Part 1:

1. The script is included in the Assignment 2 Scripts folder as icmp_monitor.py.
2. The script is included in the Assignment 2 Scripts folder as icmp_teardrop.py. The ICMP teardrop attack works by targeting the TCP/IP reassembly mechanism. Through the use of overlapping offsets on the fragmented packets, the target machine cannot correctly reassemble them due to the overlap. This can cause a denial of service, though most newer operating systems have patched their systems to prevent ICMP teardrop attacks.
3. The script is included in the Assignment 2 Scripts folder as combo_attack.py. This script works similar to the ICMP teardrop script, except with a larger payload to run the Ping of Death attack and a spoofed source IP. This means that the packet will show up as coming from a different source than the attacker. The payload is larger than normally allowed, which can cause denial of service when the packet is reassembled by the target machine.
4. When a host is targeted by the attack from the previous question, it can be harmed in several ways. The ICMP Teardrop attack targets packet reassembly processes, preventing the payload from being properly reassembled which can cause the host servers to crash or freeze. This is amplified by the ping of death attack added to the script, which will mean that the reassembled packet will be too large for the system to handle which can cause additional denial of service to the host machine and its servers as it will be unable to properly process other traffic due to the malicious packets.
5. The script is included in the Assignment 2 Scripts folder as xmas_tree.py. A Christmas Tree attack is an attack that sends a specifically crafted packet with the FIN, PSH and URG TCP flags set, a combination of flags which is not usually seen in normal traffic. This is normally used for system reconnaissance as the target machine's responses allow the attacker to gain information as different operating systems will react differently to receiving this packet. As such it can be used for OS identification, as well as gaining other information about the target which can be used by an attacker to look for more potential vulnerabilities in a system.
6. Backscatter traffic is a side effect of spoofed DDOS attacks, where the attacker spoofs the source IP in packets sent to the target. The target machine responds to these spoofed packets as it would normally. If the attacker is spoofing random IP addresses for the fraudulent packets, the responses will be sent to random IP addresses – these response packets are backscatter traffic. This can be beneficial to us for security purposes, however. Through monitoring and analysis of backscatter we can observe packets arriving at a statistically significant portion of the IP address space to determine the characteristics of the attacker and potentially enable us to begin identifying the source of the attack.
7. When selecting alternative servers for an amplification attack, there are several criteria to consider. We need to use an unwilling intermediary to deliver the attack in conjunction with a spoofed source IP. The intermediary needs to deliver a response that will go to the target instead of the attacker, and the attack needs to be asymmetric – the response has to be larger than the initial query. The servers will also need to use a protocol such as UDP or ICMP that does not require a handshake so that they cannot tell that our spoofed IP address is illegitimate. NTP, SSDP and SNMP are all valid potential alternatives that meet these criteria.

Part 2:

1. Firewalls can slow down worm propagation by implementing bandwidth throttling or rate limiting measures. By reducing the TCP Window Size parameter, they can decrease the amount of data that a sender can transmit before receiving an acknowledgement from the receiver. This limits the transmission rate, which will slow the worm's propagation speed. IPS can leverage the TCP Maximum Segment Size, which represents the maximum amount of data that can be carried in a single TCP segment. By reducing this, IPS can force the worm to send smaller chunks of data, slowing its rate of propagation. Additionally we can use IPS to analyse network traffic patterns, detect worm-like behaviour and apply throttling mechanisms similar to firewalls to limit the spread of malicious traffic. Honeypots can also be used to delay worms by emulating vulnerable systems with limited resources through the previously mentioned methods. When a worm infects such a honeypot it is limited by the firewall and IPS methods, and due to the nature of a honeypot not at risk of compromising important systems. This can give administrators time to analyse the worm's behaviour and payloads and develop countermeasures to mitigate its impact on real (non-honeypot) systems.
2. Policy table:

No	Transport Protocol	Protocol	Source IP/Network	Dest IP/Network	Source Port	Dest Port	Flags	Sig	Action
1	TCP	HTTP	ANY	130.195.1.1/24	<1024	80			DROP
2	TCP	ANY	130.195.4.1/24	130.195.1.1/24	ANY	8088		03 0C FE BB A2	DROP
3	UDP	ANY	130.195.4.1/24	130.195.1.1/24	ANY	4044		03 0C FE BB A2	DROP
4	TCP	ANY	130.195.4.0/24	130.195.1.1/24	ANY	ANY	FIN,PSH, URG		DROP
5	TCP	HTTP	192.168.2.0/24	130.195.1.1/24	ANY	80			ACCEPT
6	TCP	HTTP	192.168.3.0/24	130.195.1.1/24	ANY	80			ACCEPT
7	TCP	HTTPS	192.168.2.0/24	130.195.1.1/24	ANY	443			ACCEPT
8	TCP	HTTPS	192.168.3.0/24	130.195.1.1/24	ANY	443			ACCEPT
9	TCP	HTTP	130.195.1.1/24	192.168.2.0/24	ANY	80			ACCEPT
10	TCP	HTTP	130.195.1.1/24	192.168.3.0/24	ANY	80			ACCEPT
11	TCP	HTTPS	130.195.1.1/24	192.168.2.0/24	ANY	443			ACCEPT
12	TCP	HTTPS	130.195.1.1/24	192.168.3.0/24	ANY	443			ACCEPT
13	ANY	ICMP	130.195.4.0/24	130.195.1.1/24	ANY	ANY			DROP
14	UDP	DNS	192.168.2.0/24	ANY	ANY	53			DNAT
15	UDP	DNS	192.168.3.0/24	ANY	ANY	53			DNAT
16	TCP	SSH	130.195.4.1/24	130.195.1.1/24	ANY	22			ACCEPT
17	TCP	SSH	130.195.1.1/24	130.195.4.1/24	22	ANY			ACCEPT
18	TCP	SSH	ANY	130.195.1.1/24	ANY	22			DROP
19	TCP	SSH	130.195.1.1/24	ANY	22	ANY			DROP
20	ANY	ANY	ANY	130.195.1.1/24	2002	ANY			DROP
21	ANY	ANY	ANY	130.195.1.1/24	2004	ANY			DROP
22	ANY	ANY	130.195.1.1/24	ANY	ANY	2002			DROP
23	ANY	ANY	130.195.1.1/24	ANY	ANY	2004			DROP
24	UDP	ANY	ANY	130.195.1.1/24	ANY	7			DROP
25	TCP	ANY	130.195.1.1/24	130.195.4.100	ANY	80			ACCEPT

26	TCP	ANY	130.195.4.100	130.195.1.1/24	80	ANY			ACCEPT
27	ANY	ANY	130.195.1.1/24	ANY	ANY	ANY			DROP

IPTables Rules:

1. sudo iptables -A INPUT -p tcp --dport 80 -m tcp --sport 0:1023 -j DROP
2. sudo iptables -A INPUT -p tcp --dport 8088 -m string --string "\x03\x0c\xfe\xbb\xa2PASS : RECV"--to 40 -s 130.195.4.1/24 -j DROP
3. sudo iptables -A INPUT -p udp --dport 4044 -m string --string "\x03\x0c\xfe\xbb\xa2PASS : RECV"--to 40 -s 130.195.4.1/24 -j DROP
4. sudo iptables -A INPUT -p tcp --tcp-flags FIN,PSH,URG -s 130.195.4.1/24 -j DROP
5. sudo iptables -A INPUT -p tcp --dport 80 -s 192.168.2.0/24 -j ACCEPT
6. sudo iptables -A INPUT -p tcp –dport 80 -s 192.168.3.0/24 -j ACCEPT
7. sudo iptables -A INPUT -p tcp --dport 440 -s 192.168.2.0/24 -j ACCEPT
8. sudo iptables -A INPUT -p tcp --dport 440 -s 192.168.3.0/24 -j ACCEPT
9. sudo iptables -A OUTPUT -p tcp --sport 80 -d 192.168.2.0/24 -j ACCEPT
10. sudo iptables -A OUTPUT -p tcp --sport 80 -d 192.168.3.0/24 -j ACCEPT
11. sudo iptables -A OUTPUT -p tcp --sport 440 -d 192.168.2.0/24 -j ACCEPT
12. sudo iptables -A OUTPUT -p tcp --sport 440 -d 192.168.3.0/24 -j ACCEPT
13. sudo iptables -A INPUT -p icmp -s 130.195.4.0/24 -j DROP
14. sudo iptables -t nat -A PREROUTING -p udp -s 192.168.2.0/24 --dport 53 -j DNAT --to-destination 8.8.8.8:53
15. sudo iptables -A INPUT -p tcp --dport 22 -s 130.195.4.1/24 -j ACCEPT
16. sudo iptables -A OUTPUT -p tcp --sport 22 -d 130.195.4.1/24 -j ACCEPT
17. sudo iptables -A INPUT -p tcp --dport 22 -j DROP
18. sudo iptables -A OUTPUT -p tcp --sport 22 -j DROP
19. sudo iptables -A INPUT -p all --sport 2002 -j DROP
20. sudo iptables -A INPUT -p all --sport 2004 -j DROP
21. sudo iptables -A OUTPUT -p all --dport 2002 -j DROP
22. sudo iptables -A OUTPUT -p all --dport 2004 -j DROP
23. sudo iptables -A INPUT -p udp --dport 7 -j DROP
24. sudo iptables -A OUTPUT -p tcp --dport 80 -d 130.195.4.100 -j ACCEPT
25. sudo iptables -A INPUT -p tcp --sport 80 -s 130.195.4.100 -j ACCEPT
26. sudo iptables -A OUTPUT -m state --state NEW -j DROP

Explanations:
Rule 1 drops all traffic on port 80 from source ports 0 to 1023.
Rules 2 and 3 search incoming packets on TCP port 8088 and UDP port 4044 respectively to see if they contain the signature 03 0C FE BB A2 within the first 40 bytes, dropping them if so.
Rule 4 searches incoming packets from the external network for the FIN,PSH,URG flags, which are symptomatic of a Christmas Tree attack.
Rules 5 and 6 accept incoming traffic on port 80, the port used by HTTP, from the IP addresses of the clients.
Rules 7 and 8 do the same thing on port 440, the port used by HTTPS.
Rules 9 through 12 allow outgoing traffic from client IP addresses on ports 80 and 440
Rule 13 drops incoming ICMP packets from the external network
Rule 14 reroutes UDP traffic on port 53, the standard for DNS requests, to 8.8.8.8 on the same port.
Rule 15 accepts incoming TCP traffic on port 22, the port used for SSH requests from the administrator IP
Rule 16 allows outgoing TCP traffic on port 22 to the administrator IP
Rule 17 drops all other incoming traffic on port 22 to prevent any potential SSH attacks. Rule 18 does the same for outgoing traffic on port 22.

Rules 19 and 20 drop all incoming traffic on ports 2002 and 2004
Rules 21 and 22 drop all outgoing traffic to ports 2002 and 2004
Rule 23 drops all UDP traffic on port 7, which is the main port used for UDP Fraggle attacks
Rule 24 allows outgoing TCP traffic on port 80 to the update server IP
Rule 25 allows incoming TCP traffic on port 80 from the update server IP
Rule 26 drops all new outgoing connections from the server.