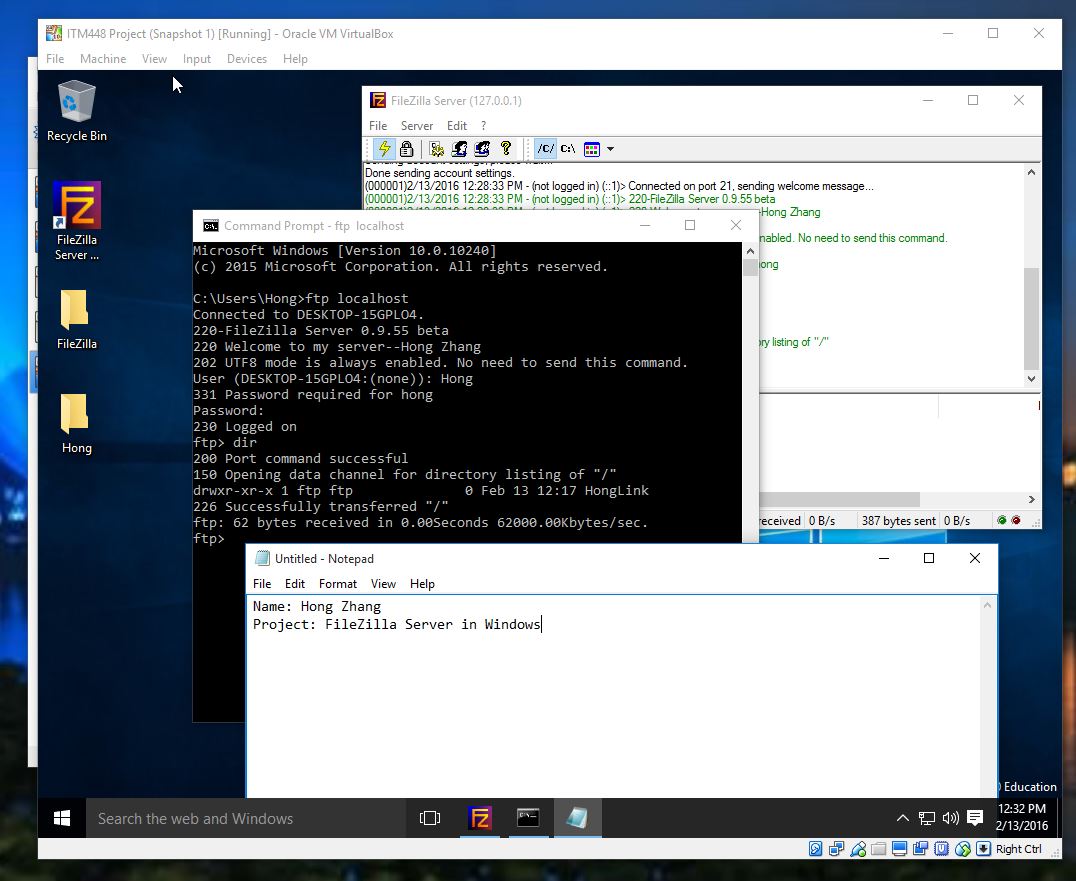
FileZilla FTP Server

FileZilla Server is a server that supports **FTP** and **FTP over SSL/TLS** which provides secure encrypted connections to the server.

## **Installation and Setup - Windows**

* Run the exe installer
* Select admin port and remember this port
* Launch FileZilla Server Interface
* Enter port from above, enter (new) password for administration, click ok/connect
* Create a user and/or group with permissions to a home directory.
* Add additional directory and set the alias name to display within home directory (e.g. /AliasName)
* Complete

<https://www.youtube.com/watch?v=m4y5_PHqXtM>



<http://www.cvedetails.com/vulnerability-list/vendor_id-2889/product_id-5062/Filezilla-Filezilla-Server.html>

<http://thehackernews.com/2014/01/warning-malicious-version-of-ftp.html>

<https://blog.avast.com/2014/01/27/malformed-filezilla-ftp-client-with-login-stealer/>

<https://www.webmasterworld.com/webmaster/4258060.htm>

<http://internet.wonderhowto.com/how-to/using-hydra-5-4-crack-ftp-passwords-352492/>

<https://integer13.wordpress.com/2011/01/12/automated-attacks-since-122010-using-compromised-credentials-chainreaction/>

<http://blog.webernetz.net/2014/05/14/filezilla-server-bug-autoban-does-not-work-with-ipv6/>

**FileZilla FTP Server Multiple Remote Buffer Overflow Vulnerabilities**

<http://www.securityfocus.com/bid/17802/info>

# Windows Gather FileZilla FTP Server Credential Collection

<https://www.rapid7.com/db/modules/post/windows/gather/credentials/filezilla_server>

# [How to Using Hydra 5.4 to crack FTP passwords](http://internet.wonderhowto.com/how-to/using-hydra-5-4-crack-ftp-passwords-352492/)

<http://internet.wonderhowto.com/how-to/using-hydra-5-4-crack-ftp-passwords-352492/>

# Install and Configure a Secure FTP Server using Filezilla with a Hostname

Step 1: Configure your network adapter to obtain a static private IP address from your router via the TCP/IP v4 properties.   
  
Step 2: Forward TCP ports 21,990 and TCP port range 3000-4000 on your router.  
  
Step 3: Install the Filezilla Server software  
  
Step 4: Allow Filezilla server software as an exception in your firewall.   
  
For example to do this in Windows Firewall you do the following: Open Windows Firewall with Advanced Security. Next, right-click on Inbound Rules and choose New Rule. Add a "program rule" and browse to the Filezilla server.exe application to add it.   
  
Step 5: Connect to the Filezilla Server interface and navigate to settings. Click on Passive Mode Settings. Click on Use Custom Port Range and input the values of 3000-4000.   
  
Step 6: Click on the SSL/TLS Settings option. On the right-side, enable the FTP over SSL/TLS support.   
  
Step 7: At the bottom click on the generate new certificate button. Create your SSL self-signed certificate. Click on OK to apply changes and close out. Ignore the port 1-65535 message if prompted.  
  
Step 8: Create a physical directory (or directories) on your server that will be used to host your files on the server. For example C:\FTP\Users\Public.   
  
Step 9: Set up your user accounts. For example you can create an anonymous user which can be used for public access.   
  
Step 10: Browse to the Shared Folders section and add the C:\FTP\Users\Public folder as the root to your FTP site.   
  
Step 11: Set up "secure only" user account by forcing SSL logon on that account and then map the shared folder to that account (C:\FTP\Users\SECUREACCOUNT)   
  
Assuming you have set up and activated your DYNDNS service simply type in ftp://YOURHOSTNAME.dyndns.org into your browser and see if your server works!   
  
To test out your secure FTP user account, download the Filezilla client software to verify that it works. In an upcoming tutorial I will show you how to do this.

<https://www.youtube.com/watch?v=-Ch2vMmaeds>

# Creating a Windows FTP Server with FileZilla Server

<https://www.youtube.com/watch?v=yMDKi2bH4Eg>

## **The problem with standard FTP**

The standard FTP protocol is unsecured. Anybody on the network (any "man in the middle", as it's called, who can read and/or modify the data on the connection) can see what you are sending, and modify the sent data. A typical FTP login goes like this:

\* TCP connection is made to the server

220 Welcome to this FTP server!

USER <username>

331 Password required for <username>

PASS <password>

<a positive or negative response>

As you can see, there is no protection of the password whatsoever. As anyone on the network can read this, an attacker can easily log in with a stolen username and password.

There are multiple ways to fix this. You could use something other than FTP, but since that's off-topic here I won't go into this. FTP itself knows two popular secured versions: FTPS and SFTP. You can keep them apart as follows: FTPS has an S in the end, just like HTTPS. And indeed, it works exactly the same as HTTPS, which is by the SSL/TLS protocol. SFTP, with a prefixed S, uses SSH to connect and is more popular on Linux servers.

I personally only have experience with FTPS, so I can't tell you the differences and advantages or disadvantages.

## **About the measures you've taken so far**

First of all, it's a good thing that you're actively seeking to secure this. Many people would simply not care!

* Changing the port number

This helps in two ways, though both are not really worth it.

* 1. Using an alternate port makes it a little bit less obvious that it's FTP traffic, but it is still very trivial to notice when anyone is actively looking to hack you and can monitor your network.
  2. I regularly see people scanning random servers on port 21 to see if there are unsecured accounts active, but a password like "1Q3XX" would already not be guessed by them (they mostly do dictionary attacks). So it's not really advantageous to change the port if your passwords are better than "admin" or "123456".
* Upload can be done from a fixed (our local) IP

This helps a little more, but it also limits you. If you are at home and notice a huge security hole in your website, you would need to drive over. With a decent password, limiting the IP should not be needed, especially if you have a server that supports automatic banning after too many invalid login attempts.

You could setup a VPN to avoid having to drive over, but what's the point if you could simply choose a strong FTP password?

## **Software suggestion**

Since you didn't mention it, I'll assume you're open to any FTP client and server implementation available for any OS.

Although I haven't tested many FTP servers, **FileZilla Server** works good for my purposes, and I guess it would work good for any small organisation. Features include:

* Multiple users and usergroups
* Allowing or denying access from given IPs, both per-user and globally for the server
* FTPS. Like HTTPS is secure HTTP, FTPS is secure FTP. It works entirely over SSL/TLS and secures the data being sent. The data (the login and the code) cannot be read or changed by someone in the middle (like a hacker on a public wifi network, or even your government or ISP). You can simply install your own certificate.
* You can force the use of FTPS per user, and also force encrypting files being sent (some clients only encrypt the login data)
* Automatic banning after too many invalid logins is possible (it bans the IP for some time, which you can set)
* Logging is supported, and you can set how long logs are stored, how large they may become, etc.
* Many more non-security-related features like speed throttling/limiting, limiting the number of concurrent client, setting which port it listens on, setting a welcome message, etc.
* The configuration is in XML, and there is an admin panel (the admin panel can be connected to remotely, and is protected by a password). This means that you can also automate adding users to the XML file if you want to.
* One disadvantage is that it is Windows-only.

If you are using a Linux server, you could look for FTP servers with similar options. **By far the most important feature to look for in a server is FTPS or SFTP, and the possibility to force that so that nobody can accidentally connect without encryption.** Besides that, autobanning and logging are rather important.

Then for **the client**, I again think that FileZilla is a very good option.

* It supports FTPS and SFTP, and even client certificates for authentication.
* You must explicitly trust the certificate sent by the server, optionally storing and trusting the certificate. If a certificate is forged by a hacker, you will know because it asks you again "do you want to trust this unknown certificate?".
* Multi-platform (runs on Windows, Mac and Linux)
* Many more options like synced directory browsing, a site manager (store logins for different servers), compare files in the local and remote directory listing based on size and modification date, limit speeds and concurrent transfers, etc.

Following steps will help you to secure FTP access

* Disable Anonymous Access
* Setup your FTP site as Blind Put
* Enable Disk Quotas
* Use Logon Time Restrictions
* Restrict Access by IP
* Audit Logon Events
* Enable Strong Password Requirement
* Enable Account Lockout and Account Lockout Threshold
* Make SSH access compulsory

<http://www.hecfblog.com/2013/09/daily-blog-93-filezilla-artifacts.html>

### Daily Blog #93: FileZilla Artifacts

With reference to the HeartBleed vulnerability found in OpenSSL (***CVE-2014-0160***) this is NOT something that affects Windows/IIS directly due to Microsoft using their own encryption component called Secure Channel (aka SChannel).

However this does not necessarily mean that your servers are not affected. The problem lies with OpenSSL (specifically versions***1.0.1***through to***1.0.1f***) So if you have any products that utilise an affected version of OpenSSL you should look to the software manufacturer as soon as possible for an update/patch.

OpenSSL v***1.0.1g***has been released which fixes the vulnerability. ([*www.openssl.org*](http://www.openssl.org/))

The most commonly used software across Windows that has been identified so far is FileZilla Server. Filezilla have just recently released version 0.9.44 of their server software which addresses this issue. Any versions prior to this are most likely vulnerable.

Other products/software may be affected but we suggest you speak to the software vendors directly with any concerns you may have.