sftp是Secure File Transfer Protocol的缩写，安全[文件传送协议](http://baike.baidu.com/item/%E6%96%87%E4%BB%B6%E4%BC%A0%E9%80%81%E5%8D%8F%E8%AE%AE)。可以为传输文件提供一种安全的加密方法。sftp 与 ftp 有着几乎一样的语法和功能。SFTP 为 [SSH](http://baike.baidu.com/item/SSH/10407)的一部分，是一种传输档案至 Blogger 伺服器的安全方式。其实在SSH软件包中，已经包含了一个叫作SFTP(Secure File Transfer Protocol)的安全文件传输子系统，SFTP本身没有单独的[守护进程](http://baike.baidu.com/item/%E5%AE%88%E6%8A%A4%E8%BF%9B%E7%A8%8B)，它必须使用sshd守护进程（[端口](http://baike.baidu.com/item/%E7%AB%AF%E5%8F%A3)号默认是22）来完成相应的连接操作，所以从某种意义上来说，SFTP并不像一个[服务器](http://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8)程序，而更像是一个客户端程序。SFTP同样是使用加密传输认证信息和传输的数据，所以，使用SFTP是非常安全的。但是，由于这种传输方式使用了加密/[解密技术](http://baike.baidu.com/item/%E8%A7%A3%E5%AF%86%E6%8A%80%E6%9C%AF)，所以[传输效率](http://baike.baidu.com/item/%E4%BC%A0%E8%BE%93%E6%95%88%E7%8E%87)比普通的[FTP](http://baike.baidu.com/item/FTP/13839)要低得多，如果您对网络安全性要求更高时，可以使用SFTP代替FTP。

连接方法

windows中可以使用Core FTP，FileZilla, WinSCP，[Xftp](http://baike.baidu.com/item/Xftp" \t "_blank)来连接SFTP进行上传，下载文件，建立，删除目录等操作。

linux下直接在终端中输入：sftp username@remote ip(or remote host name)。出现验证时，只需填入正确的密码即可实现远程链接。登入成功后终端呈现出:sftp>....

在sftp的环境下的操作就和一般ftp的操作类似了，ls,rm,mkdir,dir,pwd,等指令都是对远端进行操作，如果要对本地操作，只需在上述的指令上加‘l’变为：lls,lcd, lpwd等。当然既然是ftp，当然得说它的上传和下载咯！

上传：put /path/filename(本地主机) /path/filename(远端主机)；

下载：get /path/filename(远端主机) /path/filename([本地主机](http://baike.baidu.com/item/%E6%9C%AC%E5%9C%B0%E4%B8%BB%E6%9C%BA))。

另外提一下sftp在非正规端口（正规的是22号端口）登录：sftp -o port=1000 username@remote ip.

FTP与SFTP的区别

FTP是文件传输协议。在网站上，如果你想把文件和人共享，最便捷的方式莫过于把文件上传到FTP服务器上，其他人通过FTP客户端程序来下载所需要的文件。

FTP进行文件传输需要通过端口进行。一般所需端口为：

1.       控制链路—TCP端口21。控制器端。用于发送指令给服务器以及等待服务器响应。

2.       数据链路---TCP端口20。数据传输端口。用来建立数据传输通道的。主要用来从客户向服务器发送一个文件、从服务器向客户发送一个文件、从服务器向客户发送文件或目录列表。

FTP为了适应不同的网络环境，支持主动连接和被动连接两种模式。这两种模式都主要针对数据链路进行的，跟控制链路无关。

FTP的安全隐患：

一、FTP服务器软件漏洞。

二、明文口令。

三、FTP旗标。

四、通过FTP服务器进行端口扫描。

五、数据劫持。

FTP的安全策略：

一、使用较比安全的系统和FTP服务软件。

二、使用密文传输用户名和口令。

三、更改服务软件的旗标。

四、加强协议安全性。

# Sftp和ftp over ssh2的区别

最近使用SecureFx，涉及了两个不同的安全文件传输协议：

-sftp

-ftp over SSH2

这两种协议是不同的。sftp是ssh内含的协议，只要sshd服务器启动了，它就可用，它本身不需要ftp服务器启动。ftp over SSH2则象一个二传手。

1、SFTP的工作模式：

图1显示了SFTP的工作模式，它是作为SSH2的一个子服务工作的。

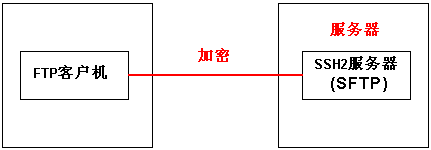


图 1 SFTP工作模式

2、FTP over SSH2

此协议还是基于ftp协议的。在此协议中SSH2服务器起了一个桥梁作用，把数据在客户端和ftp之间转发。ftp协议本身包括两个通道，一个是控制通道，另一个是数据通道。

FTP over SSH2有两种情况，半安全连接（Less Secure Connection）和安全连接（Full Secure Connection）。在半安全连接时，ftp客户端先和SSH2服务器连接，在这个连接中无论控制通道和数据通道都是加密的。但是SSH2服务器和 ftp服务器之间就不是加密的了，如果ftp服务器运行在另外一台机器上，SSH2服务器和ftp直接就是明文传输。见图2。

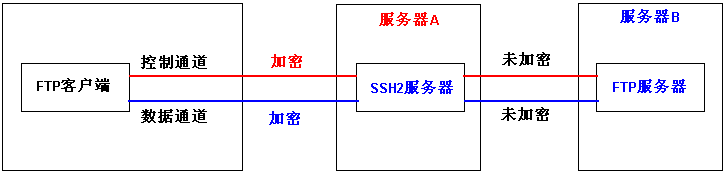


图 2半安全连接

图3是安全连接模式的情形，SSH2服务器和FTP服务器在同一台服务器上。

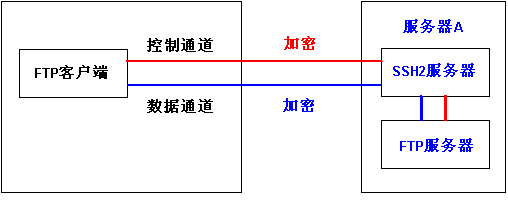


图 3 安全连接

# 

# 

# FTP(文件传输协议)工作原理

目前在网络上,如果你想把文件和其他人共享。最方便的办法莫过于将文件放FTP服务器上，然后其他人通过FTP客户端程序来下载所需要的文件。

**1、FTP**[**架构**](http://lib.csdn.net/base/architecture)

          如同其他的很多通讯协议，FTP通讯协议也采用客户机 / 服务器（Client / Server ）架构。用户可以通过各种不同的FTP客户端程序，借助FTP协议，来连接FTP服务器，以上传或者下载文件。

**2、FTP通讯端口知识**

          FTP服务器和客户端要进行文件传输，就需要通过端口来进行。FTP协议需要的端口一般包括两种：

**控制链路--------TCP端口21**

          所有你发往FTP服务器的命令和服务器反馈的指令都是通过服务器上的21端口传送的。

**数据链路--------TCP端口20**

          数据链路主要是用来传送数据的，比如客户端上传、下载内容，以及列目录显示的内容等。

**3、FTP连接的两种方式**

          在数据链路的建立上，FTP         Server         为了适应不同的网络环境，支持两种连接模式：主动模式（Port）和被动模式（Pasv)。其实这两种连接模式主要是针对数据链路进行的，和控制链路无关。

**主动模式**

          主动模式是这样工作的：客户端把自己的高位端口和服务器端口21建立控制链路。所有的控制命令比如Is或get都是通过这条链路传送的。

          当客户端需要服务器端给它传送数据时，客户端会发消息给服务器端，告诉自己的位置和打开的高位端口（一般大于1024的端口都就叫高位端口），等候服务器 的20端口和客户端打开的端口进行连接，从而进行数据的传输。当服务器端收到信息后，就会和客户端打开的端口连接，这样数据链路就建立起来了。

          采用主动模式连接服务器的客户端，当它位于NAT或者防火墙的保护之后时会碰到连接失败的问题。这是因为当防火墙接到服务器发送过来的信息的时候，并不知道应该发送给内部网络中的哪一台客户端造成的。

**被动模式**

         被动模式是这样工作的：当客户端发送数据请求后，服务器也会发信息给客户端，告诉客户端：服务器在本地打开了一个高位端口P，你现在来连接我吧。当客户端收到该信息时，就会去连接服务器端的端口P，连接成功后，数据链路就建立了。

          从上面的解释中我们可以看到，两种模式主要的不同是数据连接建立的不同。对于Port模式，是客户端在本地打开一个端口等服务器去连接建立数据连接，而Pasv模式就是服务器打开一个端口等待客户端去建立一个数据连接。

### 浅析文件传输协议 (ftp) 的工作原理

起初，FTP并不是应用于IP网络上的协议，而是ARPANEt网络中计算机间的文件传输协议， ARPANET是美国国防部组建的老网络，于1960-1980年使用。在那时， FTP的主要功能是在主机间高速可靠地传输文件。目前FTP仍然保持其可靠性，即使在今天，它还允许文件远程存取。这使得用户可以在某个系统上工作，而将 文件存贮在别的系统。例如，如果某用户运行Web服务器，需要从远程主机上取得HTML文件和CGI程序在本机上工作，他需要从远程存储站点获取文件(远 程站点也需安装Web服务器)。当用户完成工作后，可使用FTP将文件传回到Web服务器。采用这种方法，用户无需使用Telnet登录到远程主机进行工 作，这样就使Web服务器的更新工作变得如此的轻松。  
  
    FTP是TCP/IP的一种具体应用，它工作在OSI模型的第七层，TCP模型的第四层上，即应用层，使用TCP传输而不是UDP，这样FTP客户在和服 务器建立连接前就要经过一个被广为熟知的"三次握手"的过程，它带来的意义在于客户与服务器之间的连接是可靠的，而且是面向连接，为数据的传输提供了可靠 的保证。  
  
    下面，让我们来看看，一个FTP客户在和服务器连接是怎么样的一个过程（以标准的FTP端口号为例）。  
  
    首先，FTP并不像HTTP协议那样，只需要一个端口作为连接（HTTP的默认端口是80，FTP的默认端口是21），FTP需要2个端口，一个端口是作 为控制连接端口,也就是21这个端口，用于发送指令给服务器以及等待服务器响应；另一个端口是数据传输端口,端口号为20（仅PORT模式），是用来建立 数据传输通道的，主要有3个作用   
  
  
从客户向服务器发送一个文件。   
  
  
从服务器向客户发送一个文件。   
  
  
从服务器向客户发送文件或目录列表。   
  
    其次，FTP的连接模式有两种，PORT和PASV。PORT模式是一个主动模式，PASV是被动模式，这里都是相对于服务器而言的。为了让大家清楚的认识这两种模式，朗月繁星分别举例说明。  
PORT模式  
  
    当FTP客户以PORT模式连接服务器时，他动态的选择一个端口号（本次试验是6015）连接服务器的21端口，注意这个端口号一定是1024以上的，因 为1024以前的端口都已经预先被定义好，被一些典型的服务使用，当然有的还没使用，保留给以后会用到这些端口的资源服务。当经过TCP的三次握手后，连 接（控制信道）被建立（如图1和图2）。



图1：FTP客户使用FTP命令建立于服务器的连接

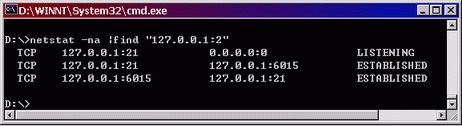


图2：用netstat命令查看，控制信道被建立在客户机的6015和服务器的20端口

    现在用户要列出服务器上的目录结构(使用ls或dir命令)，那么首先就要建立一个数据通道，因为只有数据通道才能传输目录和文件列表，此时用户会发出 PORT指令告诉服务器连接自己的什么端口来建立一条数据通道（这个命令由控制信道发送给服务器），当服务器接到这一指令时，服务器会使用20端口连接用 户在PORT指令中指定的端口号，用以发送目录的列表（如图3）。

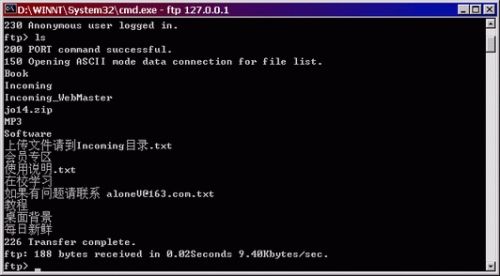


图3：ls命令是一个交互命令，它会首先与服务器建立一个数据传输通道。经验证本次试验客户机使用6044端口

    当完成这一操作时，FTP客户也许要下载一个文件，那么就会发出get指令，请注意，这时客户会再次发送PORT指令，告诉服务器连接他的哪个"新"端 口，你可以先用netstat -na这个命令验证，上一次使用的6044已经处于TIME\_WAIT状态(如图4)。

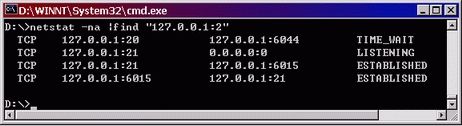


图4：使用netstat命令验证上一次使用ls命令建立的数据传输通道已经关闭

    当这个新的数据传输通道建立后(在微软的系统中，客户端通常会使用连续的端口，也就是说这一次客户端会用6045这个端口)，就开始了文件传输的工作。   
  
    PASV模式  
  
    然而，当FTP客户以PASV模式连接服务器时，情况就有些不同了。在初始化连接这个过程即连接服务器这个过程和PORT模式是一样的，不同的是，当 FTP客户发送ls、dir、get等这些要求数据返回的命令时，他不向服务器发送PORT指令而是发送PASV指令，在这个指令中，用户告诉服务器自己 要连接服务器的某一个端口，如果这个服务器上的这个端口是空闲的可用的，那么服务器会返回ACK的确认信息，之后数据传输通道被建立并返回用户所要的信息 （根据用户发送的指令，如ls、dir、get等）；如果服务器的这个端口被另一个资源所使用，那么服务器返回UNACK的信息，那么这时，FTP客户会 再次发送PASV命令，这也就是所谓的连接建立的协商过程。为了验证这个过程我们不得不借助CUTEFTP Pro这个大家经常使用的FTP客户端软件，因为微软自带的FTP命令客户端，不支持PASV模式。虽然你可以使用QUOTE PASV这个命令强制使用PASV模式，但是当你用ls命令列出服务器目录列表，你会发现它还是使用PORT方式来连接服务器的。现在我们使用 CUTEFTP Pro以PASV模式连接服务器（如图5）。

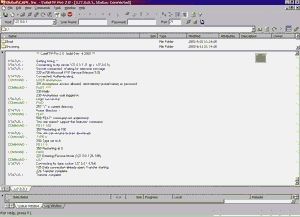


图5：使用CUTEFTP Pro以PASV模式连接服务器

    请注意连接LOG里有这样几句话：   
  
COMMAND:> PASV  
227 Entering Passive Mode (127,0,0,1,26,108)  
COMMAND:> LIST  
STATUS:> Connecting ftp data socket 127.0.0.1: 6764...  
125 Data connection already open; Transfer starting.  
226 Transfer complete.  
  
  
  
    其中，227 Entering Passive Mode (127,0,0,1,26,80). 代表客户机使用PASV模式连接服务器的26x256+108=6764端口。（当然服务器要支持这种模式）  
125 Data connection already open; Transfer starting.说明服务器的这个端口可用，返回ACK信息。  
  
    再让我们看看用CUTEFTP Pro以PORT模式连接服务器的情况。其中在LOG里有这样的记录：   
  
COMMAND:> PORT 127,0,0,1,28,37  
200 PORT command successful.  
COMMAND:> LIST  
150 Opening ASCII mode data connection for /bin/ls.  
STATUS:> Accepting connection: 127.0.0.1:20.  
226 Transfer complete.  
STATUS:> Transfer complete.  
  
  
    其中，PORT 127,0,0,1,28,37告诉服务器当收到这个PORT指令后，连接FTP客户的28x256+37=7205这个端口。  
Accepting connection: 127.0.0.1:20表示服务器接到指令后用20端口连接7205端口，而且被FTP客户接受。  
  
    比较分析  
  
    在这两个例子中，请注意: PORT模式建立数据传输通道是由服务器端发起的，服务器使用20端口连接客户端的某一个大于1024的端口；在PASV模式中，数据传输的通道的建立是 由FTP客户端发起的，他使用一个大于1024的端口连接服务器的1024以上的某一个端口。如果从C/S模型这个角度来说，PORT对于服务器来说是 OUTBOUND，而PASV模式对于服务器是INBOUND，这一点请特别注意，尤其是在使用防火墙的企业里，比如使用微软的ISA Server 2000发布一个FTP服务器，这一点非常关键，如果设置错了，那么客户将无法连接。  
  
    最后，请注意在FTP客户连接服务器的整个过程中，控制信道是一直保持连接的，而数据传输通道是临时建立的。  
  
    在本文中把重点放到了FTP的连接模式，没有涉及FTP的其他内容，比如FTP的文件类型（Type），格式控制（Format control）以及传输方式（Transmission mode）等。不过这些规范大家可能不需要花费过多的时间去了解，因为现在流行的FTP客户端都可以自动的选择正确的模式来处理，对于FTP服务器端通常 也都做了一些限制，如下：   
  
  
类型：A S C I I或图像。   
  
  
格式控制：只允许非打印。   
  
  
结构：只允许文件结构。   
  
  
传输方式：只允许流方式   
  
    至于这些内容，限于篇幅在这里就不想再介绍了。希望这篇文章能对大家有些帮助，特别是正在学习ISA Server2000的朋友和一些对FTP不很了解的朋友。

["+userLink+"](http://blog.163.com/chjxx@126/blog/static/163075679201032771155821/%22+userLink+%22) "; $('miniAd').show(); } }, on

Failure: function(){} }}); } showMiniAd();

**FTP协议安全分析**

前 言   
FTP(File Transfer Protocol,文件传输协议)是互联网上常用的协议之一，人们用FTP实现互连网上的文件传输。由于TCP/IP协议族在设计时是处在一个相互信任的 平台上的，使得在网络安全越来越被重视的今天，TCP/IP协议族的安全性也成为了安全界研究的一个重点，著名的ARP欺骗，交换环境下的数据监听，中间 人攻击，以及DDOS，都利用了TCP/IP协议的脆弱性，FTP协议也或多或少的存在着一些问题，本文从FTP协议本身出来，探讨一下FTP协议的安全 性。

第一章 FTP协议   
一、协议简介   
FTP协议和HTTP协议类似，都是采用的TCP连接，但与HTTP协议不同的是，HTTP协 议的所有数据都是通过80端口进行传输(这里不考虑SSL)，而FTP把数据和命令分开来处理，我们暂且把它们分别命名为“命令通道”和“数据通道”。命 令通道一般是在我们熟悉的21端口，而数据通道通常是一个高端口。例如客户机要从FTP服务器上获取某个文件，首先由客户机登录服务器，与服务器建立连 接，这就是我们前面讲的“命令通道”，客户机从这条通道将请求文件的命令发往服务器，服务器接到此命令后将与客户机重新建立一条连接，这就是我们面前讲的 “数据通道”，文件数据将通过数据通道传送到客户机。这里我们用图1-1来表示文件传输中的处理过程

用户接口   
|   
| 命令通道   
用户协议解释器--------------服务器协议接口   
|   
| 数据通道   
用户数据传输功能-----------服务器数据传输功能

二、数据表示   
FTP协议规范提供了控制文件传送与存储的多种选择。在以下四个方面都须作出一个选择。   
1、文件类型   
(1)ASCII码文件类型 一般情况下ASCII码文件类型是默认选择的。   
(2)EBCDIC文件类型 该类型传输方式要求两端都是EBCDIC系统。   
(3)图像文件类型(也称二进制类型) 数据发送呈现为一个比特流，通常用于传输二进制文件   
(4)本地文件类型 该方式在具有不同字节大小的主机之间传输二进制文件。

2、格式控制   
该选项只对ASCII和EBCDIC文件有效。   
(1)非打印   
(2)远程登录格式控制   
(3)Fortran回车控制

3、结构   
(1)文件结构   
(2)记录结构   
(3)页结构

4、传输方式   
(1)流方式   
(2)块方式   
(3)压缩方式

三、FTP命令   
FTP命令和应答是在命令通道以ASCII码开形式传送的，以下给出常用的命令及命令的相关说明：   
命令 说 明   
ABOR 放弃先前的FTP命令和数据转输   
LIST 列表显示文件或目录   
PASS 服务器上的口令   
PORT 客户IP地址和端口   
QUIT 从服务器上注销   
RETR 取一个文件   
STOR 存一个文件   
SYST 服务器返回系统类型   
TYPE 说明文件类型   
USER 服务器上的用户名

四、FTP应答   
FTP应答都是ASCII码形式的3位数字，并跟有报文选项。3位数字每一位都有不同的意义，这里给出一些常见的反回数字：   
125 数据通道已经打开;传输开始。   
200 就绪命令。   
214 帮助报文。   
331 用户名就绪，要求输入口令。   
425 不能打开数据通道。   
500 语法错误(未认可命令)。   
501 语法错误(无效参数)。   
502 未实现的MODE(方式命令)类型。

五、连接管理   
我们在前面讲了FTP的命令传输和数据传输是通过不同的端口进行传输的，连接管理也就理所当然的成了FTP协议的核心问题。其中最关键的又是数据通道的管理。   
数据通道有以下三大用途：   
1>从客户向服务器发送一个文件。   
2>从服务器向客户发送一个文件。   
3>从服务器向客户发送文件域目录列表。   
其中传输模式又有主动FTP和被动FTP之分。主动FTP的一般过程如下：   
1、客户发出命令建立数据连接。   
2、客户通常在客户端主机上为所在数据连接选择一个临时端口号。客户从该端口发布一个被动的打开。   
3、客户使用PORT命令从命令通道把数据通道的端口发向服务器。   
4、服务器在命令通道上接收端口号，并向客户端主机上的端口发存一个主动的打开。这时服务器的数据通道使用的是20端口。   
而被动FTP与主动FTP不同的是它不是由客户主机开一个临时端口，而是由服务器开一个临时端口等待客户机的连接。Fedora自带的vsftp就是采用的被动的传输模式。下面是从登录vsftp，到执行ls,再到退出的全过程。   
[xinhe@xinhe xinhe]$ ftp -d xxx.xxx.xxx.xxx   
Connected to xxx.xxx.xxx.xxx (xxx.xxx.xxx.xxx).   
220 Welcome to ylyz FTP service.   
Name (xxx.xxx.xxx.xxx:xinhe): xplore   
---> USER xinhe   
331 Please specify the password.   
Password:   
---> PASS XXXX   
230 Login successful.   
---> SYST   
215 UNIX Type: L8   
Remote system type is UNIX.   
Using binary mode to transfer files.   
ftp> ls   
ftp: setsockopt (ignored): Permission denied   
---> PASV   
227 Entering Passive Mode (xxx,xxx,xxx,xxx,204,73)   
---> LIST   
150 Here comes the directory listing.   
drwxr-xr-x 11 48 48 4096 Jul 21 10:52 xxx   
drwxr-xr-x 19 48 48 4096 Jul 31 14:18 xxxx   
drwx------ 2 0 0 16384 Jun 23 03:18 lost+found   
drwxr-xr-x 3 510 510 4096 Aug 03 05:42 software   
drwxr-xr-x 2 510 510 4096 Jun 30 09:34 tmp   
drwxr-xr-x 6 510 510 4096 Jun 27 08:17 xxxxx   
drwxrwxr-x 10 501 501 4096 Aug 12 20:35 xxxxxxx   
-rw-r--r-- 1 510 510 12649185 Aug 12 20:34 xxxx   
drwxr-xr-x 7 501 12 4096 Jul 03 15:13 xinhe   
drwxr-xr-x 7 510 510 4096 Aug 13 19:08 zwell   
226 Directory send OK.   
Ftp> bye   
---> QUIT   
221 Goodbye.

以上就是登录某台vsftp的服务器，执行了一个ls然后再退出的全过程，以上是采用的被动传输模式，关键看这一句：   
---> PASV   
227 Entering Passive Mode (xxx,xxx,xxx,xxx,204,73)

这一句告诉了我们服务器的IP和开临时数据端口，接着便是登录到这一临时端口 52297 ，临时端口的**[算法](http://lib.csdn.net/base/datastructure" \o "算法与数据结构知识库" \t "_blank)**：204\*256+73。为了更清楚的了解之过程，我们对这一过程的数据传送进行了监视。以下是客户机登录临时端口过程中的一段   
08/24-15:24:24.052846 0:E0:4C:F0:E0:EA -> 0:D0:F8:51:FC:81 type:0x800 len:0x4A   
192.168.10.8:32791 -> xxx.xxx.xxx.xxx:52297 TCP TTL:64 TOS:0x0 ID:39780 IpLen:20 DgmLen:60 DF   
\*\*\*\*\*\*S\* Seq: 0x42206DD2 Ack: 0x0 Win: 0x16D0 TcpLen: 40   
TCP Options (5) => MSS: 1460 SackOK TS: 849590 0 NOP WS: 0   
注：由于测式过程中的服务器是公网上的真实服务器，故屏弊其地址。

第二章 安全隐患

以上我们讨论了FTP协议本身和FTP的具体传输过程，在这一过程中，很多地方都存在着安全隐患，随着互联网和普及了深入，网络安全也越来越被人们 重视，在这里我把一些常见的关于FTP的安全隐患提出来，希望引起人们对FTP安全的重视，使FTP服务器和数据传输过程更加安全。   
一、FTP服务器软件漏洞   
这 类安全隐患不是本文讨论的重点，但是在这里必须把它提出来，因为它对于FTP服务供应商来说就是恶梦，也是倍受黑客们关注的焦点，常用的FTP服务软件有 Wu-ftpd, ProFTPD,vsftpd,以及windows下常用的Serv-U等，最常见也最可怕的漏洞就是缓冲区溢出，近来Wu-ftpd和Serv-U的溢 出漏洞层出不穷，ProFTPD也出现过缓冲区溢出，目前比较安全的还是vsftp，必竞是号称非常安全的FTP。

二、明文口令   
前面讲过了，TCP/IP协议族的设计在地相互信任和安全的基础上的，FTP的设计当然也没有采用加密传送，这样的话，FTP客户与服务器之前所有的数据传送都是通过明文的方式，当然也包括了口令。   
至 从有了交换环境下的数据监听之后，这种明文传送就变得十分危险，因为别人可能从传输过程过捕获一些敏感的信息，如用户名和口令等。像HTTPS和SSH都 采用加密解决了这一问题。而FTP仍然是明文传送，而像UINX和**[Linux](http://lib.csdn.net/base/linux" \o "Linux知识库" \t "_blank)**这类系统的ftp账号通常就是系统帐号，(vsftp就是这样做的)。这样黑 客就可以通过捕获FTP的用户名和口令来取得系统的帐号，如果该帐号可以远程登录的话，通常采用本地溢出来获得root权限。这样这台FTP服务器就被黑 客控制了。   
以下是我捕获的明文传送的数据：   
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

08/24-15:24:13.511233 0:E0:4C:F0:E0:EA -> 0:D0:F8:51:FC:81 type:0x800 len:0x4F   
192.168.10.8:32790 -> xxx.xxx.xxx.xxx:21 TCP TTL:64 TOS:0x10 ID:36423 IpLen:20 DgmLen:65 DF   
\*\*\*AP\*\*\* Seq: 0x407F7F77 Ack: 0x1BD963BF Win: 0x16D0 TcpLen: 32   
TCP Options (3) => NOP NOP TS: 848536 1353912910   
55 53 45 52 20 78 70 6C 6F 72 65 0D 0A USER xinhe..

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

08/24-15:24:13.557058 0:D0:F8:51:FC:81 -> 0:E0:4C:F0:E0:EA type:0x800 len:0x42   
xxx.xxx.xxx.xxx:21 -> 192.168.10.8:32790 TCP TTL:56 TOS:0x0 ID:29145 IpLen:20 DgmLen:52 DF   
\*\*\*A\*\*\*\* Seq: 0x1BD963BF Ack: 0x407F7F84 Win: 0x16A0 TcpLen: 32   
TCP Options (3) => NOP NOP TS: 1353916422 848536

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

08/24-15:24:13.560516 0:D0:F8:51:FC:81 -> 0:E0:4C:F0:E0:EA type:0x800 len:0x64   
xxx.xxx.xxx.xxx:21 -> 192.168.10.8:32790 TCP TTL:56 TOS:0x0 ID:29146 IpLen:20 DgmLen:86 DF   
\*\*\*AP\*\*\* Seq: 0x1BD963BF Ack: 0x407F7F84 Win: 0x16A0 TcpLen: 32   
TCP Options (3) => NOP NOP TS: 1353916426 848536   
33 33 31 20 50 6C 65 61 73 65 20 73 70 65 63 69 331 Please speci   
66 79 20 74 68 65 20 70 61 73 73 77 6F 72 64 2E fy the password.   
0D 0A ..

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

08/24-15:24:13.571556 0:E0:4C:F0:E0:EA -> 0:D0:F8:51:FC:81 type:0x800 len:0x42   
192.168.10.8:32790 -> xxx.xxx.xxx.xxx:21 TCP TTL:64 TOS:0x10 ID:36424 IpLen:20 DgmLen:52 DF   
\*\*\*A\*\*\*\* Seq: 0x407F7F84 Ack: 0x1BD963E1 Win: 0x16D0 TcpLen: 32   
TCP Options (3) => NOP NOP TS: 848542 1353916426

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

08/24-15:24:21.364315 0:E0:4C:F0:E0:EA -> 0:D0:F8:51:FC:81 type:0x800 len:0x54   
192.168.10.8:32790 -> xxx.xxx.xxx.xxx:21 TCP TTL:64 TOS:0x10 ID:36425 IpLen:20 DgmLen:70 DF   
\*\*\*AP\*\*\* Seq: 0x407F7F84 Ack: 0x1BD963E1 Win: 0x16D0 TcpLen: 32   
TCP Options (3) => NOP NOP TS: 849321 1353916426   
50 41 53 53 20 78 70 6C 6F 72 65 5F 32 30 30 34 PASS test   
0D 0A ..

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+   
这样就我们就可以看到该ftp服务器上的用户名是:xinhe和密码:test。

三、FTP旗标   
这个问题相对来说不是很严重，现在很多服务软件都有这类问题，黑客在发起攻击之前一般要先确定对方所用的版本号。这样便于选择攻击程序。以下是一个例子：   
[xinhe@xinhe xinhe]$ ftp xxx.xxx.xxx.xxx   
Connected to xxx.xxx.xxx.xxx (xxx.xxx.xxx.xxx).   
220-Serv-U FTP Server v5.1 for WinSock ready...   
220 S TEAM   
这此信息我们可知该服务器使用的服务软件可能就是Serv-U 5.1

四、通过FTP服务器进行端口扫描   
FTP客户端所发送的PORT命令告诉服务器FTP服务器传送数据时应当连向的IP和端口，通常，这就是FTP客户所在机器的IP地址及其所绑定的端口。然而FTP协议本身并没有要求客户发送的PORT命令中必须指定自已的IP。   
利用这一点，黑客就可以通过第三方FTP服务器对目标机器进行端口扫描，这种方式一般称为FTP反射，对黑客而言，这种扫描方式具有以下两个优点：   
(1)提供匿名性   
由于端口扫描的源地址为FTP服务器的IP地址，而不是黑客的机器，所以这种方式很好的隐藏了黑客的真实IP。   
(2)避免阻塞   
由于通过第三方FTP服务器进行扫描，即使目标机器通过添加内核ACL或无效路由来自动阻塞对其进行扫描的机器，但黑客可以过不过的FTP服务器来完成其扫描工作。   
Nmap就可以实现这一扫描过程，以下是一次利用ftp服务器进行扫描的实例。   
[xinhe@xinhe xinhe]$ nmap -b xinhe:test@xxx.xxx.xxx.xxx:21 -v xxx.xxx.xxx.xxx   
Hint: if your bounce scan target hosts aren't reachable from here, remember to use -P0 so we don't try and ping them prior to the scan

Starting nmap 3.48 ( <http://www.insecure.org/nmap/> ) at 2004-08-24 20:16 CST   
Resolved ftp bounce attack proxy to xxx.xxx.xxx.xxx (xxx.xxx.xxx.xxx).   
Machine xxx.xxx.xxx.xxx MIGHT actually be listening on probe port 80   
Host xxx.xxx.xxx.xxx appears to be up ... good.   
Attempting connection to [ftp://xinhe:test@xxx.xxx.xxx.xxx:21](ftp://xinhe:test@xxx.xxx.xxx.xxx/)   
Connected:220 Welcome to FTP service.   
Login credentials accepted by ftp server!   
Initiating TCP ftp bounce scan against xxx.xxx.xxx.xxx at 20:16   
Adding open port 237/tcp   
Deleting port 237/tcp, which we thought was open   
Changed my mind about port 237   
Adding open port 434/tcp   
Deleting port 434/tcp, which we thought was open   
Changed my mind about port 434   
Adding open port 1509/tcp   
Deleting port 1509/tcp, which we thought was open   
Changed my mind about port 1509   
Adding open port 109/tcp   
Deleting port 109/tcp, which we thought was open   
Changed my mind about port 109   
Adding open port 766/tcp   
Deleting port 766/tcp, which we thought was open   
Changed my mind about port 766   
Adding open port 1987/tcp   
Deleting port 1987/tcp, which we thought was open   
Changed my mind about port 1987   
Adding open port 5998/tcp   
Deleting port 5998/tcp, which we thought was open   
Changed my mind about port 5998   
Adding open port 1666/tcp   
Deleting port 1666/tcp, which we thought was open   
Changed my mind about port 1666   
Adding open port 506/tcp   
Deleting port 506/tcp, which we thought was open   
Changed my mind about port 506   
caught SIGINT signal, cleaning up

五、数据劫持   
我们在前面讲了FTP的数据传输过程，同样FTP协议本身本并没有要求传输命令的客户IP和进行数据传输的客户IP一致，这样黑客就有可能劫持到客户和服务器之间传送的数据。根据数据传输的模式可把数据劫持分为主动数据劫持和被动数据劫持。   
1、被动数据劫持   
跟据前面讲的被动传输过程我们可以看出，在FTP客户端发出PASV或PORT命令之后并且在发出数据请求之前，存在一个易受攻击的窗口。如果黑客能猜到这个端口，就能够连接并载取或替换正在发送的数据。   
要实现被动数据劫持就必须知道服务器上打开的临时端口号，然后很多服务器并不是随机选取端口，而是采用递增的方式，这样黑客要猜到这个端口号就不是很难了。   
2、主动数据劫持   
主动数据劫持比被动数据劫持要困难很多，因为在主动传输的模式下是由客户打开临时端口来进行数据传输，而黑客是很难找到客户的IP和临时端口的。

第三章 安全策略   
一、使用较比安全的系统和FTP服务软件   
这里安全的系统主要是最好不要采用windows系统作服务 器，因为系统本身的安全性就很成问题，windows每年都要暴N个漏洞，一旦有溢出漏洞很可能就能拿到管理员权限。一旦系统被入侵了，运行在此系统之上 的服务也就无安全性可言。Linux和BSD都将是不错的选择。   
服务软件采用漏洞比较少的，如vsftp，而且确保版本的更新。

二、使用密文传输用户名和口令   
这里我们可以采用scp和sftp,也可以使用SSH来转发。这样即使黑客能监听到客户与服务器之间的数据交换，没有密钥也得不到口令。使用SSH转发有一些条件限制，首先要求服务器和客户端都是主动模式，然后是服务器必须允许命令通道之外的机器向其发送PORT命令。

三、更改服务软件的旗标   
更改服务软件的旗标能起到迷惑攻击者的作用，至少能迷惑很多扫描器，造成扫描器的误报，但更改旗标并不是解决安全问题的根本办法，安全漏洞不会因为旗标不同而消失，不过更改总比不改要好一些。现在大多数的服务端软件都可以在配置文件里更改该FTP的旗标。

四、加强协议安全性   
这一点是服务软件的提供商需要做的，一是对PORT命令进行检查，PORT后的IP应和客户主机是同一IP，我们对 FTP的攻击很多都是通过构造特殊的PORT命令来实现的，所以PORT命令的使用对于攻击者来说就显得尤为重要了。做到这一点并不是很容易，Wu- ftpd就花了几年的时间。目前针对数据劫持还没什么完美的防御方法，目前能做的就是检查命令通道和数据通道的IP地址是不是一致，但这也不能百分之百地 防止数据劫持的发生。因为客户机和黑客可能处于同一内网。

后 记   
网络已深入到社会生活的方方面面，网络安全也越来越显得重要，FTP协议安全只是网络安全中一个很小的部分，网络安全还有 很多工作要做，这篇文章由于时间的关系写得比较浅显，没有把FTP服务软件的源码拿出来讲(很多好的东西都是开源的),我想如果能把源码拿来对照源码讲 FTP的传输过程，这样我们会对FTP的传输过程有一个本质的了解，也更容易从中找到某些不为人知的安全隐患。

**1、FTP架构**

    如同其他的很多通讯协议，FTP通讯协议也采用客户机 / 服务器（Client / Server ）架构。用户可以通过各种不同的FTP客户端程序，借助FTP协议，来连接FTP服务器，上传或下载文件。

**2、FTP通讯端口知识**

    FTP服务器和客户端要进行文件传输，就需要通过端口来进行。FTP协议需要的端口一般包括两种：

**控制链路--------TCP端口21**

控制器端。用于发送指令给服务器以及等待服务器响应。所有你发往FTP服务器的命令和服务器反馈的指令都是通过服务器上的21端口传送的。

**数据链路--------TCP端口20**

数据传输端口。用来建立数据传输通道的。主要用来从客户向服务器发送一个文件、从服务器向客户发送一个文件、从服务器向客户发送文件或目录列表。数据链路主要是用来传送数据的，比如客户端上传、下载内容，以及列目录显示的内容等。

**3、FTP连接的两种方式**

   在数据链路的建立上，FTP Server 为了适应不同的网络环境，支持两种连接模式：主动模式（Port）和被动模式（Pasv)。其实这两种连接模式主要是针对数据链路进行的，和控制链路无关。

**主动模式**

    主动模式是这样工作的：客户端把自己的高位端口和服务器端口21建立控制链路。所有的控制命令比如Is或get都是通过这条链路传送的。当客户端需要服务器端给它传送数据时，客户端会发消息给服务器端，告诉自己的位置和打开的高位端口（一般大于1024的端口都就叫高位端口），等候服务器的20端口和客户端打开的端口进行连接，从而进行数据的传输。当服务器端收到信息后，就会和客户端打开的端口连接，这样数据链路就建立起来了。采用主动模式连接服务器的客户端，当它位于NAT或者防火墙的保护之后时会碰到连接失败的问题。这是因为当防火墙接到服务器发送过来的信息的时候，并不知道应该发送给内部网络中的哪一台客户端造成的。

**被动模式**

    被动模式是这样工作的：当客户端发送数据请求后，服务器也会发信息给客户端，告诉客户端：服务器在本地打开了一个高位端口P，你现在来连接我吧。当客户端收到该信息时，就会去连接服务器端的端口P，连接成功后，数据链路就建立了。从上面的解释中我们可以看到，两种模式主要的不同是数据连接建立的不同。对于Port模式，是客户端在本地打开一个端口等服务器去连接建立数据连接，而Pasv模式就是服务器打开一个端口等待客户端去建立一个数据连接。

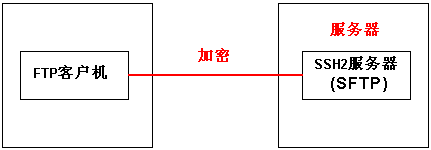
**Sftp和ftp的区别**

SFTP是Secure File Transfer Protocol的缩写，是安全文件传送协议。可以为传输文件提供一种安全的加密方法。跟ftp几乎语法功能一样。SFTP是SSH的一部分，是一种传输档案至Blogger伺服器的安全方式。它本身没有单独的守护进程，必须使用sshd守护进程来完成相应的连接操作，所以从某种意义上来说，SFTP并不像一个服务器程序，而更像是一个客户端程序。SFTP同样是使用加密传输认证信息和传输的数据，所以使用SFTP是十分安全的。但由于这种传输方式使用了加密/解密技术，所以传输效率比普通的FTP要低得多。在对网络安全性要求更高时，代替FTP使用。

Sftp和Ftp都是文件传输协议。区别：sftp是ssh内含的协议（ssh是加密的telnet协议），只要sshd服务器启动了，它就可用，而且sftp安全性较高，它本身不需要ftp服务器启动。 sftp = ssh + ftp（安全文件传输协议）。由于ftp是明文传输的，没有安全性，而sftp基于ssh，传输内容是加密过的，较为安全。目前网络不太安全，以前用telnet的都改用ssh2(SSH1已被破解)。sftp这个工具和ftp用法一样。但是它的传输文件是通过ssl加密了的，即使被截获了也无法破解。而且sftp相比ftp功能要多一些，多了一些文件属性的设置。

1、SFTP的工作模式：

图1显示了SFTP的工作模式，它是作为SSH2的一个子服务工作的。

图 1

2、FTP的工作模式：

**SFTP 和FTPS的区别**

都是为ftp连接加密，协议非常相似。**SFTP**借助ssh加密，**FTPS**借助ssl协议加密。ssh是为telnet/ftp等加密、建立传输通道而设计的，ssl是为http/smtp等加密设计的。其实ssh建立传输通道就是为了加密和传输，而且这个通道是可以用来远程登录。

功能上，ssh就像铺管子，ssl就像打包裹，铺管子和打包裹都会使数据安全，都是一个制作密钥的过程，而因为ssh是一个管子所以它很适合ftp的安全传输。原理上，sftp协议是ssh中的一条独立的协议，利用sftp服务器就可以传输数据。而ftps是ftp-over-ssl的意思，即ftp借助ssl协议加密传输，不但要用ftp服务器还要用ssl协议加密。（如果ftp-over-ssh，就是完全不同于sftp的传输方式了，就是利用ftp服务器和ssh协议加密传输数据。 ）