ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)
Volume 1, Issue 9, November 2012

"SPOOFING"…… Headache to IT world.

Mr. Shaikh Shaffiq, Miss. Paithankar Kavita, Shinde Monica D.

Abstract— **The whole world is engaged to provide the complete security to information when it is in network. But the techniques somewhere fail to overcome on attacks. One of them is *Spoofing*.**

***Spoofing* is an active security attack in which one machine on the network masquerades as a different machine. As an active attack, it disrupts the normal flow of data and may involve injecting data into the communications link between other machines.**

This masquerade aims to fool other machines on the network into accepting the impostor as an original, either to lure the other machines into sending it data or to allow it to alter data. The meaning of *"spoof"* here is not "a lighthearted parody," but rather "a deception intended to trick one into accepting as genuine something that is actually false." Such deception can have grave consequences because notions of trust are central to many networking systems.

This paper contains active attack info in the form of *"spoofing"*, and the way of doing this with avoidance techniques.

Keyword: ***Spoofing, ARP, security threat.***

SPOOFING

Spoofing can occur at all layers of the IP system. The hardware layer, the data link layer, the IP layer, the transport layer, and the application layer are susceptible. All application layer protocols are at risk if the lower layers have been compromised.

Serious problem occurs if the network interface can alter the source address and send data that appears to come from various source addresses. In the IEEE 802 standards for networking, each network interface has a 48-bit hardware address. It uses this hardware address to match the variety of destination addresses of the frames it sees. The interface copies frames with matching destination addresses into its internal buffer and notifies the operating system that they are available for further processing. Packets coming from the operating system to the interface do not typically specify a source address; the interface always puts its hardware address in the source field.

Most software does not typically control the source field of frames leaving an Ethernet interface. When another host examines a packet containing a hardware source address associated with an interface of a particular machine, it assumes that the packet originated on that machine and accepts it as authentic. An IEEE standards committee assigns each network interface manufacturer a unique 24-bit prefix for the 48-bit hardware address; the manufacturer assigns a unique 24-bit suffix to each interface it makes. Regardless, many interface cards are configurable and allow host software to specify a source address other than the one assigned by the manufacturer. This configurability makes it possible to use them to spoof the source address.[5]

To see how common it is for a network interface to be able to spoof the source address, however, recall how a bridge works. A bridge not only puts its interfaces into promiscuous mode, but it also sets the hardware source address of packets sent out on its interfaces to match the hardware source address of the originating interface. A PC with two software configurable interfaces can be configured to be used as a bridge. Clearly, such software configurability has a variety of malicious uses. The drawbridge software prevent sniffing is compatible with most Ethernet boards which means most Ethernet boards will permit source address spoofing.

Countering hardware level spoofing is difficult because it is virtually undetectable without tracing the physical wiring. You need to trace the wiring to be certain no one has connected an unauthorized machine and you also need to check to see if the authorized machines are using the hardware address they should. The latter can be checked using sufficiently "intelligent" hubs in secure locations.

All machines not in physically secure locations can be connected to hubs in secure locations. Some "intelligent" hubs can be configured to accept or send packets or both to or from specific hardware addresses on each port they service. Thus, you can configure the hub to accept only packets with hardware addresses matching the manufacturer-assigned hardware address of the interface on the authorized machine. This interface should be connected to the wall plate on the far side of the wires connected to that port. Clearly, you are still relying on physical security to be sure that the hub, wires, and authorized machine remain as they should.[1][2]

ARP Spoofing

A more common form of spoofing that is accidental is ARP spoofing. ARP (Address Resolution Protocol) is part of Ethernet and some other similar protocols (such as token-ring) that associate hardware addresses with IP addresses. ARP is not part of IP but part of these Ethernet-like protocols; ARP supports IP and arbitrary network-layer protocols. When an IP datagram is ready to go out on such a network, the host needs to know the hardware destination address to associate with the given IP destination address. For local IP destinations, the hardware address to use will be the hardware address of the destination interface. For non-local destinations, the hardware address to use will be the hardware address of one of the routers on the local network.

How ARP and ARP Spoofing Work

To find the hardware address, the host sends out an ARP request using the hardware broadcast address. A frame with the hardware broadcast address reaches every network interface on the local network, and each host on the local network has its operating system interrupted by the network interface. The ARP request is asking the question, “What is the hardware address corresponding to the IP address I have here?” Typically, only the host with the matching IP address sends an ARP reply and the remaining hosts ignore the ARP request. The ARP request contains the IP address of the sender of the request and reaches all hosts via a broadcast.

Other hosts could potentially store the association between hardware address and IP address of the sender of the request for future reference. The target of the request certainly would store the association. It will almost certainly send an IP datagram in reply to the IP datagram it is about to receive. The reply will require knowing the association between the IP address and the hardware address of the sender of the ARP broadcast.

Thus, when the ARP cache entry for a machine expires, an ARP request goes out to refresh the entry. No reply comes back if the target machine goes down. The entries for its interface’s hardware will disappear from the ARP caches in the other machines on the network. The other machines will be unable to send out IP datagrams to the downed system after the ARP cache entries expire. Before that point in time, IP datagrams are sent out but are not received. When the machine comes back up, it will again be able to reply to ARP requests. If someone replaces its interface, the now up and running machine will have a new hardware address and will use that new hardware address in ARP replies. ARP caches throughout the network will reflect the change, and IP datagrams go out using the new hardware address.

Because you expect the IP address to hardware address association will change over time, the potential exists that the change may be legitimate. Sometimes it is purely accidental. Someone may inadvertently assign a machine the same IP address held by another machine. On personal computers or special purpose devices such as network printers or X Window System terminals, the end user typically has access to a dialog box, command, or text file that sets the IP address. On multiuser systems, the system administrator is typically the only one who can set the IP addresses of the network interface(s). This arrangement is changing, however, as more inexperienced IP-based end users with PCs set addresses. In addition, bureaucracies often separate system administrators and network administrators that use the same network. Under such circumstances it is common for two machines to end up with the same IP address. Duplication can occur either by copying the network configuration from one personal computer to another without the end user knowing the need for IP addresses to be unique. Duplication can also occur if system administrators on a network do not work together when configuring system addressing.

When two machines end up with the same IP address, both of them will naturally reply to an ARP request for that address. Two replies to the request come back to the host that originated the request. These replies will arrive in rapid succession, typically separated by at most a few milliseconds. Some operating systems will not realize anything is wrong and simply file each reply in the ARP cache with the slowest response remaining in the ARP cache until the entry for that IP address expires. Other operating systems will discard ARP replies that correspond to IP addresses already in the cache. These may or may not bother to check if the second reply was a harmless duplicate or an indication an ARP spoof may be underway. Thus, depending on the mechanism used to process duplicate ARP replies, if a spoofer wants to be the target of the IP datagrams being sent to a particular IP address from a particular host, it needs to make sure it is either the first or the last to reply to ARP requests made by that particular host. An easy way to be first or last is to have the *only* machine that replies to the ARP requests. An attacker can simply use a machine assigned, via the normal operating system configuration mechanisms, the same IP address as a machine that is currently not working. An attacker attempting to masquerade his or her machine can simply turn the legitimate machine off. The attacker does not need to have direct access to the power switch on the machine. The machine can be turned off either by unplugging it or flipping the appropriate circuit breaker. An alternative to disconnecting its power is to disconnect it from the network at some point in the network wiring scheme. Third, the attacker can change the legitimate machine’s IP address

and leave it turned on if he or she can reconfigure the machine. Doing so is less likely to draw attention or result in confusion from the machine’s user or administrator.

Receiving the IP datagram sent to the IP address the old and new workstations shared. The new workstation did not know what to do with these datagram and promptly sent a TCP/IP reset message in reply, resulting in the shutdown of the demonstration program. From initial appearances, the demonstration program just stopped and the old workstation appeared to have been cut off from the network.

Needless to say, the presenter was upset. When the system administrator figured out what had gone wrong, the technician who used the IP address of an existing machine learned a valuable lesson: two machines with the same IP address cannot be connected to the network at the same time.[3]

Preventing an ARP Spoof

It is not particularly satisfying to simply detect ARP spoofing, which only identifies a problem after it has already occurred. Although it may not be possible to prevent ARP spoofing entirely, one simple precaution can be taken where it may count the most. The devious thing about an ARP spoof is that the attack is really directed at the machine being deceived, not the machine whose IP address is being taken over. Presumably, the machine or machines being deceived contain data that the ARP spoofer wants to get or modify.

The deception is useful to the ARP spoofer because the legitimate holder of the IP address is trusted in some way by the machine being deceived. Perhaps the trusted machine is allowed to NFS mount filesystems, use rlogin, or start a remote shell without being prompted for a password (particularly troublesome for privileged user accounts). Ideally, machines extending such trust should simply not use ARP to identify the hardware addresses of the machines they trust.

Stop Using ARP

Machines extending trust to other machines on the local network based on an IP address should not use ARP to obtain the hardware address of the trusted machines. Instead, the hardware address of the trusted machines should be loaded as permanent entries into the ARP cache of the trusting machine. Unlike normal ARP cache entries, permanent entries do not expire after a few minutes. Sending a datagram to an IP address associated with a permanent ARP cache entry will never result in an ARP request. With no ARP request being sent, an attacker does not have the opportunity to send an ARP reply. It seems unlikely that any operating system would overwrite a permanent ARP cache entry with an unsolicited ARP reply.

With permanent ARP cache entries for trusted machines, the trusting host will not use ARP to determine the correct hardware address and will not be fooled into sending IP data to an ARP spoofer. Of course, it will also send IP data to the machine even if the machine has been down for some time. Another downside to permanent ARP entries is that the cache entries will need revising if the hardware address changes for a legitimate reason. Finally, ARP caches may be of limited size, limiting the number of permanent entries or further limiting the time a dynamic entry spends in the cache.

Use an ARP Server

The arp command outlined in the previous section also allows one machine to be an ARP server. An ARP server responds to ARP requests on behalf of another machine by consulting (permanent) entries in its own ARP cache. You can manually configure this ARP cache and configure machines that extend trust based on this IP address to use ARP replies coming from the ARP server rather than ARP replies from other sources. However, configuring a machine to believe only in the ARP server is a difficult task for most operating systems.

Even if you do not configure other machines to trust only the ARP server for ARP replies, the type of server may still be beneficial. The ARP server will send out a reply to the same requests as a potential ARP spoofer. When machines process the ARP replies, there is at least a fair chance that the ARP spoofer’s replies will be ignored. You cannot be sure because as you have seen, much depends on the exact timing of the replies and the algorithms used to manage the ARP cache.

Detecting an ARP Spoof

Unless you have the capability to introduce the kind of hardware barriers described previously, preventing an ARP spoof is probably not practical. The best you can usually hope for is rapid detection followed by some form of intervention. When an anomaly is detected in the ARP protocol it may be legitimate, accidental, or a security breach. Policies and procedures should be in place to handle each type of incident. This chapter limits its discussion to mechanisms; it is up to the reader to decide what policies and procedures to implement after detection of a potentially serious problem takes place. Several mechanisms exist for detecting an ARP spoof. At the host level, an ordinary host may attempt to detect another machine using its own IP address either by passively examining network broadcasts or by actively probing for such a machine. At the server level, a machine providing a supposedly secure service to the network—perhaps a file server or a router—may also attempt to detect an ARP spoof by one of its clients. Finally, at the network level, a machine under control of the network administrator may examine all ARP requests and replies to check for anomalies indicating an ARP spoof is underway.

Host-Level Passive Detection

As a basic precaution, when an operating system responds to an ARP broadcast, it should
inspect both the sender IP address and the target IP address. It only needs to check the target address to see if the target IP address matches its own IP address. If so, it needs to send an ARP reply. However, once the operating system has been interrupted, it takes little extra work to check to see if the sender IP address matches its own. If so, another machine on the network is claiming to have the same IP address. Such an anomaly certainly indicates a serious configuration problem and may be the result of a simplistic ARP spoof in which the attacker simply reset the IP address of the

machine being used in the attack. Many Unix systems perform such a check.

Host-Level Active Detection

Another precaution to detect ARP spoofs is to arrange for hosts to send out an ARP request for their own IP address, both on system startup and periodically thereafter. If the host receives an ARP reply for its own IP address, the IP software should report the detection of an ARP spoof to the host user or administrator. Actively querying ARP with one's own IP address will catch inadvertent IP address misconfigurations as well as an attacker who is simply using an ordinary operating system with a deliberately misassigned IP address. However, it is possible to mount a more sophisticated attack that will thwart the active query detection method.

Server-Level Detection

Alternatively, a more elaborate precaution would be to verify an ARP reply by making an RARP request for the hardware address contained in the reply. RARP, the reverse address resolution protocol, uses the same format as ARP and also broadcasts requests. RARP requests ask the question "What is the IP address associated with the hardware address I have here?" Traditionally, the primary use of RARP is by diskless machines with no permanent modifiable memory. Such machines need to discover their own IP address at boot time. RARP relies on one or more RARP servers that maintain a database of hardware addresses and the corresponding IP addresses. Use of an RARP server is probably overly elaborate when an ARP server would do the same job.

Network-Level Detection:

The motivation for network-level detection is that host-level detection may be unable to inform the network staff that a problem exists and that server-level detection probably requires modification of IP software of the operating system source code. When a host detects that it is being impersonated by another machine, it may be able to report the fact to its user, but once an attack is underway it may be unable to inform the network administrator who is presumably using another machine.

Some popular IP system software may very well take the precaution of occasionally making ARP requests for the hardware address associated with the IP address it believes is its own. The active querying precaution is well-known and is a common textbook exercise. Most corporate system staffs are unable to modify the IP software of most of the machines on their network. If that is your situation, you probably want a software detection system that can be deployed on a single machine on your network. Building the system using software already written by someone else is preferable.

Network-Level Detection via Periodic Polling

By periodically inspecting the ARP caches on machines, you should be able to detect changes in the IP address to hardware address association on those machines. It should be routine for the network staff to keep a database of hardware addresses, IP addresses, DNS names, machine types, locations, and responsible persons. At the very least, such an inspection can probably be done manually on most hosts. It could be done more often if hosts could be configured to periodically report the contents of their ARP caches to a centralized machine. A program on that machine could look for inconsistencies between hosts, changes from previous reports, and conflicts between reported ARP cache information and the information in the manually maintained database—any of these may indicate a problem.

Standard mechanisms for periodic reporting of network configuration information from machines on an IP-based network to the network administration staff already exist. One such mechanism is SNMP—the Simple Network Management Protocol. In SNMP, each machine using IP runs an SNMP agent which both responds to information and configuration requests as well as reports certain conditions to the network management staff. Virtually all current systems provide bundled SNMP agents. To take advantage of SNMP, the network management staff must have SNMP management software to query theagents and react to the agent reports. Finding good SNMP management software may be difficult and expensive to purchase and deploy. [4]

REFERENCES :-

[1]Spoofing
http://www.articsoft.com/whitepapers/spofing.pdf
［2］ Attacks:
http://www.cigitallabs.com/resources/papers/download/arppoison.pdf
http://www.hut.fi/~autikkan/kerberos/docs/phase1/pdf/LATEST_hijaking_attack.pdf
［3］ ARP Basics:
http://ece.gmu.edu/~robohn/tcom509-l2-s03.pdf
http://www.comptechdoc.org/independent/networking/guide/netarp.html
[4]ARP SPOOFING
http: / /SPLOIT.US
http://node99.org/projects/arpspoof/arpspoof.pdf
[5] IEEE
Seung Yeob, N., K. Dongwon, and K. Jeongeun, Enhanced ARP:preventing ARP poisoning-based man-in-the-middle attacks. Communications Letters, IEEE, 2010. 14(2): p. 187-189.