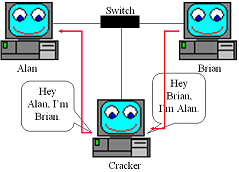
**A Quick Intro to Sniffers:  
Wireshark/Ethereal, ARPSpoof, Ettercap, ARP poisoning and other niceties.**

     When I tell some of my coworkers that I'm sniffing the network, they have a tendency to look at me funny. A Sniffer (also know as a Network Analyzer) is a piece of software that can look at network traffic, decode it, and give meaningful data that a network administrator can use to diagnose problems on a network. Sniffers are also useful tools for deviant computer users since they can be used to pull plain text passwords off a network. A few popular general purpose Sniffers are NAI Sniffer (commercial), Wireshark (previously know as Ethereal, an Open Source GUI Sniffer for Linux, Windows and other platforms), TCPDump (Open Source command line Sniffer for \*nix - any Unix like operating system like Linux or FreeBSD-) and its Windows version called WinDump.  
  
     First an explanation of some network basics is in order. Most Ethernet networks use to be of a common bus topology, using either coax cable or twisted pair wire and a hub. All of the nodes (computers and other devices) on the network could communicate over the same wires and take turns sending data using a scheme known as carrier sense multiple access with collision detection (CSMA/CD). Think of CSMA/CD as being like a conversation at a loud party, you may have to wait for quite a spell for your chance to get your words in during a lull in everybody else's conversation. All of the nodes on the network have their own unique MAC (media access control) address that they use to send packets of information to each other. Normally a node would only look at the packets that are destined for its MAC address. However, if the network card is put into what is known as "promiscuous mode" it will look at all of the packets on the wires it is hooked to.  
  
     To cut down on the number of collisions and the possibility of sniffing data that does not belong to a node, most networks use switches. On a network, a hub is a passive device that sends all traffic it receives to all of its ports. A switch on the other hand looks at the MAC address of the nodes hooked to it and what ports they are on then tries to send packets only to the nodes they are intended for. A switch cuts back on the number of collisions on the network, increasing throughput. In theory, on a switched network a node can only see broadcast messages (meant for all computers on the LAN) and packets addresses to its MAC along with the occasional stray packet whose destination is not known. Even with switches in place a LAN can be sniffed using the mirrored port on some switches (put there so administrators can use a Sniffer to diagnose network problems), by confusing the switch into mirroring traffic to all ports or by a technique know as ARP poisoning (more on this later).

    The above is about Ethernet networks, WiFi (802.11a/802.11b/802.11g/802.11n) is a bit different however. Wireless LANs act a lot like Ethernet LANs using hubs. Every computer on the LAN can see the traffic destined to others but normally they just choose to ignore it. (In reality it's a little more complicated than that, but I want this to be an article and not a book on the intricacies of 802.11 networks) However, if a network card is put into what is known as promiscuous mode, it will not ignore traffic going to other computers and will instead look at it, allowing the user of the computer running the sniffer to see the data traveling to other computers attached to the same access point. Promiscuous mode works on pretty much any wired network card in Windows and Linux (or other Unix like Operating System), but not all wireless cards support it properly (like Intel's Centrino 802.11g chipset know as IPW2200). If the sniffer's card does support promiscuous mode it will have to be attached to the wireless networks WAP (Wireless Access Point) to be able to see anything. If the attacker is using Linux (or another Unix like Operating System) the attacker may be able to use what is known as monitor mode if their card supports it. In monitor mode, the wireless network card listens to the raw packets in the radio waves without ever having to attach to a WAP. The nice thing about monitor mode from the attacker's perspective is that they leave no logs of their activities since they don't have to attach to the WAP and don't have to send any packets on the network.

     Sniffing WiFi networks is further complicated by what security protocols they use. If your card support promiscuous mode and you can attach to a wireless network using WEP (in other words, you know the WEP key) you can sniff pretty much anything you want. If the network is using WPA it's not as easy since just knowing the pass phrase won't  let you decode all traffic in a network conversation your box is not evolved in. However, it may be possible to ARP poison or use some other MitM (Man in the Middle) attack to get the data routed through you.

     Sniffers have many legitimate uses that system administrators should be aware of. They can be used to find what computers on the network are causing problems such as using too much bandwidth, having the wrong network settings or running malware. I've personally found them useful in the past for finding hack attempts as they were happening by sniffing my own servers for inappropriate traffic. Every system admin would do well to learn about using sniffers to find network problems and I'd recommend starting with Wireshark since it's free, multiplatform and well supported (see the links section of this article of more information).  
  
     Sniffers can also be used by those trying to bypass security. Many popular application protocols pass logon credentials (username and password) in plain text or using weak encryption that's easy for a Sniffer to decode. Common examples of such insecure protocols are FTP, Telnet, POP3, SMTP, and HTTP Basic Authentication. In their place use encrypted protocols like SFTP, SSH (Secure Shell), and HTTPS (SSL) when possible. Protocols like FTP may be hard to switch away from because the clients for more secure protocols like SFTP are not as readily available. FTP clients come with every recent version of Windows (ftp.exe from the command line and Explorer from a GUI), but free clients that support SFTP like FileZilla and PSFTP can be downloaded. A few sniffers that have good password extraction abilities include Cain, Dsniff and Ettercap. All three are free or Open Source. Cain is for Windows only and Dsniff and Ettercap are mostly used in \*nix environments but also have Windows versions available.  
  
**ARP Spoofing/ARP Poisoning**  
  
     ARP stands for Address Resolution Protocol and it allows the network to translate IP addresses into MAC addresses. Basically, ARP works like this: When one host using IP on a LAN is trying to contact another it needs the MAC address of the host it is trying to contact. It first looks in its ARP cache (to see your ARP cache in Windows type in "arp -a" at the command line) to see if it already knows the MAC address, but if not, it broadcasts out an ARP request asking "Yo, who has this IP address I'm looking for?" If the host that has that IP address hears the ARP query it will respond with its own MAC address and a conversation can begin using IP. In common bus networks like Ethernet using a hub or 802.11b all traffic can be seen by all hosts whose NICs (network interface card) are in promiscuous mode, but things are a bit different on switched networks. A switch looks at the data sent to it and tries to only forward packets to its intended recipient based on the MAC address. Switched networks are more secure and help speed up the network by only sending packets where they need to go. There are ways around switches though. Using a program like Arpspoof (part of the Dsniff package), Ettercap or Cain we can lie to other machines on the local area network and tell them we have the IP they are looking for, thus funneling their traffic through us.  
  
     Even with a switched network it's not hard for an attacker to use Dsniff or Ettercap from the BackTrack boot CD to do some ARP spoofing and redirect traffic through them for the purposes of sniffing. These tools can even parse out usernames and passwords automatically, making the attacker's job easy. If the attacker ARP Spoofs between the gateway and the FTP server he can sniff the traffic and extract user names and passwords as users are trying to get their data from offsite, and the same thing goes for SMTP and POP3. Even with SFTP, SSL, and SSH, passwords can still be sniffed with Ettercap because it has the ability to proxy those types of connections. The user might get a warning that the public key of the server they are trying to get to has changed or may not be valid, but how many of us just click past those kinds of messages without actually reading them?  
  
     The image in figure 1 helps to illustrate how ARP Spoofing/ARP Poisoning works. Basically, the attacker is telling Alan's box that he has the IP that corresponds to Brian's box and vice versa. By doing this the attacker receives all network traffic going between Alan and Brian. Once the attacker has ARP Spoofed his way between two nodes he can sniff the connection with whatever tool he likes (TCPDump, Wireshark, Ngrep, etc.) By ARP Spoofing between a computer and the LAN's gateway an attacker can see all the traffic the computer is sending out and receiving from the Internet. In this article I'm only giving the basics of how these tools are used.  
  
**A quick demonstration of ARP Spoofing using Dsniff Tools and Ettercap**  
  
     Let's start by using Dug Song's Arpspoof program that comes with his Dsniff package. I use the \*nix version but if you look around you may be able to find a Win32 version. The easiest way to run Dsniff is to boot from a BackTrack boot CD. The first thing you should do is make sure packet forwarding is turned on, otherwise our machine will drop all traffic between the hosts we are trying to sniff, causing a denial of service. Some of the tools I use do this automatically (Like Ettercap), but to be sure, you may want to do it yourself. Use the following commands, depending on operating system:  
  
Linux:  
echo 1 > /proc/sys/net/ipv4/ip\_forward  
  
BSD:  
sysctl -w net.inet.ip.forwarding=1  
  
     Now that your computer will forward the traffic you can start ARP Spoofing. Let's assume you want to sniff all traffic between a host and the gateway so you can see the traffic it's sending to the Internet. To get traffic in both directions you would use the following two commands:  
  
arpspoof -t 192.168.1.1 192.168.1.2 & >/dev/null  
arpspoof -t 192.168.1.2 192.168.1.1 & >/dev/null  
  
     The "& >/dev/nul" part is there to make it easier to run from one terminal, but you may want to omit it for debugging purposes. Now you can use any package you wish to sniff the connection. To start with I'd recommend using the Sniffer Dsniff that comes along with Arpspoof to sniff for plain text passwords. To start sniffing with Dsniff just drop out to a command shell and type:  
  
dsniff  
  
     As Dsniff finds passwords and usernames it will print them to the screen. To look at all sorts of other traffic I would recommend TCPDump or Wireshark. When you are ready to stop ARP Spoofing issue the following command:  
  
killall arpspoof  
  
     This should kill the two instances of Arpspoof started above.

    Another great tool is Ettercap, the Swiss army knife of ARP Poisoning and password sniffing. I usually use it in non-interactive mode, but by default it has a ncurses interface that some may find easier to use. If you would like to use Ettercap for ARP poisoning instead, the following commands should serve as good examples. If we wanted to target all hosts on the network and sniff traffic between every node, we would use the following command:  
  
ettercap -T -q -M ARP // //  
  
    Be careful with the above command, having all of the traffic on a large network going though one slow computer can really bog down network connections. If we had a specific victim in mind, let's say a host with the IP 192.168.1.1, we would use this command:  
  
ettercap -T -q -M ARP /192.168.1.1/ //  
  
If 192.168.1.1 is the gateway, we should be able to see all outgoing traffic. Here are what the command line option flags do:  
  
-T tells Ettercap to use the text interface, I like this option the best as the more GUI modes are rather confusing.  
-q tells Ettercap to be more quiet, in other words less verbose.  
-M tells Ettercap the MITM (Man in the Middle) method we want to use, in this case ARP poisoning.

For some other things you can do with Ettercap check out my video [Fun with Ettercap Filters: The Movie](http://irongeek.com/i.php?page=videos/ettercapfiltervid1).  
  
**Other Tools**  
  
     There are many other packages I would like to mention as well. The first is Cain, which Windows users will be much more comfortable with. It has some great functionality and a nice interface. I have a video tutorial on how to use it here:

[Cain to ARP poison and sniff passwords](http://irongeek.com/i.php?page=videos/cain1)

    If you like pretty GUIs, Cain is the way to go. It does not have as many options as Ettercap, but it's still pretty cool and has some other Windows specific extras built in.

    There are also specialized snuffers for certain kinds of content. Driftnet parses out the images people are seeing as they web surf .I've not done a video on Driftnet, but I have done one on NetworkActive which can also parse out images from web traffic:

[Using NetworkActiv to sniff webpages on a Wi-Fi network](http://irongeek.com/i.php?page=videos/networkactiv)

    These also sniffers like P0f that let you finger the OS of network traffic passively:

[Passive OS Fingerprinting With P0f And Ettercap](http://irongeek.com/i.php?page=videos/passive-os-fingerprinting)

    And that's just the tip of the iceberg when it comes to specialized sniffers.

**Mitigating Sniffing Attacks**  
  
     There are quite a few ways to mitigate sniffing attacks.  
  
1. Avoid using insecure protocols like Basic HTTP authentication and Telnet. As a matter of fact you should sniff your own network to see what passwords the tools listed above can pick up.  
2. If you have to use an insecure protocol, try tunneling it though something to encrypt the sensitive data. I have a video on [SSH Dynamic Port Forwarding](http://irongeek.com/i.php?page=videos/sshdynamicportforwarding) that show one way to accomplish this.  
3. Look into using Static ARP tables between critical workstations and servers. They are more trouble to maintain but limit arpspoofing.  
4. Run software like ARPWatch to detect changes in MAC addresses on your network that may point to Sniffers.  
5. Try running tools like Sniffdet and Sentinel to detect network cards in promiscuous mode that may be running sniffing software.  
6. Have outside laptops using Wi-Fi that come into your facility use a VPN to connect to the network.  
7. Lockdown workstations so users can't install sniffing software or boot from a CD like Knoppix.  
8. Keep public terminals on a separate LAN from the staff workstations and servers.  
  
I hope you have found this article useful.  
  
**Further research:**  
  
ARPWatch  
<http://www-nrg.ee.lbl.gov/>

BackTrack Live CD  
<http://www.remote-exploit.org/backtrack.html>  
  
Cain:  
<http://www.oxid.it/cain.html>

Driftnet  
<http://ex-parrot.com/~chris/driftnet/>  
  
Dsniff Package:  
<http://www.monkey.org/~dugsong/dsniff>   
  
Ethernet Networks:  
<http://en.wikipedia.org/wiki/Ethernet>

Ettercap  
<http://ettercap.sourceforge.net/>  
  
Wireshark  
<http://www.wireshark.org/>  
  
Wireshark User's Guide  
<http://www.wireshark.org/docs/wsug_html/>  
  
Ettercap:  
<http://ettercap.sourceforge.net/>   
  
P0f  
<http://lcamtuf.coredump.cx/p0f.shtml>  
  
Sentinel  
<http://www.packetfactory.net/Projects/sentinel/>   
  
Sniffdet  
<http://sniffdet.sourceforge.net/>   
  
TCPDump  
<http://www.tcpdump.org/>  
  
WinDump  
<http://windump.polito.it/>

**Change Log::**  
**02/01/2005:**Article first published. **07/30/2007:**Update Ethereal to Wireshark, added section on WiFi networks, switched to using BackTrack CD and a lot of other little tweaks.  
**04/30/2008:**Fixed a stupid error I made where I mistyped 801.11 instead of 802.11.

[[](https://www.irongeek.com/i.php?page=security/AQuickIntrotoSniffers&mode=print)Printable version of this article](https://www.irongeek.com/i.php?page=security/AQuickIntrotoSniffers&mode=print)

15 most recent posts on Irongeek.com:

* [Who's Your Hacker](https://www.whosyourhacker.com/webinars-free-cpes/)
* [BSides Tampa 2020 Videos](http://www.irongeek.com/i.php?page=videos/bsidestampa2020/mainlist)
* [Louisville Infosec 2019 Videos](http://www.irongeek.com/i.php?page=videos/louisvilleinfosec2019/mainlist)
* [BSidesCT 2019 Video](http://www.irongeek.com/i.php?page=videos/bsidesct2019/mainlist)
* [GrrCON 2019 Videos](https://www.irongeek.com/i.php?page=videos/grrcon2019/mainlist)
* [BSidesSTL 2019 Videos](http://www.irongeek.com/)
* [DerbyCon 9 Videos](http://www.irongeek.com/i.php?page=videos/derbycon9/mainlist)
* [OISF 2019 Videos](http://www.irongeek.com/i.php?page=videos/oisf2019/mainlist)
* [BSides Cleveland 2019 Videos](http://www.irongeek.com/i.php?page=videos/bsidescleveland2019/mainlist)
* [ShowMeCon 2019 Videos](http://www.irongeek.com/i.php?page=videos/showmecon2019/mainlist)
* [Circle City Con 2019 Videos](http://www.irongeek.com/i.php?page=videos/circlecitycon2019/mainlist)
* [NolaCon 2019 Videos](http://www.irongeek.com/i.php?page=videos/nolacon2019/mainlist)
* [Hacker/Infosec Con Types & Getting More Out Of Hacker/Infosec Conferences](http://www.irongeek.com/i.php?page=security/hacker-infosec-con-types-getting-more-out-of-hacker-infosec-conferences)
* [BSides Columbus 2019 Video](http://www.irongeek.com/i.php?page=videos/bsidescolumbus2019/mainlist)
* [BSides Tampa 2019 Videos](http://www.irongeek.com/i.php?page=videos/bsidestampa2019/mainlist)