Resource taken from : https://blogs.msdn.microsoft.com/mast/2013/12/12/setting-up-a-passive-ftp-server-in-windows-azure-vm/

Setting up a Passive FTP Server in Windows Azure VM

★★★★★
★★★★
★★★
★★
★

Pradeep M G December 12, 2013 9

-
- 0
- 0

This post is authored by Lalitesh Kumar, Pradeep M G and reviewed by Avinash Venkat Reddy. Also special thanks to Adam Conkle and Craig Landis for providing the "points to consider from Azure SLB perspective".

FTP may run in active or passive mode. Passive mode is extensively used to solve the issue of the client firewall blocking the FTP server data connection. Detailed information on FTP server modes here.

Setting up a Passive FTP server in Windows Azure VM involves the following steps:

1. Deploying a Windows Azure VM
2. Installing FTP service on Windows Azure VM
3. Adding the FTP site to IIS Manager on Windows Azure VM
4. Specify the data channel port for passive FTP connection on Windows Azure VM
5. Adding ports specified in the previous step as endpoint to the VM
6. Adding Firewall rules to allow traffic on the added endpoint
7. Verifying that FTP server is using the port previously specified under data channel port
8. Points to consider from Azure SLB perspective

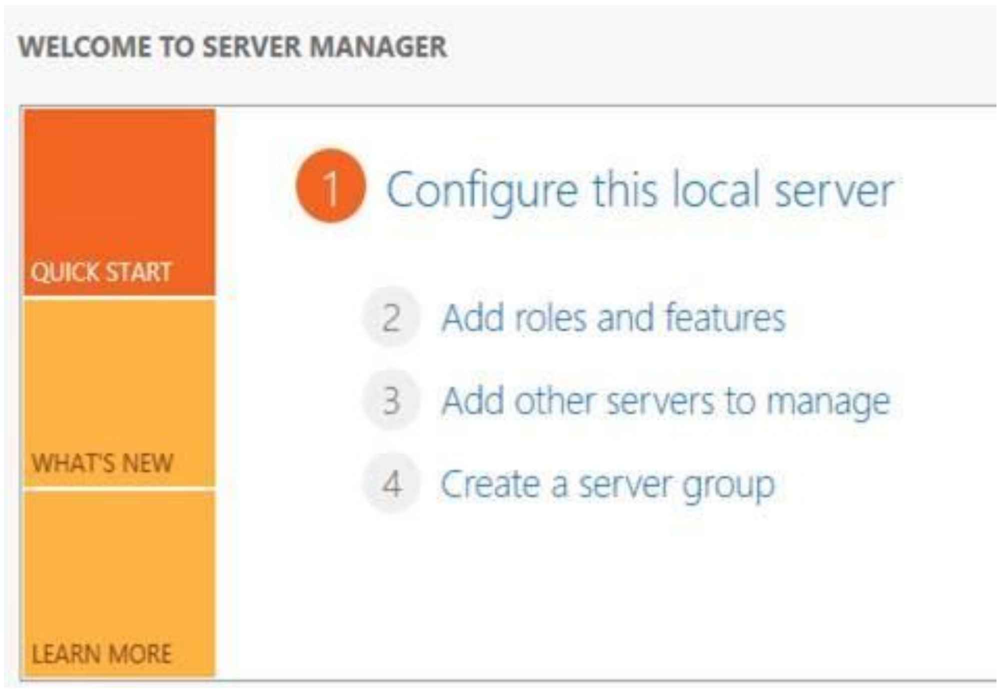
Deploying a Windows Azure VM

1. Log in to the Windows Azure management portal.
2. Create a Windows Azure VM with Windows Server 2012 or Windows Server 2008 image.
3. Fill in the appropriate details under the **Create a Virtual Machine** dialog tabs.
4. Once the VM is provisioned, RDP into the VM.

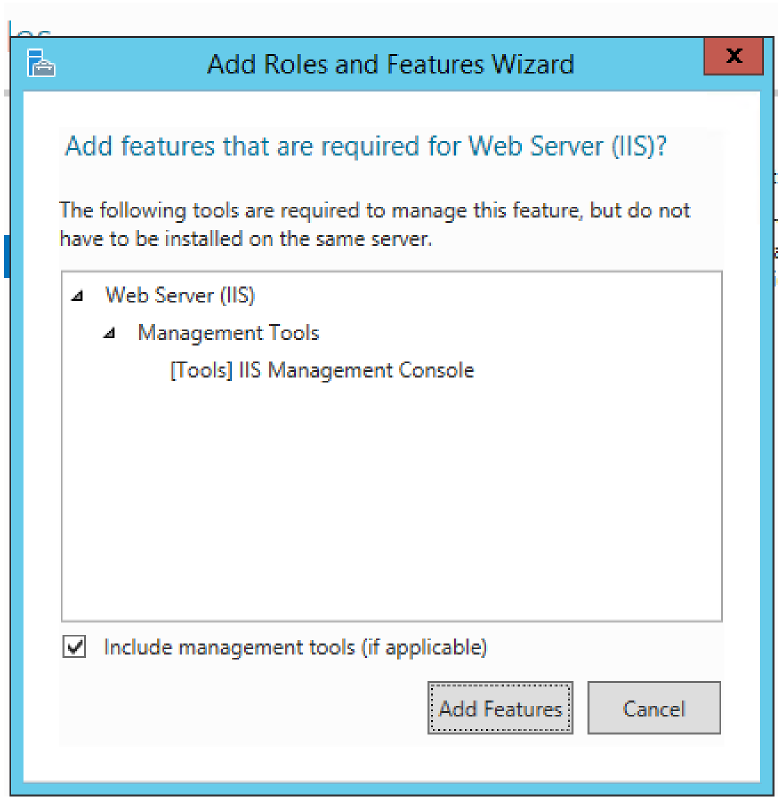
Note: If you are new to using Windows Azure then here are the detailed steps to provision a Windows Server VM and RDP to it.

Installing FTP service on a Windows Azure VM

1. Open **Server Manger** and click **Add roles and features.**


WELCOME TO SERVER MANAGER
QUICK START
WHAT'S NEW
LEARN MORE
1 Configure this local server
2 Add roles and features
3 Add other servers to manage
4 Create a server group

2. From **Installation Type** tab select **Role based or feature-based installation** and click **Next**.
3. From **Server Selection** tab select the server on which you want to enable FTP and click **Next**.
4. From **Server Roles** tab select **Web Server (IIS),** you will be presented with the **Add Roles and Features Wizard.**Click **Web Server (IIS)** and then click **Add Features**.


Add Roles and Features Wizard

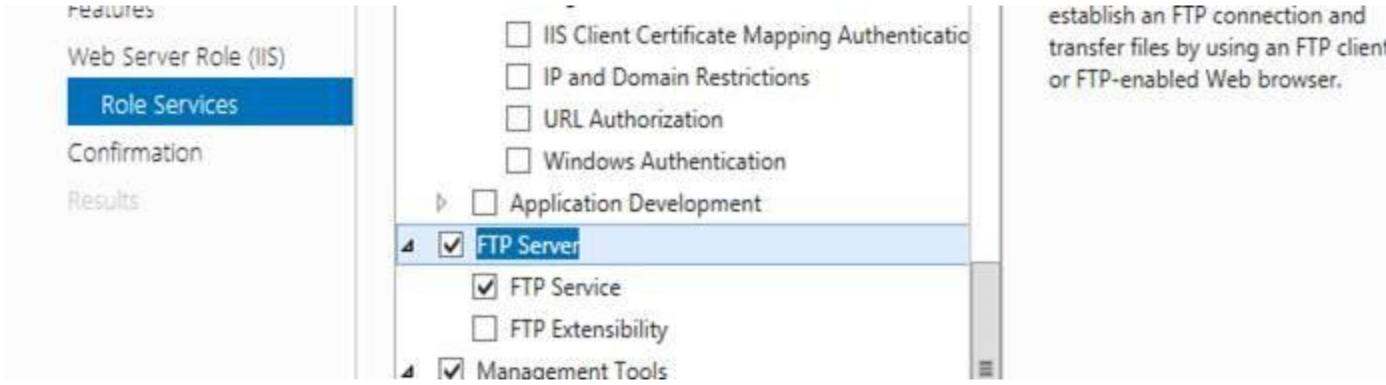
Add features that are required for Web Server (IIS)?

The following tools are required to manage this feature, but do not have to be installed on the same server.

- Web Server (IIS)
  - Management Tools
    - [Tools] IIS Management Console

☑ Include management tools (if applicable)

5. Click **Next** on the **Features** and **Web Server Role (IIS)** tabs.
6. From **Role Services** tab select **FTP Server** and **FTP Service** and click **Next.**



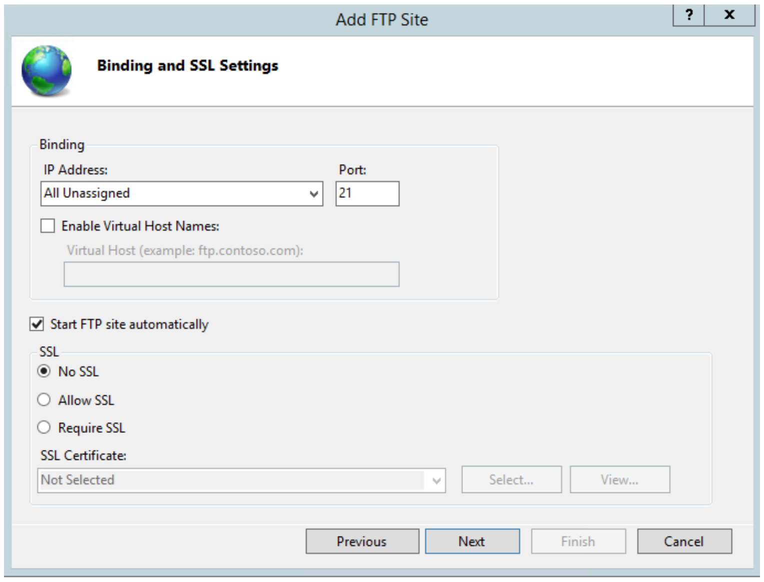
☐ IIS Client Certificate Mapping Authenticatio
☐ IP and Domain Restrictions
☐ URL Authorization
☐ Windows Authentication
☐ Application Development
☑ FTP Server
☑ FTP Service
☐ FTP Extensibility
☑ Management Tools

establish an FTP connection and transfer files by using an FTP client or FTP-enabled Web browser.

7. From **Confirmation** tab click **Install** and wait for the installation to complete.

Adding the FTP site to IIS Manager

1. From the **Control Panel> Administrative Tools** open the **IIS Manager.**
2. From **IIS Manager**, in the Connections pane, expand the **Sites** node in the tree, then right click the **Default Web Site**.
3. Now click **Add FTP Publishing**.
4. Fill the **Add FTP Site** dialog box as shown in the below figures and click **Finish**.



Binding and SSL Settings

Binding

IP Address: All Unassigned

Port: 21

☐ Enable Virtual Host Names:

Virtual Host (example: ftp.contoso.com):

☑ Start FTP site automatically

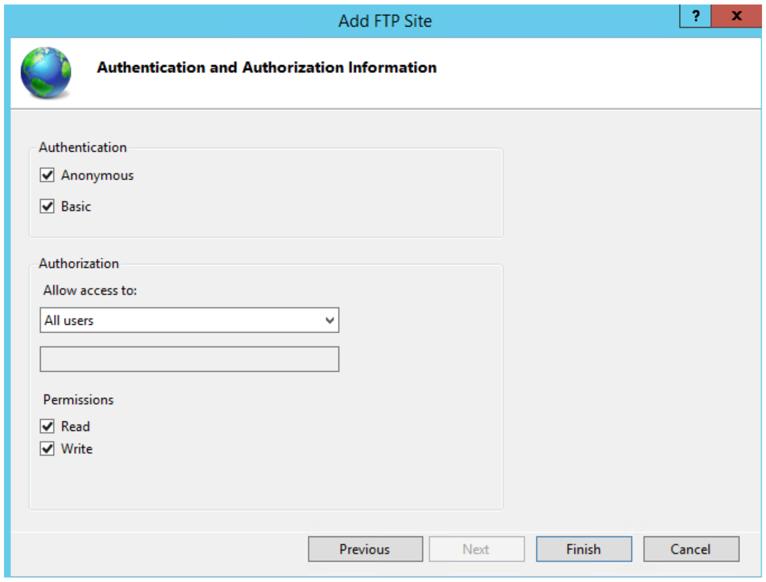
SSL

● No SSL

○ Allow SSL

○ Require SSL

SSL Certificate: Not Selected



Authentication and Authorization Information

Authentication

☑ Anonymous

☑ Basic

Authorization

Allow access to: All users

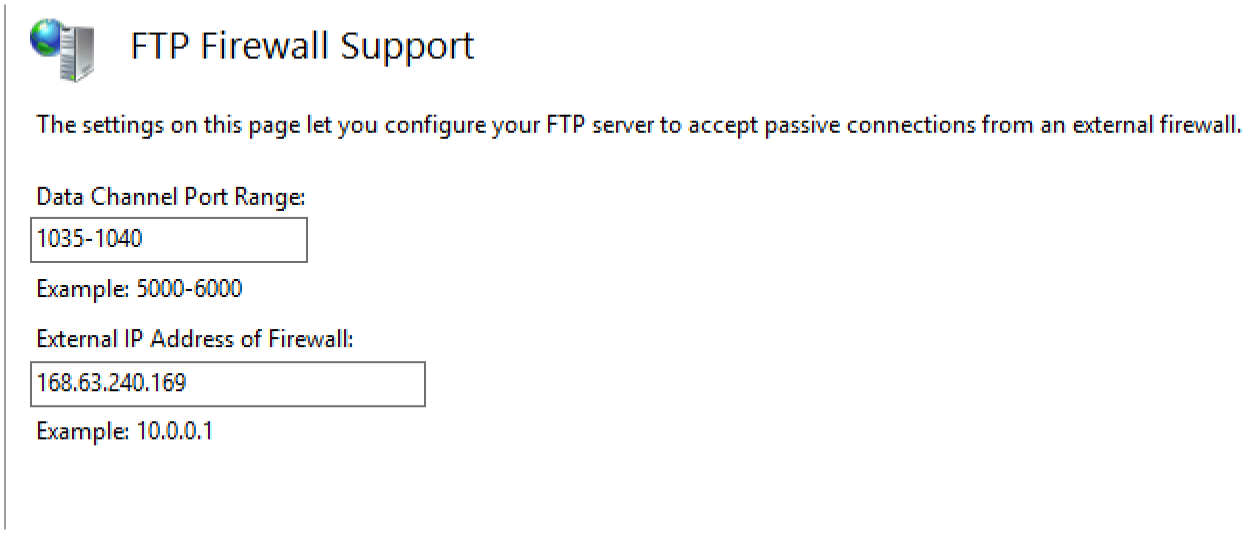
Permissions

☑ Read

☑ Write

Specify the data channel port for passive FTP connection on Windows Azure VM

1. From the **Control Panel** open the **IIS Manager.**
2. In IIS Manager, in the **Connections** pane, click local host.
3. In the Home pane, double-click the **FTP Firewall Support** feature.
4. In the **Data Channel Port Range** box specify a Port Range. In this case we have used 1035-1040. **The External IP Address of Firewall** is the VIP of your VM.



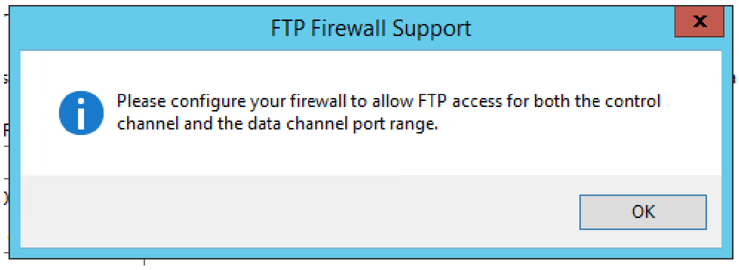
FTP Firewall Support

The settings on this page let you configure your FTP server to accept passive connections from an external firewall.

Data Channel Port Range:
1035-1040
Example: 5000-6000

External IP Address of Firewall:
168.63.240.169
Example: 10.0.0.1

5. Click **Apply**. You will be prompted to configure the firewall to allow FTP access.

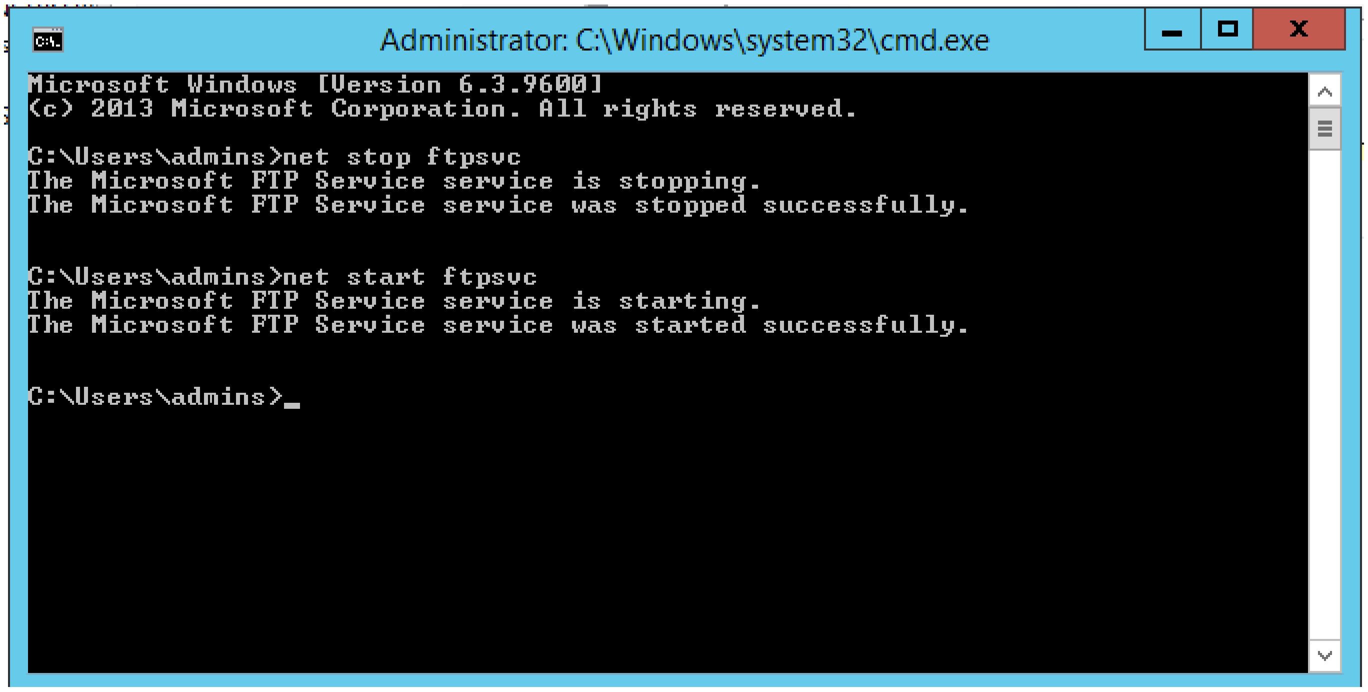


FTP Firewall Support

Please configure your firewall to allow FTP access for both the control channel and the data channel port range.

6. To make sure that FTP server has taken all the setting we added, let's stop and start the FTP service.

Note: iisreset does not restart the FTP service as it is outside the IIS.



```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\admins>net stop ftpsvc
The Microsoft FTP Service service is stopping.
The Microsoft FTP Service service was stopped successfully.


C:\Users\admins>net start ftpsvc
The Microsoft FTP Service service is starting.
The Microsoft FTP Service service was started successfully.


C:\Users\admins>_
```

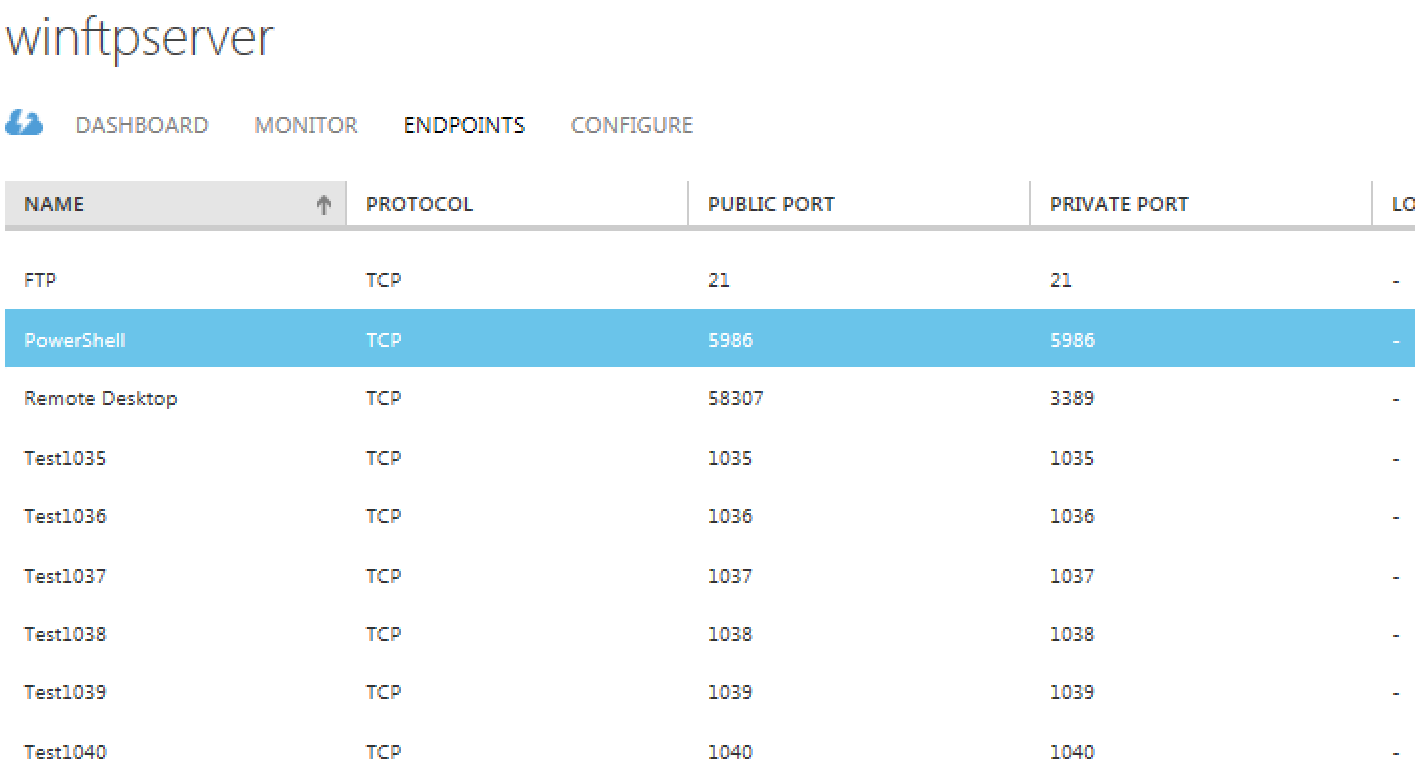
Adding ports specified in the previous step as endpoint to the VM

Ports 1035 to 1040 should also be added as endpoint to the Azure VM. You can add multiple ports as endpoint to the VM using Windows Azure PowerShell. Detailed procedure here.

You can also add endpoints using the management portal. Detailed procedure here.

To confirm that the said ports are added to the VM, please check the endpoint list on portal for the said VM.

Note: You also would need add port 21 to the endpoint list which is command port for FTP connection.


winftpserver

NAME	PROTOCOL	PUBLIC PORT	PRIVATE PORT	LO
FTP	TCP	21	21	-
PowerShell	TCP	5986	5986	-
Remote Desktop	TCP	58307	3389	-
Test1035	TCP	1035	1035	-
Test1036	TCP	1036	1036	-
Test1037	TCP	1037	1037	-
Test1038	TCP	1038	1038	-
Test1039	TCP	1039	1039	-
Test1040	TCP	1040	1040	-

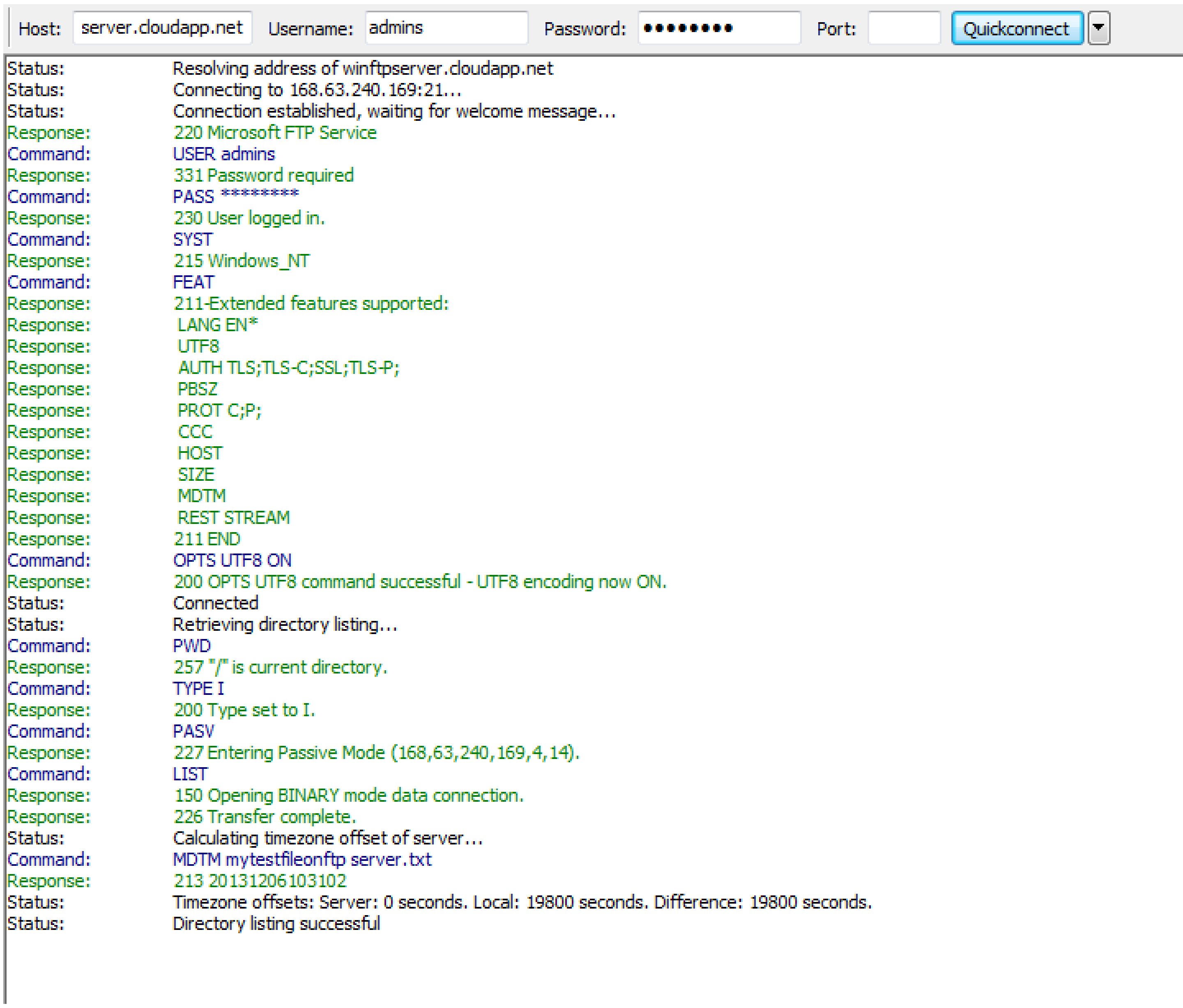
Adding Firewall rules to allow traffic on the added endpoint

For ports added as endpoints in the above procedure, no configuration is done automatically to the firewall in the guest operating system. When you create an endpoint, you'll need to configure the appropriate ports in the firewall to allow the traffic you intend to route through the endpoint.

In this case I have disabled the Windows firewall for simplicity. You can refer here to modify the firewall rules to allow traffic on the ports added as end points.

We are now done setting up the passive FTP Server on a Windows Azure VM.

Verifying that FTP server is using the port previously specified under data channel port



Host: server.cloudapp.net Username: admins Password: •••••••• Port: Quickconnect

```
Status:        Resolving address of winftpserver.cloudapp.net
Status:        Connecting to 168.63.240.169:21...
Status:        Connection established, waiting for welcome message...
Response:      220 Microsoft FTP Service
Command:       USER admins
Response:      331 Password required
Command:       PASS *********
Response:      230 User logged in.
Command:       SYST
Response:      215 Windows_NT
Command:       FEAT
Response:      211-Extended features supported:
Response:       LANG EN*
Response:       UTF8
Response:       AUTH TLS;TLS-C;SSL;TLS-P;
Response:       PBSZ
Response:       PROT C;P;
Response:       CCC
Response:       HOST
Response:       SIZE
Response:       MDTM
Response:       REST STREAM
Response:      211 END
Command:       OPTS UTF8 ON
Response:      200 OPTS UTF8 command successful - UTF8 encoding now ON.
Status:        Connected
Status:        Retrieving directory listing...
Command:       PWD
Response:      257 "/" is current directory.
Command:       TYPE I
Response:      200 Type set to I.
Command:       PASV
Response:      227 Entering Passive Mode (168,63,240,169,4,14).
Command:       LIST
Response:      150 Opening BINARY mode data connection.
Response:      226 Transfer complete.
Status:        Calculating timezone offset of server...
Command:       MDTM mytestfileonftp server.txt
Response:      213 20131206103102
Status:        Timezone offsets: Server: 0 seconds. Local: 19800 seconds. Difference: 19800 seconds.
Status:        Directory listing successful
```

1. Client connects on the command port, which is usually TCP port 21.

2. When the connection on command port is successful, the server sends a port to the client to connect to.

   If you are using FileZilla, you will see something like this*:*

   Command: PASV
   Response: 227 Entering Passive Mode (168,63,240,169,4,14).

   Where 168,63,240,169 is the IP address of the VM and 4,14 is the port on which the data traffic is routed (256*4+14=1038).

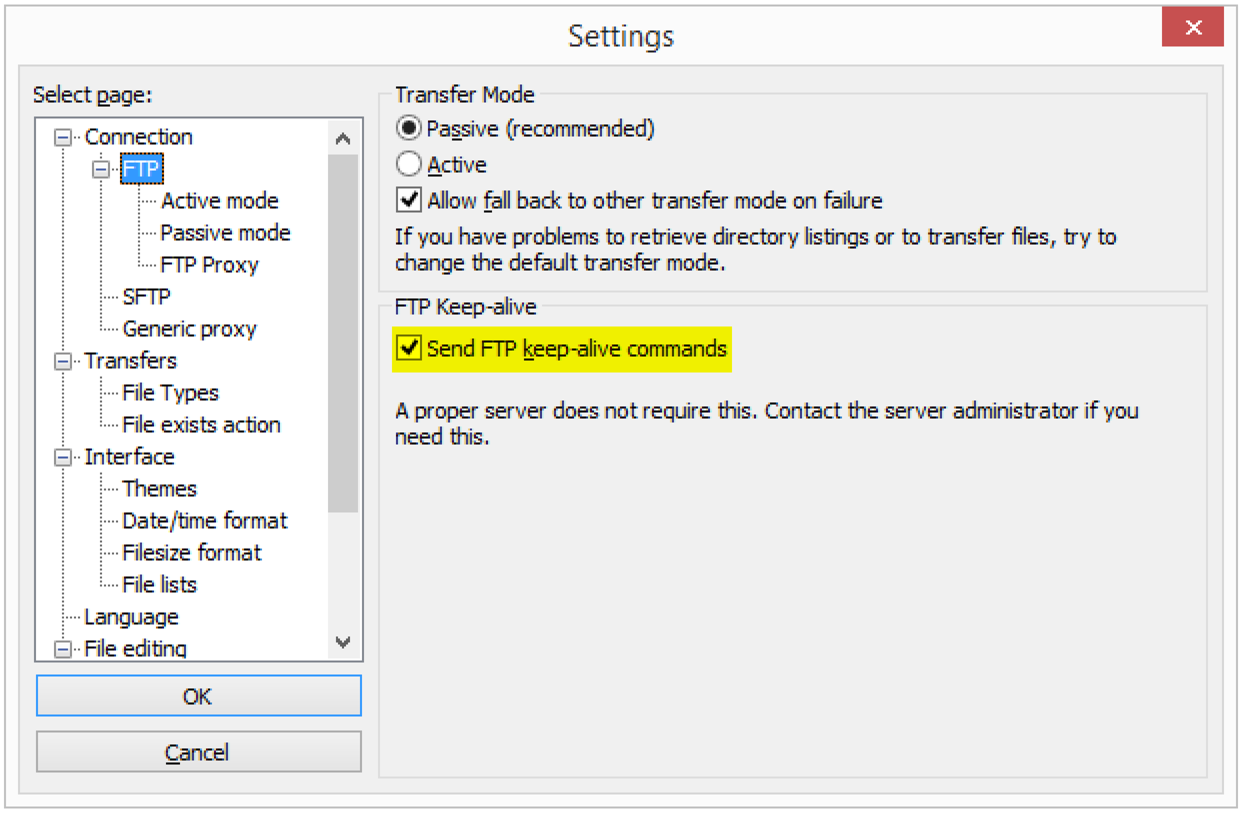
   This is how you determine that a port added in previous steps are actually being used to establish a passive FTP connection.

Points to consider from Azure SLB perspective

When FTP is transferring large files, the elapsed time for transfer may exceed 4 minutes, especially if the VM size is A0. Any time the file transfer exceeds 4 minutes, the Azure SLB will time out the idle TCP/21 connection, which causes issues with cleanly finishing up the FTP transfer once all the data has been transferred.

Basically, FTP uses TCP/21 to set everything up and begin the transfer of data. The transfer of data happens on another port. The TCP/21 connection goes idle for the duration of the transfer on the other port. When the transfer is complete, FTP tries to send data on the TCP/21 connection to finish up the transfer, but the SLB sends a TCP reset instead.

The way around this is to make the client to keep the TCP/21 connection from going idle. If using a 3rd party FTP client, there may be configuration knobs the user can turn in order to cause the FTP client to send a keepAlive. As an example of how you can set this in FTP client software, in FileZilla, you go to **Edit**, **Settings**, **Connection**, **FTP**, and check **Send FTP keep-alive commands**.



Transfer Mode

● Passive (recommended)

○ Active

☑ Allow fall back to other transfer mode on failure

If you have problems to retrieve directory listings or to transfer files, try to change the default transfer mode.

FTP Keep-alive

☑ Send FTP keep-alive commands

A proper server does not require this. Contact the server administrator if you need this.

If the FTP client is being written in .NET, customers will need to account for this client-side keepAlive in their code. Here is a sample (the keepAlive is highlighted):

```
        private void button1_Click(object sender, EventArgs e)
        {
            System.Net.ServicePointManager.SetTcpKeepAlive(true, 30000,
20000);
            try
            {
                System.Net.WebClient oFTPDownloadWebClient
= new System.Net.WebClient();

                oFTPDownloadWebClient.Proxy = null;
                oFTPDownloadWebClient.Credentials
= new System.Net.NetworkCredential("username","password");

        oFTPDownloadWebClient.DownloadFile(newSystem.Uri("ftp://contosoFTP.cl
oudapp.net/LargeFile.Dat"), "C:\\Temp\\Big.dat");
                 return;
             }
            catch (System.Exception ex)
            {
                MessageBox.Show(ex.Message);
             }
        }
```

Load balanced endpoints are not supported in Azure Passive FTP server

You will be unable to load balance your endpoints when using a Passive FTP server in Azure since there is no session stickiness between the server's Command Port (21) and the random data ports that get selected when there is data being transferred btw the client and the ftp server.

The client will contact the FTP server (Azure VM) through Port 21 (Server's Command port), and establish an FTP session. Then as soon as the client tries to download or upload data, the FTP server (Azure) will send a hashed port number to the client (from the list of data ports you select when setting up your FTP service).

It will look like:

ReplyCode: 227, Entering Passive Mode <h1,h2,h3,h4,p1,p2>

*Entering Passive Mode (<ftp_svr_ip>, **a**, **b**)*

The port can be calculated as:

Random Ephemeral Port = (**a***256) + **b**

The server will select a random ephemeral port for every data session that is opened (within the port range specified when setting up the FTP service).

Let's say you have Azure FTP Servers **FTP1** and **FTP2** behind the load balancer.

You place FTP1 and FTP2 in the same cloud service;

You load balance Port 21 and all the data ports across these two VM's;

The client tries to establish an FTP session (through Port 21), and the LB first selects **FTP1.**

The client then chooses to download a file from **FTP1**.

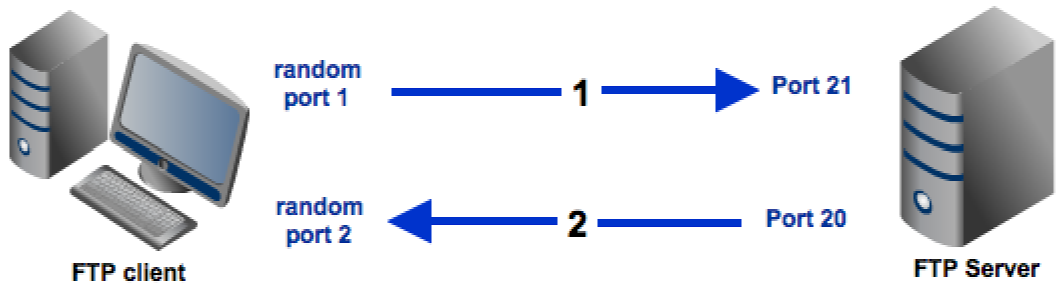
FTP1 will respond back to the client and provide a random data port that the client will need to use to access its data channel;

The client then has to set up a **new** TCP session with that target data port on FTP1, and this request goes through the Load Balancer;

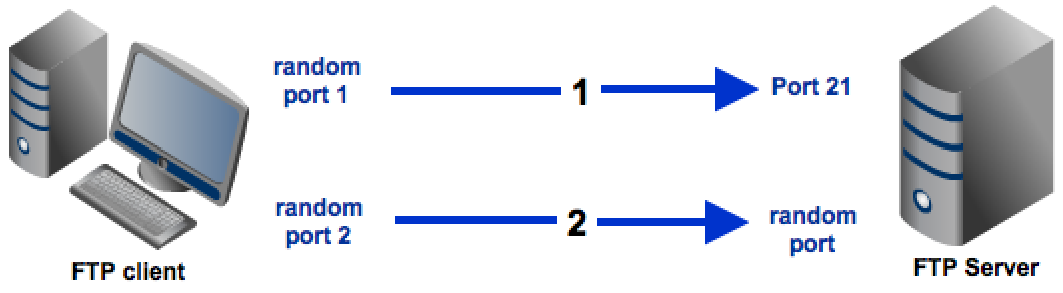
The LB, which is not configured with sticky sessions, and is just a Layer 3 round robin LB, will not keep track of the session previously established with **FTP1** and may pick **FTP2** instead.

While this can work with Active FTP (since no endpoint is required for outbound traffic, and since active FTP just uses Src Port 20 to initiate the data channel with the client).

ACTIVE FTP



PASSIVE FTP



Running concurrent traces on the client and the 2 Azure FTP servers will show this.