Defense Against ARP Spoofing and TCP Hijacking
Shantanu (2010CS50295) Nikhil (2010CS50289)

**ARP Spoofing:**

**Static ARP Entries:** This is the simplest way. It includes defining IP to MAC mappings in the local ARP cache and then configuring the hosts to ignore all ARP reply requests. This task is done at the OS level, that is, the OS should offer capability to configure ARP rules for the machine.

**Feasibility:** This scheme is practical for decently sized networks because it requires statically configuring the hosts. For example, all Linux distributions have support for a network switch to block gratuitous ARP replies, although this switch is off by default on some major distributions. To block gratuitous ARP replies, you can run the following Linux command:

```
echo 0 > /proc/sys/net/ipv4/conf/all/arp_accept
```

Similar results can be obtained in Windows 7 by enabling the Windows Firewall.

**Potential Problems with this approach:** As mentioned previously, this approach is feasible for small sized networks. Also, it limits the networks capacity of adding hosts to the network dynamically. Also this method is still stateless. We need some method to maintain the state of the table (for timed out MAC-IP mappings). This can be overcome by using a secondary static ARP cache table.

**Packet Filtering:** The basic idea here is to use a tool that inspects the packets as they are transmitted across the network. They are helpful as they are capable of filtering out and blocking packets with conflicting source address information.

**Feasibility:** This scheme is often realised using network monitoring softwares. This is covered below in ARP protection software section.

**Potential Problems with this approach:** Monitoring packets in a large network becomes a resource intensive task as this may require a lot of traffic. This might add to network latencies.

**Cryptography in ARP packets:** Transport Layer Security (TLS), Secure Shell (SSH), HTTP Secure (HTTPS), and other secure communications protocols bolster ARP spoofing attack prevention by encrypting data prior to transmission and authenticating data when it is received.

**Feasibility & Potential Problems:** This scheme is indeed feasible but it will slow down the whole process in link layer. Moreover it violates the end to end principle at the link layer level.

**ARP Protection Softwares:** ARP Guard, DroidSheep Guard, XARP (Advanced ARP Spoofing Detection) are some of the software tools which are used on various
platforms for detection and prevention of ARP spoofing attacks. Most of these softwares work on the principle of checking ARP responses or some form of certification. The unknown (uncertified ARP messages are then dropped or blocked). This technique of checking ARP responses can also be integrated as a capability in DHCP servers and switches. In DHCP servers, this technique can be used to certify both static and dynamic IP addresses.

Feasibility: These softwares can be easily installed and setup on the network infrastructure.

Potential Problems with this approach: Any significant loophole in this case is unlikely unless the software itself is compromised. Another potential problem can be that there might be need to map more than one IP to a MAC address (Network traffic monitoring application). In this case, the software can pose a problem. In all aspects, this scheme does work quite well against ARP Spoofing.

ARP-Cache Overwriting Check: If there is any change in arp table entries, the host should first check if for the existing entries the host is alive (by doing a ping to the host). Now here we can ban the MAC which advertised the wrong ARP entry.

Feasibility & Potential Problems: The problem with this approach is that if the cache is already poisoned then a legitimate host might be banned. So there can be some soft state mechanism whereby the entries are banned temporarily and then the ban is lifted.

TCP Hijacking:

Randomness: Since the attack relies heavily on the fact of forging the initial sequence number. It is important to introduce randomness in deciding the ISN. Now a days, most OS decide ISN on the basis of clock which is not a good scheme. The OS can use various heuristics for adding randomness to the ISN.

Feasibility: Randomness can be practically enhanced in the host machines via various heuristics for Pseudo Random Number Generators and can be easily implemented.

Potential Problems with this approach: This approach not always returns a true random number.

Cryptographic Protocols: Using encrypted transport protocols such as SSH, SSH, and IPSec, we can lower the risk of hijack because these encryption protocols use dynamically generated session keys to provide a secure communication channel and make it difficult for the attacker to predict or steal the keys.

Feasibility: This scheme is in use over a large scale now a days. There are various new versions of the cryptographic protocols which are in use. Hence, this scheme very much feasible.