Analysis of FTP Hijack

Phong Huynh

Analysis of FTP Hijack

Phong Huynh
GSEC Practical Requirements (v.1.2f)
9/19/01

FTP

File Transfer Protocol is a protocol that is used to transfer files from one destination to another. There are four main objectives for the use of FTP:

1) Promote sharing of files (computer programs and/or data).
2) Encourage indirect or implicit (via programs) use of remote computers.
3) Shield a user from variations in file storage systems among hosts.
4) Transfer data reliably and efficiently.

FTP is widely used throughout the world to help transfer files from one place to another and to share files between people. There is always an authentication check with FTP through the use of username and password to control the data that are held at the FTP server. Thus only reliable hosts and users are able to access any of the data within the FTP server.

TCP/IP

The File Transfer Protocol uses the TCP/IP protocol to reliably send data to its destination.

The transfer aspect is accomplished through the use of IP protocol. The IP protocol determines which LAN (Local Area Network) the destination resides in. The routers will deliver the file to its destination by evaluating the IP destination address. IP protocol has a checksum value to ensure that the file is not corrupted when it reaches its destination.

The reliability aspect is accomplished through the TCP protocol that includes sequence numbers and acknowledgement numbers to ensure that the file reaches its destination. The checksum within the TCP protocol makes sure that the data was not corrupted during the transfer in order to guarantee that the information received is the same as the

information sent. If a file was corrupted or did not reached its destination the client would not send an acknowledgement so the server would re-send the data thus providing the reliability of the service.

Since the file must travel through many routers before it reaches its destination, it can be intercepted at any of these locations and be compromised. In order for anyone to gain access to the FTP traffic they must first force the traffic to be routed through them. This can be accomplish through the use of various techniques:

1) DNS cache poisoning
2) ICMP redirect

DNS Cache Poisoning

This is one technique that can be used to redirect traffic through different networks. This technique involves injecting false information to a DNS server so that it will send network traffic to the false location instead of the actual location. DNS cache poisoning is a simple method where the malicious user would determine the DNS query number of the server, this can be done by sending various queries to the server requesting data. He would then send a recursive query to the server asking for a specific address, the server would then send out a query asking another DNS server for the response. He would then spoof the second server's response and change the data of the query to re-route through his location. Network traffic would now be routed through him, which allows him to monitor for the specific FTP session to hijack. DNS cache poisoning is mostly vulnerable to Unix/Linux boxes through their use of the bind vulnerability.

Protection from DNS Cache Poisoning

One method to protect the DNS server from cache poisoning is to harden the bind DNS server.

There are several preventive measures that can help prevent your Unix/Linux boxes from DNS cache poisoning. Earthweb Networking and Communications has shown some steps that can help harden the BIND to prevent DNS cache poisoning:

1) Resource isolation: Use a dedicated, hardened server for Internet DNS, don't share with other services, and

especially do not allow user logins. Minimal services/users means reducing the amount of software running and hence the amount exposed to network attacks. Separation prevents other services or users possibly using local weakness in the system to attack BIND.

2) Redundancy: Install a secondary on a different Internet connection (foreign branch of your company, another ISP, etc.). If your site dies, at least other sites won't think you "cease to exist"; they just think you're "not available," so that emails, for example, won't get lost but will be queued (typically up to four days).

3) Use the latest version (e.g. 8.2.2-P5 or later, which includes security fixes).

4) Access control: Restrict zone transfers to minimize the amount of information on your networks available to attackers. Consider using transaction signatures. Consider restricting/not allowing recursive queries.

5) Run BIND with minimum privileges: Run BIND as a non-root user.

6) More resource isolation: Run BIND in a "chroot" jail, so it is much more difficult for a compromised bind daemon to damage the operating system or compromise other services.

7) Detection: Monitor logs for unusual activity; monitor the system for unauthorized changes with an integrity checker; keep an eye on relevant advisories.

ICMP Redirect

ICMP redirect can be used to alter a router's routing table. The normal use of an ICMP redirect is to allow routers to reconfigure their routing table to allow for maximum efficiency by recording the best routing path. But the redirect can also be used to force a router to alter it's routing table and route to a different location. A malicious user could use this technique to monitor for a specific FTP session in order to gain access to the data. In some cases ICMP redirect lasts for a set time on the router and thus must be resent or the routing table will remove the ICMP redirected path. ICMP redirect could also

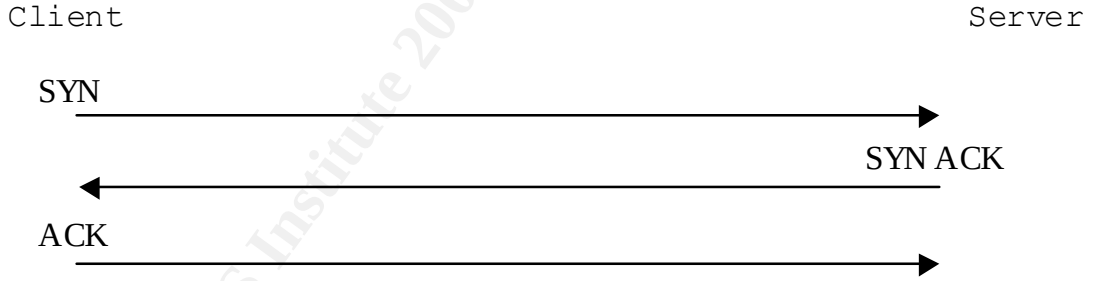
be sent to the client's machine telling it to send it's network traffic directly to the attackers path.

Protection from ICMP Redirect

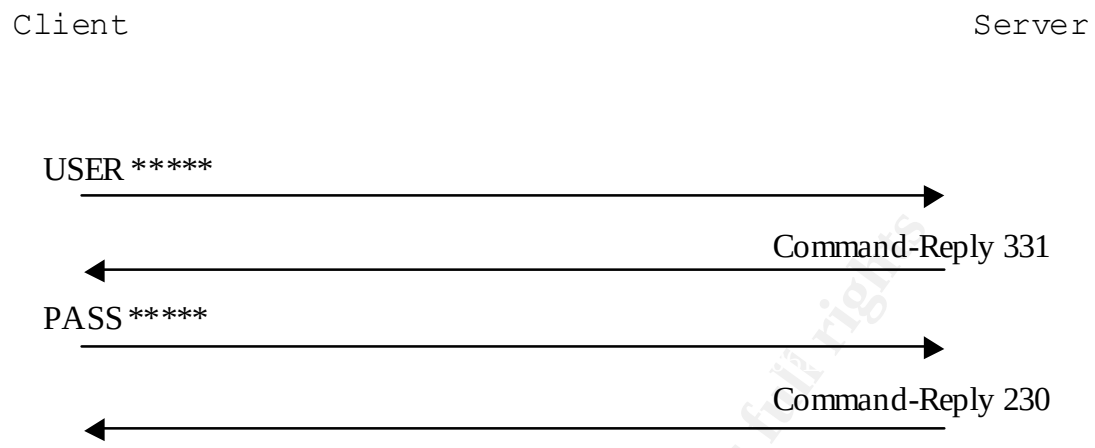
The simplest way to prevent from ICMP redirects is to disable ICMP Redirect Acceptance. Thus ICMP redirects will have no effect on the routing tables.

FTP Analysis

FTP is a powerful tool to transfer file from one destination to another. Its use of the TCP Protocol will enable a reliable connection ensuring data will be sent and received as long as the connection is alive. Below shows a three-way handshake between the client and server application. The client sends a network packet with the SYN flag requesting a connection to be made with the server. The server responds to the client's SYN with an ACK flag and sends its own SYN flag to the client. The client now sends an ACK in response to the server's SYN completing the three-way handshake and establishing a connection.

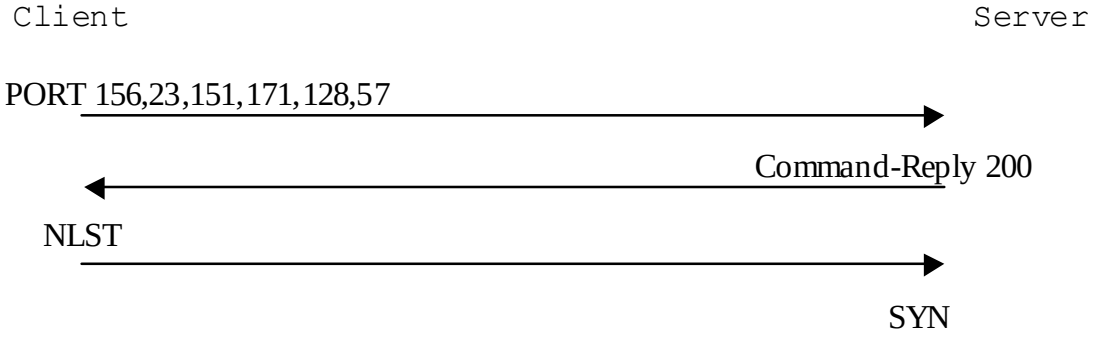


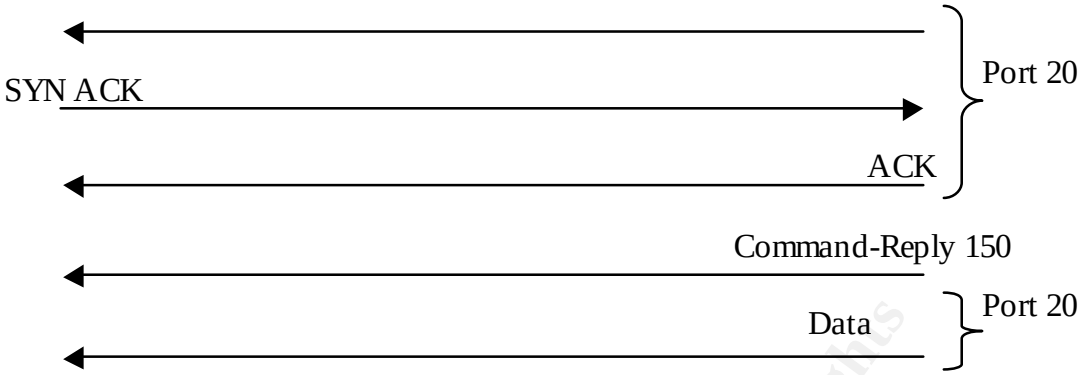
To help secure an FTP session, authentication is performed for each new session through the use of username and password. When a user starts the FTP client program they will be prompted for a username, the user's input will be transferred to the server who will then confirm it and send a request, in the form of a command-reply, for a password. When the user inputs the password the server then authenticates the session by comparing the password. After the server has confirmed the user is authentic then the service is open to the user to send commands to the server.



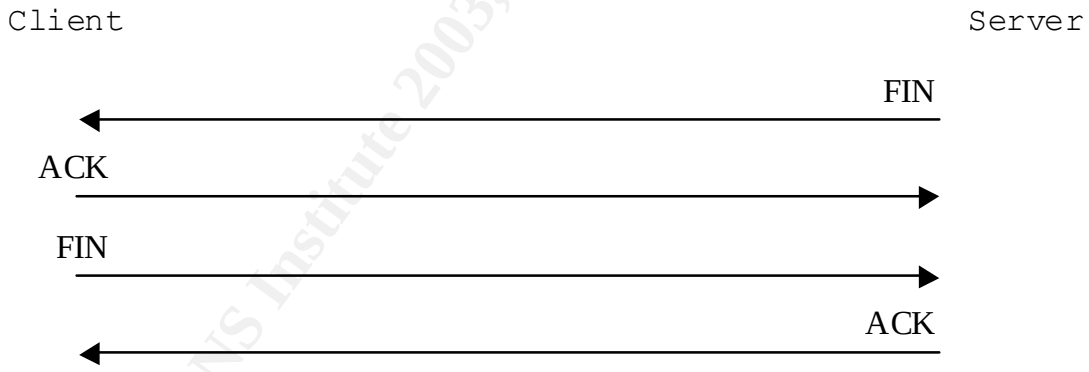
During the initial connection and authentication along with certain commands such as *pwd(print working directory)* and *cd(change directory)* the control port is used to keep track of the session. The control port consists of the server's FTP port, commonly port 21, and an available client port, randomly chosen.

The data port is used when data is transferred between the client and server. The data port consists of a server port, commonly port 20, and an available port from the client, randomly chosen. Commands such as *ls, get,* and *put* use the control port to transfer data. For every data port command that is send to the server, a new data port is open to transfer the data and then closed when it is completed. The client uses the *port* command to tell the server which port it is opening up to allow the server to send data. Thus the server must make a connection from it's own port, usually port 20, to the client specified port. Below details the network traffic of an *ls* command.





Whenever the server or client completes the transfer of data through the data port, the server then closes the data port. This is also done for the control port when the client completes the FTP session and wants to end the session. The server makes use of the FIN flag whenever it wishes to close either a data port or a control port. The sequence of packets needed to close a port connection is show below.



Hijack Analysis

Once a malicious user gains access to the FTP session traffic he can now begin to monitor the session and wait for an opportunity to hijack the session. A hijack occurs when the attacker is able to intercept the communication

between the client and server after the session has been authenticated. The simplest method to hijack the session would be to send a reset to the user forcing the client application to close the FTP session but he also have to prevent the client from resetting the port on the server end. If he does not prevent this packet from reaching the server then the connection will be terminated and he will have to wait for another opportunity to hijack a session. Once he has successfully closed the client, he now has the opportunity to send queries to the server requesting files or upload his own malicious files to the server. Since he was monitoring the entire session between the server and client he has the right sequence number and acknowledgement number so that the server thinks its still communicating with the original client.

If the attacker chooses to keep both the client and server running then he will have to keep track of the sequence number and acknowledgement number being sent between the client and server. Any command that the attacker sends to the server will change the sequence number and acknowledgement numbers and will cause the client and server to be out of synchronization and they will not be able to communicate thus causing the connection to close. This method is more difficult because the attacker now has to continually change the client and server sequence/acknowledgement numbers to reflect the commands that he injected towards the server and the data he received from the server.

Protection from FTP Hijack

The best way of protection is to put defenses around they system. Firewalls will help by blocking off certain network packets and Intrusion Detection Systems will help find any discrepancies within network packets. It is possible for the IDS to discover spoofed packets if they compare the hostname, IP, and MAC addresses making it that much more difficult for attackers to spoof addresses. Since it is difficult for the client to know that their sessions are being monitored and even hijacked, the best they can do is to check their files for any sudden changes or differences that should not be there. They can only rely on the network administrator to find and prevent any intruders from accessing protected files. During any FTP sessions it is always possible to have a connection dropped

thus making it more difficult to determine if someone is altering the data or if it was just lost.

Conclusion

Regardless of how much you harden your system or what kind of defenses you put around your system nothing is full proof. It is always a good idea to consistently audit your machine and files to make sure nothing out of the ordinary occurred. Any sudden changes or glitches should be checked for possible intrusions. If the attacker were able to disguise the spoofed FTP data and bypass both the firewall and intrusion detection system then it would be very difficult for even a network administrator to determine which is which. There is simply too much network traffic for network administrators to verify one by one. FTP hijack is far from impossible to implement. Programs such as T-Sight have the feature to allow its users to hijack ftp sessions. T-Sight is an advanced intrusion detection system that allows users to monitor the network traffic and help determine possible break-in or compromises to the system. While it serves as an intrusion detection system it also has the feature of performing an FTP hijack. This can be useful to test if your system is vulnerable of being a victim of FTP hijack.

Reference

Boran, Sean. "Hardening the BIND DNS Server" URL: http://softwaredev.earthweb.com/devtools/sdlin/article/0,,12308_625181_1,00.html

Doeppner, Thomas W. "DNS Cache Poisoning". URL: http://www.cs.brown.edu/courses/cs196-5/docs/lect/09/sld071.htm (22 April, 2001)

Postel J. / Reynolds J. "FILE TRANSFER PROTOCOL (FTP)". URL: http://war.jgaa.com/ftp/rfc/rfc959.txt (Oct. 1985)

Smith, Nicholas J. "What's Lurking in the Ether? Security in an Ethernet LAN Environment". URL: http://www.sans.org/infosecFAQ/firewall/ethernet.htm (4 July, 2000)

"Explanation of ICMP Redirect Behavior". URL: http://support.microsoft.com/support/kb/articles/Q195/6/86.ASP (10 Aug. 2001)

"Network Security". URL: http://www.idatanet.net/networking.html

"T-Sight-On Target Security". URL: http://www.engarde.com/software/t-sight