3-way handshake both parties establish a starting ISN with the other party. Once that is established and the other party knows the ISN, both sides of the communication can send that data across. If there is a 2-way handshake, then one party would establish the ISN, and the receiver party would acknowledge that. This would be inefficient because it only allows one party to send data to the other party, whereas with a 3-way handshake both parties are able to send data to the other party. An example of this being inefficient would be if you connected to a server to send data using a two-way handshake. You could send the data to the server, but in order for the server to send the data back to you, the server would have to initiate a new 2-way handshake. If a 3-way handshake was used in this instance, then the user would be able to send the data to the server and the server could send data back to the user.

2.

SYN cookies are used mainly to prevent a SYN Flood attack on a victim system. This is done by having the SYN cookie reply to the TCP SYN requests with SYN ACKs without establishing a connection. By doing this, it prevents the SYN packets from remaining open and hanging up the server's resources thus preventing the SYN Flood attack. A one-way hash function is a fixed-length binary sequence that is difficult to generate the original string from the hash. The SYN cookie uses some information from the client's packet and some information from the server to crease a random ISN. The SYN cookie is then able to compare the sequence number received with some server information to see if it is a valid ACK number. If the number is valid a connection is established, if it is invalid the connection is refused. A one-way hash function is used to determine if the SYN cookie is valid and since a one-way hash function is very difficult to generate the original string from, it would then mean that the SYN cookie is valid. When the SYN+ACK packet returns to the client, the server can subtract 1 from the acknowledgement number to reveal the original SYN cookie that was sent to the client. If the SYN cookie matches and is valid, then the connection will be established. The one-way hash needs to be used like a secret key onto the authorization tag of the packet to ensure that the packets are valid. If one-way hash was not used with SYN cookies, the original string could most likely be used and then spoofed back in the packet which would make SYN cookies ineffective as the SYN Flood attack would still take place because the system would think that the SYN cookie matched and was valid. The one-way hash functions create a unique value that represents the state of the connection request and is included in the server's response to the client's request. Then when the client requests to the server to establish the connection, it needs to use that same value for the connection to be valid and established. If the request does not contain the one-way hash value, then the connection is refused. If one-way hash is not used, then the SYN cookie would be ineffective because the attacker could send a forged SYN cookie value that could not be verified and would lead to a DoS attack.

3.

Since the basic firewall is installed on Host B's internet gateway, a direct attack from outside of the network would not work to cause a denial of service on Host B. One attack that the attacker could attempt is to flood the ssh server with a large amount of traffic through connections by use of a botnet.

This would generate a high volume of traffic through the ssh server and could cause a denial of service to legitimate users trying to access the internet through the ssh server. If the attacker was able to use Host A and Host C to flood Host B with a large amount of traffic, this could overwhelm the ssh server like outlined earlier. If the attacker was able to launch a phishing attack on Host A or B to gain access to their user account this could lead to the attacker being able to infect that host with malware that takes control of Host A or C to launch a DoS attack against Host B by flooding that host with traffic that makes the service unavailable to valid connections.

4.

If the client host was trying to send a packet to a blocked IP address through the firewall, they would be able to use their VPN server (130.16.10.2) to evade that firewall. A secure tunnel would be established from the client host (212.13.14.16) to the VPN server (130.15.10.2). The client host would then have their packet created which would have the original destination IP of 156.10.10.2 but would be sent first through the tunnel to the VPN, the VPN server would then forward the packet to the destination. The response would then first go to the VPN server which would then forward the response packet back to the client host. This would bypass the firewall rule blocking the packet being sent to the destination. SSL tunneling takes place at the transport layer of the OSI model. The packet will originate at the client host, 212.13.14.16 and will then travel to the VPN server, 156.10.10.2 through the VPN tunnel that was established between the VPN server and the client host. The packet would then travel to it’s destination 156.10.10.2, bypassing the firewall at the client’s gateway at 212.13.14.1. Now adding that pathway taking the OSI model into consideration. The client sends a request to the VPN when trying to access the destination of 156.10.10.2, then the VPN connects to the VPN server 130.16.10.2 establishing a connection at the transport layer. The VPN will create a new packet with the original destination request in the payload with the destination of the packet going to the VPN server at 130.16.10.2. The packet will then go through the application layer, to the transport layer, to the network layer where it routes through the gateway at 212.13.14.1 and connecting to the internet. The VPN server would then decapsulate the original client’s request and connect to the web server at 156.10.10.2 at the transport layer. The response would then follow a similar return by using the VPN tunnel. The relevant hops would be:

Client host(212.13.14.16) -> Client gateway(212.13.14.1) -> Internet(156.10.10.2) -> VPN Server(130.16.10.2)

VPN server (130.16.10.2) -> Internet(156.10.10.2) -> Client gateway(212.13.14.1) -> Client host(212.13.14.16)

5.

Fragment	**ID**	**IP Header Length**	**Fragmentation Flag**	**Offset**	**Upper layer protocol**	**Relevant data in the payload**
1	6200	20	1	0	TCP	976
2	6200	20	1	122	TCP	976
3	6200	20	0	244	TCP	28

Final Exam
Gunnar Yonker

6.

A DNS cache poisoning attack allows an attacker to replace DNS data with data that will redirect the victim to a malicious website of the attacker's choosing. DNS cache poisoning is when the DNS cache is injected with an entry that it accepts as valid, but is really created by the attacker. The DNS cache poisoning can also be set to last for a fixed time frame using the TTL so that the attacker could poison the cache for a certain amount of time before it would refresh and the attacker would hope to go unnoticed. A DNS cache poisoning attack starts with the victim user requesting an entry from the DNS server, if the DNS server does not have that entry available for the user it will forward the look up to the root authoritative server of the top level domain. While this is taking place, the attacker will inject their own malicious entry into the local DNS server before the reply from the root authoritative server is delivered to the user. If the attackers malicious DNS entry reaches the victim user before the root authoritative servers reply then it will be cached into the DNS cache until the TTL expires or the DNS cache is cleared. At this point the DNS cache is poisoned and when the user goes to access that webserver again the DNS cache will load the attacker's DNS record leading the user to a malicious website that they did not intend to visit. This attack is less likely to succeed if the attacker is further away from the user because the attacker's DNS entry would take more time to reach the user. If the attacker's DNS entry doesn't reach the user before the root authoritative DNS reply, the attack will not be successful. The closer the attacker, the faster their entry injection can be in an attempt to reach the user before the real valid DNS response.

7.

An event counter would count the number of commands or actions to see if there is an unusually high frequency of commands which would signal a possible attack. A resource measure is a quantitative list of elements that give the amount used for some resources such as the CPU time. If there is an abnormal of resources being used it could indicate that there is a malicious program running The Markov Process Model applies to the event counters and it regards each even as a state variable, then uses a state transition matrix to characterize the transition frequencies between states. This can then be used to detect anomalies by looking at its probability which is determined by the previous state and if the transition matrix is too low. An example of this would be if you were looking at transitions between commands where the sequence was important. This means that the model can be applied to the event counter where the number of events of a certain action or command can be compared to the previous state and given a probability of the event being an anomaly. For example, a user could run a given command very infrequently such as accessing the root user through the sudo su command. Then on a given day the user accessed the sudo su command at a much higher frequency. The Markov Process Model could be applied to this situation to label this event as an anomaly due to the previous state of the command being infrequently used and the matrix between the previous state and current state being too low. That in conjunction with the probability can define this observation as abnormal and be labeled as an anomaly.

8.

HIDS would be able to detect a malware infection that is running a malicious process on the system. This would take place already past NIDS and would not be detected by NIDS. However, HIDS would be able to detect this process through monitoring processes on the system. NIDS is only able to monitor network traffic and is unable to monitor the processes running on the system. Another attacks that can be detected by HIDS and not NIDS is if an attacker is able to access a system using a valid username and password. HIDS would be able to detect this because if the attacker was using this account in an abnormal way that was out of the usual pattern, it would be alerted to as an anomaly and could then be taken care of by a system administrator. NIDS would not be able to detect this usage because NIDS monitors the network traffic and not the patterns of the user on the system.

NIDS would be able to detect a DoS attack like a SYN Flood attack whereas HIDS would not be able to. NIDS is monitoring the network traffic and upon seeing a large influx of SYN packets from spoofed IP addresses, NIDS would be able to send out an alert to the system about an imminent SYN Flood attack through the increased traffic on the network to a system. HIDS would not be able to detect this kind of attack because it is monitoring the system itself and does not have the ability to monitor the network traffic. Another attack that NIDS could detect would be a scanning attack through the use of a program like nmap where the attacker could be looking for open ports. This would be visible to a NIDS where the traffic of the ports being scanned could be detected and alerted as abnormal. HIDS would not be able to detect a port scanning attack because it monitors the system activity and not the traffic coming into the network.

9.

a) alert tcp 212.16.0.0/24 -> 212.16.0.0/24 any (msg:"TCP Reset Attack from Internal Host"; flags:R; classtype:attempted-dos; priority:1;)

b) alert icmp any any -> 212.16.0.0/24 (msg:"ICMP Redirect Attack on Client Hosts"; icode:5; classtype:attempted-recon; priority:2;)

10.

a) There's a few different "observable phenomena" that could be used to conclude that the system has been compromised, what you would use would depend on the type of attack that was carried out. One phenomenon that could be observed is if there is unusual system activity such as a high CPU usage or programs/process that are running that aren't usually. Along with this you could also look for unexpected network services that are running that you don't recognize. Another phenomenon that I could look for is if there were any new files created or any new user accounts created that I do not recognize, this could also indicate that there was an attack. If there are any files that have been modified or deleted that I did not do myself, this could also indicate that the system has been compromised.

b) The next course of action after identifying the break-in would first be to contain. This would be done depending on what the attack was and how severe it is. One of the first steps to contain this attack would be to disconnect the server from the network, this would prevent the attacker from continuing to access data or other systems on the network for further attacks or to infect other systems. I could also change the filtering rules of the firewalls to stop any incoming and outgoing traffic from the system. This could stop the transmission of any data out from the infected system. If any accounts or files were found that were compromised, those files could be deleted to ensure that the attacker does not have access to them any longer. If the attack is too severe, the system itself could be shut down until a solution was developed to remedy the attack.

c) Honeypots could be used to minimize the occurrence of such attacks in the future that would lure the potential attack away from the critical systems, collect information about the attacker, and encourage the attack to stay on the system long enough for the administrators to figure out how to stop the attacker. Either a low interaction honeypot could be used where the honeypot is a software package that emulates the systems to provide an initial interaction but it less realistic. This honeypot would not help to prevent an attack but would help to warn of an imminent attack. A high interaction honeypot if possible could be used where it is a real system that is more realistic and made to be an easier target to attack which can encourage the hacker to target that system rather than the actual systems. This type of honeypot would require more resources. Also there is a risk that if this system, since it is a real system, if compromised could be used to initiate attacks on the other systems if it was not properly set up as a honeypot and contained as such. Depending on the type of attack we are trying to negate with the honeypot we could deploy it outside of the external firewall to reduce the risk for the internal network and reduce the alerts issues by the IDS and firewall. It could be deployed in the DMZ if there is one which would reduce false positives and detecting probing attempts. Lastly, it could be deployed as a fully internal honeypot like the high interaction one mentioned earlier.