V.12

Cerberus, LLC

Cerberus FTP Server

User Manual

Detailed steps and help on configuring Cerberus FTP Server.

Contents

1.0 Introduction

1.1 Description

Cerberus FTP Server provides a secure and reliable file transfer solution for the demanding IT professional or the casual file sharer. Supporting SFTP, FTP/S, and HTTP/S, Cerberus can authenticate against Active Directory and LDAP, run as a Windows service, has native x64 support, and includes a robust set of integrity and security features, and offers an easy-to-use manager for controlling user access to files and file operations.

1.2 Guide

For additional help and troubleshooting information, take a look at the Cerberus FTP FAQ.

You can also access the most recent help documentation online.

2.0 Minimum System Requirements

This section describes the minimum hardware and software requirements to install and run Cerberus FTP Server.

2.1 Hardware Requirements

- 2GHz x86 or x64 processor
- 2 GB RAM (4 GB or higher recommended
- WXGA (1280 x 768) or higher-resolution monitor

2.2 Operating Systems

Note: The latest Service Packs for your operating system are required in all cases.

2.2.1 Cerberus FTP Server 12 and 11

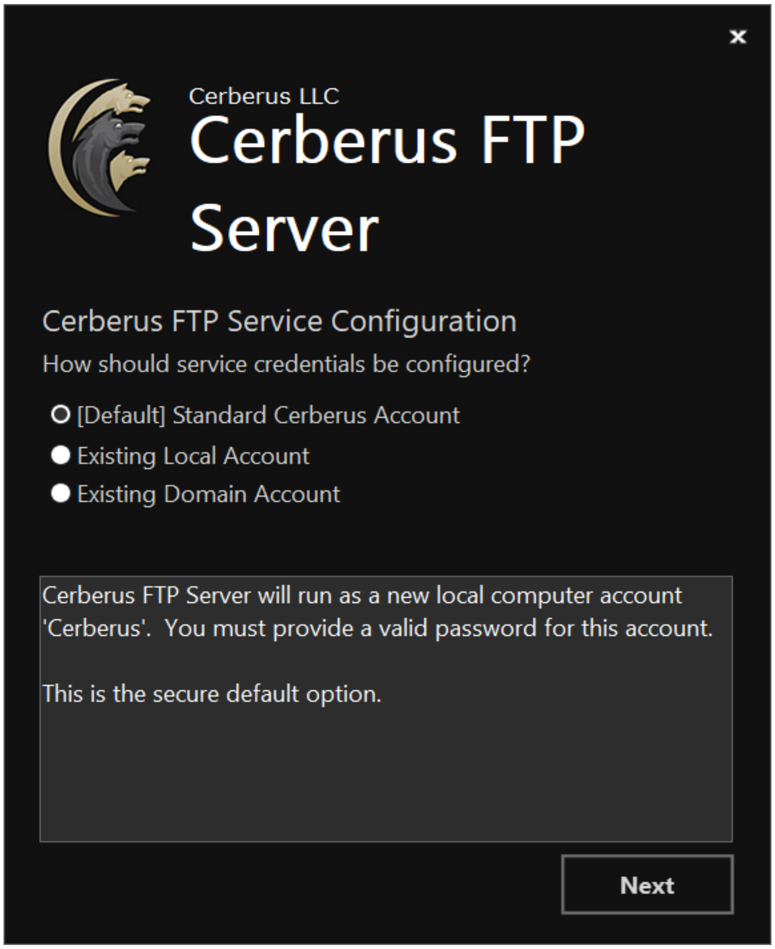
- Windows 7
- Windows 8
- Windows 10
- Windows 11
- Windows Server 2012 and R2
- Windows Server 2016
- Windows Server 2019
- Windows Server 2022

The latest Service Packs for your operating system are **highly** recommended.

3.0 Installation

Close all other programs (recommended) before installing Cerberus FTP Server and make sure that you install it logged in as Administrator or a member of the Administrators group if you are installing it on a Windows Server system.

1. Download the Latest Cerberus FTP Server installer
2. Double click or run the **CerberusInstall64.exe** self-extracting installer. You may be prompted “Do you want to allow the following program to make changes to this computer” click **Yes** (or **Allow**). Clicking **Yes** will give the Cerberus FTP Server Installer Administrator privileges to install (required on most operating systems).
3. After the Welcome screen, you will see the screen where you will select your preferred service account option. The default option will create a limited privilege dedicated Windows account to run the service. See below for detailed explanations of the options.



Cerberus FTP Service Configuration

How should service credentials be configured?

○ [Default] Standard Cerberus Account
● Existing Local Account
● Existing Domain Account

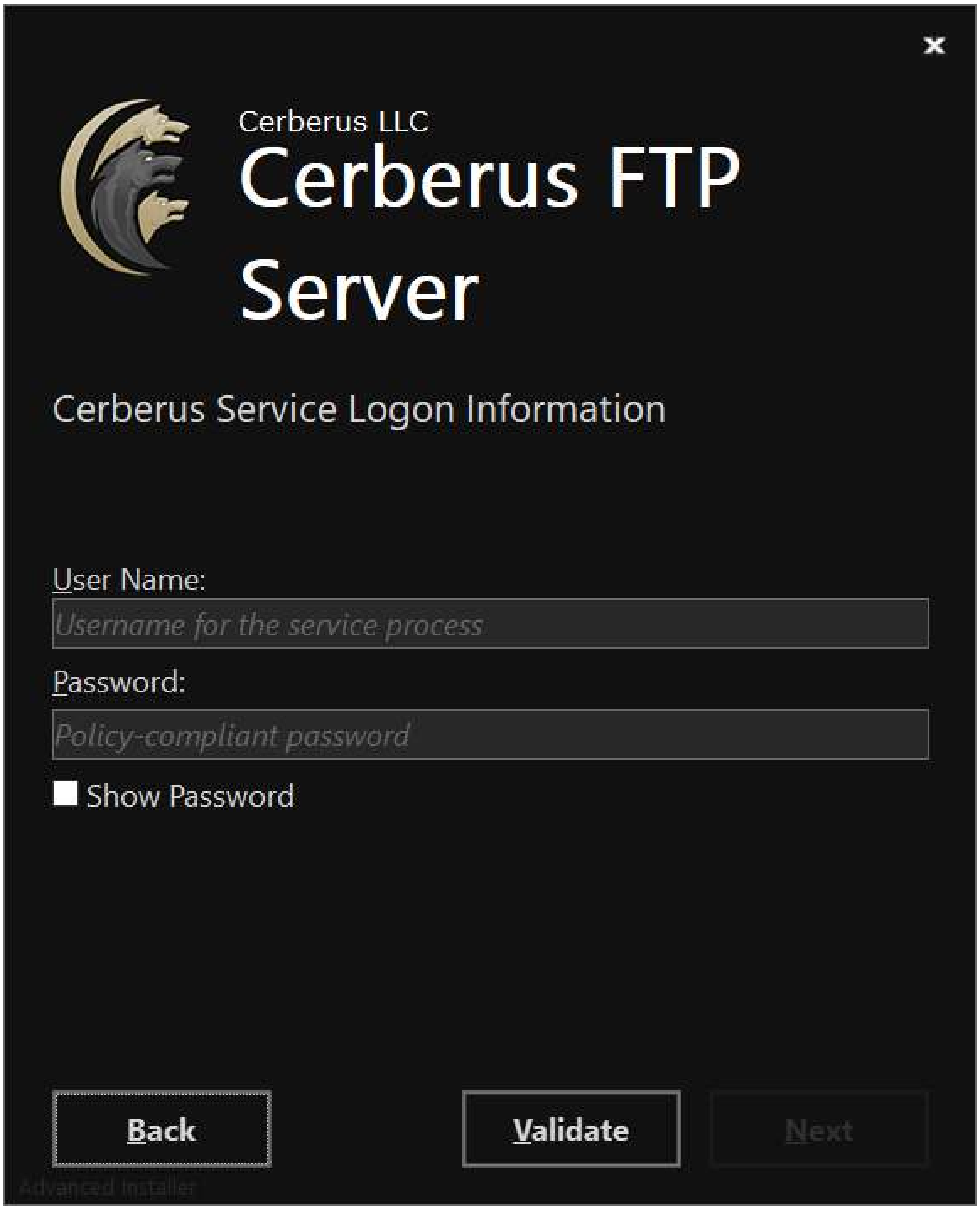
Cerberus FTP Server will run as a new local computer account 'Cerberus'. You must provide a valid password for this account.

This is the secure default option.

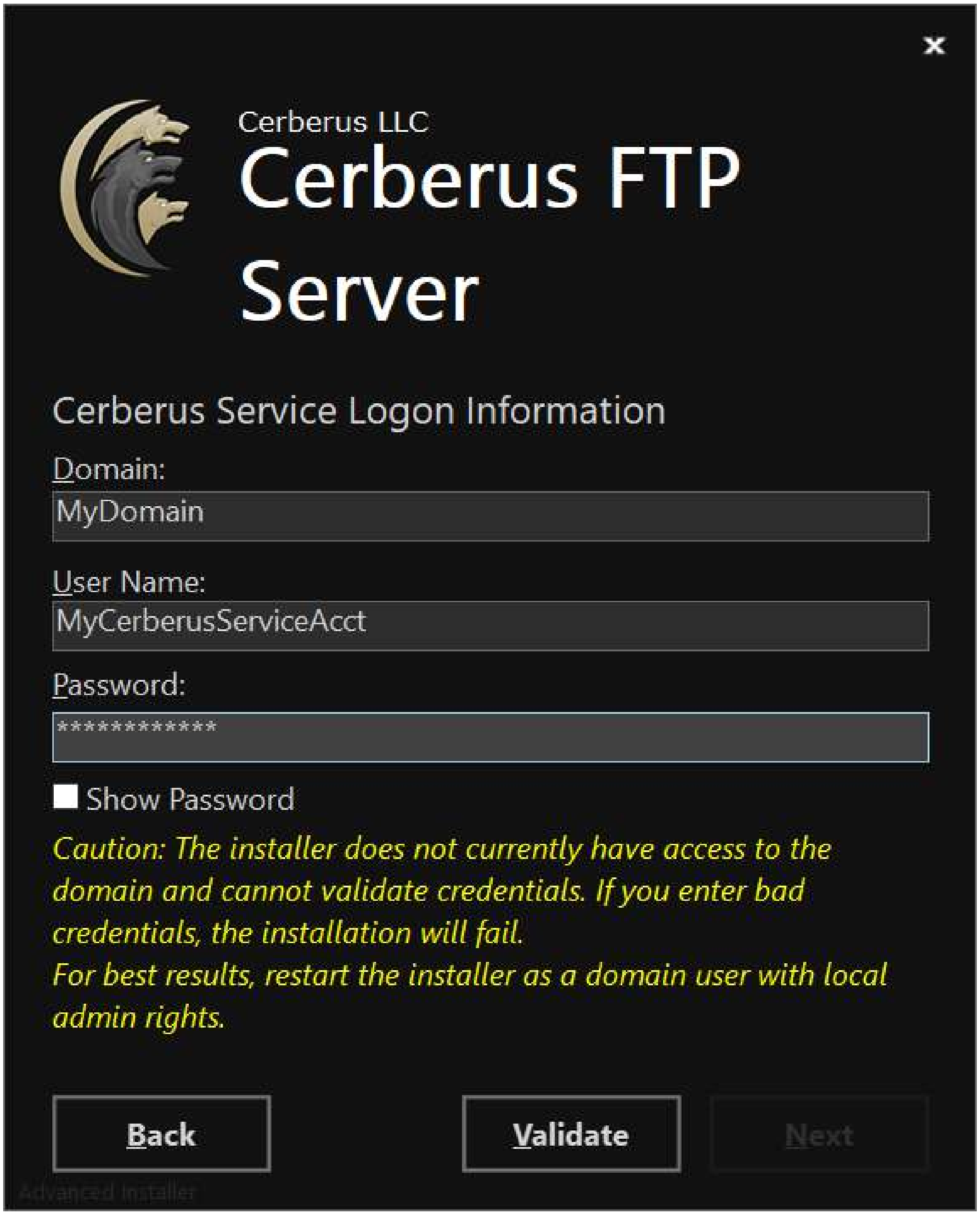
The installer offers three options during first-time installation:

1. **Standard Cerberus Account**
   This option creates a new, unprivileged local computer account named ‘Cerberus’ and configures the Cerberus FTP Server service to run as this user. You’ll be prompted to create a password for this account.
2. **Existing Local Account**
   For security purposes, some administrators choose to run services as specific local accounts. This option allows you to configure the service to run with these local computer accounts.
3. **Existing Domain Account**
   Likewise, Windows Domain environments may require that specific accounts be used. This account may, for instance, be granted access to the domain directory.

4. When username, password, or domain are required, the installer requests this information:



5. The ‘Validate’ button checks the username and password and alerts you if the user can’t be found, or if the password is incorrect. Both Local Computer and Domain credentials can be validated, however, validating domain credentials requires that the installer be run by a domain user:



Cerberus LLC

Cerberus FTP Server

Cerberus Service Logon Information

Domain: MyDomain

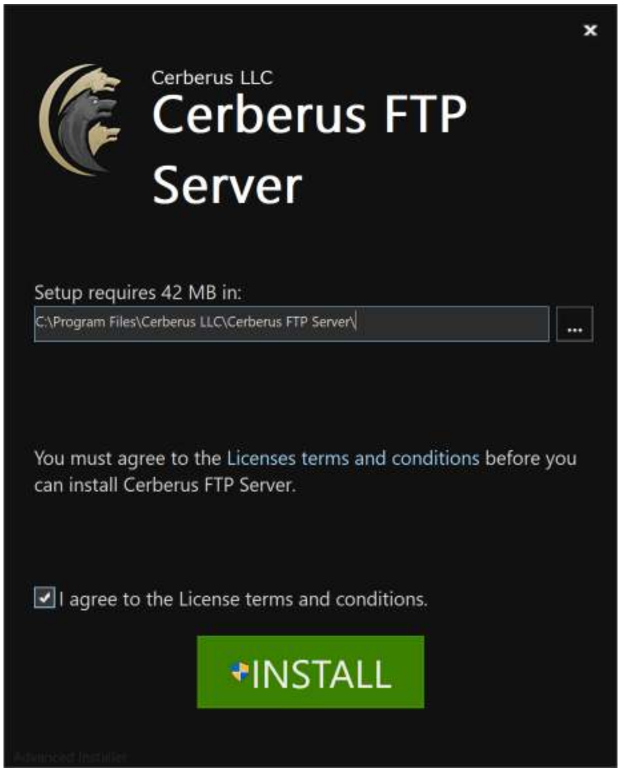
User Name: MyCerberusServiceAcct

Password: ************

□ Show Password

Caution: The installer does not currently have access to the domain and cannot validate credentials. If you enter bad credentials, the installation will fail.
For best results, restart the installer as a domain user with local admin rights.

6. After choosing your option and validating username and password (if that's what you chose). Click Next to continue.
7. On the next screen you will need to check the box to confirm your acceptance of the licensing agreement. Select the "**I agree to the License terms and conditions**" checkbox and click **Install.**



Cerberus LLC

Cerberus FTP Server

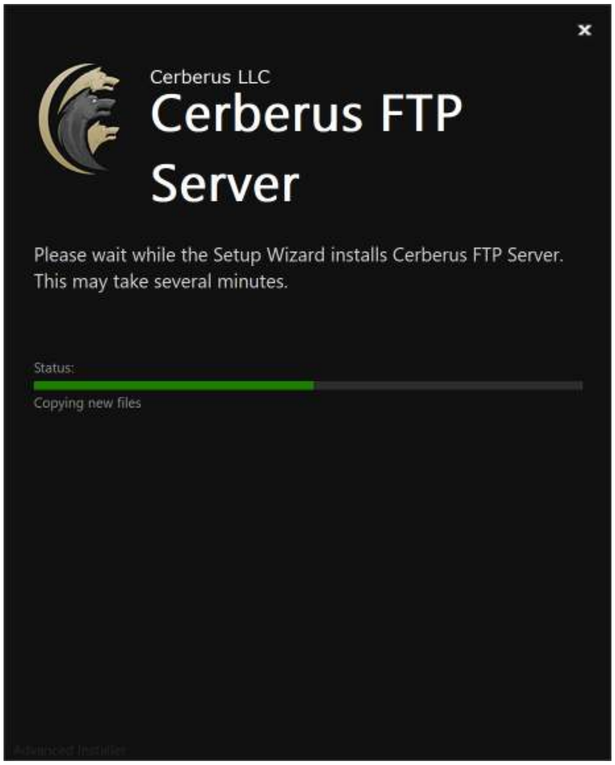
Setup requires 42 MB in:

C:\Program Files\Cerberus LLC\Cerberus FTP Server\

You must agree to the Licenses terms and conditions before you can install Cerberus FTP Server.

☑ I agree to the License terms and conditions.

8. Wait for the installer to finish.



Cerberus LLC

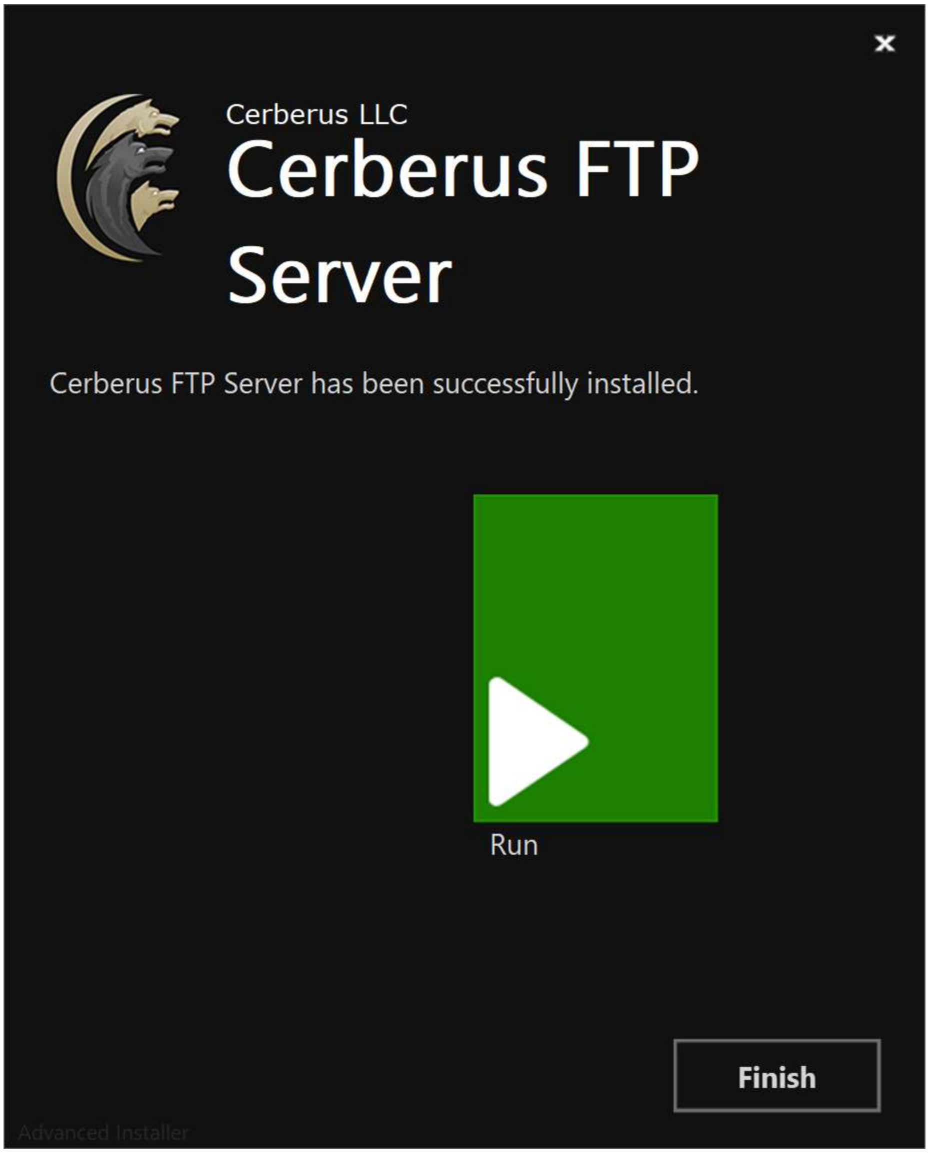
Cerberus FTP Server

Please wait while the Setup Wizard installs Cerberus FTP Server. This may take several minutes.

Status:

Copying new files

9. Click **Finish** or press the **Run** button to launch the Cerberus FTP Server Administration Tool.



Cerberus LLC

Cerberus FTP Server

Cerberus FTP Server has been successfully installed.

Cerberus FTP Server Installation Complete Page

4.0 Updating an Existing Installation

There are two methods for updating an existing installation of Cerberus FTP Server. You can use the built-in auto-updater, or you can download the latest installer and run it to manually update your installation. Both methods are discussed below.

When updating using either method, the installer first stops the Cerberus FTP Server service, uninstalls the existing Cerberus FTP Server installation, then installs the latest release. The uninstallation only affects the actual program files. The server configuration and user settings are never removed.

An update usually takes about 5 minutes, and seldom requires a reboot. The Cerberus FTP Server service will be unavailable during the update.

NOTE: We always recommend making a backup of your users and settings from the Cerberus **Tools** menu before updating. Select the **Backup Users and Settings** option to create a zip file of all of your Cerberus settings and users.

4.1 Method 1: Using the Auto-updater

The Cerberus FTP Server automatic updater will check for a new release of Cerberus FTP Server, and allow you to download and run the updater to update your installation. To check for an update and, optionally, install it:

1. Select the **Help** menu option from the main menu.
2. Select the **Check for Update** menu option.
3. You will see the **Update Check** dialog. It will list the current version installed and any available updates.
4. If no updates are available, the Current Version and Latest Version will be the same, and the **Update** button will be disabled. If this is the case, you have the latest release and can press the **Close** button to end the update process.
5. If a new release is available, the **Update** button will become enabled and a list of changes since your current version was released will be listed in the release notes list box.
6. Press the **Update** button to automatically download the latest release and begin the update process.
7. Once the download has completed, the Cerberus UI will close and shutdown and the installer will automatically launch. You should select the default options for any questions in the installer. The installer will automatically remove your existing installation (users and settings are never removed during an uninstallation) and then install the latest release. All of your existing users and settings will be preserved.
8. Finish the installation and you are done. The latest version of Cerberus FTP Server should now be installed and running.

4.2 Method 2: Manually downloading and running the latest Installer

You can manually download the latest installer and use it to update your installation if you cannot use the auto-updater. The installer will update an existing installation to the latest release. To download and run the latest installer:

1. Go to the Cerberus FTP Server Support Downloads page
2. Download the latest installer.
3. Close the Cerberus FTP Server UI. Go to the File menu and select the Exit menu option. You should also shut down the Cerberus FTP Server Window Service. The installer will normally be able to shut down the service, but on rare occasions, the automatic shutdown will not work. Shutting down the Cerberus Windows Service before installation ensures a restart will not be necessary after the installation completes.
4. Launch the installer. You should select the default options for any questions in the installer. The installer will automatically remove your existing installation and install the latest release. All of your existing users and settings will be preserved.
5. Finish the installation and you are done. The latest version of Cerberus FTP Server should now be installed and running.

5.0 Getting Started - Initial Setup Wizard

5.1 The Wizard

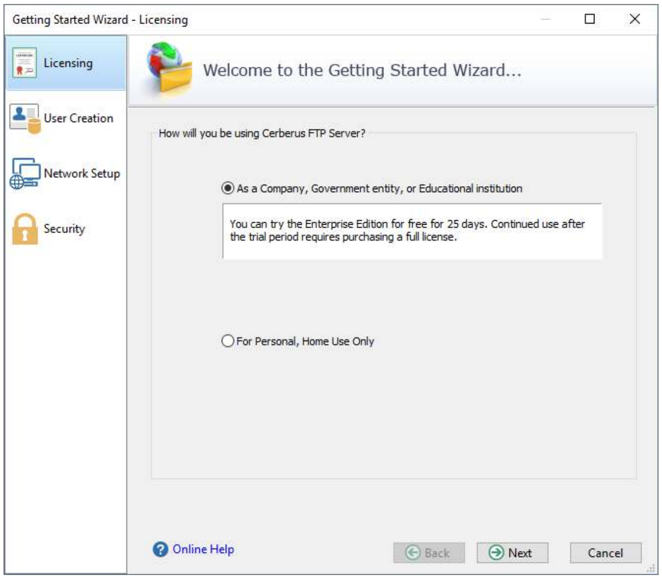
The Getting Started Wizard will appear when you start Cerberus FTP Server for the first time. The wizard is designed to walk you through the basic steps of configuring the server to allow clients to connect. At the end of the Getting Started Wizard, your server should be ready to accept connections from FTP, FTPS, SSH SFTP, and HTTP clients.

5.1.1 Step 1 - Licensing

The Licensing page allows the administrator to select the licensing option most appropriate for their intended use of Cerberus FTP Server. There are two options:

- Selecting “**As a Company, Government entity, or Educational institution”** enables a 25-day trial period of the Enterprise edition of Cerberus FTP Server. During the trial period, the server will perform and function as the Enterprise edition. **Cerberus FTP Server reverts to the Home edition after the evaluation period expires** and a message indicating that the server is unregistered will be added to the server welcome message for each connection. At any time, including after the trial period has expired or even if "For Personal Use" was selected at startup, Cerberus may be licensed as the full commercial Personal, Standard, Professional, or Enterprise edition by entering a valid registration code into the license dialog.

- Selecting the **“For Personal, Home Use Only”** option immediately causes Cerberus to function as the Home edition. This license is only permitted for at home, personal use of the FTP server. The Home edition is limited to at most 5 simultaneous FTP or FTPS connections. A message indicating that the server is Cerberus FTP Server Home edition will also appear in the FTP welcome message whenever a client connects to the server. In all other respects, Cerberus FTP Server Home edition is functionally equivalent to the licensed Personal edition.



Welcome to the Getting Started Wizard...

How will you be using Cerberus FTP Server?

● As a Company, Government entity, or Educational institution

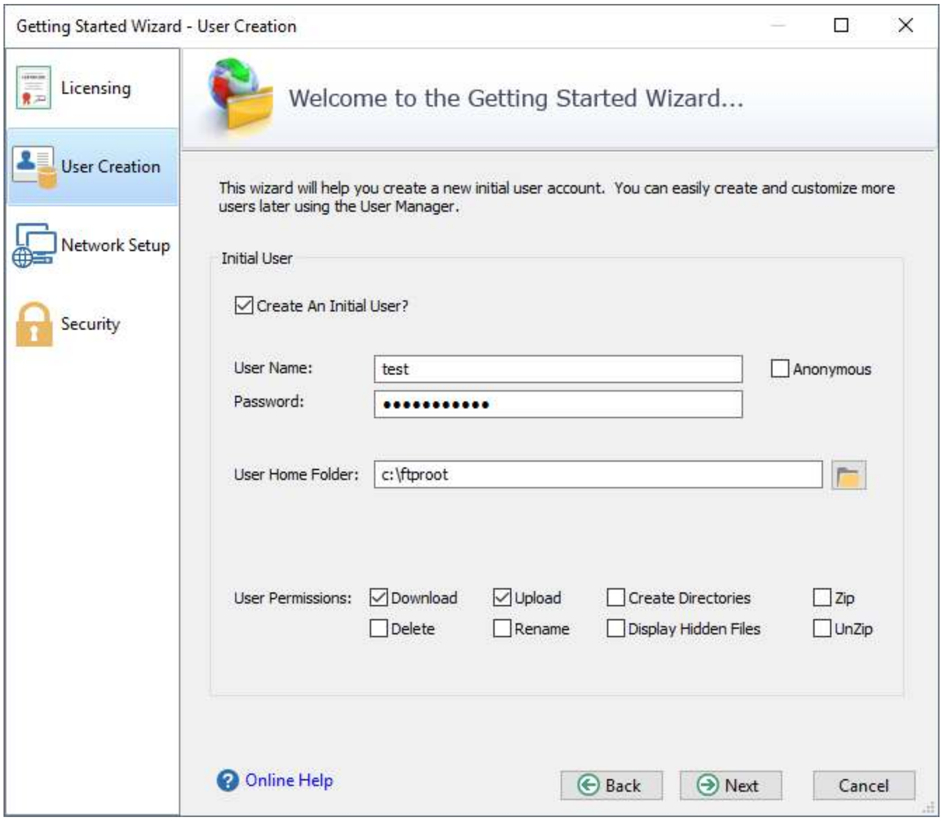
You can try the Enterprise Edition for free for 25 days. Continued use after the trial period requires purchasing a full license.

○ For Personal, Home Use Only

Licensing Page of the Getting Started Wizard

5.1.2 Step 2 - Initial User Creation

The User Creation page will allow you to automatically create a simple user account with access to a directory on the local machine. You can use this account to test out your initial connection to the server. You can turn off the creation of the user account by un-checking the "Create an Initial User?" checkbox.


Getting Started Wizard - User Creation
Licensing
User Creation
Network Setup
Security
Welcome to the Getting Started Wizard...
Online Help
Back
Next
Cancel

This wizard will help you create a new initial user account. You can easily create and customize more users later using the User Manager.

Initial User

☑ Create An Initial User?

User Name: test ☐ Anonymous

Password: ●●●●●●●●●●●●

User Home Folder: c:\ftproot

User Permissions: ☑ Download ☑ Upload ☐ Create Directories ☐ Zip
☐ Delete ☐ Rename ☐ Display Hidden Files ☐ UnZip

Enter the User Name and Password for the test account.

If desired, check the ‘Anonymous’ box to create an *anonymous* user account. ***Please note, creating an anonymous user allows anyone to connect to your FTP server without specifying a password. Anyone who has just the user name can access the directory specified and, if granted, can upload and download files to that directory and any subdirectories of that directory.***

The account created will have access to the directory in the ‘User Home Folder’ box. Cerberus sets this to ‘c:\ftproot’ by default and this directory will be created if it does not already exist. You may also press the folder icon to specify any fold you wish

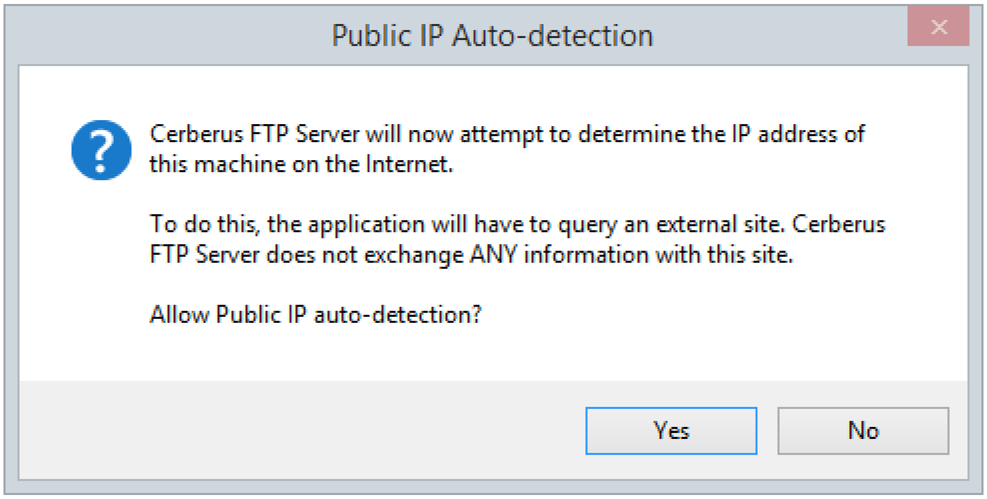
You can further customize the newly added user, or create and manage additional users, through the User Manager after the "Getting Started" wizard has finished.

5.1.3 Step 3 - Network Setup

The Network Setup page detects basic network settings and tries to provide advice on any changes that may need to be made because of the computer's network configuration.

5.1.3.1 Public IP Auto-detection for Passive Mode FTP

The most complex task in configuring basic FTP access to your server is preparing the machine to accept FTP data connections. Unlike SSH SFTP or HTTP/S protocols, FTP is complicated by the need for two connections for each client session. The first connection is established when the client initially connects and is used to exchange commands and status between the FTP server and the client. A second connection is created every time a directory listing or file transfer takes place. Whenever a directory listing or file transfer is requested, the FTP server has to respond with an IP address and port that the client can connect over to establish the secondary data connection. To aid the server in determining what IP address to give to the client, the server can be configured to automatically detect the IP address of the server on the internet and use this IP address when sending the client connection instructions.



Public IP Auto-detection

Cerberus FTP Server will now attempt to determine the IP address of this machine on the Internet.

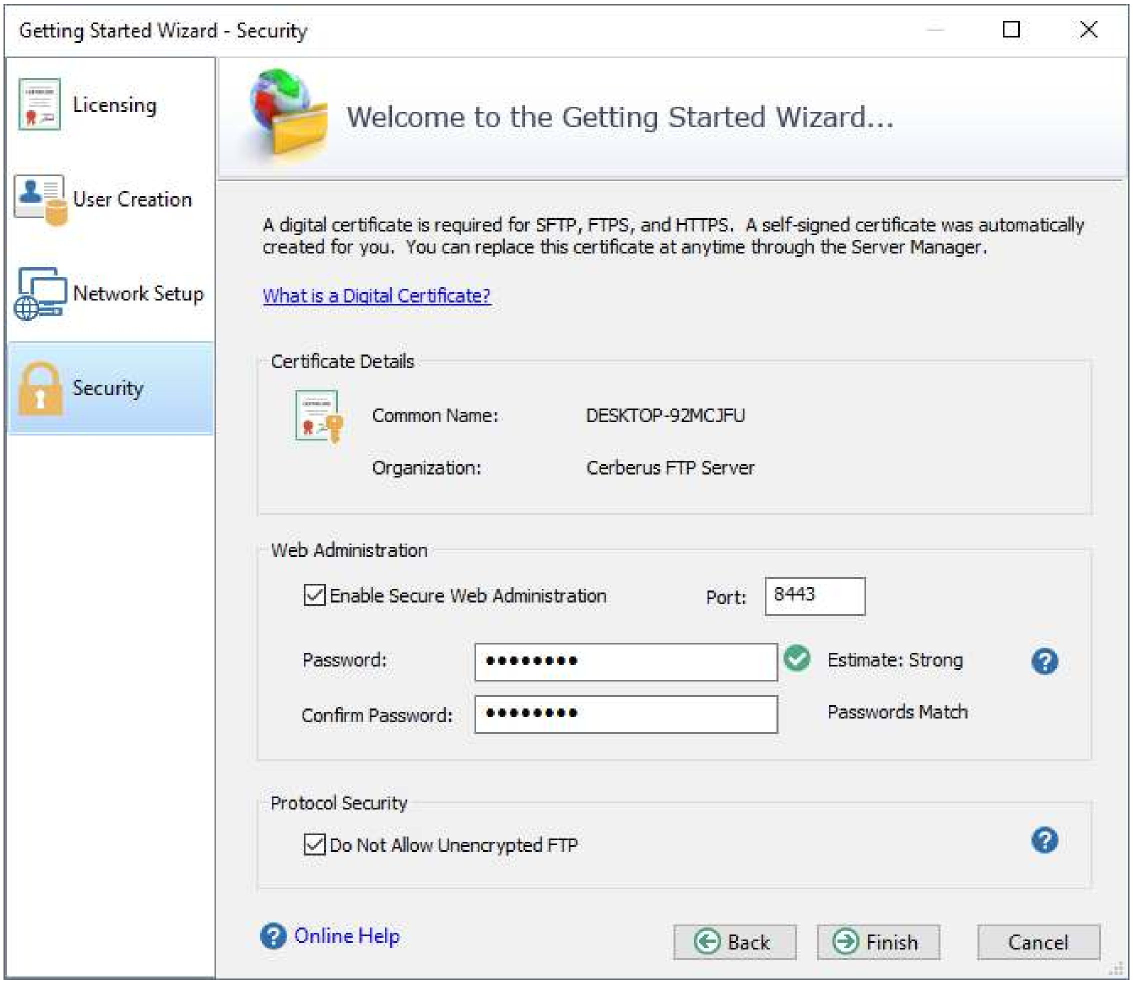
To do this, the application will have to query an external site. Cerberus FTP Server does not exchange ANY information with this site.

Allow Public IP auto-detection?

After clicking the Next button on the Network Setup page a dialog prompt will ask whether you want to allow Cerberus to automatically attempt to detect your public IP address. We normally recommend you answer **Yes** here. Answering yes will instruct Cerberus to automatically attempt to detect and use the correct external IP address when clients request passive FTP data connections.

5.1.4 Step 4 – Security

The last page of the Getting Started Wizard will allow the administrator to configure a few basic server security settings.



Welcome to the Getting Started Wizard...

A digital certificate is required for SFTP, FTPS, and HTTPS. A self-signed certificate was automatically created for you. You can replace this certificate at anytime through the Server Manager.

What is a Digital Certificate?

Certificate Details

Common Name: DESKTOP-92MCJFU

Organization: Cerberus FTP Server

Web Administration

☑ Enable Secure Web Administration Port: 8443

Password: •••••••• Estimate: Strong

Confirm Password: •••••••• Passwords Match

Protocol Security

☑ Do Not Allow Unencrypted FTP

Cerberus FTP Server fully supports TLSv2 encryption over FTP (FTPS), HTTPS, and SSH SFTP.

To enable FTPS, HTTPS, and SSH SFTP support, a digital certificate must be generated for the server. This digital certificate contains the necessary security data to allow the server to establish encrypted connections with clients

Cerberus FTP Server will automatically generate a new self-signed certificate for you the first time you run the Getting Started Wizard. You can replace the certificate at any time through the Security page of the Server Manager.. See section 19 for details.

5.1.4.1 Web Administration Password

You have the option to configure a web administration and remote API access password on the Security Wizard page. You will need this password to use the HTTPS Web Admin Console, the Cerberus SOAP API, or you can configure the desktop interface to ask for this password when opening the user interface. You should set a strong password here even if you are not using web administration. Please note that the password strength estimation meter is only meant as a guide. It will flag poor passwords but there is no official weighting system and this meter should only be utilized as a loose guide to improving your password.

5.1.5.2 Protocol Security

The last option allows you to configure the server to only accept encrypted FTP connections. Normal FTP has no encryption and therefore allows passwords and data to be transmitted unencrypted over a network.

Fortunately, it is possible to establish a normal unencrypted FTP connection and then "upgrade" the connection to secure encryption through special FTP commands (this enhanced protocol is called FTPES). This type of connection depends on the client issuing FTP commands instructing the server to establish encryption before accepting login credentials. However, the client can also continue as a normal FTP connection without enabling encryption. This situation allows for unencrypted connections and presents a security issue for servers.

If you wish to allow FTPES secure connections, but not FTP, then you must instruct the server to require encryption before allowing a connection to proceed.

Checking this option does exactly that. It requires the client to upgrade the connection to use encryption before allowing login.

5.1.5.3 Final Steps

Click the Finish button to complete the Getting Started Wizard. Your server is now ready to accept local network FTP/S, SSH SFTP, or HTTP/S web client connections. Please take a look at the next section for any changes that might need to be made to your firewall or router to allow connection from outside of your local network to reach your server.

6.0 Getting Started - Network Setup

6.1 Basic Setup so users can connect from the Internet

FTP connections within your local network usually works without any problems. However, when you want the FTP server to be available outside of your local network, additional steps are often necessary to make the server visible to the outside world. The following steps are usually required to allow Cerberus FTP Server to be accessed from the Internet:

6.1.1. Step 1 - Control Connection

The control connection port Cerberus FTP Server is listening on needs to be forwarded from your router and/or firewall to the machine hosting Cerberus. The default port that Cerberus listens on is port 21. Consult your router and/or firewall documentation for instructions on how to set up port forwarding. Finishing this step will allow Internet users to establish a connection with your server. The next step is making sure **passive mode** is configured so that directory listings and file transfers work.

6.1.2 Step 2 - Passive Mode

To allow passive mode to work properly, you must forward the passive range of ports from your router to the machine running Cerberus. See "My IP address begins with 192.168.xxx.xxx. Is there anything special I have to do for people to see my FTP Server on the Internet?" for detailed instructions on how to make sure passive mode is set up properly. If you don't perform this step, users may be able to log in but directory listings may hang and timeout.

6.1.3 Step 3 - Firewalls

Make sure any firewalls you are running are allowing connections on port 21. Cerberus will automatically attempt to add itself to the Windows Firewall Exception list (you will be prompted to allow this). However, you may still have to manually add an exception to allow port 21 connections into your computer.

7.0 How many types of FTP are there?

There are three types of FTP connections possible (Cerberus FTP Server supports all three):

FTP: Plain, unencrypted FTP that defaults over port 21. Most web browsers and Windows Explorer support basic FTP, but it is insecure and not recommended as passwords and data are not encrypted.

FTPS: **Implicit** SSL/TLS encrypted FTP that works just like HTTPS. Security is enabled with SSL as soon as the connection starts. The default FTPS port is 990. This protocol was the first version of encrypted FTP available, and while considered deprecated, is still widely used. None of the major web browsers support FTPS.

FTPES: **Explicit** FTP over SSL/TLS. This starts out as plain FTP over port 21, but through special FTP commands is upgraded to TLS/SSL encryption. This upgrade usually occurs before the user credentials are sent over the connection. FTPES is a somewhat newer form of encrypted FTP and is considered the preferred way to establish encrypted connections because it can be more firewall friendly. None of the major web browsers support FTPES.

7.1 Controlling what types of FTP are Allowed

You can control the types of FTP connections allowed at both the user level, and at the listener level.

7.1.1 Restricting FTP connections at the User level

For a user or group account, the **Require Secure Control** and **Require Secure Data** constraints are meant to enforce that the connection is encrypted using either FTPS or FTPES. If **Require Secure Control** is checked, FTP over port 21 will be denied login if the user attempts to authenticate without upgrading the connection to use encryption. If the FTP connection is upgraded to use encryption (upgraded to FTPES), then the user will be allowed to send login credentials and attempt to log in. Cerberus requires an FTP listener to allow FTP or FTPES connections.

FTPS connections are always encrypted, and connections that come through on an FTPS listener will always be allowed to attempt to login.

The user and group constraints **Allow FTP** and **Allow FTPS** is meant to control what protocol a user can log in over. If **Allow FTP** is selected for a user, then both FTP and FTPES connections will be allowed to attempt to login over an FTP listener. This can be further restricted to only allowing FTPES connections by selecting the **Require Secure Control** and **Require Secure Data** constraints for the user.

You can create combinations of these options to allow exactly the type of protocol and security settings that you prefer.

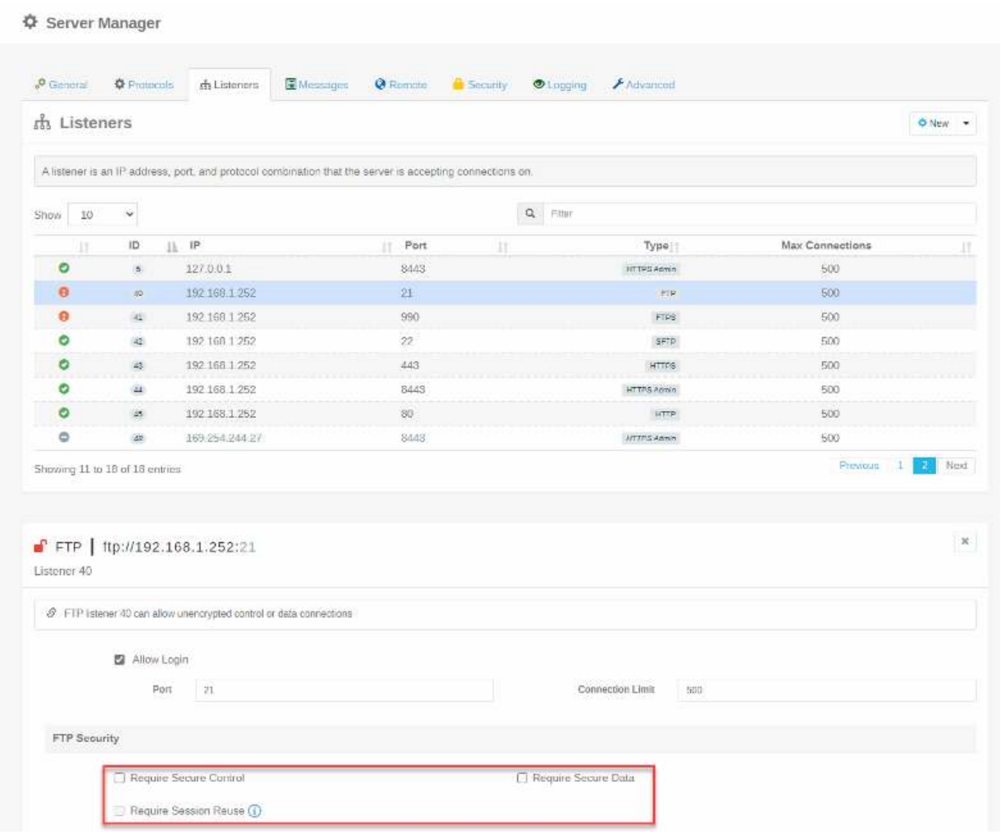
For example:

To allow any protocol, as long as it is secure, leave **Allow FTP** and **Allow FTPS** checked, and make sure **Require Secure Control** and **Require Secure Data** are checked.

This will allow connecting over implicit FTPS listeners on port 990, and explicit FTPES connections over FTP listeners on port 21 (as long as the connection gets upgraded to TLS/SSL encryption before the user attempts to log in).

7.1.2 Restricting FTP connections at the Listener level

In addition to the fine-grain control, administrators have at the user level, broader restrictions can be enforced at the listener level. FTP listeners also have the **Require Secure Control** and **Require Secure Data** settings. These settings are checked first before a user even attempts to log in. If the **Require Secure Control** and **Require Secure Data** options are specified for an FTP listener, then only secure FTPES connections will be allowed. These settings are enforced before the individual user settings are checked. The **Require Session Reuse** setting ensures passive mode connections are always resumed from the correct control session, preventing man in the middle attacks, and is recommended.



Listeners

A listener is an IP address, port, and protocol combination that the server is accepting connections on.

Show 10

Filter

	ID	IP	Port	Type	Max Connections
✓	5	127.0.0.1	8443	HTTPS Admin	500
!	40	192.168.1.252	21	FTP	500
!	41	192.168.1.252	990	FTPS	500
✓	42	192.168.1.252	22	SFTP	500
✓	43	192.168.1.252	443	HTTPS	500
✓	44	192.168.1.252	8443	HTTPS Admin	500
✓	45	192.168.1.252	80	HTTP	500
⊖	48	169.254.244.27	8443	HTTPS Admin	500

Showing 11 to 18 of 18 entries

FTP | ftp://192.168.1.252:21

Listener 40

FTP listener 40 can allow unencrypted control or data connections

☑ Allow Login

Port 21

Connection Limit 500

FTP Security

□ Require Secure Control

□ Require Secure Data

□ Require Session Reuse

8.0 SSH2 SFTP Setup

8.1 About SSH SFTP Support in Cerberus FTP Server

Cerberus FTP Server Professional edition and higher supports the SSH2 File Transfer Protocol, also known as SFTP. SFTP is a network protocol that provides secure and reliable file access, file transfer, and file management functionality. Features of the protocol include resuming interrupted file transfers, directory listings, getting and setting file attributes, and remote file removal.

There are currently 6 different versions of the SFTP protocol, with versions 3 - 6 being in common use by modern SFTP clients. Cerberus supports SFTP version 3, 4, 5, and 6 clients.

Cerberus also supports SSH public key authentication.

8.2 Supported SSH2 Key Exchange Methods

Cerberus supports both Diffie-Hellman and Elliptic Curve Diffie-Hellman (ECDH) SSH2 key exchange methods. The following exchange methods are supported:

- diffie-hellman-group1-sha1
- diffie-hellman-group14-sha1
- diffie-hellman-group-exchange-sha1
- diffie-hellman-group14-sha256
- diffie-hellman-group16-sha512
- diffie-hellman-group18-sha512
- diffie-hellman-group-exchange-sha256
- ecdh-sha2-nistp256
- ecdh-sha2-nistp384
- ecdh-sha2-nistp521

8.3 Supported SSH2 Ciphers

The following SSH ciphers are supported:

- 3des-cbc
- aes256-cbc
- aes192-cbc
- aes128-cbc
- aes256-ctr
- aes192-ctr
- aes128-ctr

8.4 Supported SSH2 MAC Algorithms

The following SSH MAC algorithms are supported:

- hmac-sha1
- hmac-sha1-96
- hmac-sha2-256
- hmac-sha2-256-96
- hmac-sha2-512
- hmac-sha2-512-96

8.5 Adding an SSH2 SFTP Listener

You must first have at least one SFTP listener for Cerberus to be able to accept SFTP connections. Cerberus FTP Server will automatically add and enable SFTP listeners on each available IP address the first time it is run so you normally do not need to add an SFTP listener. However, if you've previously removed an SFTP listener you can add a new one from the Listeners page of the Server Manager.

To add a new SFTP listener:

1. Open the Server Manager
2. Select the **Listeners** page
3. Press the “New” button The "Add New Listener" dialog box will appear to ask for the interface details (interface IP, type, and port combination)
4. Select the IP address that you want to listen for connections on
5. Select the **SSH SFTP** interface type
6. Enter the port you wish to listen on (the default for SSH2 SFTP is 22). Cerberus will automatically pre-populate the port with the default port for the type of listener you are adding
7. Press the **Add Listener** button to add the listener
8. The listener should now be added to the Interfaces list.

8.6 Allowing SSH2 SFTP Connections through a Firewall

SFTP connections use port 22 by default. You may need to allow that port through your firewall to the machine running Cerberus FTP Server. You may also need to make sure your router is forwarding incoming connections on that port to the machine running Cerberus FTP Server.

8.7 Enabling or Disabling Existing SFTP Listeners

In addition to adding and deleting interfaces, Cerberus allows an administrator to disable or enable an existing listener. This feature can be used to temporarily disable a listener or to re-enable a listener that has become disabled because of a port conflict or trial license expiration.

9.0 Configuring the Server

9.1 Allowing External Access to your Server

Depending upon your connection to the Internet, you may need to configure your router or firewall before users outside of your local network can see your FTP server. Communication with an FTP server is done through two connections, a control connection, and a data connection. Ensuring these connections can be established are the two areas where special attention is usually needed.

9.1.1 The FTP control connection

The control connection is always the first connection established with an FTP server. The control connection's purpose is to allow clients to connect and send commands to the server (and receive server responses). **Port 21** is considered the default control connection port, and this is the default port that Cerberus FTP Server will configure your IP interfaces to listen on for new connections. Using the default port is not mandatory - the administrator is free to change the interface to use any free port on the system as the listening port. However, if the administrator is running a software-based firewall, the administrator must be certain that incoming connections are not blocked on the port chosen for the control connection. If the port that Cerberus is listening on is blocked, no one will be able to see or connect to the FTP server.

9.1.2 The FTP data connection

The second type of connection is called the data connection. This is the connection that an FTP server uses to exchange file listings and transfer files on. When an FTP client uses the control connection to instruct Cerberus FTP Server to send a file listing or transfer a file, the actual data exchange takes place on the data connection. The data connection is usually where most of the confusion and problems arise for FTP server administrators.

There are two different ways a data connection can be established between an FTP client and an FTP server. The first is commonly called **active** FTP. In this mode, an FTP client sends the IP address and port that the client is currently listening for data connections on to the FTP server. The client accomplishes this by sending the server a *PORT* command over the control connection. Using the address and port from the *PORT* command, the FTP Server then connects to the client and sends the file or file listing. When using **active** FTP, the administrator has to make sure that port 20 on the machine that Cerberus FTP Server is running on is open for outgoing connections. The reason for this is because when using **active** FTP, the server always establishes connections from port 20. Most firewalls allow outgoing connections automatically, so manually opening up port 20 for outgoing connections is usually not necessary.

The other way to establish a data connection between client and server is to use **passive** FTP. **Passive** mode was introduced to get around common problems with client firewalls. Instead of the FTP server connecting to the FTP client, the client connects to the FTP server using a port previously communicated using the *PASV* command. When a client issues the *PASV* command, the FTP server responds with a port that the server is currently listening on for data communication. Problems occur with **passive** FTP when the firewall that Cerberus FTP Server is running on is blocking the selected ports. To get around this problem, the administrator is required to open up the range of ports that

Cerberus has reserved for **passive** FTP connections. You can configure what range of ports Cerberus FTP Server uses for **passive** FTP mode by looking under the 'Advanced' tab of the Server manager.

Failures during *LIST, NLST, MLSD, MLST, RETR*, or *STOR* operations can usually be attributed to problems with the data connection.

9.2 Common Network Configurations

A server or PC running Cerberus FTP Server with access to the Internet often fits into one of two configurations:

9.2.1 Configuration 1: Your computer is connected directly to the Internet

This is the simplest network configuration you can have and usually requires little or no configuration to Cerberus FTP Server to allow full access. This configuration is most common with dial-up, DSL, cable modem, and other broadband users. However, machines connected to the Internet directly often employ a software firewall to provide some protection against unwanted intrusion attempts. While some firewall software can automatically detect an FTP server and properly configure itself, the administrator usually has to manually configure the firewall. See the explanation above about the control and data connection for common ports that have to be allowed through a firewall.

9.2.2 Configuration 2: Your computer is connected to a router, and the router is connected to the Internet

Routers usually act as firewalls, so the same problems that can occur in Configuration 1 can occur here. Follow the advice in Configuration 1 to resolve firewall problems.

In addition to the firewall problems that can occur in this network configuration, there is now the problem that the IP address you are using on your machine is not the IP address that the Internet sees for your machine. Other users on the Internet usually see your router's IP address instead of your PC's private address. Routers are devices on your network, just like your PC, and they have their own IP address, and that is the IP address the router tells other computers is your address when you go out on the Internet. When a user attempts to connect to the FTP server, they need to use the Internet-facing IP address of the router (the router is where the connection is really happening), not the private address of the computer Cerberus FTP Server is running on. When the router receives the connection attempt it is then able to forward the connection to your computer.

The first thing to check in this configuration is that your router is sending all of the FTP traffic to the computer Cerberus FTP Server is running on. Most routers have a web-based configuration utility that you can use to configure **Port Forwarding**. Specifically, you will want to make sure you forward the control and possible data connection ports to the computer running Cerberus FTP Server.

There is one more problem that crops up in this network configuration. To properly allow **passive** transfer mode, the administrator will have to make sure Cerberus is giving out the router address in response to PASV requests. You can automatically enable this by making sure "WAN IP Autodetection" is enabled in the 'General' tab of the Server Manager. Alternatively, you can enter the IP address of the router manually for each interface in the "Use different IP for PASV mode" IP box under the Server manager's 'Listeners' tab.

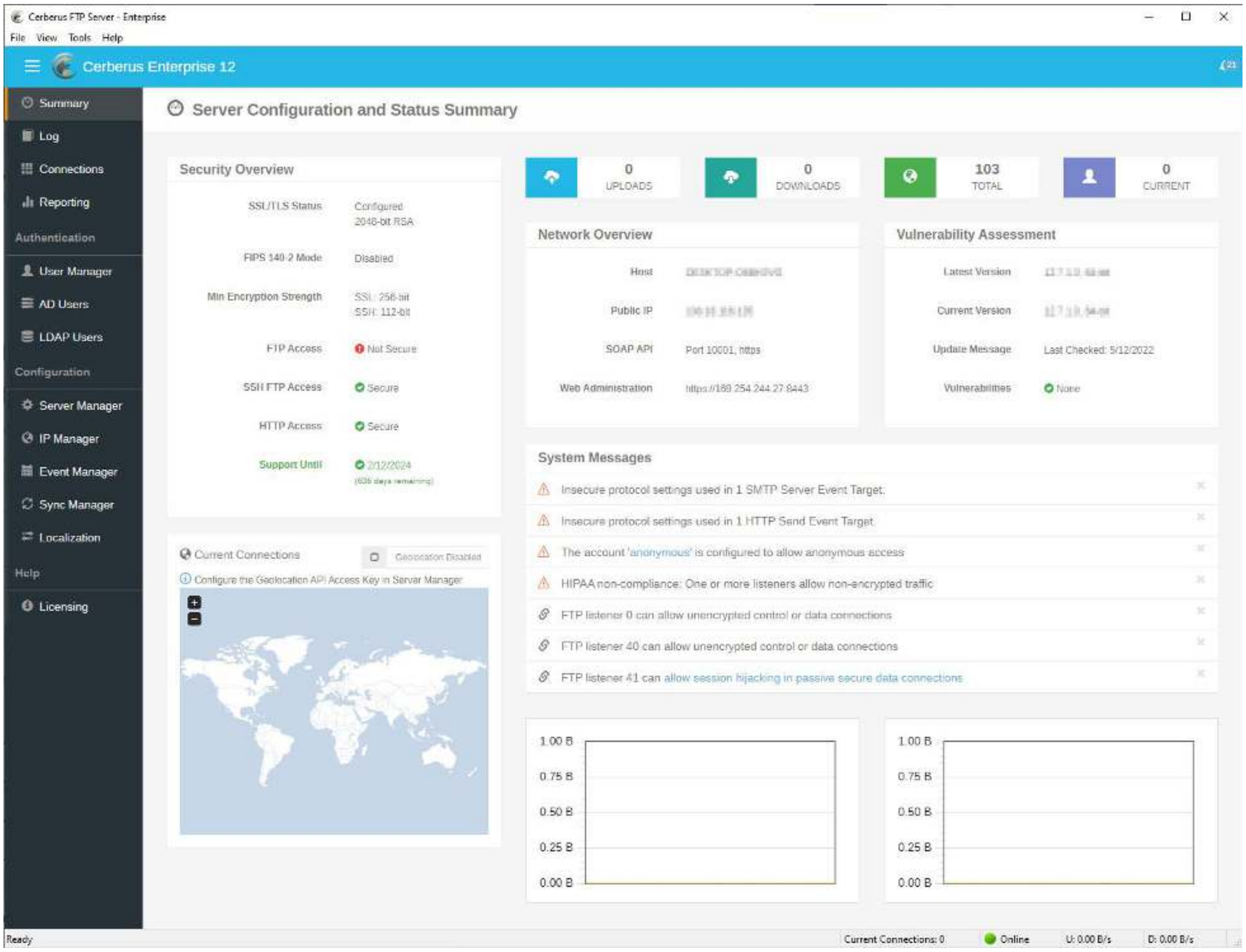
While more complicated network configurations are possible, most users will fall into one of the above configurations.

10.0 The Summary View

10.1 Understanding the Summary View

Provides the administrator with a one-page overview of the server's configuration and any potential security issues that may be present.

Every time a configuration change is made the server scans the current Cerberus configuration at startup to look for any potential security issues that might result from the current system configuration. System warnings and messages are displayed in the System Messages list and each protocol type is given an overall security status indicator.



Server Configuration and Status Summary

Security Overview

SSL/TLS Status	Configured 2048-bit RSA
FIPS 140-2 Mode	Disabled
Min Encryption Strength	SSL: 256-bit SSH: 112-bit
FTP Access	Not Secure
SSH FTP Access	Secure
HTTP Access	Secure
Support Until	2/12/2024 (636 days remaining)

0 UPLOADS	0 DOWNLOADS	103 TOTAL	0 CURRENT

Network Overview

Host	[redacted]
Public IP	[redacted]
SOAP API	Port 10001, https
Web Administration	https://169.254.244.27:8443

Vulnerability Assessment

Latest Version	[redacted]
Current Version	[redacted]
Update Message	Last Checked: 5/12/2022
Vulnerabilities	None

System Messages

- Insecure protocol settings used in 1 SMTP Server Event Target.
- Insecure protocol settings used in 1 HTTP Send Event Target.
- The account 'anonymous' is configured to allow anonymous access
- HIPAA non-compliance: One or more listeners allow non-encrypted traffic
- FTP listener 0 can allow unencrypted control or data connections
- FTP listener 40 can allow unencrypted control or data connections
- FTP listener 41 can allow session hijacking in passive secure data connections

Current Connections

Geolocation Disabled

Configure the Geolocation API Access Key in Server Manager

Cerberus FTP Server Summary View

The possible status for each protocol type is

Secure	All listeners currently active for this protocol type are configured to accept only encrypted connections.

Not Secure	Some or all listeners currently active for this protocol type are configured to allow unencrypted connections.
Disabled	There are no listeners currently active on the server for this protocol.

10.2 Common System Messages

There are generally two types of system messages displayed in the System Messages list - general messages and security messages.

Anytime a protocol is listed as Not Secure there will be a system security message detailing the reason. Common system messages, their explanation, and resolution, if applicable, are detailed below.

10.2.1 “We recommend disabling TLS 1.0/1.1 for SSL-based SOAP (Remote) connections”

Explanation: TLS 1.0 and TLS 1.1 are now considered insecure, so TLS 1.2 should now be used for all connections. No impact on customers as this is internal to Cerberus. The only impact would be if you plan to use our SOAP API for Cerberus server administration and you are connecting to it from third party software. If that is the case check what TLS protocols are supported.

Resolution: To resolve this issue navigate to 'Server Manager' > 'Remote' > 'SOAP TLS Settings'. Uncheck TLS 1.0 and 1.1 and 'Update'.

10.2.2 “Server is configured to allow the weak 3DES/ RC4 encryption cipher with SSL. 3DES/ RC4 should be disabled” and/or "HIPAA non-compliance: Your current SSL settings allow encryption that is less than 128-bits”

Explanation: Ciphers that are now considered 'weak' and insecure are being permitted. Most modern clients are compatible with newer ciphers. This would only impact old clients and our recommendation would be for you to have your customers upgrade their clients.

Resolution: To resolve this issue Navigate to Server Manager > Security > Advanced TLS Security Settings

Select one of the security profiles from the drop down and 'Update'

Navigate to 'Server Manager' > 'Protocols' > 'SSH SFTP' > 'SSH Security Defaults'.

Press the '128-bit Min' button and 'Update'. This will bring you up to 128-bit standard as required. You also have the option to use 256-bit Min.

10.2.3 “[SOAP] Server is configured to allow the weak 3DES/ RC4 encryption cipher with SSL. 3DES/ RC4 should be disabled”

Explanation: Ciphers that are now considered 'weak' and insecure are being permitted.Most modern clients are compatible with newer ciphers. This would only impact old clients and our recommendation would be for you to have your customers upgrade their clients.

Resolution: To resolve this issue Navigate to 'Remote' > 'SOAP TLS Settings' > 'SOAP SSL Cipher String'

Select one of the security profiles from the drop-down and 'Update'

10.2.4 “FTP Listener X can allow unencrypted control or data connections”

Explanation: Normal FTP has no encryption and therefore allows passwords and data to be transmitted in the clear over a network. To address this security issue, two secure forms of FTP were developed called implicit FTPS and explicit FTPES. Implicit FTPS is very similar to HTTPS and takes place on a completely separate port from typical FTP. Interfaces of this type are always encrypted and considered secure. Explicit FTPES, however, starts on a normal unencrypted FTP connection and is then "upgraded" to a secure connection through special FTP commands. This type of connection depends on the client issuing commands instructing the server to enable encryption. However, the client can also continue as a normal FTP connection without enabling encryption. This situation allows for unencrypted connections and presents a security issue for servers.

Resolution: To resolve this issue and still allow FTP access there are two possible solutions. One is to remove all FTP listeners and only enable FTPS listeners. FTPS listeners only accept encrypted communications and are considered secure.

If you wish to also allow FTPES secure connections then you must instruct the server to require encryption before allowing a connection to proceed. To require the FTP listener to require encryption, go to the Listeners page of the Server Manager and for each FTP interface, select the Require Secure Control and Require Secure Data options.

For more detailed information, please take a look at our information page describing the different forms of FTP and secure FTP.

10.2.5 “FTP Listener X can allow session hijacking in passive secure data connections”

Explanation: This warning highlights a risk that arises from how the FTP protocol works with its separate control and data channels in passive mode. This FTP setting prevents another current user from hijacking a new passive data connection. Session reuse takes advantage of TLS features to verify that a resumed data connection pairs with the same active control connection.

FTP is unusual in that it has a control channel and a data channel. Data connections can be made from the server back to the client or vice versa, but today, due to firewalls, the client usually connects back to the server using passive mode. While only the client will know the port number to be used for the data connection, the problem is only a subset of available ports are typically used for the data channel. If you have a busy server, attackers can try random ports and may eventually find an open data connection and hijack it.

Reusing the TLS session protects you from the possibility that an attacker could hijack an FTP data connection. If the server requires that the same TLS session be used for the data connection resumption, the attacker will not be able to start their own TLS session, preventing them from accessing any data.

This works because the server and the client share an encrypted session key. The client can pass that key back to the server to connect to the data channel. The server, when this feature is enabled, checks the key from the control channel and only allows the data connection if they match.

Resolution: **As of Cerberus 12.7.0, administrators have three options:**

1. *Secure your FTP and FTPS listeners* by turning on the option and make sure all the clients are updated to handle session reuse;
2. *Create a separate FTP/FTPS listener* on a custom port for the old client only. On that listener, you can keep the reuse option off and/or turn off 'require secure data' and 'require secure control'. It is strongly recommended to restrict connections to it by IP to prevent others from connecting to it, or;
3. *Turn off require session reuse and dismiss the Summary message* as a last resort if you have clients that cannot update their software. Again, it's strongly recommended to IP restrict access to this listener. Ensure that 'require secure control' and 'require secure data' are still ON where possible.

10.2.6 "HTTP Listener X only accepts unencrypted connections"

Explanation: Connections of type HTTP are always unencrypted and are therefore very susceptible to inspection on a network. System administrators are encouraged to disable HTTP listeners in favor of secure HTTPS listeners.

Resolution: To resolve this issue the system administrator must disable any HTTP listeners in the system, or set the redirect to HTTPS flag on the HTTP listener to make sure the connection is immediately redirected to HTTPS. HTTPS listeners will not trigger a security issue.

10.2.7 "HIPAA Non-compliance: One or more listeners allows non-encrypted traffic"

Explanation: HIPAA requires all data to be encrypted before being sent over a network. You have an active listener that allows data to be transmitted without encryption.

An FTP listener without the Require Secure Control and Require Secure Data settings will trigger this warning. An HTTP listener that is not configured to redirect to HTTPS will also result in a warning. Allowing SSH SFTP to use no encryption (configured from the **Advanced** section on the **Security** page of the Server Manager) will also result in a warning.

Resolution: To resolve this issue the system administrator must disable any HTTP listeners in the system (or redirect them to HTTP), configure FTP listeners to require encryption, and make sure SSH SFTP listeners are not allowed to use any encryption for connections.

10.2.8 “FXP is enabled and could leave the server vulnerable to an FTP bounce attack”

Explanation: FTP bounce attack is an exploit of the FTP protocol whereby an attacker is able to use the PORT command to request access to ports indirectly through the use of the victim machine as a middle man for the request.

Resolution: Go to the **Advanced** page of the Server Manager and check the option to Deny FXP Transfers.

10.2.9 “Server is configured to allow FTP data connection to reserved ports”

Explanation: You will receive this warning if you have configured Cerberus to allow FTP data connections to ports less than 1025. Ports 1 through 1024 are intended for system services, so those ports are called reserved ports. FTP should normally not be allowed to establish data connections within that port range.

Resolution: Go to the **Advanced** page of the Server Manager and check the option to Deny Reserved Ports.

10.2.10 “You should set a Remote Access password”

Explanation: Web administration and SOAP API remote access use an admin password to control or deny access to the server.

Resolution: Go to the Remote page of the Server Manager and set an admin password.

10.2.11 “Password Policy is weak”

Explanation: This is just an advisory warning. We have made the recommended minimum password policy stricter in more recent versions of Cerberus and we recommend increasing your password standards for security. If you change this it affects NEWLY set passwords for new and existing users only.

Resolution: Navigate to 'User Manager' > 'Policy' > 'Password Complexity Requirements'. Change the password requirements and press 'Update'

11.0 The User Manager

11.1 About Cerberus FTP Server Authentication

Cerberus FTP Server can manage user accounts from three different sources. The first is the default Cerberus FTP Server user manager. The Cerberus user manager is displayed in the 'Users' tab of User Manager. The accounts within User Manager are users created just for Cerberus FTP Server. The directions on this page are for adding a user to this list.

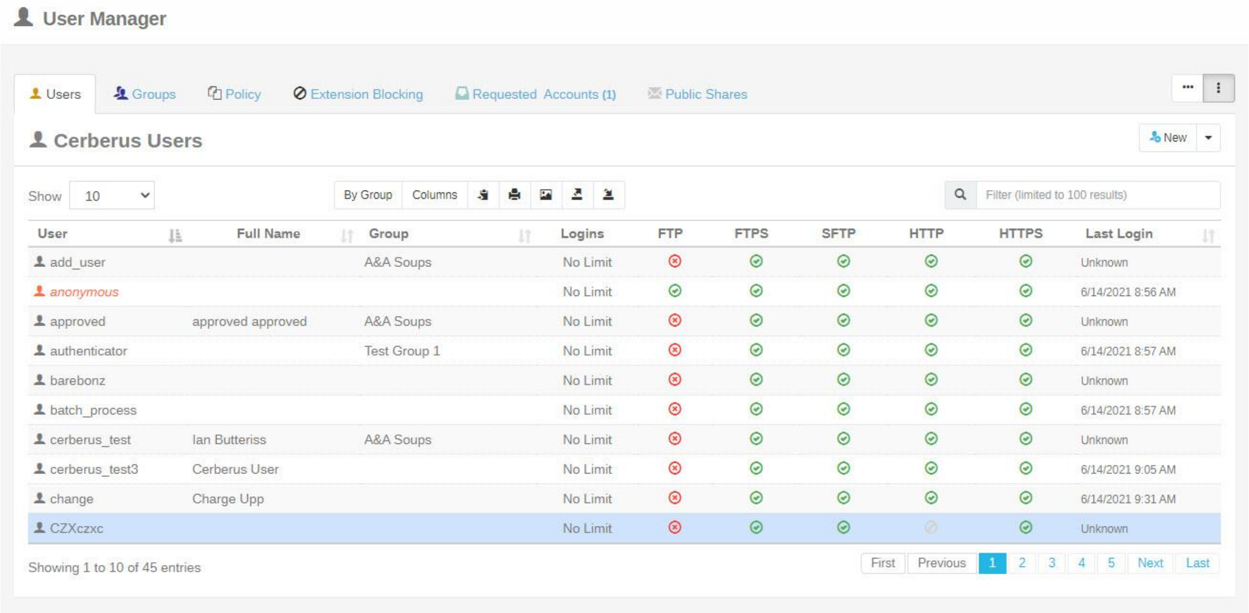
You may also use Cerberus FTP Server to authenticate Active Directory users when the machine hosting Cerberus is part of a domain (or the local NT account database), even if the computer Cerberus FTP Server is installed on is not the domain controller. See the section on Active Directory Authentication for more information on how to configure Cerberus to allow authentication of Active Directory domain users.

Finally, users can also be authenticated against an LDAP service. See the section on configuring Cerberus for LDAP authentication for more information.

NOTE: Active Directory and LDAP authentication are only available in the Professional and Enterprise editions of Cerberus FTP Server.

11.2 Adding a new user

Users can be added and modified in Cerberus FTP Server by opening up **User Manager** and selecting the **Users** tab.



User	Full Name	Group	Logins	FTP	FTPS	SFTP	HTTP	HTTPS	Last Login
add_user		A&A Soups	No Limit	☒	✓	✓	✓	✓	Unknown
anonymous			No Limit	✓	✓	✓	✓	✓	6/14/2021 8:56 AM
approved	approved approved	A&A Soups	No Limit	☒	✓	✓	✓	✓	Unknown
authenticator		Test Group 1	No Limit	☒	✓	✓	✓	✓	6/14/2021 8:57 AM
barebonz			No Limit	☒	✓	✓	✓	✓	Unknown
batch_process			No Limit	☒	✓	✓	✓	✓	6/14/2021 8:57 AM
cerberus_test	Ian Butteriss	A&A Soups	No Limit	☒	✓	✓	✓	✓	Unknown
cerberus_test3	Cerberus User		No Limit	☒	✓	✓	✓	✓	6/14/2021 9:05 AM
change	Charge Upp		No Limit	☒	✓	✓	✓	✓	6/14/2021 9:31 AM
CZXczxc			No Limit	☒	✓	✓	⊘	✓	Unknown

Showing 1 to 10 of 45 entries

To add a user, click the **New** button from the button group along the right side of the page. A new user form will appear under the user list box. All usernames must be unique and are case insensitive. Once you have entered the new username, continue filling out the remaining fields. Once the user is created, their settings can be further configured by selecting the user and using the settings categories above the details. The categories are Profile, Constraints, Authentication, Allowed Protocols, and Virtual Directories

Some of the configurable properties for users are:

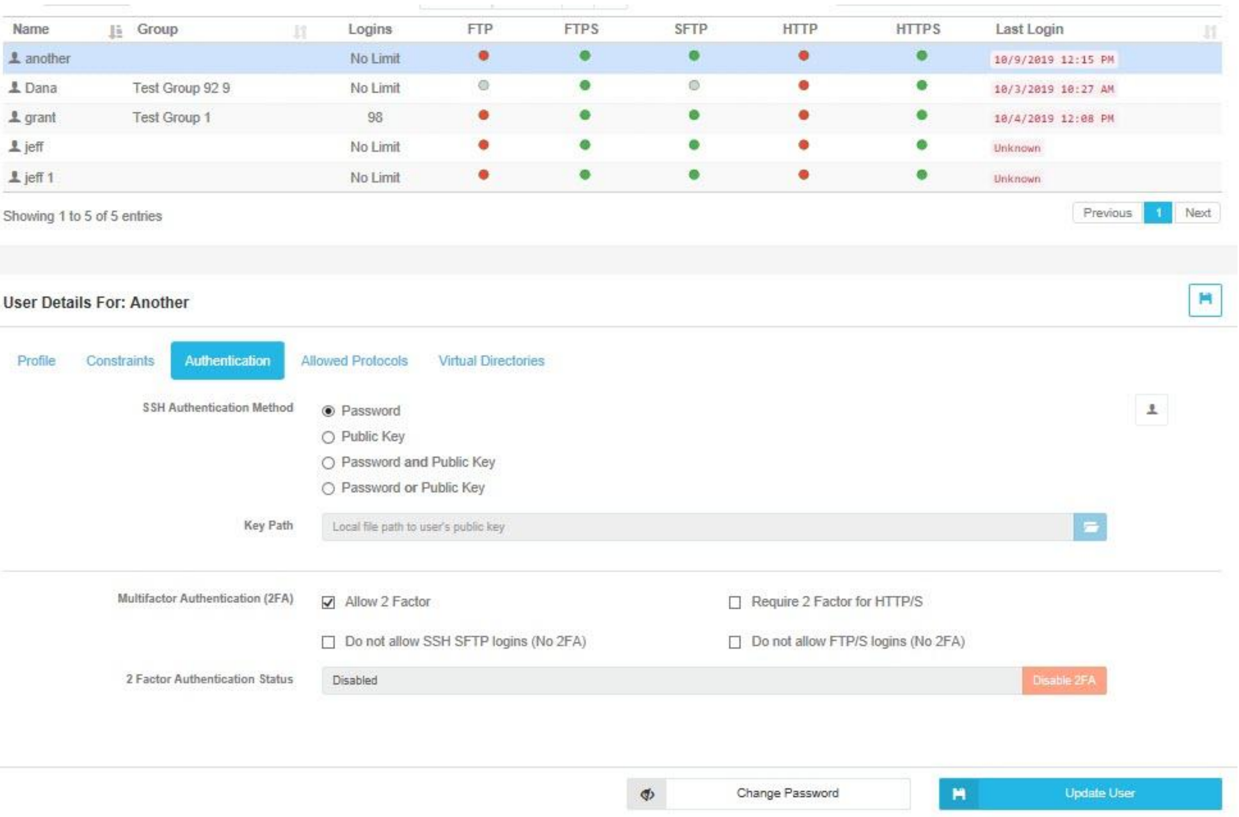
Profile	
Primary Group	A Cerberus FTP Server Group this user has been assigned to. When a user has been assigned to a primary group, all the group settings override the user's settings.
Secondary Group	A Cerberus FTP Server Group this user has been assigned to. When a user has been assigned to a secondary group, just the virtual directories assigned to the group are now assigned to the user. The user's other settings remain and are not overridden by the group settings.
Constraints	
Password Never Expires	Overrides the password expiration policy if you have set passwords to expire after a given number of days.
User Can Change Password	Defines if a user can change their own password
Anonymous	If checked, the user password is ignored and the user can be logged in using any password.
Disabled	Determines whether the account can log in or not. A disabled account cannot log in into the server.
Max Logins	The maximum number of connections this user can make to the server at the same time.
Disable Date	If a date is set here then the account will become disabled after the date specified. **Note**: The granularity of the timer is 30 minutes. The account will be disabled within 30 minutes of the time set.
Maximum Upload Filesize	This field can be used to limit the maximum size of an uploaded file. This value defaults to unlimited. The file size is specified in bytes. Specify 0 or any non-positive value to reset the maximum file size to unlimited.
Allowed IP Addresses	A comma-separated list of IP addresses that this user can login from. If no IP addresses are specified then no per-user IP address filtering is enforced. IP addresses can be specified as a single IP, a range of IP addresses separated by a dash, e.g. 192.168.0.100 - 192.168.0.150, or a CIDR-formatted IP address range. Multiple formats can be combined, with each single IP or range separated by a comma. Note, global IP address deny lists or allow lists are always enforced first, regardless of this setting.
Authentication	
SSH Authentication Method	Determines the authentication requirements for logging into an SFTP interface. Valid options are: ● **Password Only**: Require only a password for authentication. ● **Public Key Only**: Require only a valid public key for authentication ● **Public Key and Password**: Require both a valid public key and a valid password for authenticating a user ● **Public Key or Password**: Require either a valid public key or a valid password for authenticating a user

Multifactor Authentication (2FA)	Determines if 2 Factor authentication is allowed or required. ● **Allow 2 Factor:** This option allows users to set up 2fa if they choose to ● **Require 2 Factor for HTTP/S:** This makes 2fa a requirement when using the HTTP/S web client. ● **Do not allow SSH SFTP logins (No 2FA):** This option will not allow users to log in via SSH SFTP when 2FA is enabled. ● **Do not allow FTP/S logins (No 2FA) :** This option will not allow users to login via FTP/S when 2FA is enabled.
Allowed Protocols	
Permitted Login Protocols	Controls which protocols a user is allowed to log in with. If a protocol is not checked then the user will not be allowed to log in using that protocol.
Require Secure Control	(Applies to FTP only) If enabled, this user can only log in to the server using a secure TLS/SSL encrypted connection.
Require Secure Data	(Applies to FTP only) If enabled, file transfers will only be allowed over secure TLS/SSL encrypted connections.

11.3 Configuring a user for SSH Public Key Authentication

The procedure for configuring a user for SSH Public Key Authentication in Cerberus FTP Server is:

1. Open the Cerberus FTP Server **User Manager**. The default page is the **Users** tab.
2. Select the User from the **Cerberus Users** list that you wish to configure for Public Key Authentication.
3. Click on the **Authentication** tab for the selected user. The **Authentication** Requirements dialog will appear.



Name	Group	Logins	FTP	FTPS	SFTP	HTTP	HTTPS	Last Login
another		No Limit	🔴	🟢	🟢	🔴	🟢	10/9/2019 12:15 PM
Dana	Test Group 92 9	No Limit	⚪	🟢	⚪	🔴	🟢	10/3/2019 10:27 AM
grant	Test Group 1	98	🔴	🟢	🟢	🔴	🟢	10/4/2019 12:08 PM
jeff		No Limit	🔴	🟢	🟢	🔴	🟢	Unknown
jeff 1		No Limit	🔴	🟢	🟢	🔴	🟢	Unknown

Showing 1 to 5 of 5 entries

User Details For: Another

SSH Authentication Method
● Password
○ Public Key
○ Password and Public Key
○ Password or Public Key

Key Path: Local file path to user's public key

Multifactor Authentication (2FA)
☑ Allow 2 Factor
□ Require 2 Factor for HTTP/S
□ Do not allow SSH SFTP logins (No 2FA)
□ Do not allow FTP/S logins (No 2FA)

2 Factor Authentication Status: Disabled

4. Select the **Public Key Only** or **Public Key and Password** radio option. The **Key Path** edit box and file selection button will become visible/enabled.
5. Select the folder button next to the **Key Path** edit box. A file selection dialog box will appear.
6. Select the public key file you wish to use for the selected user. Press the **Open** button to select the file.
7. Press the **OK** button on the Change **SSH Authentication Requirements** dialog to close and save the new SSH authentication settings.
8. Press the **Close** button on the **User Manager** to save the changes to the selected user.

11.4 The Virtual Directory System

The virtual directory (VD) system allows the administrator to attach any directory or drive to the root. When a client requests the root directory from the server, the VDs you specify are sent to the client. The client can also navigate to any of the VD directories' subdirectories. The VD system takes care of all path translation. Security settings can be specified for each virtual directory. All subdirectories under the VD inherit the security settings of the VD.

Shared (remote) resources can be accessed using the UNC path as long as the account running the Cerberus service has permission to access the resources.

There are 2 modes that a user account can operate in with respect to the virtual file system. The two modes are Simple Mode and Standard Mode.

11.5 Simple Virtual Directory mode

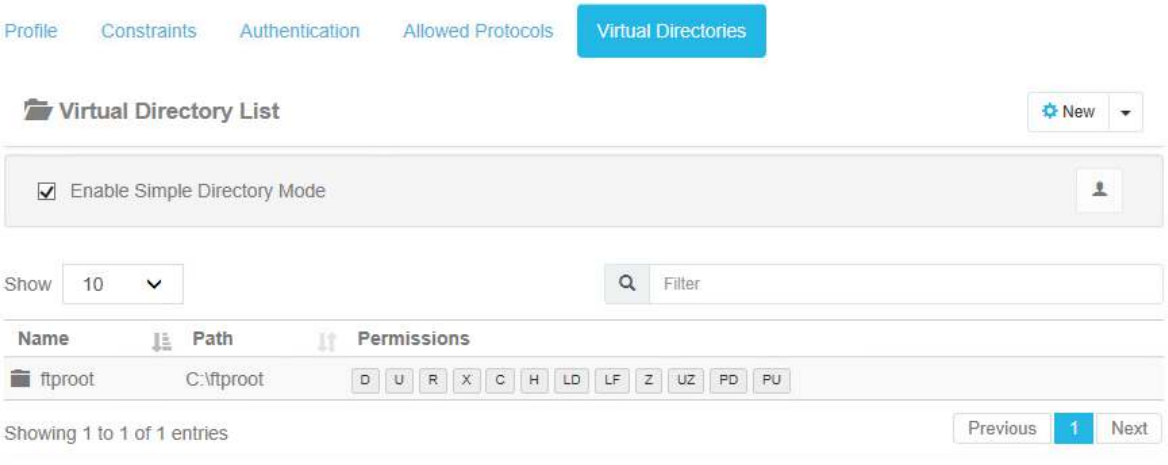
When a user account uses simple directory mode (the **Simple Directories** option is checked), the administrator can only assign ONE directory to represent the virtual directory for that user. Instead of that directory being seen as a subdirectory off of the root, the virtual directory selected will be the directory the user is placed in when they first log into the server. In other words, the directory selected as the virtual root directory will be the root directory.

11.6 Standard Virtual Directory mode

In standard mode (the **Simple Directories** option is unchecked), the administrator may add as many directories as virtual directories to a user account as desired. The directories selected will appear as subdirectories off of the root when the designated user logs into the server.

11.6.1 A Virtual Directory Mode Example

Let's take a user with one simple virtual directory called **ftproot** that maps to **C:\ftproot**.



Virtual Directory List

☑ Enable Simple Directory Mode

Name	Path	Permissions
ftproot	C:\ftproot	D U R X C H LD LF Z UZ PD PU

Showing 1 to 1 of 1 entries

Virtual Directory Settings for a User

In **Simple Directory** mode, the remote root directory that the user sees, "**/**", is mapped directly to **C:\ftproot** on the server. The actual virtual directory name is ignored (you can think of it as always being named "**/**"). The user will see all files and folders in **C:\ftproot** listed in their root directory. They can upload and download files directly into the root directory and they will be uploaded or downloaded to **C:\ftproot** on the server.

When not in simple directory mode, the root directory "**/**" doesn't map to anything. Instead, the root directory "**/**" becomes a virtual file system to that you can attach sub-directories. When not in simple directory mode, you can add as many virtual directories to a user account as you like and the virtual directory name will become a sub-directory in the virtual root. However, you have to change to that sub-directory before you can upload or download anything. If you try to upload a file to the root folder "**/**" then the operation is invalid because the path "**/**" doesn't map directly to a folder on the server. You would need to specify the path **/ftproot** to upload or download files from the virtual directory **ftproot**.

11.6.2 Variables that can appear in Virtual Directory Names and Paths

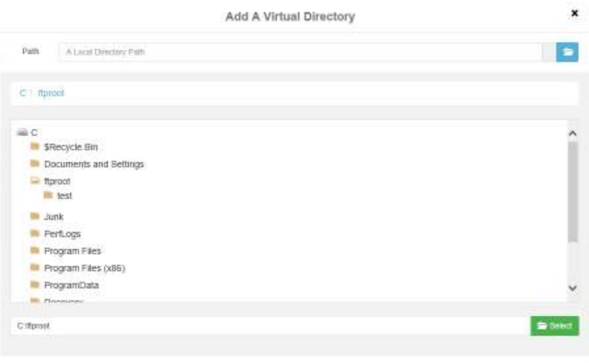
The special variable **%USER%** can be present in a virtual directory name or path. When present, the **%USER%** variable is replaced by the user's username during login.

11.7 Adding a virtual directory to a user account

Each user can be assigned different virtual directories. A virtual directory is added to a user account by using the User Manager. To add a virtual directory to a user, first:

1. Select the user in the "Cerberus Users" list
2. Next, scroll down to see the user details for the selected user. Click on the button labeled "**Virtual Directories**".
3. Click “New” to open the “Add a Virtual Directory” window.
4. Enter the path, or UNC path to the directory, or use the folder select option to navigate to the directory you wish to add. If using the folder select option, select the folder you want, and press the "**Select**" button on the dialog box. The directory you selected should appear in the “Path”

section. Please note Cerberus does NOT support mapped drive letters to access remote resources, you must use UNC path.
5. Enter a name for your Virtual Directory.
6. Next, select the permissions for your Virtual Directory. And then click “Add”



The directory should appear in the "Virtual Root list" list box. To configure the newly added directory, double-click on the directory name in the list box. The Edit a Virtual Directory window will appear. Place a check beside any permission that you would like to grant to the virtual directory and all of that directory subdirectories.

11.8 Virtual Directory Permissions

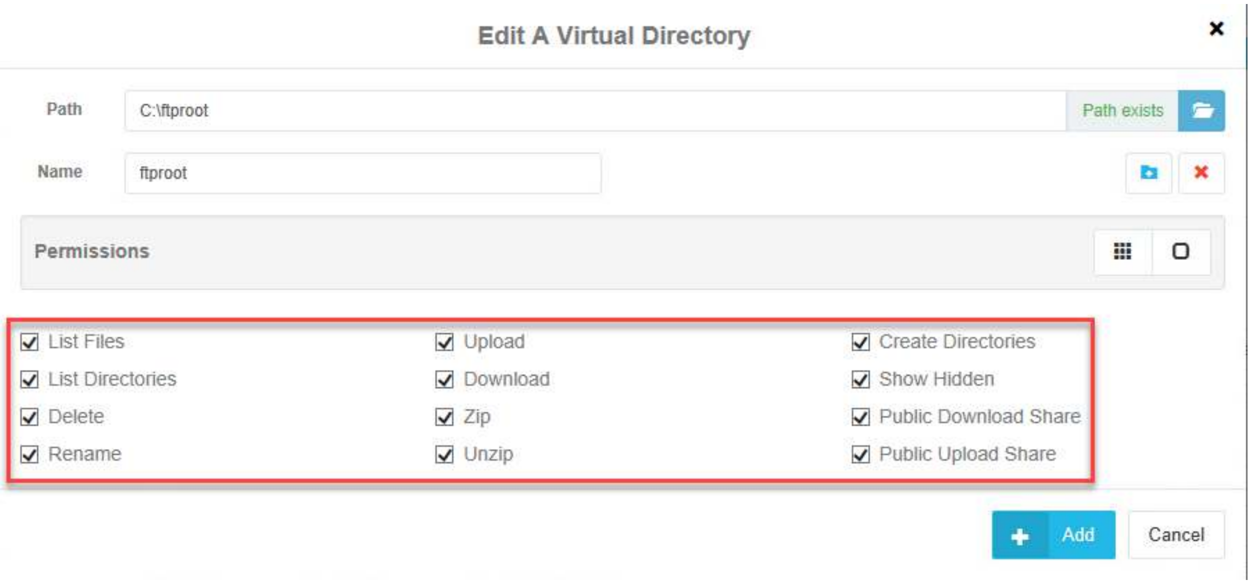
Each virtual directory that you add for a user can have a separate and distinct set of access permissions. The settings applied to a top-level virtual directory filter down to all of that root directory's subdirectories.

Permissions can only be assigned at the top, root level. To edit the permissions for a virtual directory:

1. Select the user in the Users page of the User Manager

2. Scroll down to see the user details for the selected user. Click on the button labeled "**Virtual Directories**".

3. Double click on the virtual directory name in the list box. The ‘Edit a Virtual Directory’ window will appear. Place a check beside any permission that you would like to grant to the virtual directory and all of that directories subdirectories



Edit A Virtual Directory

Path: C:\ftproot (Path exists)

Name: ftproot

Permissions

☑ List Files
☑ List Directories
☑ Delete
☑ Rename
☑ Upload
☑ Download
☑ Zip
☑ Unzip
☑ Create Directories
☑ Show Hidden
☑ Public Download Share
☑ Public Upload Share

Permissions for virtual directories

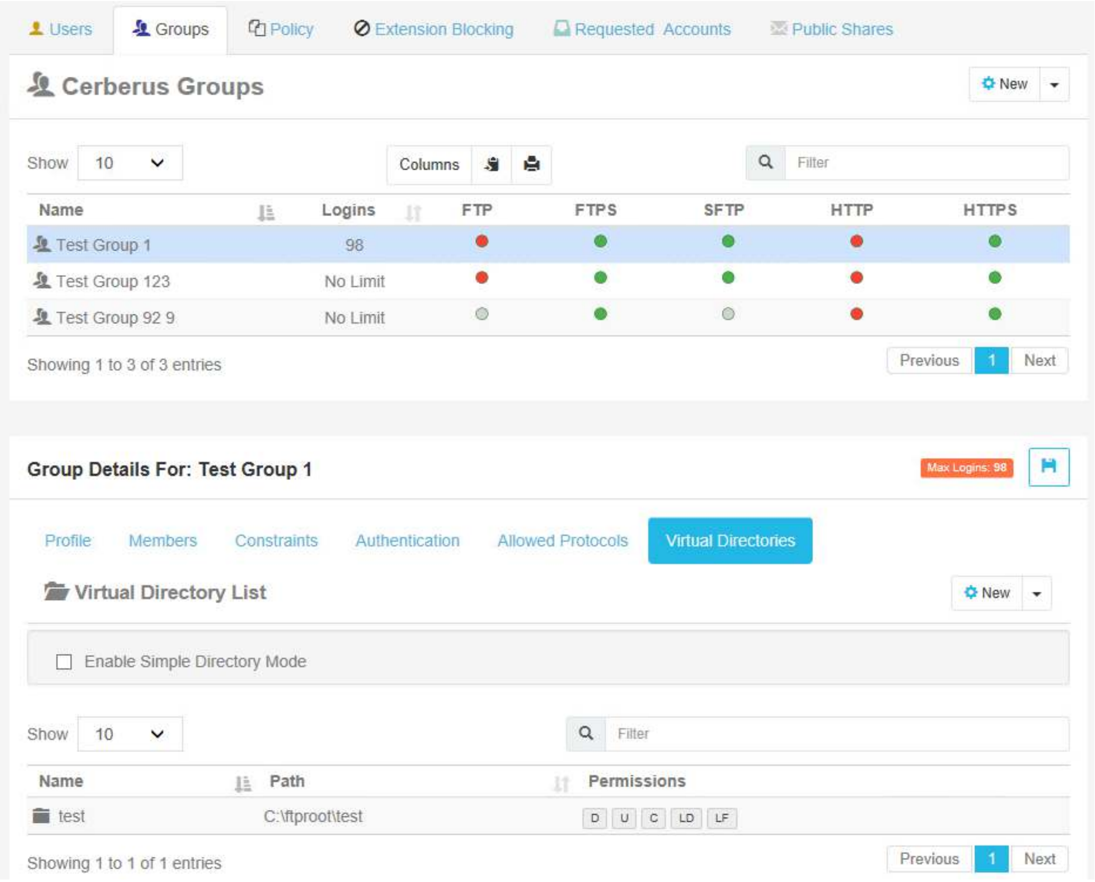
12.0 Cerberus Group Accounts

12.1 About Groups

Using groups simplifies the administration of multiple accounts by letting you assign permissions once to a group, instead of multiple times to each individual user. You can add Virtual Directories and basic user settings to a group and have users inherit those permissions. By default, when a user is assigned a PRIMARY group, that user inherits all of the group's settings. However, those settings can still be overridden for the user account. When a user is assigned a SECONDARY group, that user just the virtual directories assigned to the group. This is useful to add virtual directories to a subset of users all at once.

When a user has been assigned to a primary group, the user's settings on the Users page will be grayed out, and the actual value displayed for each grayed setting is the value of the primary group that the user belongs to.

Virtual directories for the user account are a combination of virtual directories you have specifically assigned to the user account and those assigned to the user by their group memberships.



Cerberus Groups

Name	Logins	FTP	FTPS	SFTP	HTTP	HTTPS
Test Group 1	98	🔴	🟢	🟢	🔴	🟢
Test Group 123	No Limit	🔴	🟢	🟢	🔴	🟢
Test Group 92 9	No Limit	⚪	🟢	⚪	🔴	🟢

Showing 1 to 3 of 3 entries

Group Details For: Test Group 1

Virtual Directory List

☐ Enable Simple Directory Mode

Name	Path	Permissions
test	C:\ftproot\test	D U C LD LF

Showing 1 to 1 of 1 entries

Cerberus FTP Server User Manager- Groups page

12.2 Overriding Group settings for a User

You can override the primary group settings for a user. Click on that user in the User Manager, and then click on the group icon (gray with two heads) to the right of the setting to the user icon. Once you have toggled to the user setting (blue with one head), select the desired setting different from the group value and click ‘Update User’. You can revert back to the group setting by clicking on the user icon and toggling it back to the group icon.

12.3 Adding a new group

A group can be added and modified in Cerberus by opening up the User Manager and selecting the **Groups** tab. To add a group, select the **New** button. A new group will appear under the group list box. All group names must be unique and are case insensitive. Once you have entered the new group name, press "Update Group" to commit the change. The group can then be configured by clicking on the group name in the group list box. A list of configurable properties for that group will appear below the Cerberus Group list.

Those properties are:

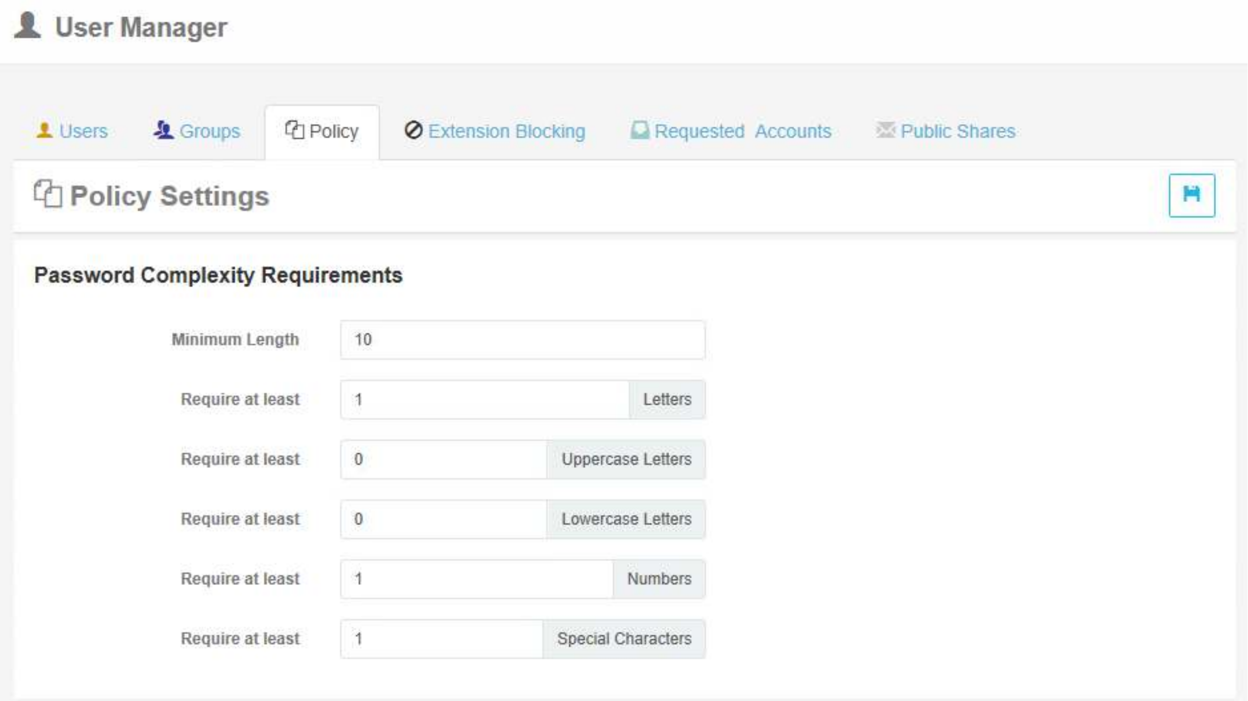
Profile	
Group Name	The unique name for the group
Description	A brief summary or way to identify the group
Members	
Group Member List	This list displays native Cerberus members of the group as well as any LDAP and AD users mapped to the group.
Constraints	
Anonymous	If checked, the password for any user that is part of this group is ignored and the user can be logged in using any password.
Disabled	Determines whether the account can log in or not. A disabled account cannot login to the server.
User Can Change Password	Controls whether a user that belongs to the group can change their password through the HTTP/S web client or through SSH SFTP or FTP commands.
Max Logins	The maximum number of connections this user can make to the server at the same time.
Disable Date	If a date is set here then the group will become disabled after the date specified. All users that are members of this group will also become disabled. **Note**: The granularity of the timer is 30 minutes. The account will be disabled within 30 minutes of the time set.

Maximum Upload File Size	This field can be used to limit the maximum size of an uploaded file. This value defaults to unlimited. The file size is specified in bytes. Specify 0 or any non-positive value to reset the maximum file size to unlimited.
Allowed IP Addresses	A comma-separated list of IP addresses that members of this group can log in from. If no IP addresses are specified then no per-group IP address filtering is enforced. IP addresses can be specified as a single IP, a range of IP addresses separated by a dash, e.g. 192.168.0.100 - 192.168.0.150, or a CIDR-formatted IP address range. Multiple formats can be combined, with each single IP or range separated by a comma. Note, global IP address deny lists or allow ists are always enforced first, regardless of this setting.
Authentication	
SSH Authentication	Determines the SSH authentication requirements for users that are members of this group. Valid options are ● **Password Only**: Require only a password for authentication. ● **Public Key Only**: Require only a valid public key for authentication ● **Public Key and Password**: Require both a valid public key and a valid password for authenticating a user
Allowed Protocols	
Allow FTP	Both FTP and FTPES connections will be allowed to attempt to login over an FTP listener
Require Secure Control	(Applies to FTP only) If enabled, members of this group can only log in to the server using a secure TLS/SSL encrypted connection.
Require Secure Data	(Applies to FTP only) If enabled, members of this group can only initiate file transfers over secure TLS/SSL encrypted connections.
Permitted Login Protocols	Controls which protocols a member of this group is allowed to log in with. If a protocol is not checked then the user will not be allowed to log in using that protocol.
Virtual Directories	
Is Simple Directories	In simple directory mode the administrator can only assign one directory to represent the virtual directory for a user that is a member of this group.

13.0 User Policy Settings

13.1 Password Complexity Requirements

Note: These settings only apply to Cerberus Native accounts.



Password Complexity Requirements

Minimum Length: 10

Require at least: 1 Letters

Require at least: 0 Uppercase Letters

Require at least: 0 Lowercase Letters

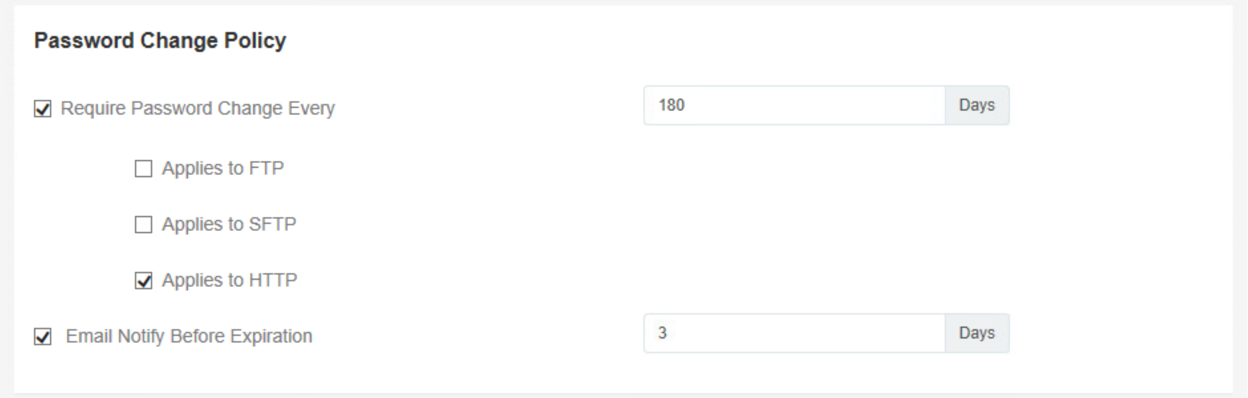
Require at least: 1 Numbers

Require at least: 1 Special Characters

Minimum Length	The password must be at least x characters long.
Require at Least *x* Letters	The password must contain at least x count of letters.
Require at Least *x* Numbers	The password must contain at least x count of numbers.
Require at Least *x* Special Characters	The password must contain at least x count of special characters (e.g.: %, $, #).

13.2 Password Change Policy

These settings only apply to Cerberus Native accounts.



Password Change Policy

☑ Require Password Change Every 180 Days

□ Applies to FTP

□ Applies to SFTP

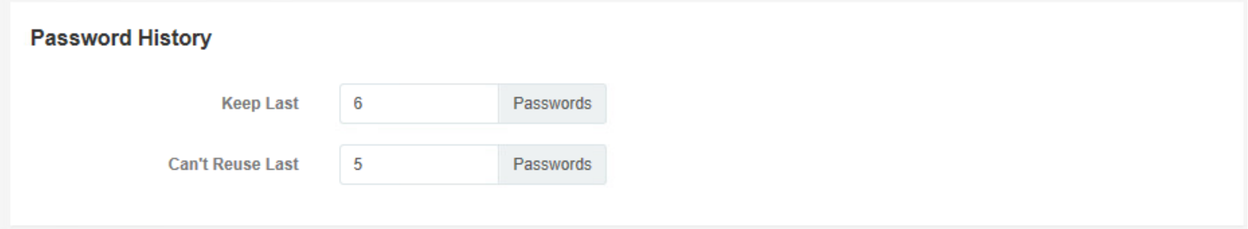
☑ Applies to HTTP

☑ Email Notify Before Expiration 3 Days

Require Password Change Every *x* Days	The server will require that native account passwords be changed after this number of days. Not all protocols have standard support for password changing, and not all clients implement that support when it does exist. To overcome this limitation, you can disable password expiration checking for specific protocols. Note, marking a user account password as requiring a change on the next login requires the password change option to be checked. **Applies to FTP** - When checked, this policy is enforced for FTP/S account access. Note, that FTP does not have a standard way of changing or prompting the user to change an account password. Cerberus supports a common extension that allows changing the user password using the SITE PSWD oldpassword newpassword command. However, using that command requires the user to be logged in. The protocol does not have a mechanism for informing the user of an expired password during login. As a result, there is no way to change an expired password via FTP once it has expired. The user will be unable to log in via FTP. **Applies to SSH SFTP** - When checked, this policy is enforced for SSH SFTP account access. SSH has a standard method of allowing users to change their passwords, but many SFTP clients do not implement it. **Applies to HTTP** - When checked, this policy is enforced for HTTP/S account access. Cerberus handles the logic of making sure the user is properly prompted for changing an expired password during login, so this method is supported by all web browsers.

13.3 Password History

These settings only apply to Cerberus Native accounts.



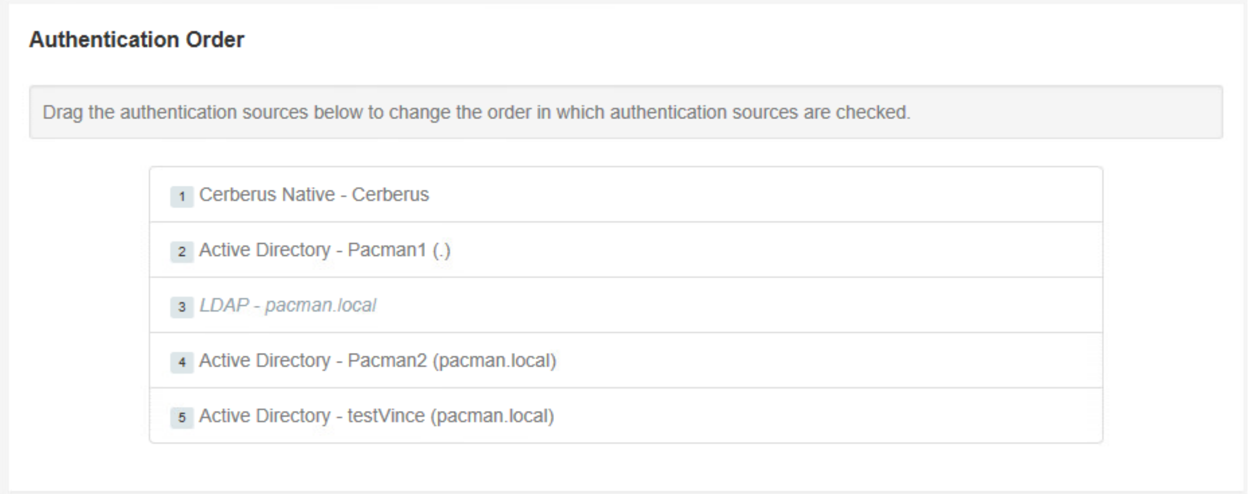
Password History

Keep Last: 6 Passwords

Can't Reuse Last: 5 Passwords

Remember Last *x* Passwords	Cerberus will save a secure hash of the last specified number of passwords that the user has used.
Can't Reuse Last *x* Passwords	Cerberus will prevent a user from changing their password to any password used within the specified history count.

13.4 Authentication Order



Authentication Order

Drag the authentication sources below to change the order in which authentication sources are checked.

1 Cerberus Native - Cerberus

2 Active Directory - Pacman1 (.)

3 *LDAP - pacman.local*

4 Active Directory - Pacman2 (pacman.local)

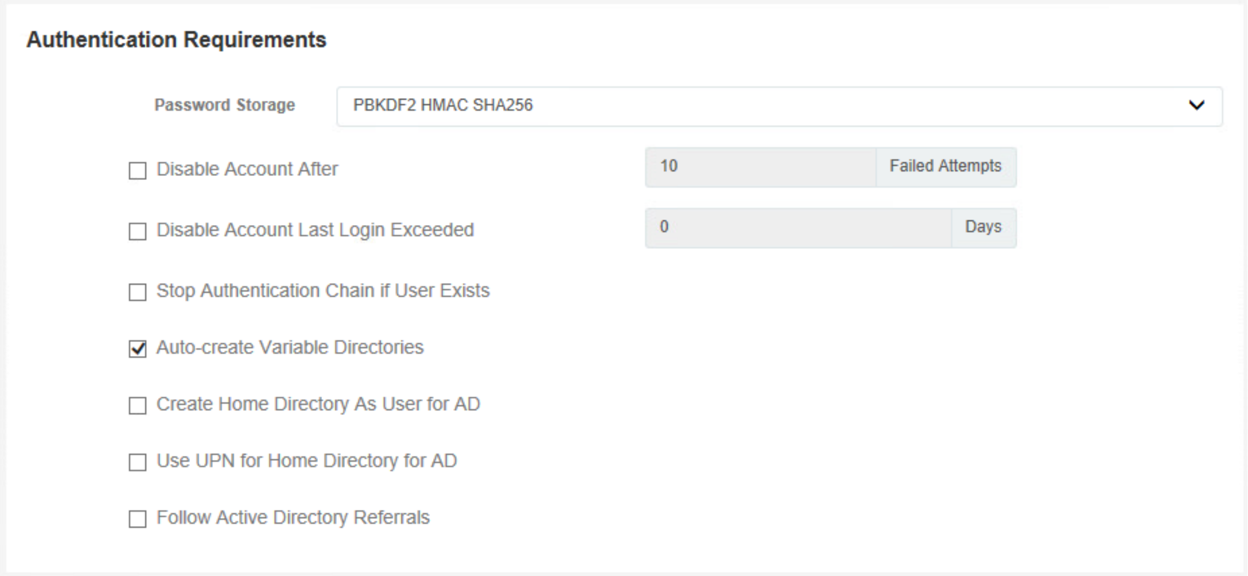
5 Active Directory - testVince (pacman.local)

Cerberus FTP Server can authenticate against several different types of data sources. The current possible authentication sources include the **Native user system**, **Active Directory (AD)**, and **LDAP**. You can have multiple AD and LDAP servers configured and Cerberus will check each one and attempt to match a username and password. Cerberus will try each authentication source in order until a successful authentication occurs or until all sources fail authentication.

The order that authentication sources are checked is determined by the Authentication Order list box. You can move authentication sources up and down in order depending upon your needs.

13.5 Authentication Requirements

The Disable Account and Password Storage Format options only apply to Cerberus Native accounts.



Authentication Requirements

Password Storage: PBKDF2 HMAC SHA256

□ Disable Account After 10 Failed Attempts

□ Disable Account Last Login Exceeded 0 Days

□ Stop Authentication Chain if User Exists

☑ Auto-create Variable Directories

□ Create Home Directory As User for AD

□ Use UPN for Home Directory for AD

□ Follow Active Directory Referrals

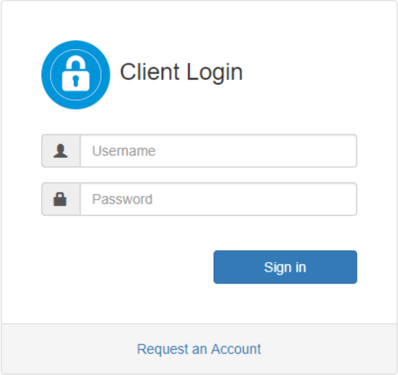
Disable Account After *x* Failed Attempts	The Native account becomes disabled after *x* number of consecutive failed login attempts. The counter is reset on a successful login.
Password Storage Format	This is the method Cerberus uses to store user account password information. Options are SHA1, SHA256, and SHA512. All options are salted and are performed using FIPS compliant crypto routines if the server is in FIPS mode.
Disable Account Last Login Exceeded	Native accounts become disabled if they exceed x number of days without successful login.
Stop Authentication Chain if User Exists	If a user is found in an authentication source but the password is incorrect, don't proceed to check the other authentication sources. Just fail the authentication request.
Auto-Create Variable Directories	The variable **%USER%** can be used in virtual directory names and paths. This variable is evaluated to the account's name when the user logs in. Selecting this option ensures that virtual directory paths with the **%USER%** variable in them will be automatically created when the user account is evaluated during login.
Create Home Directory As User For AD	This setting influences how home directories are created for Active Directory users when the default virtual directory mapping mode in AD is set to **Global Home/%USER%** mode. Normally, Cerberus creates the home directory while under the service account. If this option is enabled, Cerberus will impersonate the AD user before creating the directory. This ensures the

	home directory is owned by the AD user instead of the service account.
Use UPN for Home Directory for AD	This setting influences how home directories are created for Active Directory users when the default virtual directory mapping mode in AD is set to Global Home/%USER% mode. If this option is checked, Cerberus will always use the AD user's UPN name as the home directory name, instead of the user's login name. AD users can usually use either their SAMAccount or their UPN name. Checking this option will ensure the user is always placed in the same home directory, regardless of whether they log in with their SAMAccount or UPN name.
Follow Active Directory Referrals	When querying a domain controller, a referral is a way that a directory server communicates that it does not contain the data required to complete a query, but has a reference to a server that may contain the required data. If this option is selected, Cerberus will query other domain controllers to get a complete set of results.

14.0 Web Account Requests

14.1 Allowing Users to Request Accounts through the Web

Users can request new accounts through the HTTP/S Web Client. A "Request a New Account" link will appear on the login page if the administrator decides to allow web account requests.

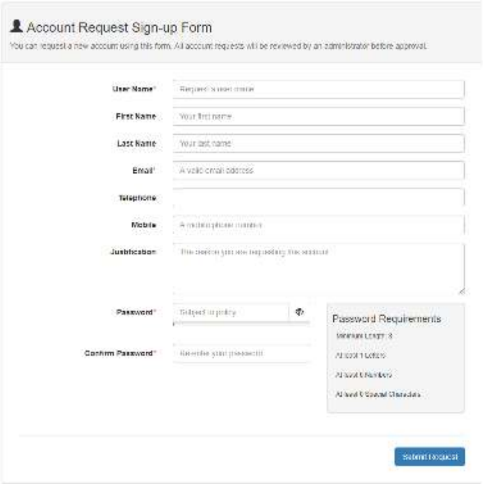


HTTP/S Login Page with "Request a New Account" Link

14.1.1 Requesting a New Account

The account request page allows a user to submit a request for a new account to the Cerberus FTP Server system administrator. The user can set a password for the account (subject to password policy rules) at the time of the account request. This relieves the administrator from having to set a new password for the user and from having to securely distribute that password.

Event Rules can be enabled on the server to automatically email the administrator whenever a new account request is made.



Account Request Sign-up Form

You can request a new account using this form. All account requests will be reviewed by an administrator before approval.

Password Requirements

Minimum Length: 8

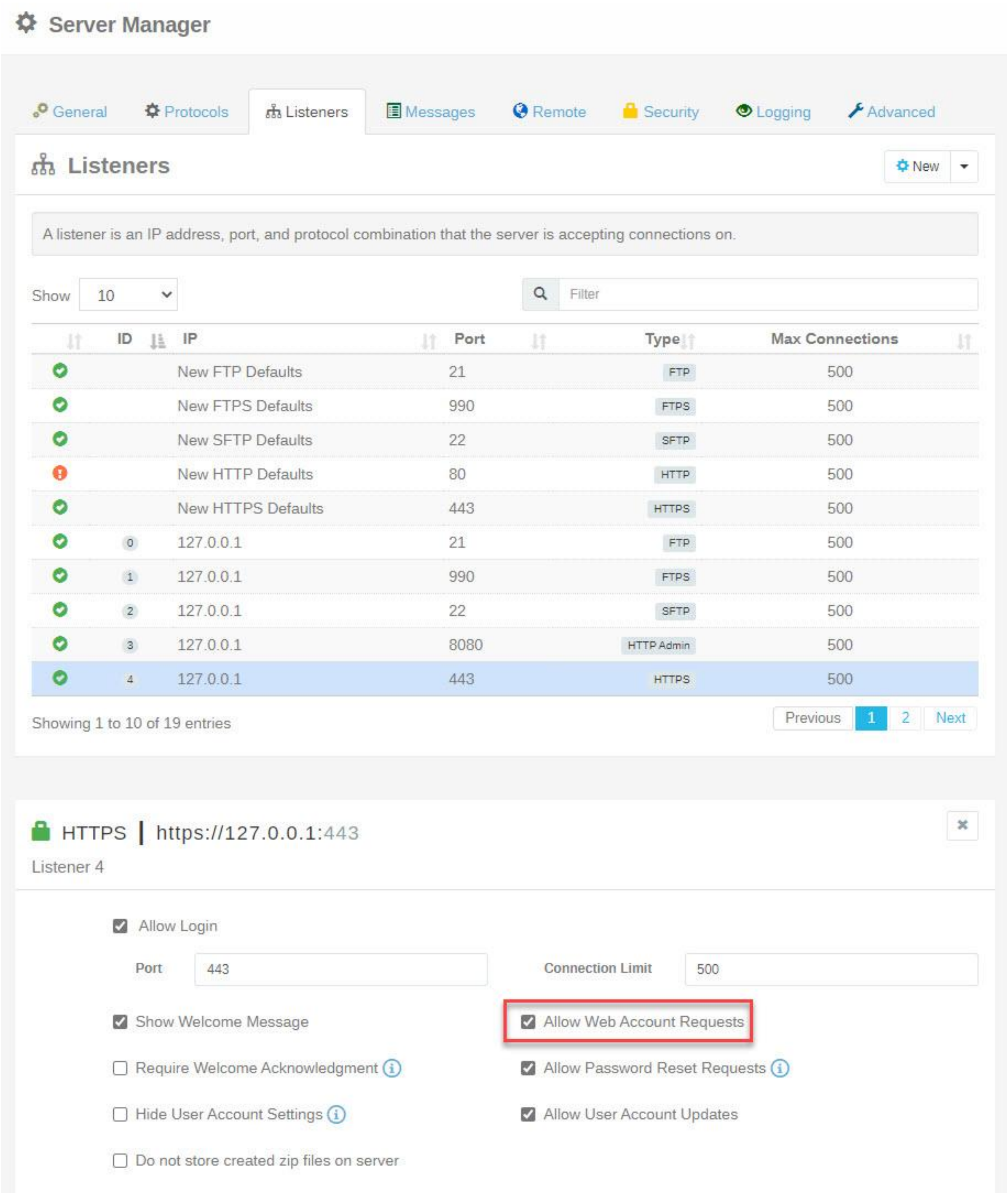
At least 1 Letters

At least 0 Numbers

At least 0 Special Characters

14.1.2 Enabling or Disabling Account Requests

The link can be enabled or disabled for any HTTP or HTTPS listener by selecting that listener in the Listeners page of the Server Manager.



Listeners

A listener is an IP address, port, and protocol combination that the server is accepting connections on.

Show 10

Filter

	ID	IP	Port	Type	Max Connections
✓		New FTP Defaults	21	FTP	500
✓		New FTPS Defaults	990	FTPS	500
✓		New SFTP Defaults	22	SFTP	500
!		New HTTP Defaults	80	HTTP	500
✓		New HTTPS Defaults	443	HTTPS	500
✓	0	127.0.0.1	21	FTP	500
✓	1	127.0.0.1	990	FTPS	500
✓	2	127.0.0.1	22	SFTP	500
✓	3	127.0.0.1	8080	HTTP Admin	500
✓	4	127.0.0.1	443	HTTPS	500

Showing 1 to 10 of 19 entries

HTTPS | https://127.0.0.1:443

Listener 4

☑ Allow Login

Port 443

Connection Limit 500

☑ Show Welcome Message

☑ Allow Web Account Requests

☐ Require Welcome Acknowledgment ⓘ

☑ Allow Password Reset Requests ⓘ

☐ Hide User Account Settings ⓘ

☑ Allow User Account Updates

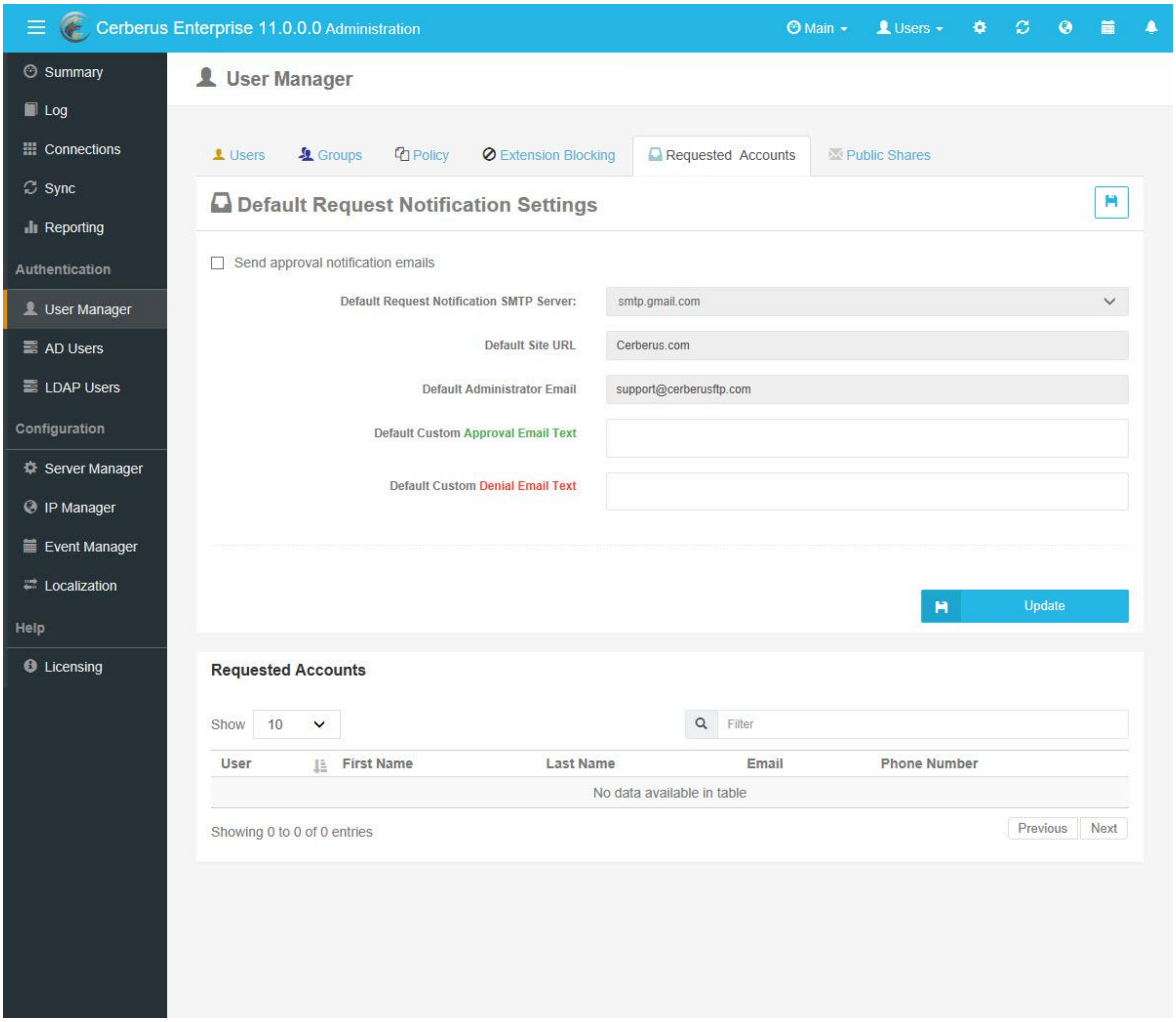
☐ Do not store created zip files on server

Enable Account Requests

14.2 Approving or Denying Account Requests

Administrators can view pending account requests through both the **Account Requests** page of the **User Manager** in the Cerberus GUI, or through the Account Requests administrator web page. Accounts can be approved or denied through either method by selecting an account and using the **Approve** or **Delete** button.

Approved accounts are automatically created and activated on the **Users** page of the **User Manager** and can be further customized there.



Default Request Notification Settings

□ Send approval notification emails

Default Request Notification SMTP Server: smtp.gmail.com

Default Site URL: Cerberus.com

Default Administrator Email: support@cerberusftp.com

Default Custom Approval Email Text:

Default Custom Denial Email Text:

Requested Accounts

Show 10

Filter

User	First Name	Last Name	Email	Phone Number
No data available in table				

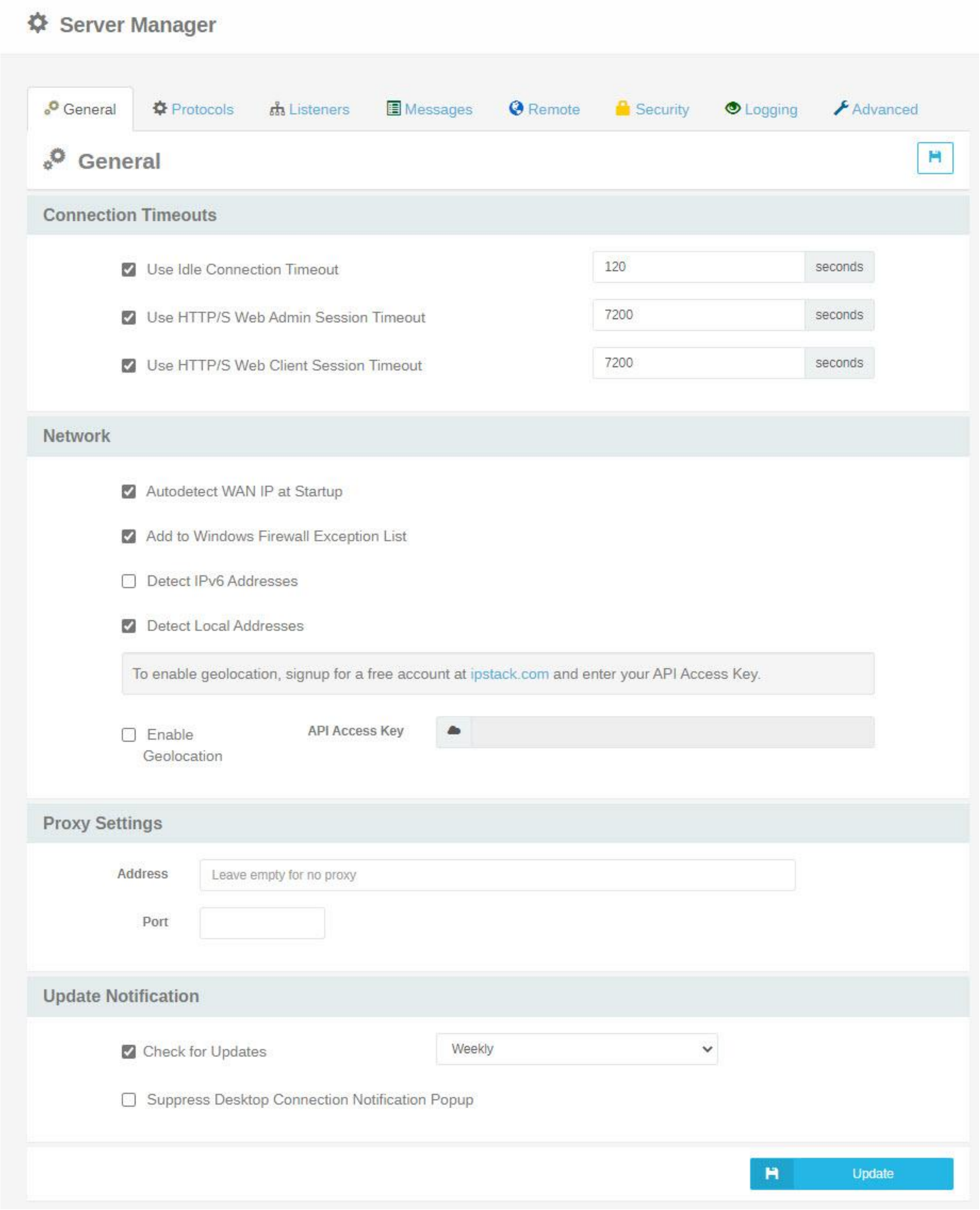
Showing 0 to 0 of 0 entries

The Cerberus FTP Server Account Request Page of the User Manager

15.0 General Settings

15.1 Configuring General Settings

The general settings page contains options for connection timeout, network detection, login notifications, and auto-update settings.



Connection Timeouts

☑ Use Idle Connection Timeout 120 seconds

☑ Use HTTP/S Web Admin Session Timeout 7200 seconds

☑ Use HTTP/S Web Client Session Timeout 7200 seconds

Network

☑ Autodetect WAN IP at Startup

☑ Add to Windows Firewall Exception List

□ Detect IPv6 Addresses

☑ Detect Local Addresses

To enable geolocation, signup for a free account at ipstack.com and enter your API Access Key.

□ Enable Geolocation API Access Key

Proxy Settings

Address: Leave empty for no proxy

Port:

Update Notification

☑ Check for Updates Weekly

□ Suppress Desktop Connection Notification Popup

15.2 General

The general settings page contains options for connection timeout and hiding the main Cerberus window.

Use idle connection timeout	Controls whether idle connections should be terminated after a period of inactivity. The **Idle Connection Timeout (seconds)** value controls how long a connection to the server can remain idle without being terminated.
Use HTTP/S web admin session timeout (in seconds)	Controls how long (in seconds) a web admin session can remain idle before the session becomes invalid and the user has to login again.
Use HTTP/S web client session timeout (in seconds)	Controls how long (in seconds) a web client session can remain idle before the session becomes invalid and the user has to login again.

15.3 Network

Controls general network settings.

Detect WAN IP at Startup	If enabled, Cerberus will attempt to detect the external address that Internet computers see for connecting to the network this machine is located on. This is usually the external router address. Enabling this option is important for ensuring passive connections work correctly.
Add to Windows Firewall Exception List	If selected, Cerberus FTP Server will attempt to add itself to the Windows Firewall Exception list. This setting is disabled on operating systems that do not support the Windows Firewall (Windows 2000 and below).
Detect IPv6 Addresses	If selected, Cerberus FTP Server will attempt to detect any IPv6 addresses that the system has initialized. You can leave this setting disabled if you are not using IPv6.
Detect Local Addresses	If selected, the server will bind to the IPv4 loopback addresses 127.0.0.1, and (if IPv6 is enabled) the ::1 loopback address.

15.4 Proxy Settings

Controls user login notification settings.

Address	Proxy address

Port	Proxy Port

15.4 Notification

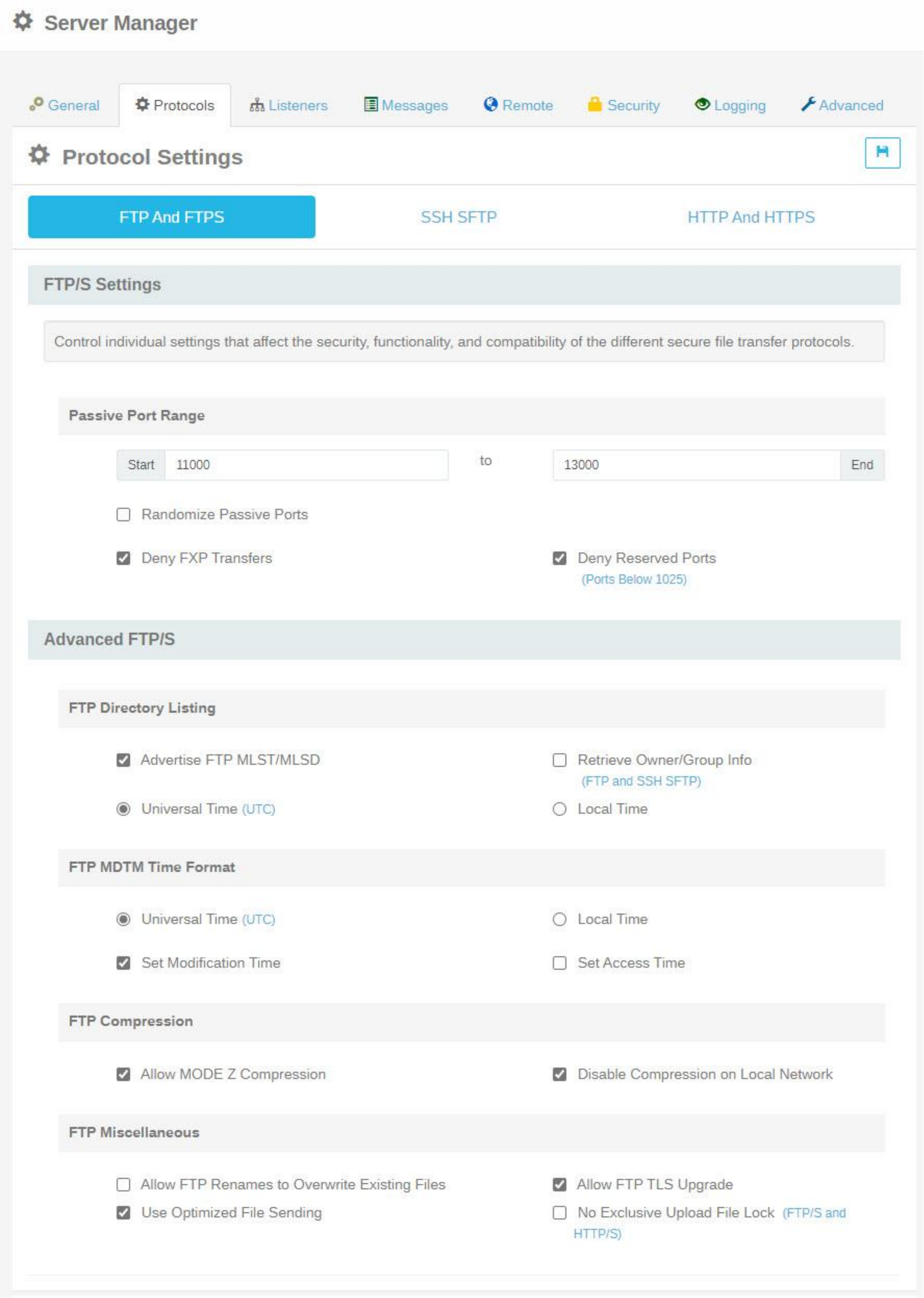
Controls user login notification settings.

Check for Updates	Controls how often the server will check for updates. Possible values are: Never, Daily, Weekly, or Monthly.
Suppress desktop notification popup	If enabled, Cerberus will not display a small notification window on the bottom-right corner of the desktop whenever a user attempts to log in to the server.

16.0 Protocol Settings

The Protocols page allows you to control individual settings that affect the security, functionality, and compatibility of the different secure file transfer protocols.

16.1 FTP/S Settings



FTP/S Settings

Control individual settings that affect the security, functionality, and compatibility of the different secure file transfer protocols.

Passive Port Range

Start 11000 to 13000 End

□ Randomize Passive Ports

☑ Deny FXP Transfers

☑ Deny Reserved Ports (Ports Below 1025)

Advanced FTP/S

FTP Directory Listing

☑ Advertise FTP MLST/MLSD

□ Retrieve Owner/Group Info (FTP and SSH SFTP)

● Universal Time (UTC)

○ Local Time

FTP MDTM Time Format

● Universal Time (UTC)

○ Local Time

☑ Set Modification Time

□ Set Access Time

FTP Compression

☑ Allow MODE Z Compression

☑ Disable Compression on Local Network

FTP Miscellaneous

□ Allow FTP Renames to Overwrite Existing Files

☑ Allow FTP TLS Upgrade

☑ Use Optimized File Sending

□ No Exclusive Upload File Lock (FTP/S and HTTP/S)

16.1.1 Passive Port Range

These settings control passive FTP options.

Start	First port in the port range to use for passive connections.
End	Last port to use for passive connections before wrapping back around to the Start port.
Randomize Passive Ports	A security option that when enabled causes the server to choose a cryptographically random, unused passive port from the passive port range. When this option is disabled the server selects a passive port from the passive port range incrementally.
Deny FXP Transfers	File eXchange Protocol (FXP) is a method of data transfer that uses the FTP protocol to transfer data from one remote server to another (inter-server) without routing this data through the client's connection. Conventional FTP involves a single server and a single client; all data transmission is done between these two. In the FXP session, a client maintains a standard FTP connection to two servers and can direct either server to connect to the other to initiate a data transfer.
Deny Reserved Ports	Do not allow passive or active port requests below port 1024.

16.1.2 Advanced FTP/S Settings

16.1.2.1 FTP Directory Listing Time Format

This setting determines the time zone format for the file list returned in response to the LIST and NLST commands. Most clients expect dates and times to be in UTC format.

Universal Time (UTC)	The default, send the file date/time in UTC format.
Local Time	Send file date/time in local time.
Advertise FTP MLST/MLSD	Allow the FTP server to advertise to clients that it supports the MLST/MLSD command (recommended).
Retrieve Owner/Group information for file listings	Includes the owner and group of each file in responses to the LIST and NLST command. NOTE: This will slow down file listings.

16.1.2.2 FTP MDTM Time Format

The FTP command, *MODIFICATION TIME (MDTM)*, can be used to determine when a file in the server file system was last modified. This command has existed in many FTP servers for many years, as an adjunct to the REST command for STREAM mode. As a result, this command is widely available.

This command is also frequently used in a non-standard fashion to set file modification times. Cerberus supports both the standard MDTM command for retrieving file times and the non-standard use for setting the date/time on a file.

NOTE: Setting dates and times requires FTP client support. There is often a setting that has to be enabled in many FTP clients before an uploaded or downloaded file will have its date/time set. Consult your FTP client documentation on how to enable this setting. Cerberus automatically supports setting a file date/time without any additional configuration.

Universal Time (UTC)	Most FTP clients expect the *MDTM* command to process date/time values in UTC format and this is the default. Selecting this option will cause Cerberus to interpret and send dates in UTC format.
Local Time	Interpret and send dates in local time (not RFC compliant).
Set Modification Time	When clients attempt to use the non-standard MDTM extension to set a date/time for a file, this setting determines whether the file modification time will be set.
Set Access Time	When clients attempt to use the non-standard MDTM extension to set a date/time for a file, this setting determines whether the file access time will be set.

16.1.2.3 FTP Compression

Cerberus FTP Server 5.0 and higher support MODE Z compression for FTP directory listings, uploads, and downloads.

Allow MODE Z Compression	The default, send file date/time in UTC format.
Disable Compression on Local Network	The benefits of compression on the local network can often be outweighed by the time it takes to compress that data. It is recommended that compression be disabled for local network connections. (recommended)

16.1.2.4 FTP Miscellaneous

These are FTP settings that don't fit anywhere else.

Allow FTP Renames to Overwrite Existing Files	When this option is enabled an FTP client can issue a rename command and overwrite an existing file.
Use Optimized File Sending	Uses the built-in Windows API for potentially faster file sending on Windows Server machines. This option only applies to plain FTP transfers. It provides no benefit for encrypted file transfers.
Allow FTP TLS Upgrade	The FTP server will advertise and allow clients to upgrade plain FTP connections to encrypted FTP connections (FTPES) when this option is enabled (recommended).
No Exclusive Upload File Lock	If this option is checked, the server will open files for upload in non-exclusive mode during file transfer. This allows other processes to open the same file for read-only access and be able to read from the file as it is being uploaded.

16.2 SSH SFTP Settings

16.2.1 SSH SFTP Settings

Ignore SSH Window Size	Some SFTP clients do not correctly request an increase in the SSH channel window size. Enabling this option will allow those connections to continue even after exceeding the available channel window space.
Require Encryption on SFTP	Although most clients won't request an unencrypted connection, the SSH protocol does allow it. Check this option to disallow unencrypted SSH connections. This option should always be enabled for production servers.
Use Legacy Handles for SFTP	If you are connecting to Cerberus using a very old FTP client that only supports legacy algorithms, and Cerberus is refusing to connect, this is an option to try.SSSS This option switches Cerberus to use the legacy SSH library.
Mask Server Identification	If this option is checked, the server will use a generic identification string for the welcome message during SSH connections. The server will also omit the server header for HTTP/S connections.
No Exclusive Upload File Lock	If this option is checked, the server will open files for upload in non-exclusive mode during file transfer. This allows other processes to open the same file for read-only access and be able to read from the file as it is being uploaded. Applies to SFTP Version 4 and below clients only.

16.2.2 SSH Security Settings

Active Key Exchange	The SSH key exchange algorithms that the server will advertise as supported to SSH clients.
Active SSH SFTP ciphers	The cipher algorithms advertised by Cerberus to clients during secure connection negotiation for SSH2 SFTP. You can select the algorithms you want advertised using this list.
Active MAC	The HMAC algorithms advertised by Cerberus to clients during secure connection negotiation for SSH2 SFTP. You can select the algorithms you want advertised using this list.

16.3 HTTP/S Settings

These are advanced settings for controlling HTTP/S web client defaults for all users.

16.1.1 HTTP/S Settings

Public Domain Name	This option is used for sending out Account Request email notifications and password reset emails.

16.3.2 HTTP/S Client Header Security

Client Domain Allow List	To prevent host header attacks when sharing public file links or client-initiated password reset requests, you can add a list of allowed public domain names for your server.

16.3.3 HTTP/S Temporary Files Settings

Temp Upload Directory	HTTP/S web client uploads are stored in this directory as they are uploaded. When the upload completes, the file is moved to its final destination. If this edit box is left blank, the temporary upload directory defaults to the temporary files directory for the account running the Cerberus FTP Server Windows Service for native accounts and LDAP accounts. For AD accounts, it defaults to the temporary folder for that AD user on the server machine. This field can be used to override the defaults for all account types.

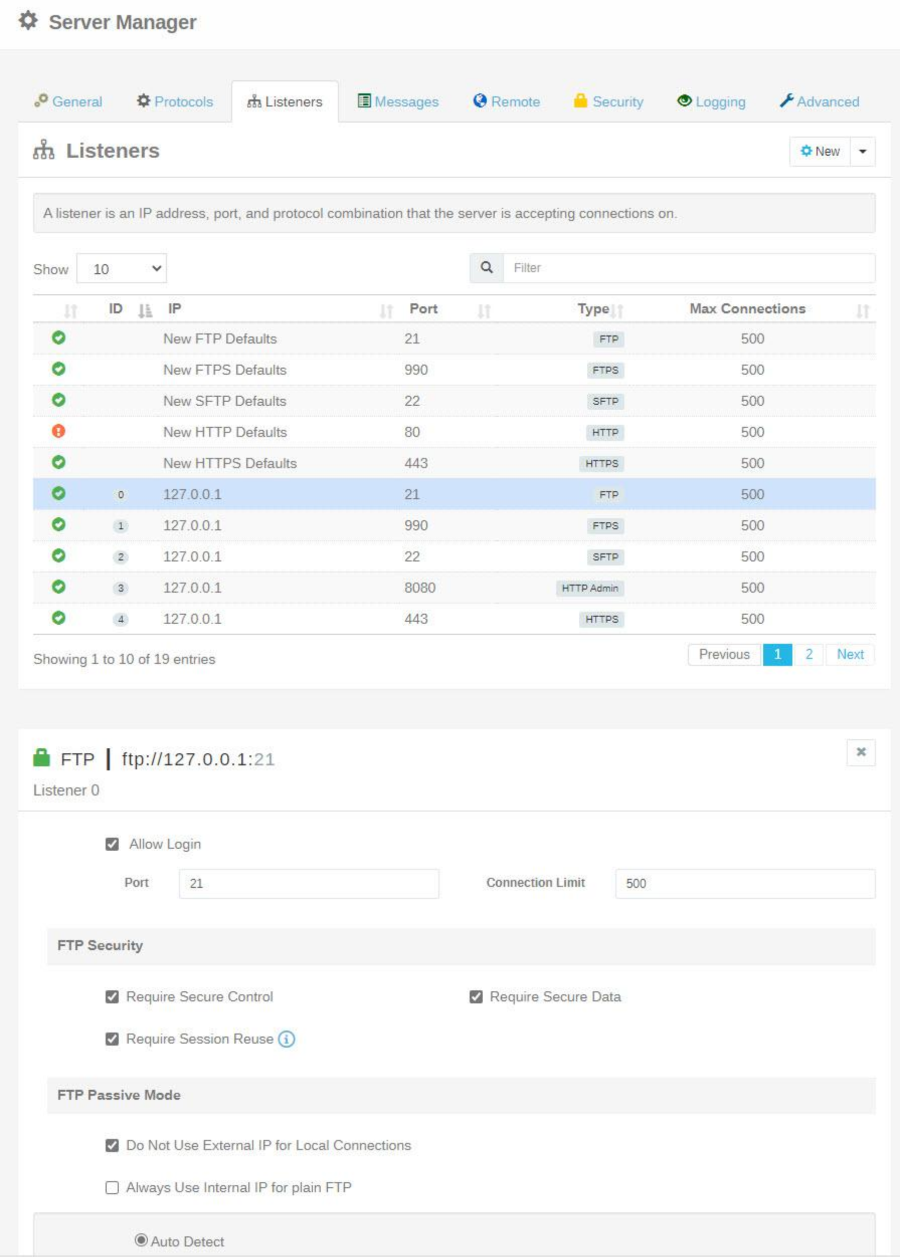
16.3.4 Advanced HTTP/S

Optional Headers to Include	Allows the administrator to determine if the listed HTTP headers should be sent to clients for HTTP/S web client connections.
Zip Options	Select Zip Compression Level if desired

17.0 Listener Settings

17.1 Configuring Listener Settings

A listener is simply an IP address, port, and protocol combination that the server is accepting connections on. For example, you can add an FTP listener on port 21 and attach it to an IP address. It can be an IPv4 or IPv6 address. The "Default" interfaces represent the settings that will be applied for newly detected interfaces. There are several different parameters that each interface can have:



Listeners

A listener is an IP address, port, and protocol combination that the server is accepting connections on.

	ID	IP	Port	Type	Max Connections
✓		New FTP Defaults	21	FTP	500
✓		New FTPS Defaults	990	FTPS	500
✓		New SFTP Defaults	22	SFTP	500
!		New HTTP Defaults	80	HTTP	500
✓		New HTTPS Defaults	443	HTTPS	500
✓	0	127.0.0.1	21	FTP	500
✓	1	127.0.0.1	990	FTPS	500
✓	2	127.0.0.1	22	SFTP	500
✓	3	127.0.0.1	8080	HTTP Admin	500
✓	4	127.0.0.1	443	HTTPS	500

Showing 1 to 10 of 19 entries

FTP | ftp://127.0.0.1:21

Listener 0

☑ Allow Login

Port: 21 Connection Limit: 500

FTP Security

☑ Require Secure Control ☑ Require Secure Data

☑ Require Session Reuse

FTP Passive Mode

☑ Do Not Use External IP for Local Connections

☐ Always Use Internal IP for plain FTP

● Auto Detect

17.2 Types of Listeners

There are seven types of listeners that you can add to an IP address:

FTP	Traditional FTP, the default port 21
FTPS	Implicit FTP with TLS/SSL encryption, default port 990

SSH SFTP	SSH2 File Transfer Protocol, default port 22
HTTP	HTTP, default port 80
HTTPS	HTTP with TLS/SSL encryption, default port 443
HTTP Admin	HTTP for web administration, default port 8080
HTTPS Admin	HTTP with TLS/SSL encryption for web administration, default port 8443

The first two allow regular FTP as well as different forms of secure FTP while the SSH2 SFTP listener is for establishing connections over the SFTP protocol (a completely different protocol from FTP, despite the similar name). The HTTP and HTTPS listeners allow web client connections to the server using either the unsecure HTTP protocol or encrypted HTTPS protocol.

There are two types of secure FTP connections possible, FTPS and FTPES. FTPS is usually referred to as implicit FTP with TLS/SSL security. Its closest analog is HTTPS. It is basically the FTP protocol over a TLS/SSL secured connection. This form of secure FTP is deprecated but widely supported and still in use. This is what Cerberus FTP Server **FTPS listener** is for and this type of listener typically listens on port 990. Note, that the settings "Require Secure Control" and "Require Secure Data" are meaningless for this type of listener. Connections established to an FTPS listener can only be established securely.

FTPES, which is often referred to as **explicit FTP** with TLS/SSL security, is a modification of the FTP protocol that starts out over an insecure, normal FTP connection and is then upgraded to a secure connection through FTP command extensions during login. This is the preferred method of secure FTP because it allows SPI firewalls to know that there is FTP traffic occurring on the connection. You establish FTPES sessions using a normal Cerberus FTP Server **FTP listener**, typically over port 21. Both unencrypted FTP and explicit TLS/SSL connections can be established to this type of listener. You cannot establish an implicit FTPS connection over this type of listener.

17.3 Adding a New Interface Listener

Cerberus FTP Server supports adding multiple listening interfaces for a given IP address. This allows you to have Cerberus accepting connections from different protocols on multiple ports. The only requirement is that each listener is on a unique IP/port combination. You can add FTP, FTPS (for implicit secure FTP only), SSH2 SFTP, HTTP, or HTTPS listeners.

Press the "New" button in the interface list box to add a new interface. A new dialog box will appear to ask for the interface details (interface IP, type, and port combination). Selecting a listener from the list and right-clicking will give you a menu where you can delete the selected interface listener.

17.4 Interface Settings

Listen Port	This setting is the port that this interface will listen on for the control connection.
Max Connections	The setting determines the maximum number of simultaneous connections that can connect to this interface listener.
Require Secure Control	(Applies to FTP only) If enabled, only secure control connection will be allowed. This is required to protect passwords from compromise on unsecured networks with FTP.
Require Secure Data	(Applies to FTP only) If enabled, only secure data connections will be allowed. All directory listings and file transfers will be required to be encrypted.
Don't Use External IP for Passive connections	If this option is checked, Cerberus will always use the internal IP address when the incoming connection originates on the local network.
Always Use Internal IP for plain FTP	If this option is checked, Cerberus will always use the internal IP address when the incoming connection is plain FTP to ensure insecure FTP connections remain inside your network.
Passive IP Options	● **Auto Detect** - If WAN IP auto detection is enabled then use the WAN IP for the PASV command, otherwise use the interface's IP. ● **Specify PASV IP** - Allows the administrator to specify what IP address is returned in response to a PASV command ● **Use DNS service** - Allows use of DNS names like www.cerberusftp.com. The address specified will be examined at regular intervals and the IP address that represents that DNS name will be used in PASV commands.
Show Welcome Message	If checked, the server will send a welcome message during user login for FTP/S, SSH SFTP, and the HTTP/S web client (note, some FTP and SFTP clients won't display the welcome message).
Allow User Updates	(Applies to HTTP/S only) If checked, the user will be allowed to update his or her personal account information (first name, last name, email, or telephone number) through the HTTP/S web client.
Allow Web Account Requests	(Applies to HTTP/S only) If checked, users can request new accounts through the HTTP/s web client.
Company Name	(Applies to HTTP/S only) The company name to display in the web client page title
Logo Image	(Applies to HTTP/S only) The logo image to display in the web client header. This file's dimensions should be 230 by 70.
Login Image	(Applies to HTTP/S only) The image to display on the web client login page. This file's dimensions should be 70 by 70.
Default Web Directory List Count	(Applies to HTTP/S only) The default number of entries that appear in the web client file list.
Show Time zone on Dates	(Applies to HTTP/S only) Toggles displaying time zone information for files and directories in the web client
Display Local Time	(Applies to HTTP/S only) Toggles between displaying server local time or UTC time for files and directories in the web client

Configure CAPTCHA	(Applies to HTTP/S only) Configures Google reCaptcha for the web client login and web requests pages.
Redirect requests to HTTP/S listener	(Applies to HTTP only) Any requests that come in over this HTTP listener will be redirected to the same address using HTTPS.

17.5 The "Default" Interfaces

There is a Default interface for each type of listener (FTP, implicit FTPS, SFTP, HTTP, and HTTPS). When a new interface (IP address) is detected, that interface will receive an FTP, FTPS, and SFTP listener and each of those listeners will be assigned the values of the appropriate "Default" interface at the time of detection. For example, If the "Default FTP" interface was defined to be on port 21, then when a new interface is detected for the first time it will receive an FTP listener on port 21 with the values of the Default FTP interface. Those settings then become the settings for the newly detected interface. Note that the new interface's settings are not linked to the "Default" interface in any way. The "Default" interface simply represents the values that newly detected interfaces will be initialized with. Changing the values of the "Default" interface wouldn't change any values on existing or previously detected interfaces.

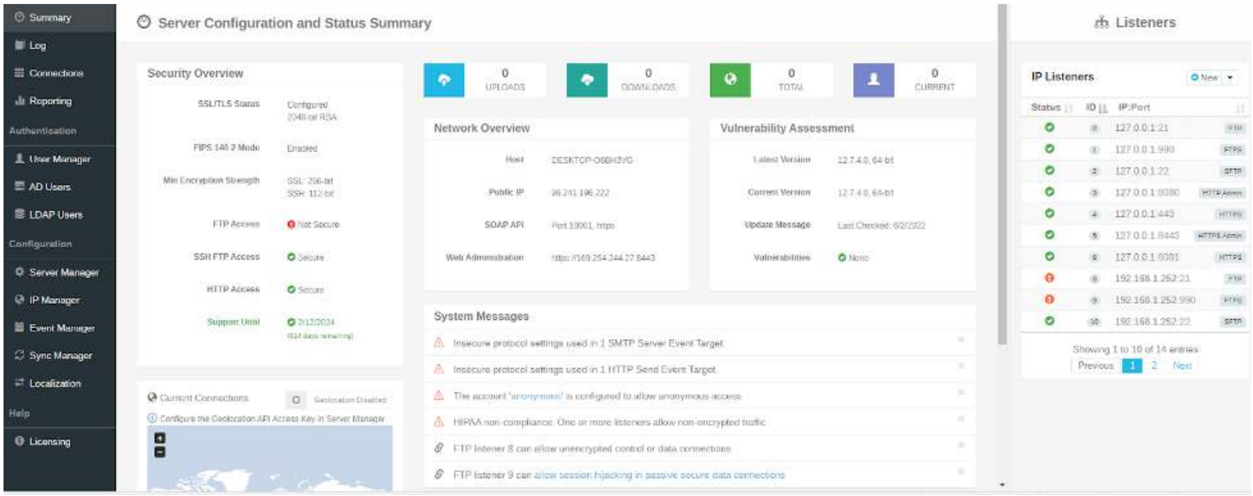
For example, when you first install Cerberus FTP Server, the "Default FTP" interface is set to port 21 (the default FTP listening port) and all interfaces detected during that first start will receive FTP listeners with that port value. If you later change the "Default FTP" interface settings then that change will have no effect on existing interfaces.

It is also worth noting that Cerberus remembers the settings for interfaces that were previously detected but might have changed. For servers that have dynamic addresses that constantly change or cycle between a range of addresses, Cerberus will "remember" the old values and apply those instead of the "Default" settings if that interface address is later detected again.

Un-checking the box next to each Default interface will disable automatic listener activation for that interface type when a new interface is detected.

17.6 Interface Status Controls

Interfaces can also be enabled or disabled from the main Cerberus FTP Server user interface:



Security Overview

SSL/TLS Status: Configured 2048-bit RSA
FIPS 140-2 Mode: Enabled
Min Encryption Strength: SSL: 256-bit SSH: 112-bit
FTP Access: Not Secure
SSH FTP Access: Secure
HTTP Access: Secure
Support Until: 2/12/2024 (614 days remaining)

Current Connections — Geolocation Disabled

Configure the Geolocation API Access Key in Server Manager

Network Overview

Host: DESKTOP-OSBH3VG
Public IP: 96.241.196.222
SOAP API: Port 10001, https
Web Administration: https://169.254.244.27:8443

Vulnerability Assessment

Latest Version: 12.7.4.0, 64-bit
Current Version: 12.7.4.0, 64-bit
Update Message: Last Checked: 6/2/2022
Vulnerabilities: None

System Messages

- Insecure protocol settings used in 1 SMTP Server Event Target
- Insecure protocol settings used in 1 HTTP Send Event Target
- The account 'anonymous' is configured to allow anonymous access
- HIPAA non-compliance: One or more listeners allow non-encrypted traffic
- FTP listener 8 can allow unencrypted control or data connections
- FTP listener 9 can allow session hijacking in passive secure data connections

IP Listeners

Status	ID	IP:Port	
✓	0	127.0.0.1:21	FTP
✓	1	127.0.0.1:990	FTPS
✓	2	127.0.0.1:22	SFTP
✓	3	127.0.0.1:8080	HTTP Admin
✓	4	127.0.0.1:443	HTTPS
✓	5	127.0.0.1:8443	HTTPS Admin
✓	6	127.0.0.1:8081	HTTPS
!	8	192.168.1.252:21	FTP
!	9	192.168.1.252:990	FTPS
✓	10	192.168.1.252:22	SFTP

Showing 1 to 10 of 14 entries

Previous 1 2 Next

Enabling or disabling a listener

Select a listener and right-click. Click the Enable/Disable menu item to toggle enabling or disabling a listener. Disabled listeners will no longer accept connections.

17.7 The HTTP/S Web Client

Available in Cerberus FTP Server Enterprise edition, the **HTTP/S web client** capability allows any user with access to a common web browser to easily connect to the server to perform file operations (uploading, downloading, deleting, renaming, creating directories, and zipping and unzipping files and directories) using a desktop or mobile web browser.

You can also grant users the ability to generate a public link to any file and email that link to someone from directly within the web client.

The web client is a native web application that requires no plug-ins or external tools to use. The web client relies on HTML and JavaScript for all of its functionality and will run on any modern web browser.

17.7.1 Adding an HTTP/S Listener

The Cerberus FTP Server web client can be accessed by adding an HTTP or HTTPS listener to Cerberus FTP Server's listener list. You can add a new HTTP/S listener from the **Listeners** page of the **Server Manager**.

To add a new HTTP or HTTPS listener:

1. Open the Server Manager
2. Select the **Listeners** page
3. Select the "plus" icon next to the interface list box to add a new interface. The "Add New Listener" dialog box will appear to ask for the interface details (interface IP, type, and port combination)
4. Select the IP address that you want to listen for connections on
5. Select the interface type (HTTP or HTTPS for web client access)
6. Enter the port you wish to listen on. Cerberus will automatically pre-populate the port with the default port for the type of listener you are adding
7. Press the **Add** button to add the listener

The listener should now be added to the Interfaces list. Press **Ok** to close the Server Manager and save your changes.

17.7.2 Web Client Customizations

The HTTP/S web client can be customized in several ways. Options for changing the default settings are discussed in the following sections.

17.7.2.1 Changing the Company Logo and Login Image

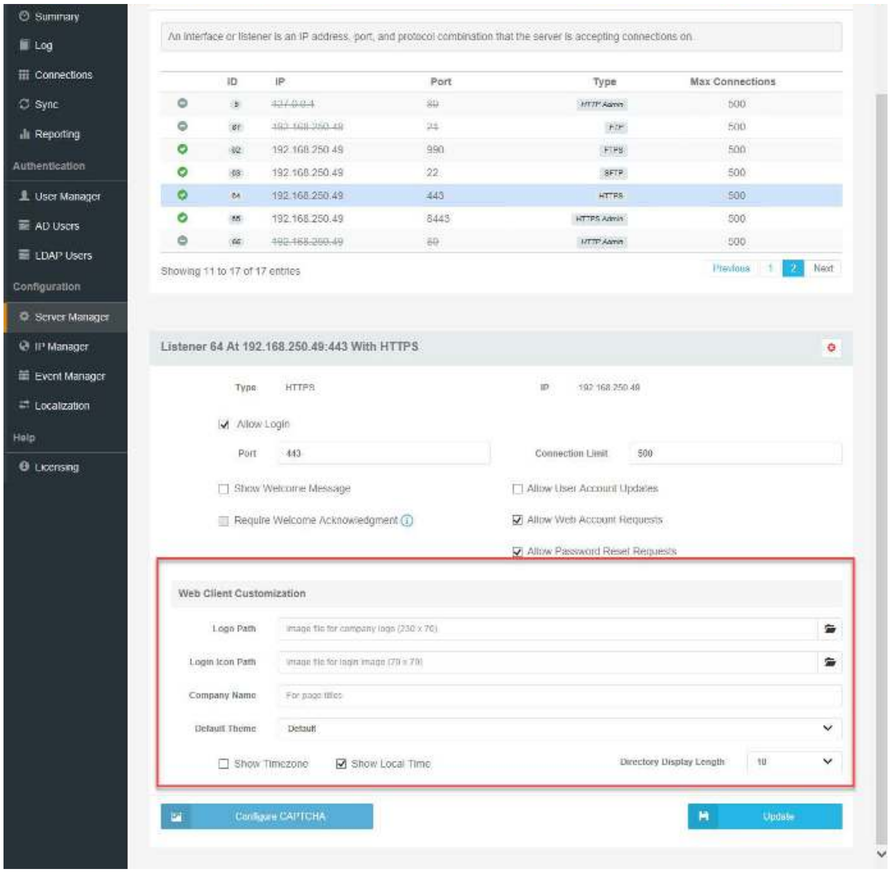
You can change the company logo displayed on the web client by specifying your own logo file.

1. Go to the **Listeners** page of the **Server Manager** (pictured above)

2. Select the HTTP/S interface you wish to change (not the default interface)
3. Press the file selection button across from the Logo Image edit box
4. Select the image file you wish to use and press Ok. The preferred image size is 230 x 70.

The login image displayed on the login page is also customizable using the same procedure as for the company logo. The preferred login image size is 70 x 70 pixels.

The image format for both logos should be one that is supported by all web browsers. We recommend PNG, GIF, or JPEG.



An interface or listener is an IP address, port, and protocol combination that the server is accepting connections on.

	ID	IP	Port	Type	Max Connections
●	5	~~127.0.0.1~~	~~80~~	HTTP Admin	500
●	61	~~192.168.250.49~~	~~21~~	FTP	500
✓	62	192.168.250.49	990	FTPS	500
✓	63	192.168.250.49	22	SFTP	500
✓	64	192.168.250.49	443	HTTPS	500
✓	65	192.168.250.49	8443	HTTPS Admin	500
●	66	~~192.168.250.49~~	~~80~~	HTTP Admin	500

Showing 11 to 17 of 17 entries

Previous 1 2 Next

Listener 64 At 192.168.250.49:443 With HTTPS

Type HTTPS IP 192.168.250.49

☑ Allow Login

Port 443 Connection Limit 500

☐ Show Welcome Message ☐ Allow User Account Updates

☐ Require Welcome Acknowledgment ⓘ ☑ Allow Web Account Requests

☑ Allow Password Reset Requests

Web Client Customization

Logo Path: image file for company logo (230 x 70)

Login Icon Path: image file for login image (70 x 70)

Company Name: For page titles

Default Theme: Default

☐ Show Timezone ☑ Show Local Time Directory Display Length 10

Configure CAPTCHA Update

17.7.2.2 Changing the Login Welcome Message

If you select the Show Welcome Message option for the HTTP/S listener then the server welcome message is displayed next to the login credentials box when a client logs in on that listener. This message can be customized from the Messages page of the Server Manager.

17.7.2.3 Custom Web Client Themes

The HTTPS web client comes installed with several themes, but administrators can easily adapt and add their own. The web client was redesigned in version 7.0 to use the popular Bootstrap 3 framework. You can develop your own custom CSS theme file and drop it in:

C:\Program Files\Cerberus LLC\Cerberus FTP Server\webadmin\client\custom

Then, restart the Cerberus FTP Server Windows Service to have it automatically detect and make available the new theme.

A theme file is simply a CSS file that contains your own custom overrides of the default Bootstrap 3 theme. Any files your CSS file references should be relative to the custom folder. Cerberus will detect the new CSS file during startup and make it available as a theme (the theme name is based on the file name) on the Accounts page of the web client.

17.7.2.4 Further Web Client Customizations

The HTTP/S web client can be further customized by modifying the underlying template files. However, any changes made to those template files will be overwritten whenever Cerberus FTP Server is updated. We are working on ways to allow more permanent and lasting changes to the web client. The relevant template files are in

C:\Program Files\Cerberus LLC\Cerberus FTP Server\webadmin

and

C:\Program Files\Cerberus LLC\Cerberus FTP Server\webadmin\client

The client-index.tpl file is probably the best place to start for modifying the overall look of the web client. The template files are cached in memory in Cerberus after the first time they are read, so a restart of the underlying Cerberus FTP Server Windows Service is required before any changes to these files will take effect.

18.0 Messages

18.1 Configuring Protocol Messages

You can configure common protocol messages on this tab, but they cannot be set per language. Note that the "Show Welcome Message" checkbox on each IP listener controls whether the server sends the welcome message or not. Not all clients will display a welcome message.

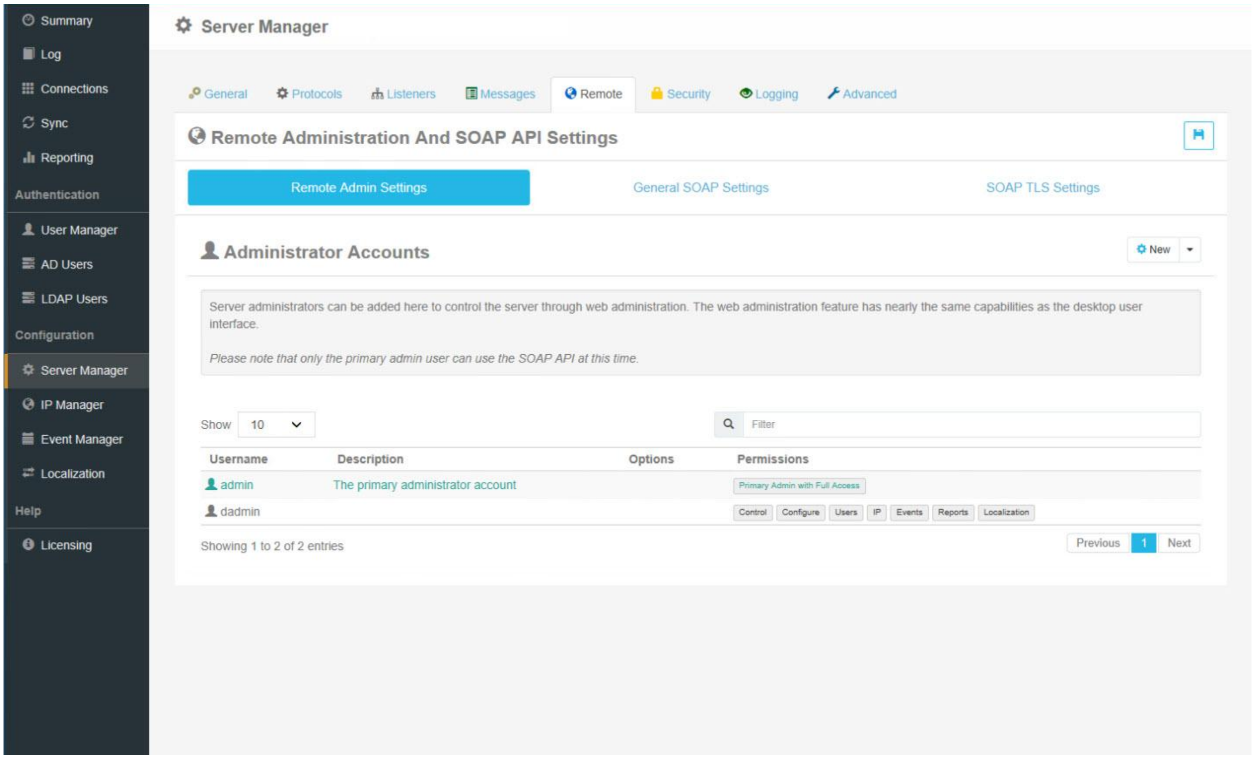
The messages you can configure are the ‘Welcome Message’, the ‘Goodbye Message’, and the ‘Max Connection Limit’ message

19.0 Remote Settings

19.1 Configuring Remote Settings

The remote settings page allows the administrator to configure web administration access and remote Application Programming Interface (API) access to Cerberus FTP Server. Cerberus allows remote access to the server administrator via a web browser-based interface and via the normal Cerberus FTP Server Graphical User Interface (GUI) when running in Windows Service mode.

For software developers, Cerberus exposes several APIs for controlling all aspects of the server using the SOAP web services.



Server administrators can be added here to control the server through web administration. The web administration feature has nearly the same capabilities as the desktop user interface.

Please note that only the primary admin user can use the SOAP API at this time.

Show 10

Filter

Username	Description	Options	Permissions
admin	The primary administrator account		Primary Admin with Full Access
dadmin			Control Configure Users IP Events Reports Localization

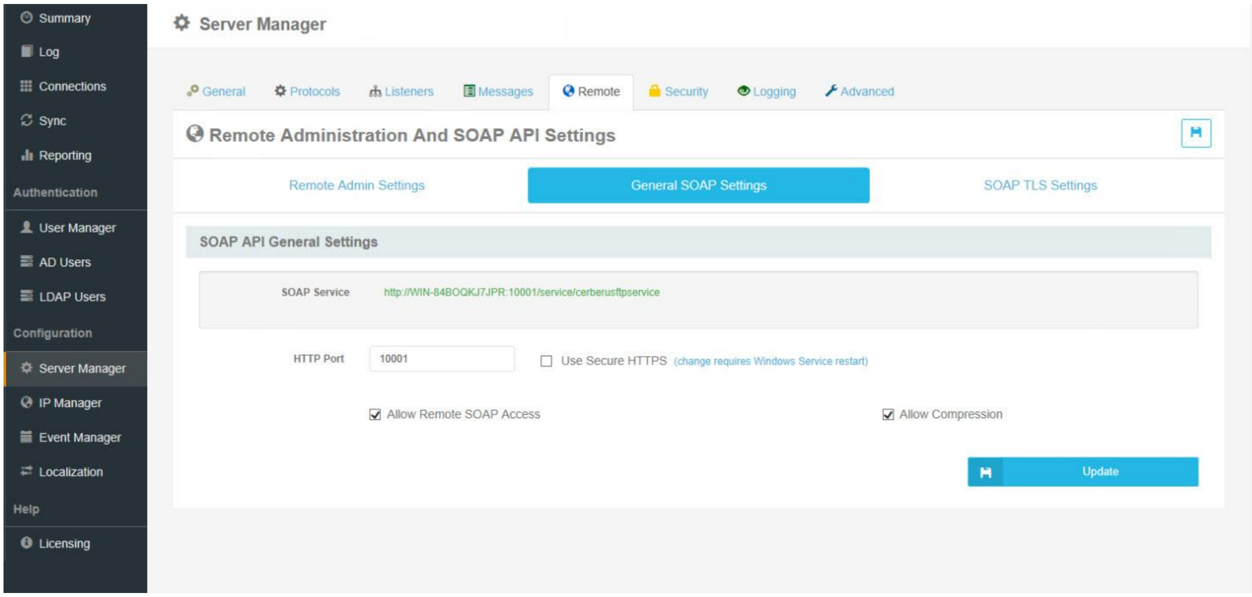
Showing 1 to 2 of 2 entries

Previous 1 Next

Remote settings page of the Server Manager

19.1.1 General SOAP Settings

The remote access settings control HTTP and HTTPS web administration, as well as SOAP API access to Cerberus FTP Server.



SOAP API General Settings

SOAP Service http://WIN-84BOQKJ7JPR:10001/service/cerberusftpservice

HTTP Port 10001

□ Use Secure HTTPS (change requires Windows Service restart)

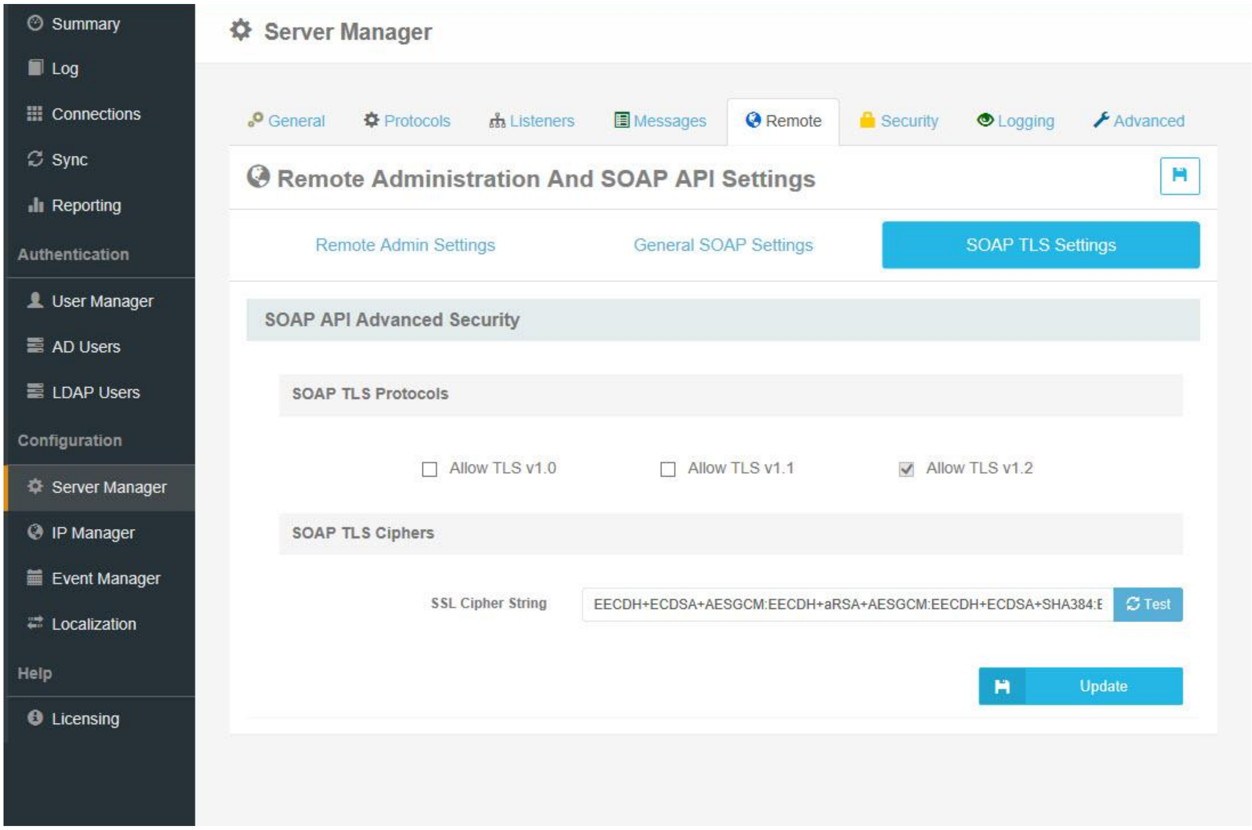
☑ Allow Remote SOAP Access

☑ Allow Compression

When Cerberus is running as a Windows Service, the GUI connects to and communicates with the Cerberus Windows Service through a remote access API called SOAP. The Cerberus Windows Service listens for SOAP connections on the Port specified under the Remote Settings page. That port must be available for Cerberus to listen on, or the GUI will be unable to connect to the service.

HTTPS Port	The port that the SOAP service and web administration pages will be served from.
Use Secure HTTPS	Select this option to allow only secure HTTPS connections for the web administration and SOAP access. A restart of the underlying Cerberus FTP Server Windows Service is required after changing this parameter.
Allow Remote SOAP Access	Enable SOAP-based remote access. SOAP is an API for connecting programmatically to the server. When this setting is enabled, applications can make SOAP calls to the server from outside the local machine (subject to authentication). **NOTE:** Local SOAP access is always enabled. The Cerberus UI requires SOAP access to enable communication between the UI and the underlying Cerberus Windows Service.

19.1.2 SOAP TLS Settings



SOAP API Advanced Security

SOAP TLS Protocols

□ Allow TLS v1.0 □ Allow TLS v1.1 ☑ Allow TLS v1.2

SOAP TLS Ciphers

SSL Cipher String: EECDH+ECDSA+AESGCM:EECDH+aRSA+AESGCM:EECDH+ECDSA+SHA384:E

You can control what SSL protocols are supported, as well as what ciphers to allow for SOAP-based SSL connections. Changes to these settings require a service restart.

19.1.3 Administrator Accounts

There is always a primary admin account, with full permissions to all server functions. The primary admin account is highlighted in green lettering in the administrator list.

Primary Admin Username	The username used to access the web administration page. This username is also used for basic authentication when using the SOAP web services API to access the server.
Primary Admin Password	The password used to access the web administration page. This password is also used for basic authentication when using the SOAP web services API to access the server. **NOTE:** This is also the username and password used when accessing Cerberus as a Windows Service from the Cerberus GUI. Normally, administrators won't be prompted for this password and the GUI will automatically connect to the service whenever it is started.

The administrator can also control the server through web administration. The web administration feature has nearly the same capabilities as the desktop user interface. Most server functions can be controlled through web administration.

19.1.4 Secondary Web Administration Accounts

You can assign additional web administration users, and limit their access to different aspects of the server like user management, reporting, etc.

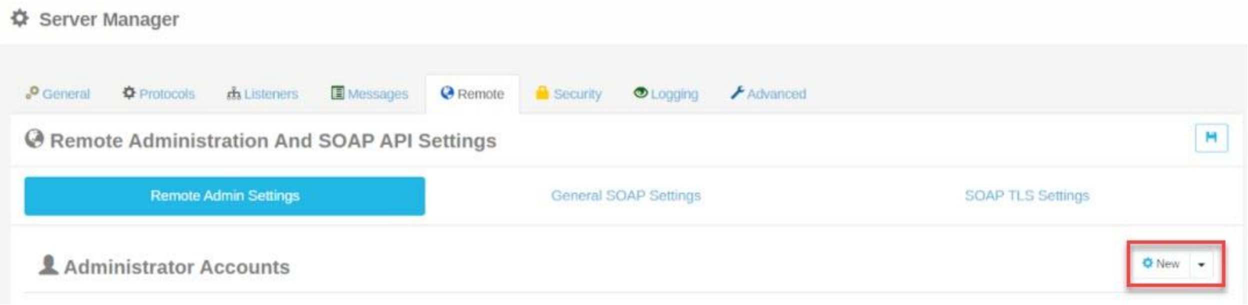
Secondary web administration users can be managed on the Remote page.

Note: Secondary web administration users cannot access the SOAP API. Only the primary admin user can use the SOAP API at this time.

You can also assign additional web administration users, and limit their access to different aspects of the server like user management, reporting, etc.

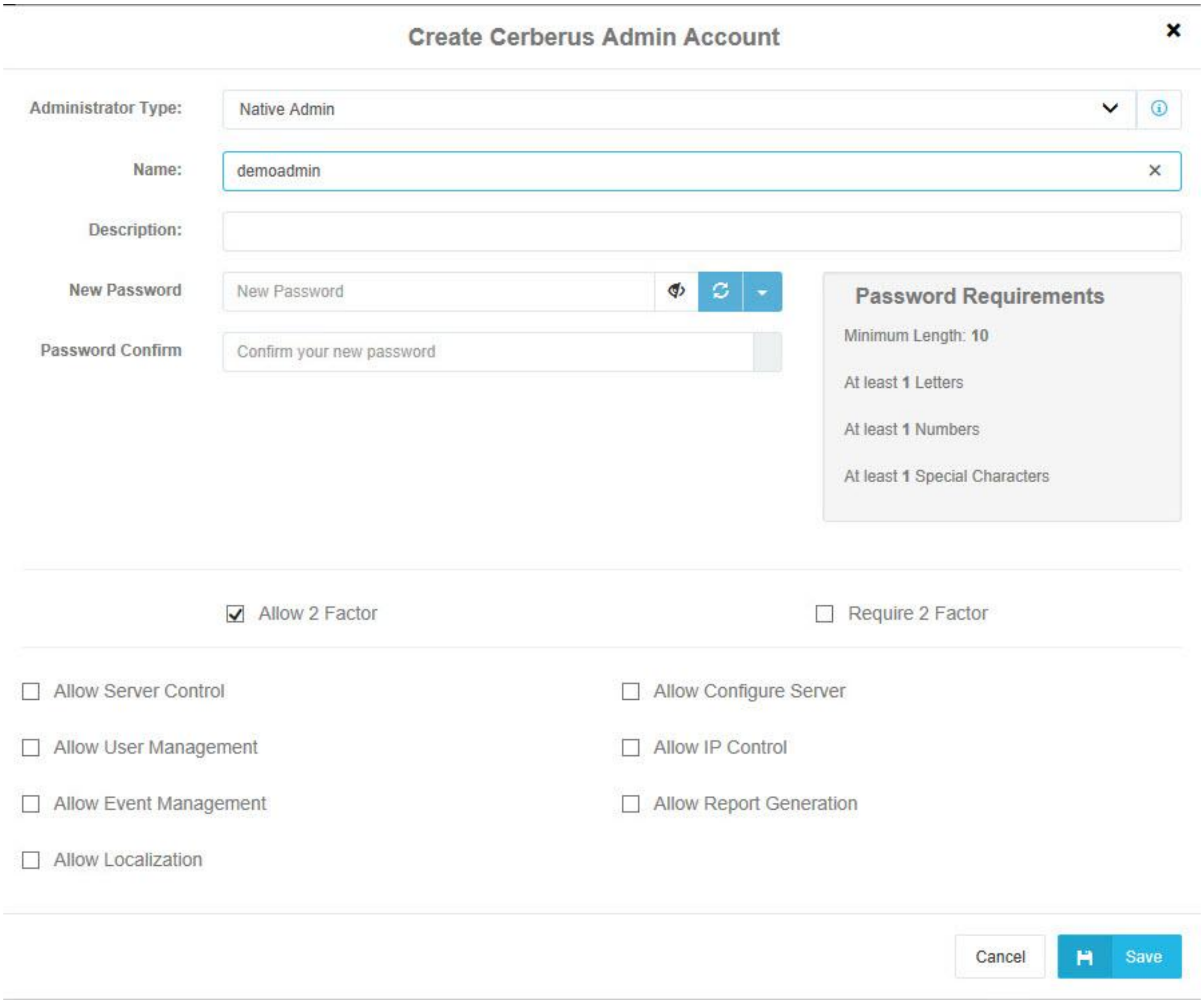
Secondary web administration users can be managed on the **Remote** page.

Press the **New** button to create new admin users.



*The **New** Administrator button on the Remote page*

Fill in the admin user's information in the New Cerberus Admin Account dialog that appears.



Create Cerberus Admin Account

Administrator Type: Native Admin

Name: demoadmin

Description:

New Password: New Password

Password Confirm: Confirm your new password

Password Requirements

Minimum Length: **10**

At least **1** Letters

At least **1** Numbers

At least **1** Special Characters

☑ Allow 2 Factor

□ Require 2 Factor

□ Allow Server Control

□ Allow Configure Server

□ Allow User Management

□ Allow IP Control

□ Allow Event Management

□ Allow Report Generation

□ Allow Localization

The New Cerberus Admin Account Dialog in the Server Manager

There are two types of administrators to choose from on the *Administrator Type* drop-down:

- **Native Admin** creates an admin account whose details and credentials are managed entirely within Cerberus FTP Server.
- **Directory Admin** type. This admin type allows you to extend Cerberus Administration rights to Active Directory users and groups.

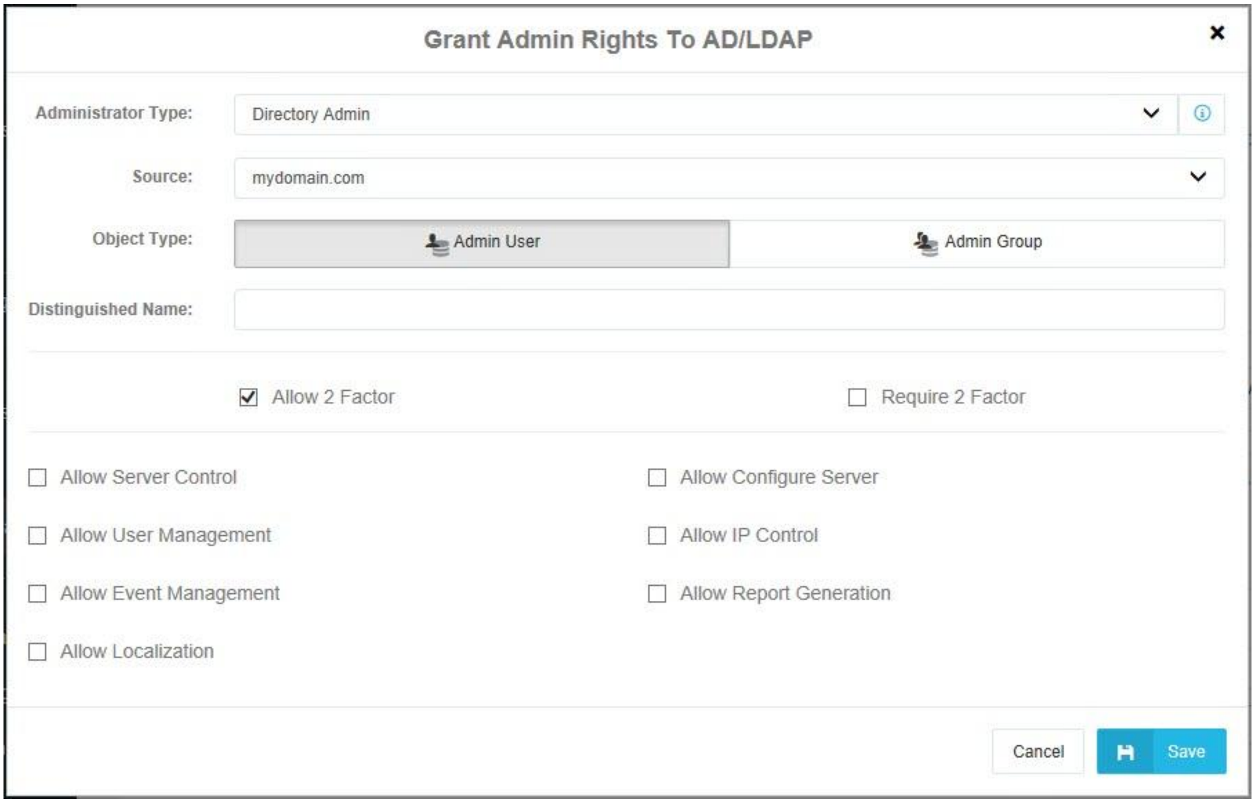
19.1.4.1 Directory Administrator Options

This window for **Directory Admin** displays all the options you'll need to grant Web Admin rights to directory-based users and groups:

- ***Source***
  The domain of the user/group to receive Admin access. The pull-down lists contain only *AD*

Admin Connections. All Admin Connections appear in the pull-down, but most deployments will need only one.

- ***Object Type***
  Admin User grants access to a single domain user.
  Admin Group grants access to all members of the group. Nested groups (and their members) also inherit the assigned permissions.
- ***Distinguished Name***
  The DN of the user or group, for example, “*CN=DirAdmin,CN=Users,DC=mydomain,DC=com”*
  It is best to copy and paste from an AD administration tool like **Active Directory Users and Groups** or PowerShell cmdlets **Get-ADUser** and **Get-ADGroup**



Grant Admin Rights To AD/LDAP

Administrator Type: Directory Admin

Source: mydomain.com

Object Type: Admin User | Admin Group

Distinguished Name:

☑ Allow 2 Factor □ Require 2 Factor

□ Allow Server Control □ Allow Configure Server

□ Allow User Management □ Allow IP Control

□ Allow Event Management □ Allow Report Generation

□ Allow Localization

The remaining options are common to both **Native Admins and Directory Admins**, and control two-factor policy and fine-grained administrative rights:

- ***Allow 2 Factor, Require 2 Factor***
  Allow or *Require* users and groups to setup two-factor authentication.
- ***Permissions***
  Admin roles allowed to this user or group.

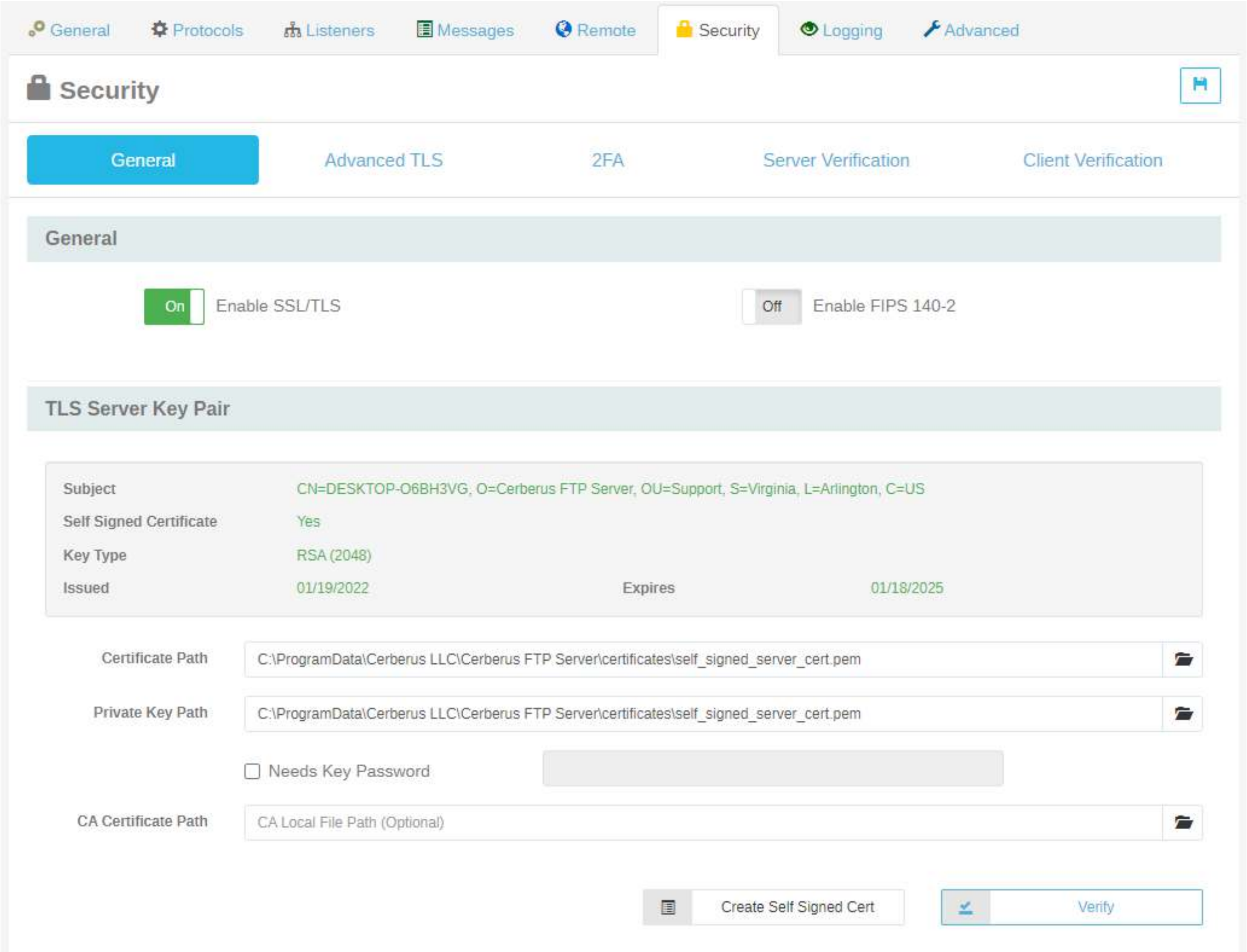
Admin access that can be granted to a user or group	
This permission...	**...grants access to:**
Allow Server Control	● Log ● Connections ● Sync Manager ● Licensing
Allow Configure Server	● Server Manager
Allow User Management	● User Manager ● AD Users ● LDAP Users
Allow IP Control	● IP Manager
Allow Event Management	● Event Manager
Allow Report Generation	● Reporting
Allow Localization	● Localization

Please note that secondary web administration users cannot access the SOAP API. Only the primary admin user can use the SOAP API at this time.

20.0 Security Settings

The security settings page allows the administrator to configure all aspects of Cerberus FTP Server SSL/TLS and SSH security. To enable TLS/SSL connections between FTP and HTTP clients and the server, you need a server certificate and a private key.

Cerberus uses the settings here for all secure connections.



General

On Enable SSL/TLS

Off Enable FIPS 140-2

TLS Server Key Pair

Subject	CN=DESKTOP-O6BH3VG, O=Cerberus FTP Server, OU=Support, S=Virginia, L=Arlington, C=US		
Self Signed Certificate	Yes		
Key Type	RSA (2048)		
Issued	01/19/2022	Expires	01/18/2025

Certificate Path: C:\ProgramData\Cerberus LLC\Cerberus FTP Server\certificates\self_signed_server_cert.pem

Private Key Path: C:\ProgramData\Cerberus LLC\Cerberus FTP Server\certificates\self_signed_server_cert.pem

☐ Needs Key Password

CA Certificate Path: CA Local File Path (Optional)

Security settings page of the Server Manager

20.1 General

These are basic TLS/SSL settings applicable to secure client FTPS, HTTPS, and SSH connections.

20.1.1 General Settings

Enable TLS/SSL	This must be enabled to allow secure access to the server. NOTE: A certificate and private key must be available before TLS/SSL encryption will be available.
Enable FIPS 140-2	Enable the FIPS 140-2 certified encryption module for Cerberus FTP Server. Selecting this option enables encryption using only FIPS 140-2 certified algorithms. *Only available in the Professional and Enterprise edition*.

20.1.2 TLS Server Key Pair

Cerberus FTP Server supports RSA, DSA, and Elliptical Curve (EC) keys.

There are generally two options for obtaining a digital certificate (with a private key):

1. You can generate your own self-signed certificate using the Cerberus **CreateSelf Signed Cert** button.

2. You can obtain a certificate from a recognized certificate authority (CA)

Which is more appropriate really depends upon your goals. If you just want to make sure that client and server connections are securely encrypted then a self-signed certificate is all you need. It has the benefit of being easily created through Cerberus and completely free.

If your goal is to make sure that your clients can verify that the server they are connecting to is legitimate and ensure they don't see any warning messages about being "unable to verify the server" then using a certificate signed by a trusted certificate authority is required. You will have to contact one of the recognized Certificate Authorities such as GoDaddy, Digicert, Sectigo, Comodo, Thawte, Verisign or one of the many other recognized Certificate Authorities and request a server certificate (for a price).

A note about secure connections: Cerberus supports FTPS, FTPES, SFTP, and HTTPS encryption. To establish a secure connection you must connect to the server with a client that supports one of those secure methods. For secure FTPES, FTPS, or SFTP, this will require a dedicated FTP client, not a web browser. *No* web browsers natively support any type of secure FTP.

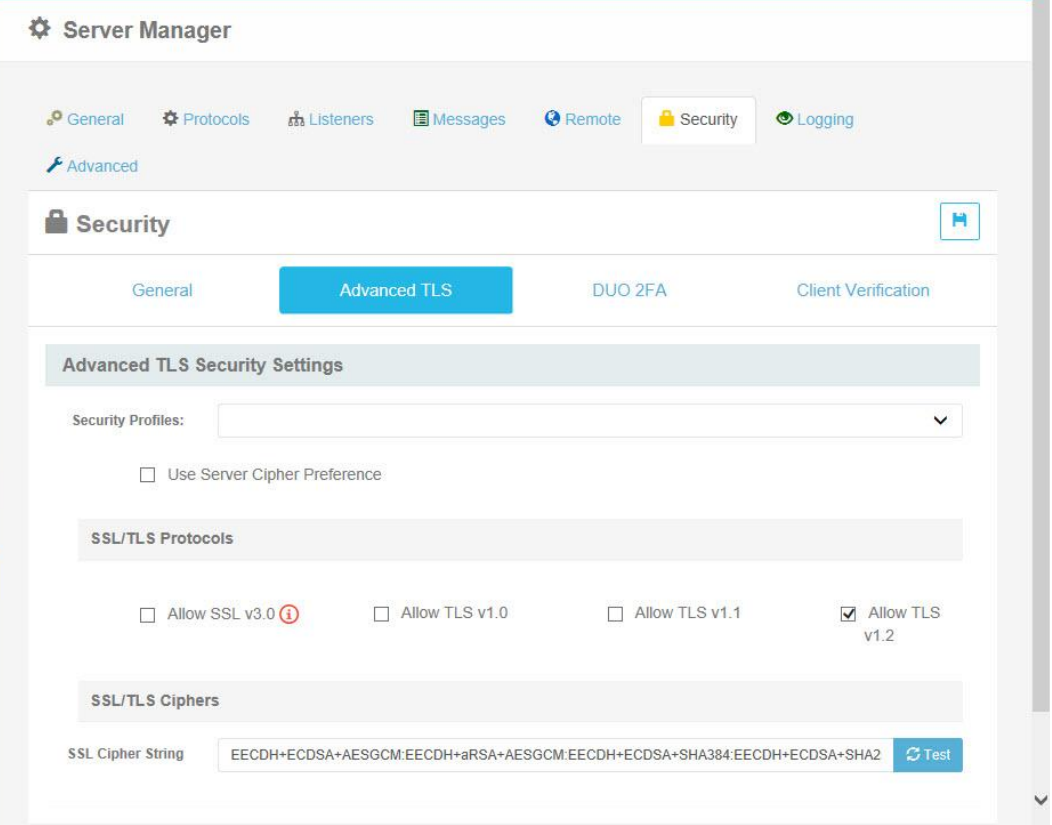
Certificate Path	The full path to your public certificate. The public certificate is exchanged with the client during TLS/SSL encryption and is examined by the client to verify the server. Supported key types include RSA, DSA, and Elliptical Curve keys.
Private Key Path	This is the server's private key. The private key is used to encrypt messages to the client. The client can use the server's public key to decrypt messages encrypted with the server's private key. The private key is not sent to the client. If your public and private key are in the same file then set this path to be the same as the Public Certificate.

	NOTE: The public and private key can be in the same file. If your public and private key are in the same file then set this path to the same path as your Public Certificate path. Cerberus understands both DER and PEM encoded certificate formats.
Needs Key Password	Check this option if the digital certificate is encrypted.
Password	The key password used to decrypt your digital certificate.
CA Certificate Path	A file containing a PEM-encoded list of Certificate Authorities with which to verify client certificates against. Cerberus FTP Server will also use this file to load and send the entire certificate chain for the server certificate when a client connects. Many CAs call this a CA bundle file.
Create Cert	Cerberus will generate a Self-Signed Certificate that will allow encrypted connections.
Verify	Cerberus will attempt to verify that the certificate at the Public and Private key path is recognized and readable with the given password.

20.1.3 SSH Host Key Pair

As of Cerberus 12.4, the application will now create a separate SSH Host Key Pair derived from the TLS/SSL certificate. The SSH Host Key Pair created is static and will not change when and if an entirely new TLS/SSL certificate is installed. This has the benefit of not requiring SFTP client software to recache or accept a new host key fingerprint if the TLS/SSL certificate changes.

20.2 Advanced TLS Options



Advanced TLS Security Settings

Security Profiles:

□ Use Server Cipher Preference

SSL/TLS Protocols

□ Allow SSL v3.0 □ Allow TLS v1.0 □ Allow TLS v1.1 ☑ Allow TLS v1.2

SSL/TLS Ciphers

SSL Cipher String: EECDH+ECDSA+AESGCM:EECDH+aRSA+AESGCM:EECDH+ECDSA+SHA384:EECDH+ECDSA+SHA2

Security Profiles	These are common security settings. Selecting a security profile from the dropdown list will immediately modify the server's security settings to match that profile.
Server Cipher Preference	During SSL/TLS session negotiation, the connecting client sends an ordered list of cipher suites to the server. The first suite in the list is the one most preferred by the client. Normally, the server honors the client's preference by selecting the suite most preferred by the client among the list of suites that both the client and server support. If this option is selected, the server selects the suite that the server itself most prefers among those that both the client and server support. This can be used to, for example, enforce that the strongest cipher that both the server and client support be used for the connection.
SSL Cipher String	The ciphers that Cerberus uses during secure connection negotiation for TLS/SSL can be controlled through a text string. This option can be accessed by pressing the **Advanced** button. An example string: **ALL:!LOW:!EXP:!ADH:@STRENGTH** The string follows the same cipher string format as the OpenSSL ciphers string.
Allow SSLv3, TLS1.0, TLS1.1, TLS1.2	These 4 settings allow you to enable or disable support for SSLv3.0, TLSv1.0, TLSv.1.1, and TLSv1.2 respectively.

20.3 2FA (DUO)

Keep your accounts safe with two-factor authentication by Duo.

Duo combines modern two-factor authentication with advanced endpoint security solutions to protect users from account takeovers and data breaches.

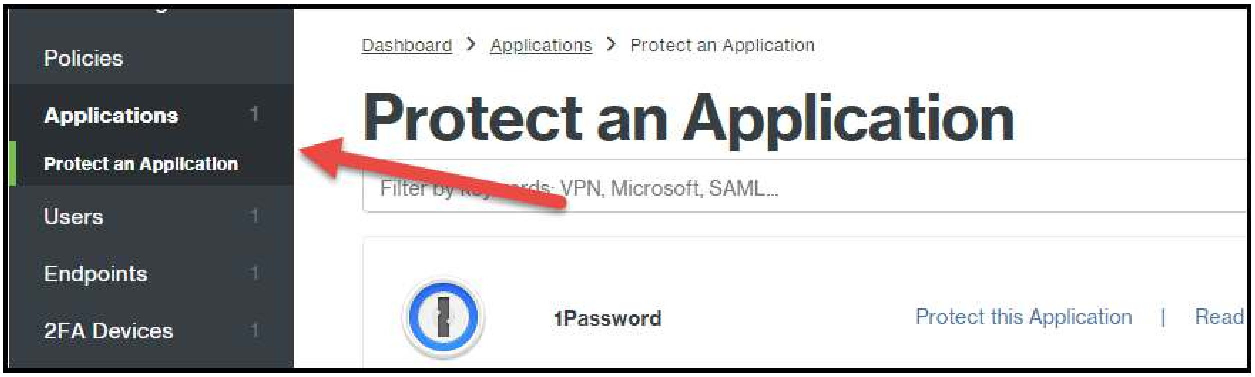
Two-factor authentication is one of the best ways to protect against remote attacks such as phishing, credential exploitation, and other attempts to take over your accounts. Without your physical device, remote attackers cannot pretend to be you in order to gain unauthorized access to corporate networks, cloud storage, financial information, etc.

After successful primary authentication, users simply approve a secondary authentication request pushed to the Duo Mobile smartphone app. Users may also authenticate by answering a phone call or by entering a one-time passcode generated by the Duo Mobile app, a compatible hardware token, or received via SMS (Short Message Service).

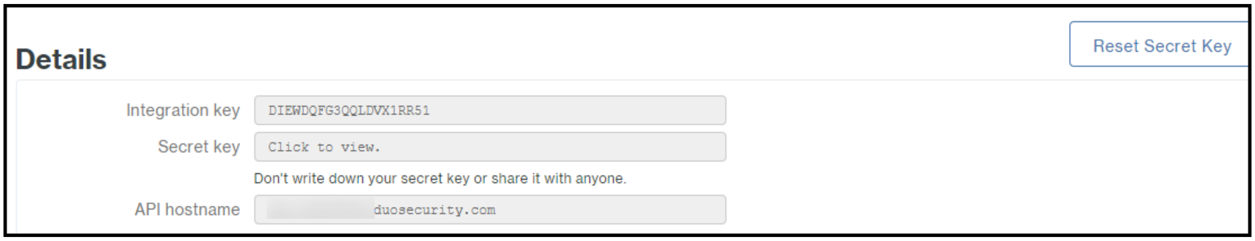
Before starting

1. Sign up for a DUO Account.

2. Log in to the Duo Admin Panel and navigate to **Applications**.


Policies
Applications 1
Protect an Application
Users 1
Endpoints 1
2FA Devices 1
Dashboard > Applications > Protect an Application
Protect an Application
Filter by keywords: VPN, Microsoft, SAML...
1Password
Protect this Application | Read

3. Click **Protect an Application** and locate **Auth API or Web SDK** in the applications list. Click **Protect this Application** to get your **integration key**, **secret key**, and **API hostname**.



Details

Integration key: DIEWDQFG3QQLDVX1RR51

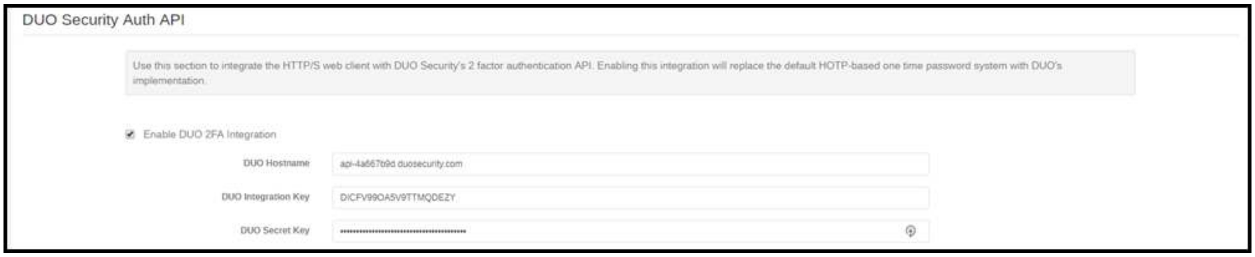
Secret key: Click to view.

Don't write down your secret key or share it with anyone.

API hostname: [redacted]duosecurity.com

Setting up DUO in Cerberus

In the **Server Manager**, Open **Security,** Click on **“2FA”,** and Check "**Enable DUO 2FA Integration**"



DUO Security Auth API

Use this section to integrate the HTTP/S web client with DUO Security's 2 factor authentication API. Enabling this integration will replace the default HOTP-based one time password system with DUO's implementation.

☑ Enable DUO 2FA Integration

DUO Hostname: api-4a667b9d.duosecurity.com

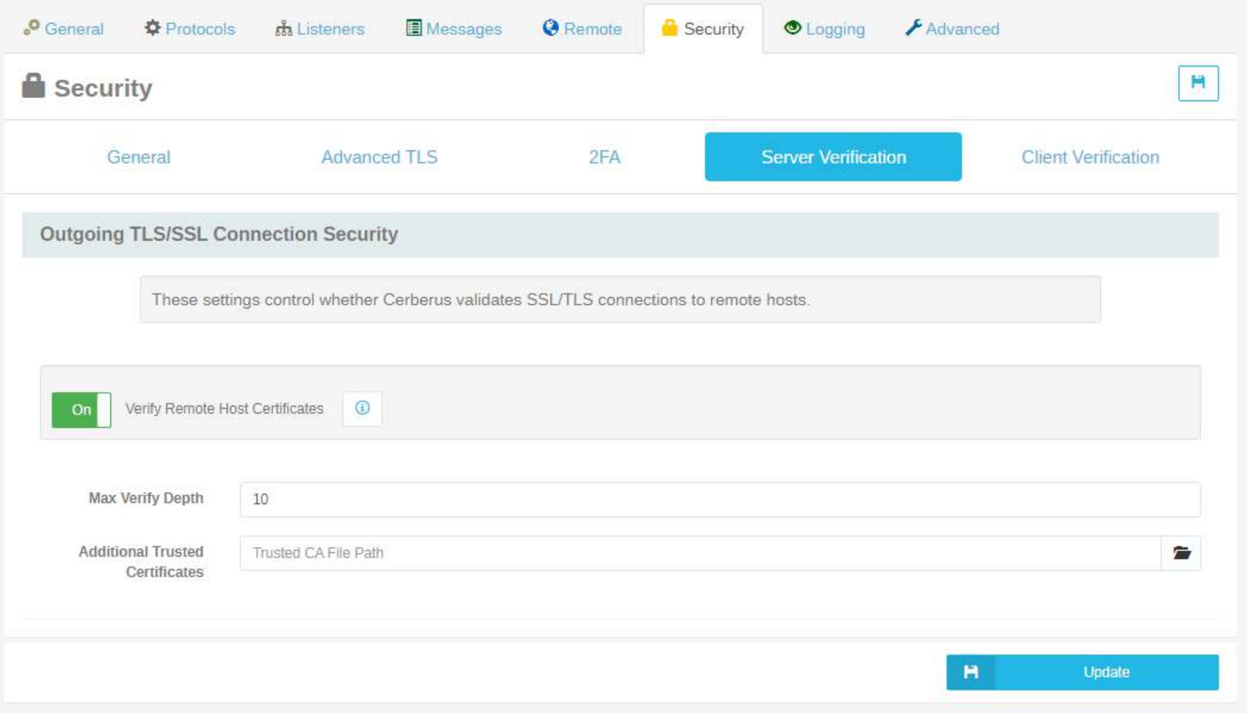
DUO Integration Key: DICFV99OA5V9TTMQDEZY

DUO Secret Key: ••

Enter the details of your Duo account. (**Hostname, Integration Key, Secret Key**) and select **Update**.

DUO is now enabled and will replace the default HOTP implementation.

20.4 Server Verification



Outgoing TLS/SSL Connection Security

These settings control whether Cerberus validates SSL/TLS connections to remote hosts.

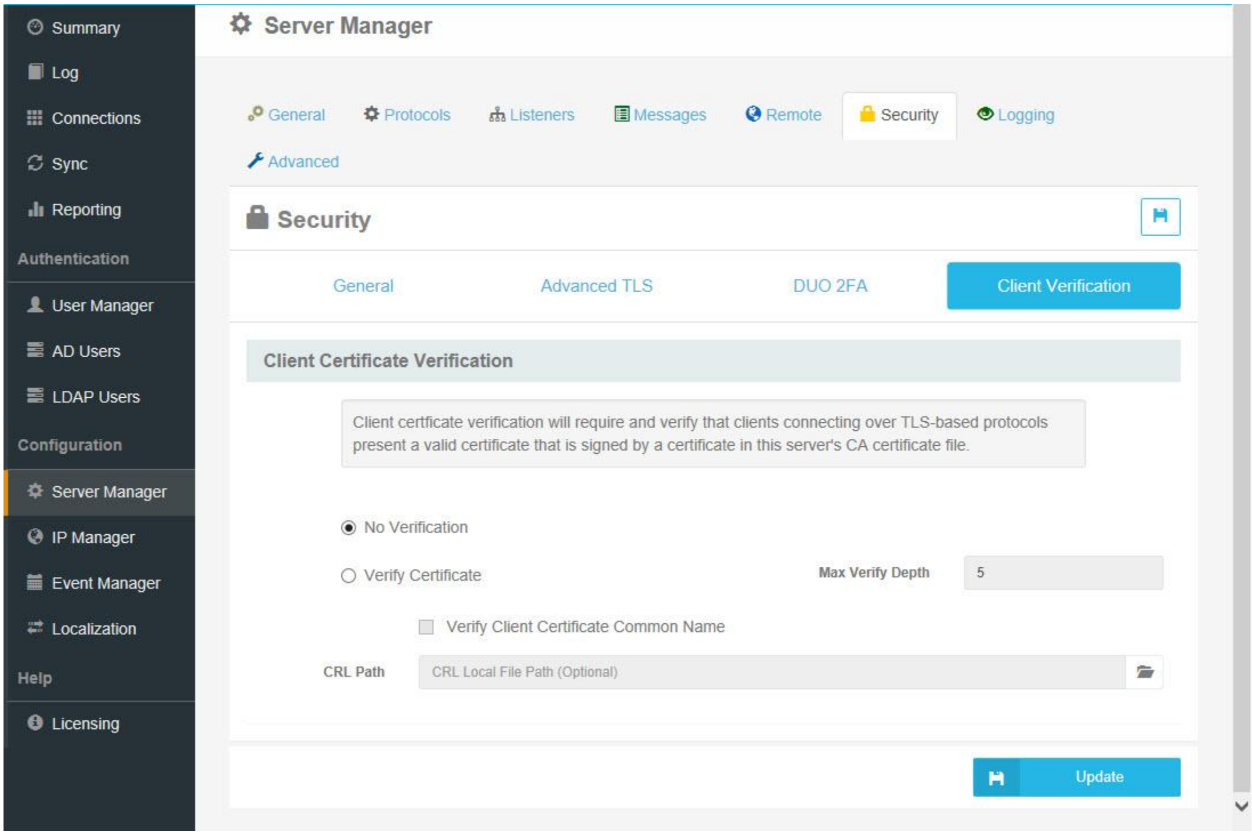
On Verify Remote Host Certificates

Max Verify Depth: 10

Additional Trusted Certificates: Trusted CA File Path

Verify Remote Host Certificate	Turning Server Verification off is global, overriding all other settings throughout Cerberus. *Turning verification off is the less secure option and is only provided as a temporary fail-safe, such as a certificate issue causing a critical service outage.*
Max Verify Depth	This determines how many issuers' certificates Cerberus will follow when verifying. Administrators may increase this value if remote hosts have long certificate chains.
Additional Trusted Certificates	Administrators may provide a path to a PEM file containing additional certificates that Cerberus should trust when verifying remote servers. Use this option when Cerberus should trust certificates that cannot or should not be imported to the operating system certificate store.

20.5 Client Verification



Client Certificate Verification

Client certficate verification will require and verify that clients connecting over TLS-based protocols present a valid certificate that is signed by a certificate in this server's CA certificate file.

● No Verification

○ Verify Certificate

Max Verify Depth: 5

□ Verify Client Certificate Common Name

CRL Path: CRL Local File Path (Optional)

Cerberus FTP Server can be configured to require FTPS and HTTPS clients to verify themselves using digital certificates. When given a CA file, Cerberus will verify that the client certificate is signed and valid for the given certificate authorities. Cerberus will also make sure the certificate hasn't been revoked if a CRL is specified. **This feature is only available in Cerberus FTP Server Professional and Enterprise edition and currently only applies to FTPS, FTPES, and HTTPS connections.**

You only need to worry about setting up and validating against a certificate authority if you (the server) want to authenticate the certificates coming from your FTPS and HTTPS clients. If you are not concerned with verifying your FTPS and HTTPS clients using certificates, then you can safely ignore all of the certificate authority configuration information. Select the **No verification** setting (the default) under ‘Client Certificate Verification’. **Note:** Client certificate verification is completely separate from SSH SFTP public key authentication. SSH SFTP public key authentication is configured on a per-user basis.

No Verification	This is the default option. Cerberus will not require nor will it verify digital certificates
Verify Certificate	Cerberus will attempt to verify that the certificate presented by the client is signed and valid. It will compare the certificate against the certificate authorities present in the CA Certificates File. Any FTPS or HTTPS

	connection attempts without a valid certificate will be denied when this option is selected.
CRL File	A file containing a PEM or DER-encoded list of key serial numbers that have been revoked. Note, that the CRL must have been signed by the CA certificate.

20.6 DSA Certificates and Ephemeral Diffie-Hellman Keys

Cerberus FTP Server includes support for **DSA certificates**. Unlike RSA certificates, DSA certificates cannot be used for key exchange (a necessary part of establishing an SSL or SSH connection), and additional pieces of information, known as Diffie-Hellman (DH) parameters, are required to allow key exchange using DSA.

DH parameters are computationally very expensive to generate, and it isn't feasible (or necessary) to generate those parameters in real-time. Cerberus FTP Server includes DH parameters for 512, 1024, 2048, and 4096-bit keys. The parameters were pre-generated using strong sources of pseudo-random entropy, and are used during DH key exchange to generate new, temporary keys for each SSL session.

Cerberus looks for the DH parameter files in the **C:\ProgramData\Cerberus LLC\Cerberus FTP Server\certificates** directory. You can freely replace the included parameter files with your own, pre-generated versions if you desire. If the existing files are deleted, Cerberus will attempt to re-create the missing files during startup by generating new ones. This can take a *very* long time, and Cerberus will appear to hang during startup while the files are generated. Deleting the existing DH parameter files is **not recommended**.

20.6 Elliptic Curve SSH Support

Cerberus FTP Server 4.0.9 and higher support Elliptic Curve Diffie-Hellman (ECDH) key agreement, Elliptic Curve Digital Signature Algorithm (ECDSA), and elliptic curve public keys for SSH SFTP as specified in RFC 5656. Only the required NIST curves at 256, 384, and 521 bits with uncompressed points are currently supported. Please see this page for more information on elliptic curve cryptography support.

21.0 Configuring Logging Support

21.1 Auditing

Cerberus FTP Server provides comprehensive logging of all file and user operations and provides both on-screen logging, file logging, and Syslog support. File-based logging can be managed through an XML configuration file that can control nearly all aspects of how log data is written to a file.

21.2 Log File Location

Cerberus FTP Server logging is implemented through the **Apache Log4cxx** framework, a robust logging package modeled after the popular log4j Java logging package. The default configuration logs up to 5000 KB of data to a single file and then rolls over to a new log file. The past 10 log files are kept by default but log file size, naming, and history are all completely configurable through the log4j.xml file.

The log file is located at the following location:

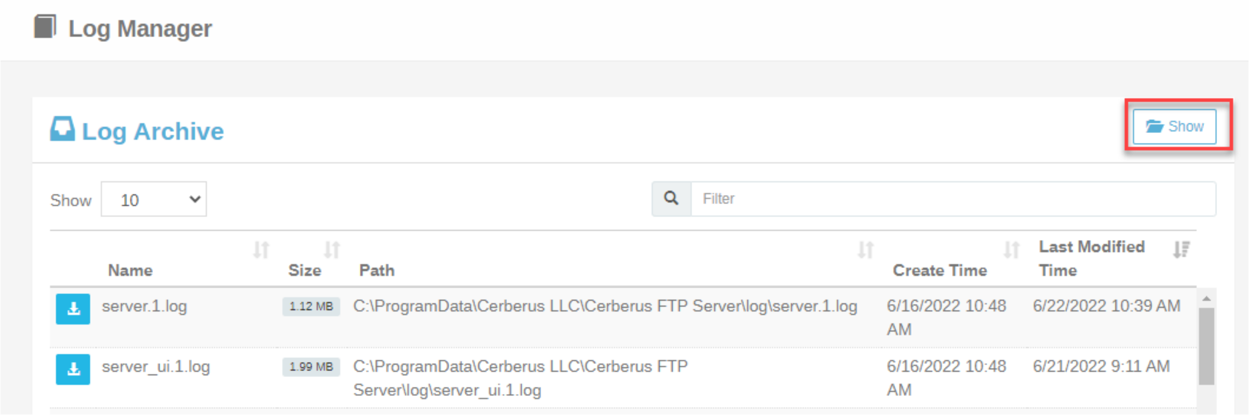
On Windows Vista, Windows 2008 and above

C:\ProgramData\Cerberus LLC\Cerberus FTP Server\log

On Windows 2003, XP, and 2000

C:\Documents and Settings\All Users\Application Data\Cerberus LLC\Cerberus FTP Server\log

You can also open the log file by simply clicking on the **Show** button on the **Log** tab of the main user interface:



	Name	Size	Path	Create Time	Last Modified Time
⬇	server.1.log	1.12 MB	C:\ProgramData\Cerberus LLC\Cerberus FTP Server\log\server.1.log	6/16/2022 10:48 AM	6/22/2022 10:39 AM
⬇	server_ui.1.log	1.99 MB	C:\ProgramData\Cerberus LLC\Cerberus FTP Server\log\server_ui.1.log	6/16/2022 10:48 AM	6/21/2022 9:11 AM

21.3 Configuring Logging

The **log4j.xml** configuration file is one level above in the "Cerberus FTP Server" folder. An example log4j.xml file is below.

There is an example of a size-based log appenders which roll over after the log file reaches a certain maximum size and that limit the number of log files that are kept. These types of loggers are limited to at most 13 saved log files.

There is also a daily log file appender example (with no maximum number of kept log file limits), and a Syslog log appender example.

21.3.1 Example Size Based log4j.xml Configuration File

```
<?xml version="1.0" encoding="UTF-8" ?>
 <log4j:configuration xmlns:log4j='http://logging.apache.org/' debug="false">

        <!-- A Size-based log file that rolls over to a new file after 5000KB and
keeps
                        at most 5 log files -->
        <appender name="FILE" class="org.apache.log4j.rolling.RollingFileAppender">
                <rollingPolicy
class="org.apache.log4j.rolling.FixedWindowRollingPolicy" >
                        <param name="activeFileName" value="log/server.log" />
                        <param name="fileNamePattern" value="log/server.%i.log" />
                        <param name="minIndex" value="1" />
                        <param name="maxIndex" value="5" />
                </rollingPolicy>
                <triggeringPolicy
class="org.apache.log4j.rolling.SizeBasedTriggeringPolicy">
                        <param name="maxFileSize" value="5000KB" />
                </triggeringPolicy>
                <layout class="org.apache.log4j.PatternLayout">
                        <param name="ConversionPattern" value="[%d{yyyy-MM-dd
HH:mm:ss}]:%7.7p [%6.6x] - [%X{client.ip}]:%X{client.user} - %m%n" />
                </layout>
        </appender>


        <!-- Add an appender that logs all errors to a separate log file -->
        <appender name="ERROR_FILE"
class="org.apache.log4j.rolling.RollingFileAppender">
                <rollingPolicy
class="org.apache.log4j.rolling.FixedWindowRollingPolicy">
                        <param name="activeFileName" value="log/server_error.log"/>
                        <param name="fileNamePattern"
value="log/server_error.%i.log"/>
                </rollingPolicy>
                <triggeringPolicy
class="org.apache.log4j.rolling.SizeBasedTriggeringPolicy">
                        <param name="maxFileSize" value="5000KB"/>
                </triggeringPolicy>
                <layout class="org.apache.log4j.PatternLayout">
                        <param name="ConversionPattern"
                                value="[%d{yyyy-MM-dd HH:mm:ss}]:%7.7p [%6.6x] -
%m%n"/>
                </layout>
                <filter class="org.apache.log4j.varia.LevelRangeFilter">
                        <param name="LevelMin" value="ERROR" />
                </filter>
        </appender>
        <root>
                <level value="INFO" class="org.apache.log4j.xml.XLevel" />
                <appender-ref ref="FILE"/>
                <appender-ref ref="ERROR_FILE"/>
        </root>
</log4j:configuration>
```

Possible values for the **<level value="LEVEL" class="org.apache.log4j.xml.XLevel" />** tag's ***level*** parameter are:

- TRACE
- DEBUG
- INFO
- WARN
- ERROR

21.3.2 Example Syslog log4j.xml Configuration File

The below log file example shows a Syslog logger.

```
<?xml version="1.0" encoding="UTF-8" ?>
 <log4j:configuration xmlns:log4j='http://logging.apache.org/' debug="false">


       <!-- Add a Syslog appender -->
       <appender name="syslog" class="org.apache.log4j.net.SyslogAppender">
              <param name="SyslogHost" value="127.0.0.1"/>
              <param name="Facility" value="USER"/>
              <param name="FacilityPrinting" value="true"/>
              <layout class="org.apache.log4j.PatternLayout">
                     <param name="ConversionPattern"
                         value="%t %5r %-5p %-21d{yyyyMMdd HH:mm:ss,SSS} %c{2} [%x] %m
%n"/>
              </layout>
       </appender>


       <root>
              <level value="INFO" class="org.apache.log4j.xml.XLevel" />
              <appender-ref ref="syslog"/>
       </root>
 </log4j:configuration>
```

21.3.3 Example Daily Rollover log4j.xml Configuration File

The below log file example shows a simple daily rollover logger.

```
<?xml version="1.0" encoding="UTF-8" ?>
 <log4j:configuration xmlns:log4j='http://logging.apache.org/' debug="false">


       <!-- Add a Daily log file appender that will roll over to a new log file each
night -->
       <appender name="DAILY_ROLL"
class="org.apache.log4j.rolling.RollingFileAppender">
           <rollingPolicy class="org.apache.log4j.rolling.TimeBasedRollingPolicy">
                    <param name="FileNamePattern"
value="log/daily_server.%d{yyyy-MM-dd}.log"/>
           </rollingPolicy>
           <layout class="org.apache.log4j.PatternLayout">
                    <param name="ConversionPattern" value="[%d{yyyy-MM-dd
HH:mm:ss}]:%7.7p [%6.6x] - [%X{client.ip}]:%X{client.user} - %m%n" />
           </layout>
       </appender>

       <root>
             <level value="INFO" class="org.apache.log4j.xml.XLevel" />
             <appender-ref ref="DAILY_ROLL"/>
       </root>
 </log4j:configuration>
```

21.3.4 Log File Location and Naming

You can change the location and name of the file created under the various log appenders using the appropriate field. For the default RollingFileAppender logger with a FixedWindowRollingPolicy you will need to change both the activeFileName and fileNamePattern parameters in the **log4j.xml** file. For the DailyRollingFileAppender you will just need to change the File parameter associated with the logger.

If using a relative log file path, the path is relative to the **C:\ProgramData\Cerberus LLC\Cerberus FTP Server** folder.

21.4 Screen Logging Settings

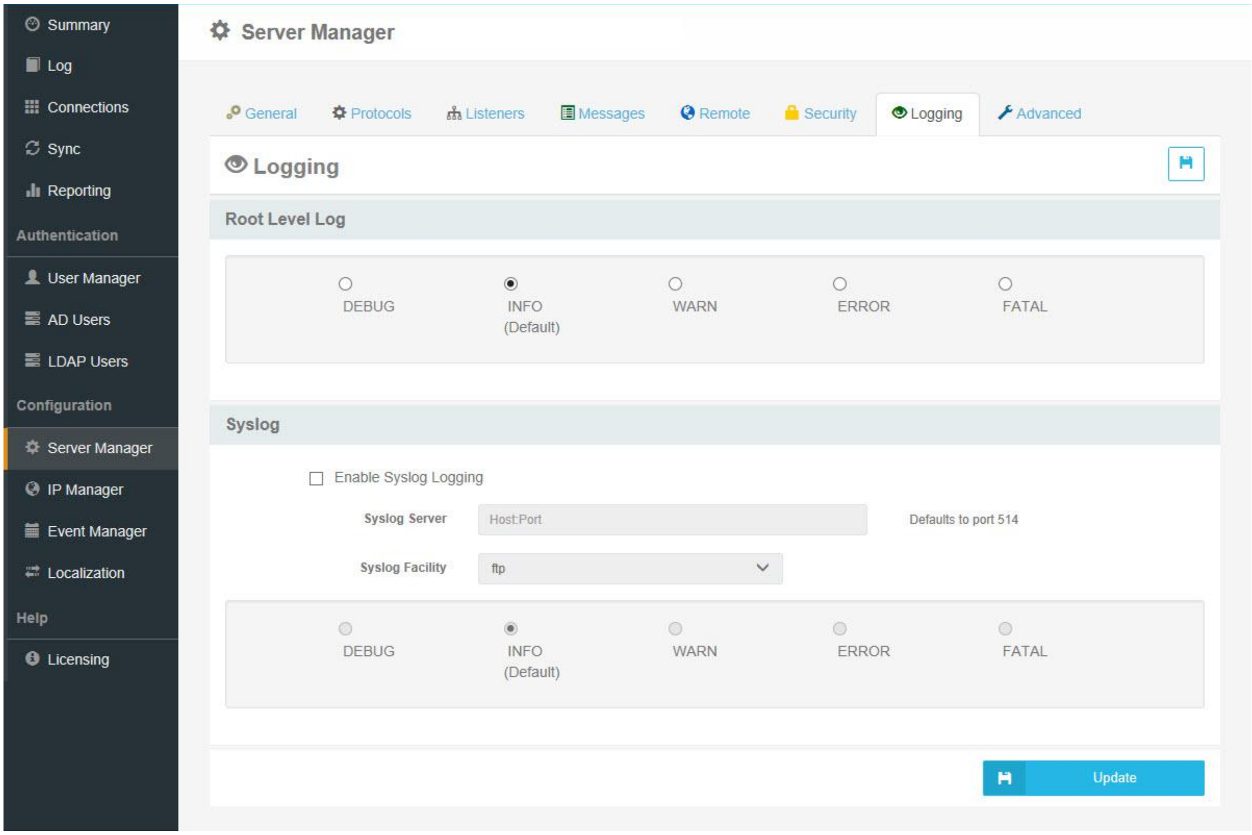
In addition to the file-based log, Cerberus also displays the current log output to the graphical user interface while the server is running. Options for the screen-based logging can be controlled through the Logging settings tab of the Server Manager.

21.4.1 Root Log Level

Controls the root log level for all log appenders. All log appenders inherit the root log level as their lowest threshold. The default level is INFO.

Log appenders can be set at a higher log level threshold than the root logger, but they cannot be set at a lower level. For example, if the Syslog appender is set to DEBUG, but the root log level is set to INFO, the Syslog will still only write out log information at the INFO level.

The DEBUG level is for troubleshooting, and the root log level should not be left at this level for regular production use because of the excessive logging produced.



Logging

Root Level Log

○ DEBUG ● INFO (Default) ○ WARN ○ ERROR ○ FATAL

Syslog

□ Enable Syslog Logging

Syslog Server: Host:Port — Defaults to port 514

Syslog Facility: ftp

○ DEBUG ● INFO (Default) ○ WARN ○ ERROR ○ FATAL

21.4.2 Syslog Support

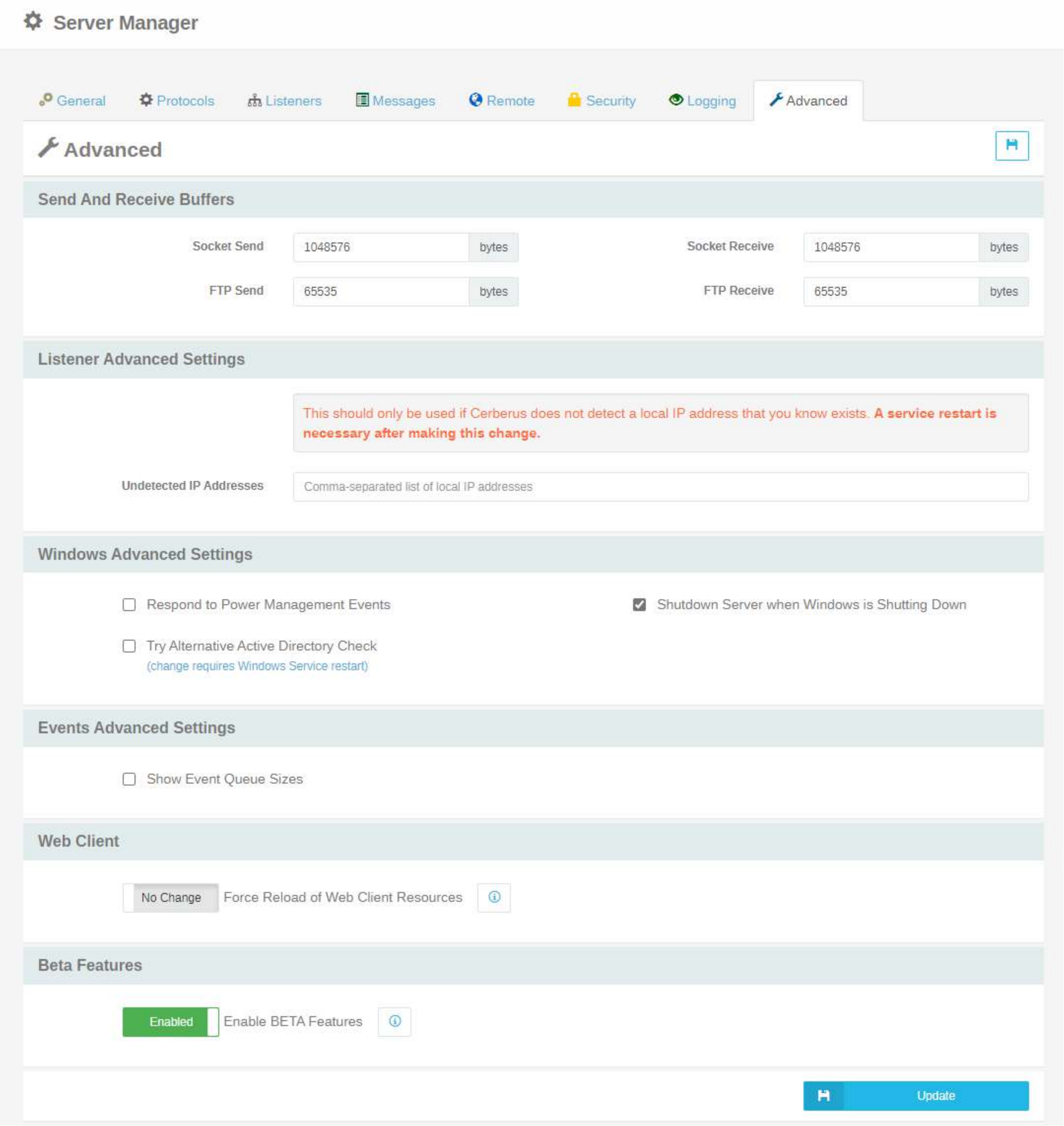
Cerberus FTP Server supports Syslog integration. Control Syslog settings from this page.

Enable Syslog logging	Enable syslog logging
Syslog Host	The address of the machine hosting the syslog server.
Syslog Facility	The syslog facility value that should be associated with the syslog events.

22.0 Advanced Settings

22.1 Configuring Advanced Settings

The advanced settings page contains options for network buffers, listener, windows and events advanced settings as well as Web Client reload and Beta features settings.



Server Manager

Advanced

Send And Receive Buffers

Socket Send 1048576 bytes

Socket Receive 1048576 bytes

FTP Send 65535 bytes

FTP Receive 65535 bytes

Listener Advanced Settings

This should only be used if Cerberus does not detect a local IP address that you know exists. **A service restart is necessary after making this change.**

Undetected IP Addresses: Comma-separated list of local IP addresses

Windows Advanced Settings

☐ Respond to Power Management Events

☑ Shutdown Server when Windows is Shutting Down

☐ Try Alternative Active Directory Check
(change requires Windows Service restart)

Events Advanced Settings

☐ Show Event Queue Sizes

Web Client

No Change Force Reload of Web Client Resources

Beta Features

Enabled Enable BETA Features

Advanced page of the Server Manager

22.1.1 Send and Receive Buffers

These settings control the size of the buffers used for data transfers. Cerberus will read and write packets of this size for send and receive operations.

Socket Send	Specifies the total per-socket buffer space reserved for sends. This value is in bytes.
Socket Receive	Specifies the total per-socket buffer space reserved for receives. This value is in bytes.
FTP Send	The size of the send buffer used for reading file data and writing data to the network for unencrypted FTP transfers, in bytes
FTP Receive	The size of the receive buffer for reading network data and writing data to files for unencrypted FTP transfers, in bytes.

22.1.2 Listener Advanced Settings

These are advanced settings for Interfaces.

Undetected IP Address	If Cerberus fails to detect an IP Address, you can input the undetected IP address here. A service restart is required after adding the IP.

22.1.3 Advanced Windows Settings

These settings are only available on Windows NT and higher.

Respond to power management events	If enabled, Cerberus will attempt to gracefully shut down and startup in response to power suspend and resume events. May allow more graceful recovery from suspending and resuming the system.
Shutdown Server when Windows is shutting down	Detects operating system shutdown or restarts and tries to gracefully terminate all connections and ensure all server settings are saved.
Try Alternative Active Directory Group Check	Uses an older method of checking an AD Active Directory user's group information. This fallback method may work in some situations when Cerberus cannot reliably detect all of the groups an AD user is a direct member of.

22.1.4 Events Advanced Settings

These are advanced settings for Interfaces.

Show Event Queue Sizes	Show Event Queue Sizes

22.1.5 Web Client

These are advanced settings for Interfaces.

Force Reload of Web Client Resources	Cache Busting If you modify any Web Client templates or JavaScript outside of Cerberus, you can force users' browsers to refresh these resources and not use previously cached versions. This control is a one-time action; after saving, this value will automatically be turned off again. **Once saved, this change is immediate.**

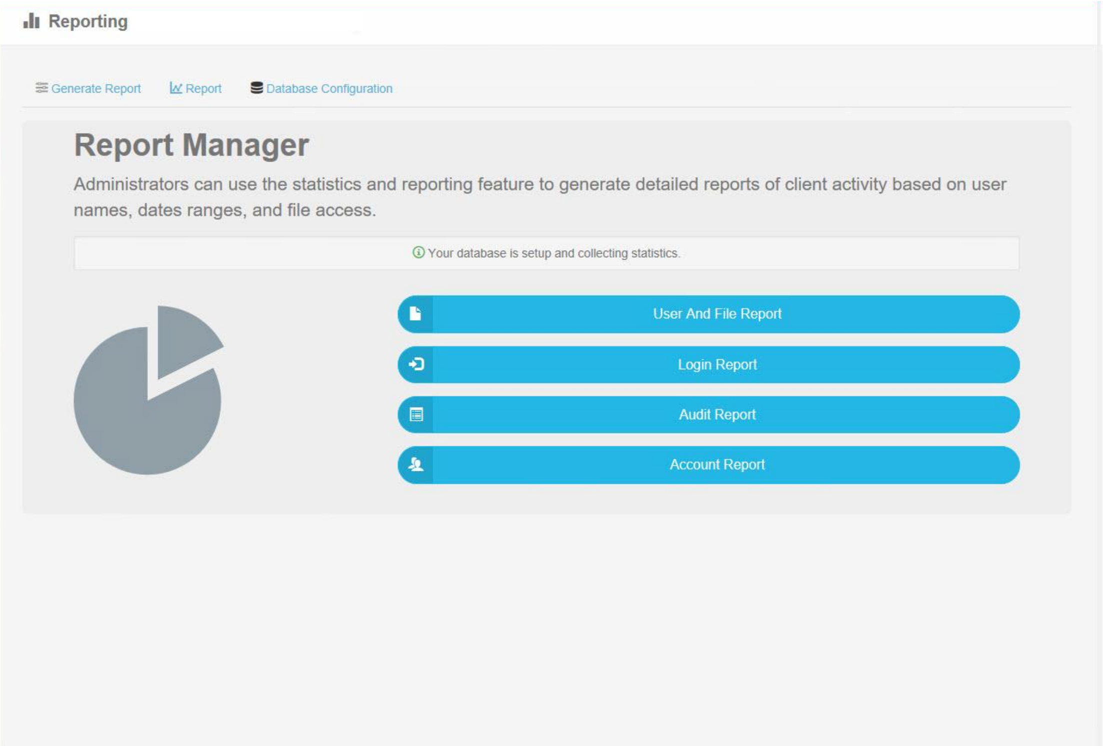
22.1.6 Beta Features

These are advanced settings for Interfaces.

Enable Beta Features	Toggle this option to enable experimental BETA features. NOTE: There is no official support for these features. We do recommend signing up for our BETA release mailing list, and we welcome any and all feedback about our new, experimental features.

23.0 Report Manager

Administrators can use the statistics and reporting feature to generate detailed reports of client activity based on user names, date ranges, and file access.



Report Manager

Administrators can use the statistics and reporting feature to generate detailed reports of client activity based on user names, dates ranges, and file access.

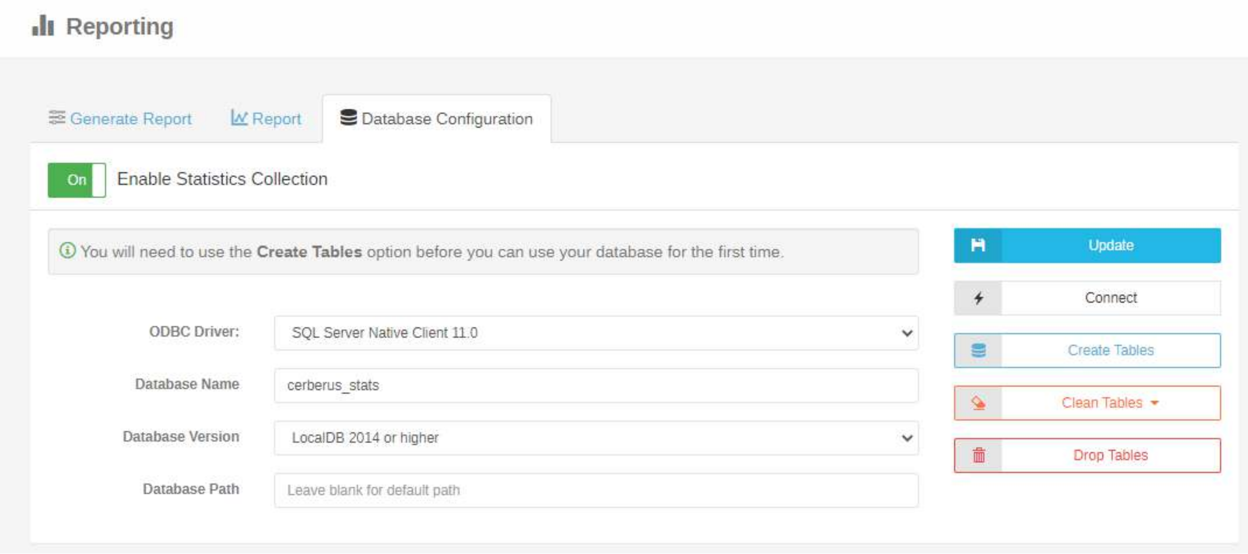
Your database is setup and collecting statistics.

The Report Manager Overview Page

Cerberus FTP Server supports collecting and reporting detailed session and file access statistics using an ODBC-compliant database. A database connection must be configured in Cerberus before the server will begin collecting statistics. The reporting database connection will also be used by the Reporting Manager for generating reports.

The following databases are currently supported (others may work with the appropriate ODBC driver installed):

- **Microsoft SQL Server 2014 Enterprise and higher**
- **Microsoft SQL Express LocalDB 2014 and higher**
- **Microsoft Azure SQL Server**
- **MySQL Server 5.2 and higher**
- **PostgreSQL version 10 and higher**



On Enable Statistics Collection

ⓘ You will need to use the **Create Tables** option before you can use your database for the first time.

ODBC Driver: SQL Server Native Client 11.0

Database Name: cerberus_stats

Database Version: LocalDB 2014 or higher

Database Path: Leave blank for default path

Database Configuration page for Statistics Collection

23.1 Installing Microsoft SQL Server 2019 LocalDB

The quickest and easiest database option is Microsoft Server 2019 LocalDB. LocalDB is a lightweight, embedded database option from Microsoft that is suitable for local, low-utilization database traffic. It has a relatively small footprint and installs quickly. You will also need the Microsoft SQL Server Native Client 11.0 ODBC driver for connecting to LocalDB. Links to both products are below.

To download and install LocalDB:

SQL Server Express LocalDB

SQL Server Native Client 11.0 ODBC

After installing SQL Server LocalDB and the SQL Server Native Client, you can go to the Server Manager's Reporting page and select the **SQL Server Native Client 11.0** driver.

23.2 Selecting a Database

If you are setting up a new database connection for the first time you will need to enable statistics collection and select a database ODBC driver. You can accomplish these tasks using the steps below:

1. Open the **Server Manager** and go to the **Reporting** page.
2. Open the Drivers select box and select the ODBC database driver appropriate for your database type (i.e. ODBC Driver 17.1 for SQL Server for a remote connection to Microsoft SQL Server).

> For Microsoft SQL Server installations other than LocalDB, we recommend downloading and installing the **Microsoft ODBC Driver 17.0 for SQL Server**. Some reporting features may not work with the default SQL Server ODBC driver installed on most machines.

3. The connection parameters available for your driver type will now appear and must be filled in.
4. After filling in the ODBC driver connection parameters, press the Connect button to test your connection.
5. If there are no errors after pressing the **Connect** button, press the **Create Tables** button to create the necessary database tables for Cerberus to write to the database.
6. If **Create Tables** was successful then you are finished setting up your connection.
7. Check the "**Enable Statistics Collection**" checkbox.

The **Connect** button will test that Cerberus can establish a connection to the database, and run a quick search for the necessary Cerberus statistics tables. If this is the first time connecting to the database, and the tables do not exist, click the **Create Tables** button to allow Cerberus to try to create the necessary tables on the database.

If you are using a database that requires a username and password, the user account must have permission to create a database, and tables in the database. Statistics collection and reporting will not work if the user account does not have create database and create table permissions.

Once you have verified a working database connection, and that the database and tables exist, select the **Enable Statistics Collection** checkbox to enable statistics collection.

23.3 Database Backup and Restore

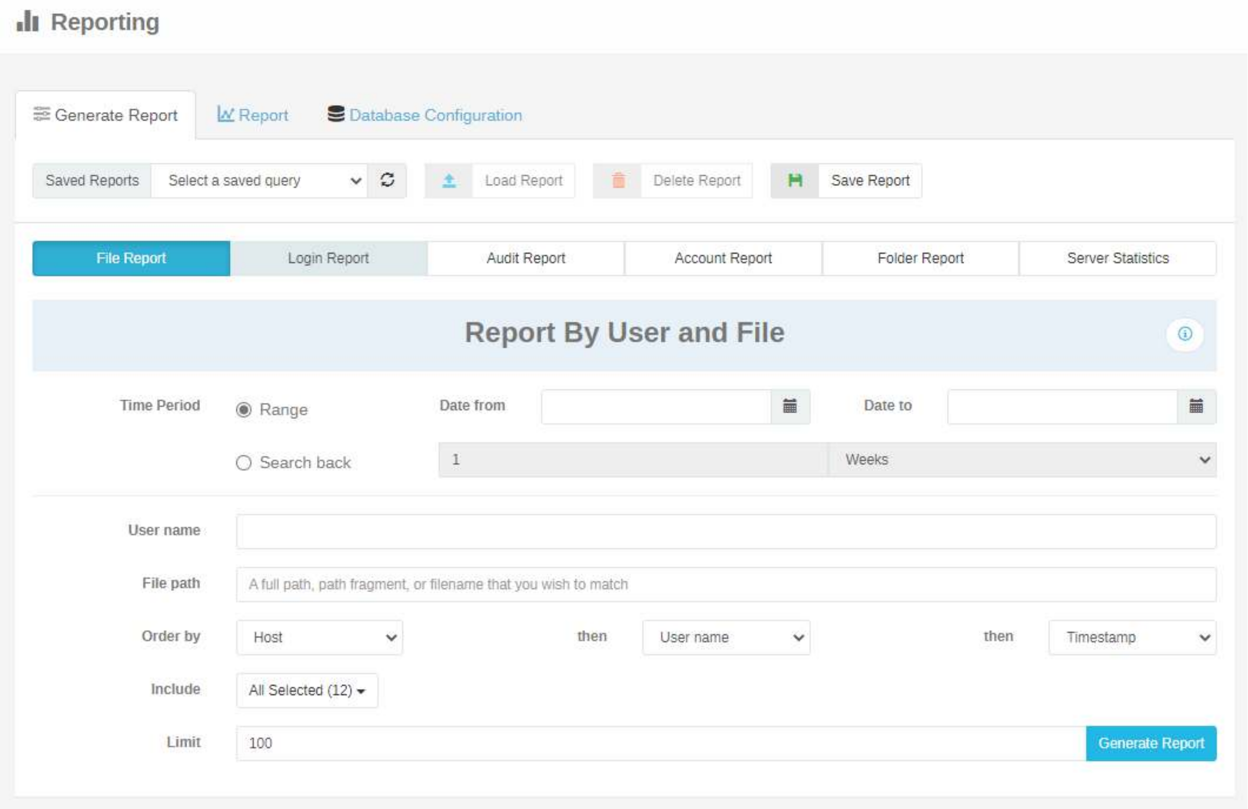
The Backup and Restore buttons are currently only supported when connecting to Microsoft SQL Server databases. The buttons will be disabled when any other database type is selected. The buttons allow downloading a copy of a local database, and later restoring it. Note, that the database will be saved to the local machine where the database is running. If you click the Backup button for a remote database, the database will be saved to the selected path *on the remote server running the database*. The same goes for restoring a database. The database must be on the local machine.

Non-SQL Server databases should be backed up using whatever backup system is native to that database.

23.4 Generating a Report

Administrators can use the statistics and reporting features in Cerberus FTP Server Enterprise edition to generate detailed reports of client activity based on user names, date ranges, and file access. In addition to client activity, the administrator can also generate native account reports indicating account creation and last login dates.

NOTE: Using the Report Manager requires that a report database be configured.



Time Period: ● Range — Date from: — Date to:

○ Search back: 1 Weeks

User name:

File path: A full path, path fragment, or filename that you wish to match

Order by: Host then User name then Timestamp

Include: All Selected (12)

Limit: 100

Cerberus FTP Server Generate Report Page

An administrator can generate three types of reports:

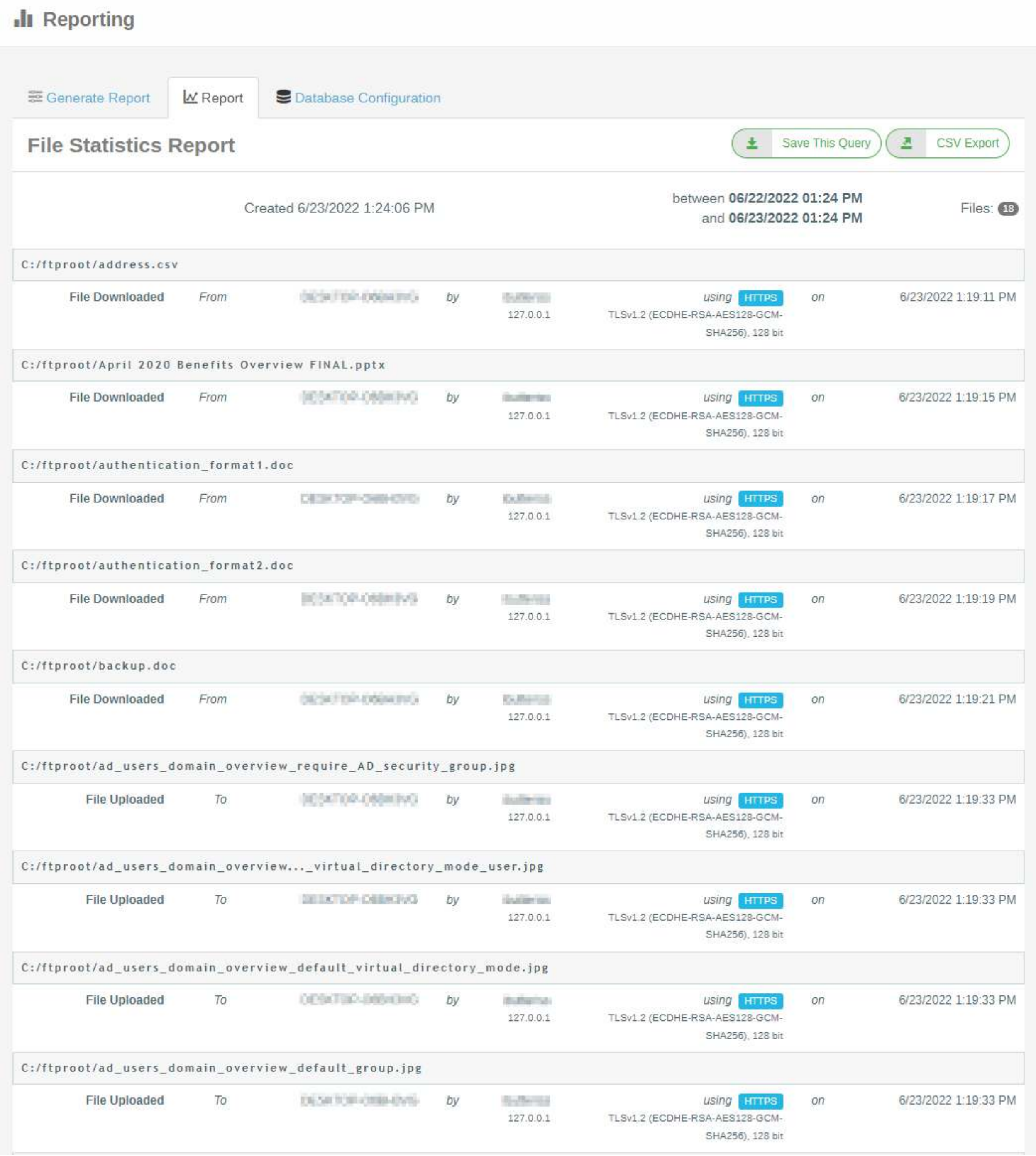
- Login Sessions
- File Access
- User Account Status

23.4.1 File Report

File access reports can be filtered by file name, timestamp, user name, and host. The file reports contain important information about a file transfer, including:

- The full local path of the file
- The type of file operation performed on the file (upload, download, rename, delete, copy)
- The user that accessed the file
- The IP address the user accessed the file from
- The date and time the access occurred
- The protocol used to access the file (FTP, FTPS, SSH SFTP, HTTP, or HTTPS)
- The type of encryption used, if any, to perform the file operation

A date range can be specified, or use the ‘Search Back’ function to search back a defined number of hours, days, weeks, months or years



File Statistics Report

Created 6/23/2022 1:24:06 PM

between **06/22/2022 01:24 PM** and **06/23/2022 01:24 PM**

Files: 18

C:/ftproot/address.csv

File Downloaded *From* [redacted] *by* [redacted] 127.0.0.1 *using* HTTPS TLSv1.2 (ECDHE-RSA-AES128-GCM-SHA256), 128 bit *on* 6/23/2022 1:19:11 PM

C:/ftproot/April 2020 Benefits Overview FINAL.pptx

File Downloaded *From* [redacted] *by* [redacted] 127.0.0.1 *using* HTTPS TLSv1.2 (ECDHE-RSA-AES128-GCM-SHA256), 128 bit *on* 6/23/2022 1:19:15 PM

C:/ftproot/authentication_format1.doc

File Downloaded *From* [redacted] *by* [redacted] 127.0.0.1 *using* HTTPS TLSv1.2 (ECDHE-RSA-AES128-GCM-SHA256), 128 bit *on* 6/23/2022 1:19:17 PM

C:/ftproot/authentication_format2.doc

File Downloaded *From* [redacted] *by* [redacted] 127.0.0.1 *using* HTTPS TLSv1.2 (ECDHE-RSA-AES128-GCM-SHA256), 128 bit *on* 6/23/2022 1:19:19 PM

C:/ftproot/backup.doc

File Downloaded *From* [redacted] *by* [redacted] 127.0.0.1 *using* HTTPS TLSv1.2 (ECDHE-RSA-AES128-GCM-SHA256), 128 bit *on* 6/23/2022 1:19:21 PM

C:/ftproot/ad_users_domain_overview_require_AD_security_group.jpg

File Uploaded *To* [redacted] *by* [redacted] 127.0.0.1 *using* HTTPS TLSv1.2 (ECDHE-RSA-AES128-GCM-SHA256), 128 bit *on* 6/23/2022 1:19:33 PM

C:/ftproot/ad_users_domain_overview..._virtual_directory_mode_user.jpg

File Uploaded *To* [redacted] *by* [redacted] 127.0.0.1 *using* HTTPS TLSv1.2 (ECDHE-RSA-AES128-GCM-SHA256), 128 bit *on* 6/23/2022 1:19:33 PM

C:/ftproot/ad_users_domain_overview_default_virtual_directory_mode.jpg

File Uploaded *To* [redacted] *by* [redacted] 127.0.0.1 *using* HTTPS TLSv1.2 (ECDHE-RSA-AES128-GCM-SHA256), 128 bit *on* 6/23/2022 1:19:33 PM

C:/ftproot/ad_users_domain_overview_default_group.jpg

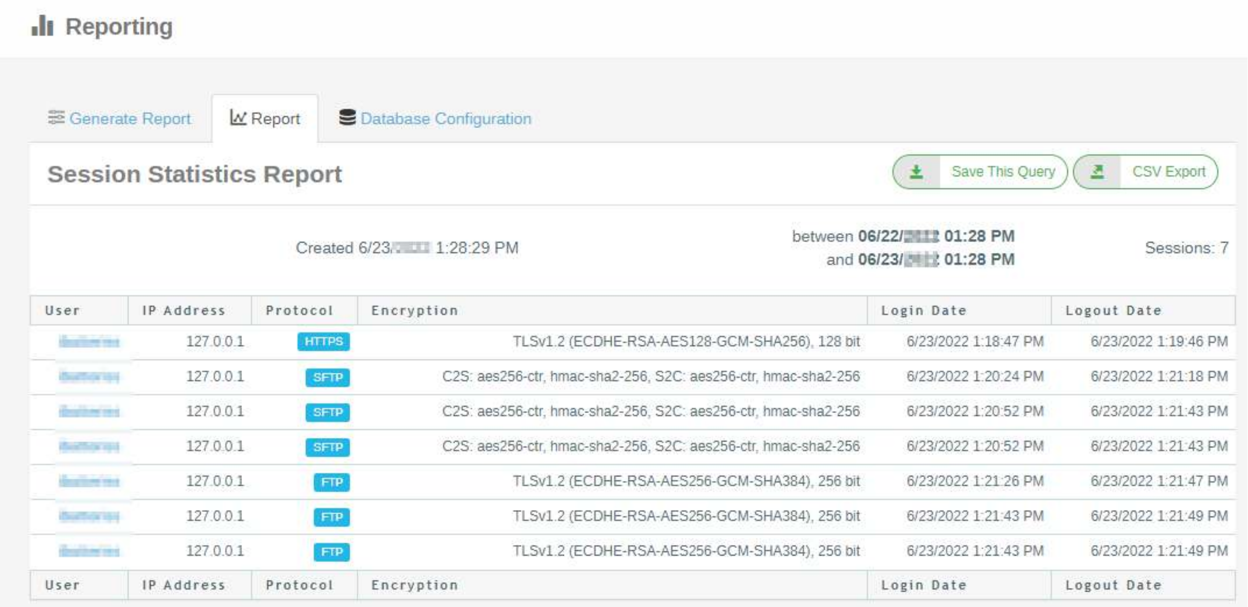
File Uploaded *To* [redacted] *by* [redacted] 127.0.0.1 *using* HTTPS TLSv1.2 (ECDHE-RSA-AES128-GCM-SHA256), 128 bit *on* 6/23/2022 1:19:33 PM

The administrator can also use the Include feature to decide what type of file activity to include in the report (downloads, uploads, file renames/moves, public file shares, etc.).

23.4.2 Login Report

The Login Report displays a list of user logins for the time period specified. File access reporting can be filtered by host, username, and date and time. The login reports contain important information about user sessions, including

The username
The IP address the user logged in from
The protocol used (FTP, FTPS, SSH SFTP, HTTP, or HTTPS)
The type of encryption used for the session
The Login date and time
The Logout date and time



Session Statistics Report

Created 6/23/[redacted] 1:28:29 PM

between **06/22/[redacted] 01:28 PM**
and **06/23/[redacted] 01:28 PM**

Sessions: 7

User	IP Address	Protocol	Encryption	Login Date	Logout Date
[redacted]	127.0.0.1	HTTPS	TLSv1.2 (ECDHE-RSA-AES128-GCM-SHA256), 128 bit	6/23/2022 1:18:47 PM	6/23/2022 1:19:46 PM
[redacted]	127.0.0.1	SFTP	C2S: aes256-ctr, hmac-sha2-256, S2C: aes256-ctr, hmac-sha2-256	6/23/2022 1:20:24 PM	6/23/2022 1:21:18 PM
[redacted]	127.0.0.1	SFTP	C2S: aes256-ctr, hmac-sha2-256, S2C: aes256-ctr, hmac-sha2-256	6/23/2022 1:20:52 PM	6/23/2022 1:21:43 PM
[redacted]	127.0.0.1	SFTP	C2S: aes256-ctr, hmac-sha2-256, S2C: aes256-ctr, hmac-sha2-256	6/23/2022 1:20:52 PM	6/23/2022 1:21:43 PM
[redacted]	127.0.0.1	FTP	TLSv1.2 (ECDHE-RSA-AES256-GCM-SHA384), 256 bit	6/23/2022 1:21:26 PM	6/23/2022 1:21:47 PM
[redacted]	127.0.0.1	FTP	TLSv1.2 (ECDHE-RSA-AES256-GCM-SHA384), 256 bit	6/23/2022 1:21:43 PM	6/23/2022 1:21:49 PM
[redacted]	127.0.0.1	FTP	TLSv1.2 (ECDHE-RSA-AES256-GCM-SHA384), 256 bit	6/23/2022 1:21:43 PM	6/23/2022 1:21:49 PM
User	IP Address	Protocol	Encryption	Login Date	Logout Date

Cerberus FTP Server Session Statistics Report, viewed via the Web Admin Portal, Filtered by User

The administrator can also use the Include feature to decide what type of file activity to include in the report (downloads, uploads, file renames/moves, public file shares, etc.).

23.4.3 Audit Report

The admin audit report provides a list of server activities undertaken by Cerberus administration users.

23.4.4 Account Report

The account status report provides a report on all native Cerberus user accounts as well as Active Directory or LDAP users if you have integrated Cerberus with AD/LDAP.

The report lists

- User
- Any Primary Groups the user is a member of (As a member of a Primary Group, the user's settings are overridden with the settings and virtual directory access set in the group)
- Any Secondary Groups the user is a member of (As a member of Secondary Group, the user is granted access to the virtual directories assigned to the group in addition to their own, but the user retains their own settings and virtual directory access)
- Virtual directories assigned to the user and their permissions
- Disabled status (An 'x' will appear in this column for disabled users)
- Date when password last changed ('Anonymous' will appear in this column if the user has been set to not require a password)
- Date when the password expires if applicable (or 'Never')
- Date when the user was created
- Date of last login ('Unknown' means the user has never logged in)

23.4.5 Folder Report

The Folder report provides a report on folders that have been assigned to Cerberus and AD/LDAP users.

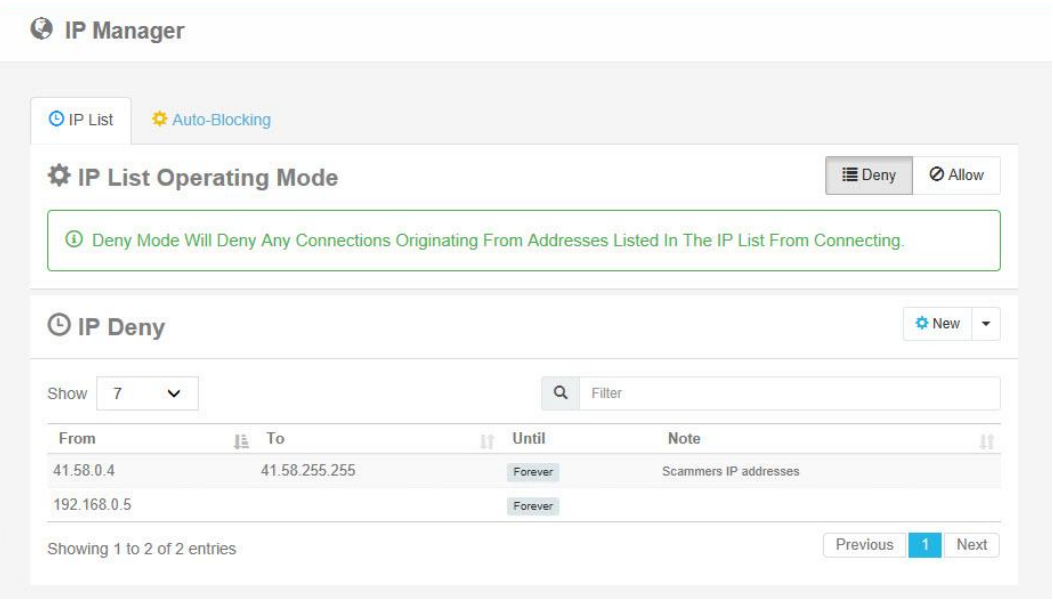
The report lists

- Folder path
- Folder Access (Lists the user accounts that have access to the folder. The users are clickable. If you click on a user, you are taken to their user account

24.0 The IP Manager

24.1 General Settings

The Cerberus FTP Server IP Manager allows an administrator to selectively allow or deny access to the FTP server based upon IP address. The IP manager functions in one of two policy modes, either denying any IP addresses listed from logging into Cerberus FTP Server (functioning as a Deny list), or only allowing IP addresses listed to log in (an Allow list). The policy mode is controlled by a radio button at the bottom of the **General** tab page.



IP List Operating Mode

Deny Mode Will Deny Any Connections Originating From Addresses Listed In The IP List From Connecting.

IP Deny

Show 7

Filter

From	To	Until	Note
41.58.0.4	41.58.255.255	Forever	Scammers IP addresses
192.168.0.5		Forever	

Showing 1 to 2 of 2 entries

General page of the IP Manager

The IP list shows the IP address or IP address range and how long that address or address range is blocked for. Possible options for block time are "Forever" (Deny mode), "Never" (Allow mode), or a date/time value.

If a date/time value is present, the IP address or IP address range is blocked from connecting until that date/time has elapsed (Deny or Allow mode). You can change how long an IP address entry is blocked for by right-clicking on that IP entry and selecting "Change Time" from the menu that appears.

24.1.1 Adding a single IP address to the IP manager policy

IP addresses can be managed individually, or whole ranges of addresses can be affected by the current policy. To add a single address to the current policy, make sure the "Assign a range of addresses" checkbox is unselected. Then, enter the IP address you wish to add to the first IP address box. Finally, click the "Add" button immediately below the IP address box.

24.1.2 Adding a range of IP addresses to the IP manager policy

To add a range of addresses, first ensure the "Assign a range of addresses" checkbox is selected. Then, enter the beginning IP address in the "IP From" box and the ending IP address in the "IP To" box. The range will be interpreted as a contiguous range of addresses to block or allow. Finally, click the **Add** button immediately below the IP address box.

24.1.2.1 CIDR Support

You can also enter a range of IP addresses in CIDR notation using the CIDR edit box. You can enter one CIDR range or multiple CIDR ranges. To enter multiple CIDR ranges, separate each CIDR range with a space or comma. The CIDR address will be converted to a contiguous range and added to the IP Manager list.

24.1.3 Deleting an IP addresses from the current policy

To delete either an IP address or a range of IP addresses from the current policy, select the item from the "IP Addresses" list view box. Once selected, press the Delete button. You can also select and delete multiple items at once from the IP manager by ctrl or shift-clicking multiple items in the list box. NOTE: You can also delete an IP address or a range of IP addresses by right-clicking on the selected IP and selecting "Delete" from the menu that appears.

24.1.4 Searching for an IP Address

You can use the "Find" button at the top of the IP list box to search for an IP address in the list box. The "Find" button will select the first IP address or range of IP addresses containing the IP address you are searching for.

24.2 The "Auto-Blocking" page

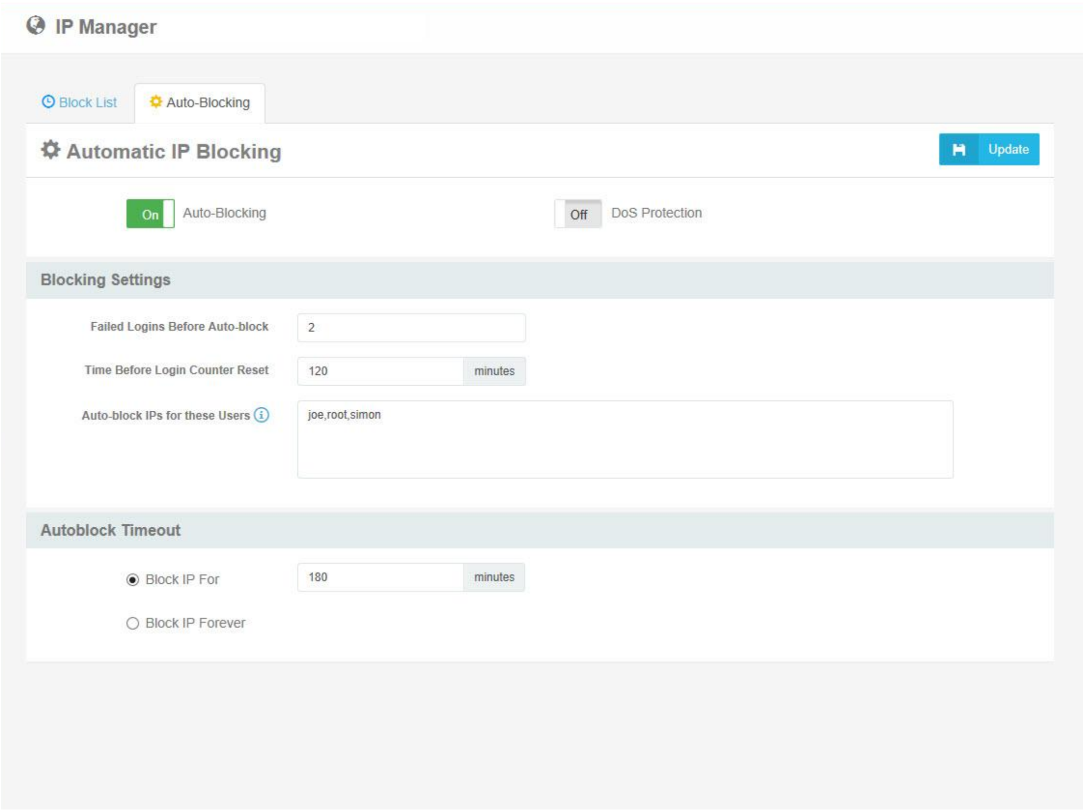
The other use for the IP manager is the ability to configure an auto-blocking policy for the FTP server. Administrators can use the auto-blocking policy to help prevent DoS (Denial of Service) and brute force password guessing. If the auto-blocking policy is enabled, a user that continually fails to log into the server will be blocked from trying after a certain number of failed attempts. The number of failed attempts and the length of time the IP address will be blocked from attempting to log in can be configured from the "Auto-Blocking" page.

When **Enable Auto-Blocking** is enabled a failed attempt is logged whenever a user enters an incorrect password or tries to log in with an invalid username. If **Enable DoS Protection** is selected then any attempt to connect to the server will be counted towards auto-blocking, even if the connection doesn't attempt to authenticate. This can help prevent DoS attacks that try to tie up connections and overwhelm the server. DoS Protection can also be useful for services continuously probing the server with garbage data attempting to find security vulnerabilities. However, a successful login from an IP address resets the "Failed login attempts" counter to zero for the IP address.

The number of failed login attempts can be configured from the **Pre-Blocked Settings** frame. The **Time before login counter reset** edit control can be used to set the amount of time that must elapse before the **Failed login attempt** counter is reset.

The length of time an address is blocked can be configured using the **Auto-Block Timeout** setting. Select the Forever radio button to block a flagged IP address indefinitely, or select the "Block for X minutes" radio button to set the length of time the address is blocked. Once an address is blocked, the timeout period must elapse before the address is allowed to log in again.

IP addresses that have recently failed logins, but have not yet exceeded the **Failed login attempt** threshold, are displayed in the **IP Addresses being "watched"** list view. You can freely delete an address from the list view. Deleting the address has the effect of resetting the **Failed Login attempt** counter for that address to zero.



On Auto-Blocking

Off DoS Protection

Blocking Settings

Failed Logins Before Auto-block: 2

Time Before Login Counter Reset: 120 minutes

Auto-block IPs for these Users: joe,root,simon

Autoblock Timeout

● Block IP For 180 minutes

○ Block IP Forever

Auto-blocking page of the IP Manager

24.2.1 Immediately Ban these Users

Certain usernames are often tried by automated bots. You can configure Cerberus to automatically block the IP of any connection that attempts to log in using one of these banned usernames.

24.2.2 Differences in Auto-blocking between Allow and Deny mode

How auto-blocking works differs depending upon whether the IP manager is functioning in Deny or Allow mode. If the IP manager is functioning in Deny mode (denying addresses listed in the IP manager), then whenever a connection exceeds the failed login attempt threshold, that connection's IP address is added to the deny list.

Auto-blocking works differently for Allow mode (allowing only addresses listed to login to the server). In Allow mode, whenever a failed login attempt exceeds the failed login threshold, the IP address is either removed from the IP manager's list of allowed IP addresses (if auto-blocking is set to block failed logins forever) or blocked for the Auto-Block Timeout period. The exception is if the IP address is part of a range of IP addresses. If an IP address is part of a range of allowed IP addresses, that range is not deleted.

25.0 The Event Manager

25.1 About Event Rules

Available in **Cerberus FTP Server Enterprise edition**, the Event Manager allows an administrator to configure email notifications, perform file operation or batch file actions, and carry out certain server operations based on server events.

Event rules are based on the simple premise that a logged event occurs that triggers an action. There are several different rule types, and for each rule type, there is a corresponding event that can trigger that rule.

You can further restrict a rule by specifying additional conditions on the event that must exist before the rule's actions are taken.

For example, suppose you have a folder into which customers can upload files. You can set up an event rule that monitors that folder, and when someone uploads a file into that folder, the rule moves the file to another folder, and then sends an email to an administrator informing them that a file has been moved.

You can also set up a rule that only moves particular files. For example, you can configure the rule to move only the files that end in .zip, or you can route particular files to different folders.

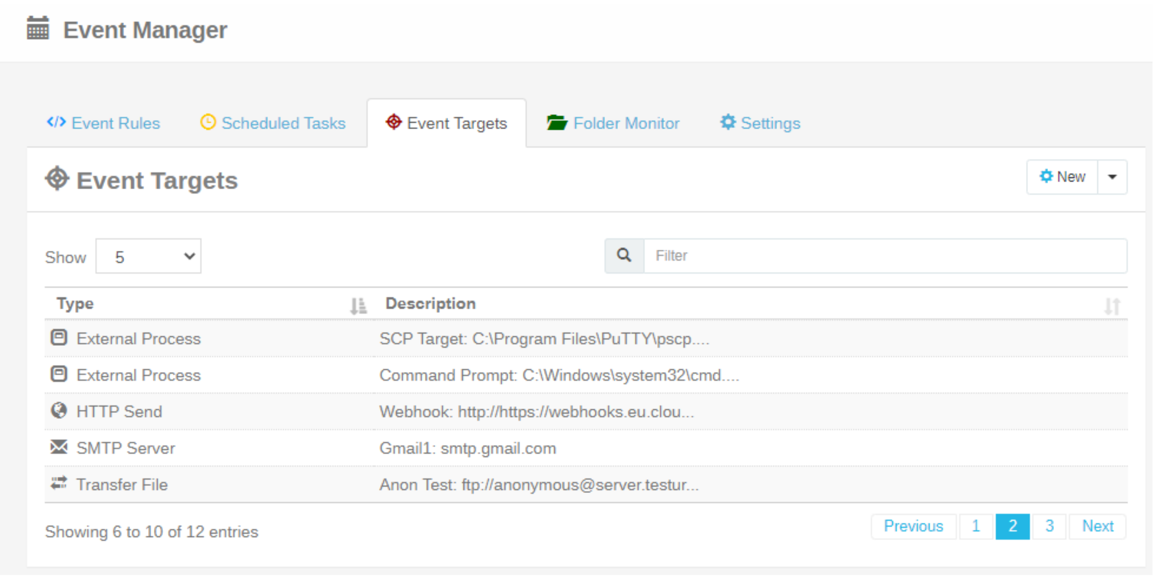
An event rule consists of a triggering event (e.g. a File Transfer), any optional conditions affecting that event (e.g. uploaded by a specific user), and the resulting actions that are carried out (e.g. moving the file, or sending an email to an administrator). You can modify your rules any time in the event manager.

25.2 The Event Targets page

Allows an administrator to add email servers, executable files, and HTTP endpoints as event targets. Many of the actions you can invoke as part of an event rule, or scheduled task, require an event target. For example, the "Email someone" action requires an email server, and the "Launch an executable" action requires the file path to the executable file. Those event targets can be defined here.

There are also certain server actions that can require an SMTP server, like public file sharing, or password expiration notification. You will first need to add at least one SMTP server here before the server can carry out those operations.

25.2.1 Available Target Types



Type	Description
External Process	SCP Target: C:\Program Files\PuTTY\pscp....
External Process	Command Prompt: C:\Windows\system32\cmd....
HTTP Send	Webhook: http://https://webhooks.eu.clou...
SMTP Server	Gmail1: smtp.gmail.com
Transfer File	Anon Test: ftp://anonymous@server.testur...

Showing 6 to 10 of 12 entries

There are four different types of event targets you can add for use in event rules and scheduled tasks.

25.2.1.1 SMTP Server Targets

You can add SMTP servers using the SMTP Server Target box. Cerberus currently supports the SMTP protocol, including SMTP with SSL encryption and STARTTLS. If your server requires it, SMTP server credentials can be configured by selecting the **SMTP Authentication** checkbox.

25.2.1.2 Executable Targets

Cerberus can be configured to launch a .exe, .bat, or .com file as an action for any event. Just select a file path and press the "**Update**" button to make an executable target available for selection when adding and editing rules. Command-line options for the executable are specified on a per-action basis from the rule editing page.

25.2.1.3 HTTP Post Targets

This option allows you to specify a URL that will receive an HTTP or HTTPS POST containing all of the rule's variables. Variables are included in a POST request using **application/x-www-form-urlencoded** encoding.

25.2.1.4 Transfer File Targets

The Transfer File Target in Event Manager allows the transfer of files to or from other servers via SFTP, FTP, FTPS, or HTTP/S PUT or GET in Cerberus FTP Server. This feature brings integrated functionality making it easy to send or receive a file and capture any messages directly within Cerberus.

25.2.2 Adding a New Event Target

Press the **New** button at the top of the Event Targets page. A dialog will prompt you for the type of target you wish to add.

25.2.3 Modifying an existing event target.

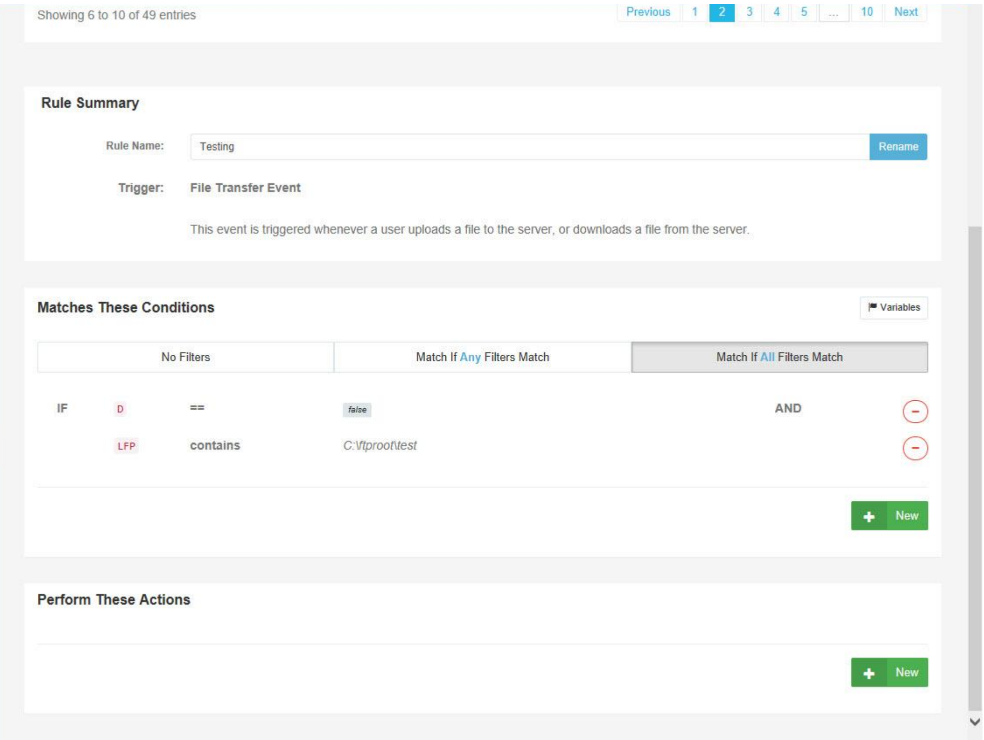
Select the event target in the Targets list. An edit section for that target will appear below the event targets list. Press the **Update** button after making your changes to save those settings to the server.

25.3 Event Rules

The Rules page provides an overview of all of the rules you have added. From this page, you can Add, Delete, Clone, or Enable and Disable a rule.

You can enable or disable a rule from this page. Whenever a rule is disabled, that rule is no longer checked whenever the system generates an event that would normally trigger the rule.

Selecting a rule from the Event Rules table will open up a summary of the rule for editing.



Showing 6 to 10 of 49 entries

Rule Summary

Rule Name: Testing

Trigger: File Transfer Event

This event is triggered whenever a user uploads a file to the server, or downloads a file from the server.

Matches These Conditions

IF D == false AND

LFP contains C:\ftproot\test

Perform These Actions

Rule Editing in the Event Manager

25.3.1 Adding a New Rule or Editing an Existing Rule

25.3.1.1 To add a new rule:

1. Go to the **Event Rules** page of the Event Manager
2. Click the **New** button. The **Add a New Rule** dialog will appear.
3. Select the **Rule Type** for your new rule option. The rule type will determine what server event triggers this rule.
4. Enter a name for your rule in the **Rule Name** edit box.
5. Press the **Add New Rule** button on the Add A New Rule dialog to save and add the new Event Rule. The event rule will be selected and ready for editing on the Event Rules page.

25.3.1.2 Available Event Rule Types

A rule is defined by the type of event that triggers it. Each rule has a single event type associated with it. When that event occurs, any rules associated with that event type are triggered. The following rule event types are available:

File Transfer Event	This event is triggered whenever a user uploads a file to the server or downloads a file from the server through an authenticated Cerberus account. This event is not generated for public share file downloads. There is a separate event for public file downloads.
IP Blocked Event	This event is triggered whenever the server adds an IP address to the block list.
User Account Blocked Event	This event is triggered whenever a user account is locked out because of a policy violation (too many failed login attempts).
User Disable Date Elapsed	This event is triggered whenever a user account is disabled because the disable date for the account has elapsed, or because the account has exceeded the last login time threshold.
Account Password Expiring Event	This event is triggered when an account password is set to expire. The number of days before expiration that this event is sent is based upon the password expiration policy settings.
New Account Request Event	This event is triggered when a new account request is submitted through the HTTP/S web client.
Login Event	This event is triggered whenever a user attempts to log into the server.

Logoff Event	This event is triggered whenever a user attempts to log in to the server.
Directory Created Event	This event is triggered whenever a user creates a directory on the server.
File Deleted Event	This event is triggered whenever a user deletes a file or folder on the server.
File Move/Copy Event	This event is triggered when a file or directory is moved or copied by a user.
Upgrade Available Event	This event is triggered whenever the server detects that a new version of Cerberus FTP Server is available.
Public File Share Event	This event is triggered whenever a public file share link is generated for a file by a user. A public file share link is generated whenever a user uses the Share or Email button in the HTTPS web client to generate a new public link.
Public File Download Event	This event is triggered whenever a publicly shared file is downloaded from the server. This event will not be generated for a file download by an authenticated (logged in) Cerberus user.
Backup Server Synchronized	This event is triggered after the server attempts to synchronize settings to a backup server.

25.4.2.2 To edit an existing rule:

1. Go to the **Event Rules** page of the Event Manager
2. Select the name of the existing rule you wish to edit from the event rules table. The event rule should appear and be ready for editing.

25.3.2 Changing the Name of a Rule

You can change the name of an existing rule by selecting it in the rules table. You can then modify the Rule Name under the Rule Summary section. After entering the new Rule Name, press the Update button attached to the Rule Name text field.

25.3.3 Rule Conditions

You can add a new condition to an event rule by pressing the Create button in the Event Conditions header. The new condition section will appear below the header.

A rule's actions are carried out whenever that rule's event trigger happens. For example, a Login Event rule will be triggered whenever a user logs into the server. Conditions (also called filters) can be placed on rules to further modify if an event matches a rule. For example, a Login Event rule can have a filter placed on it that requires the username of the user logging in to match a specific name, or be in a list of names, before the rule's actions are invoked. There are three modes that influence how conditions or filters are applied.

25.3.3.1 Rule Matching Modes

The three rule matching modes are:

Match All Events	This rule will always be triggered whenever the rule's event occurs.
Match If Any Filters Match	This rule will be triggered whenever the rule's event occurs and if **any** of the conditions listed are fulfilled
Match If All Filters Match	This rule will only be triggered whenever the rule's event occurs and if **all** of the conditions listed are fulfilled

25.3.3.2 Rule Variables

Each event type has specific variables that can be used as part of a condition or action. A rule condition consists of a variable, a comparison operation to perform on that variable, and a set of values to compare the variable to. For example, an IP Blocked event has an **{{IP}}** variable associated with it that contains the IP address that was blocked. You can use the variable in a condition to help decide if the event should trigger the rule.

You can determine what rule variables are available for each event type by looking in the **Rule Variables** combo box.

25.3.3.3 Comparison Operations

A condition is basically a comparison operation of an event variable to a set of values. The comparison operations you can perform are detailed below:

- > (Greater than or Equal To)
- ≥ (Greater than)
- < (Less than)
- ≤ (Less than or Equal To)
- = (Equal To)
- != (Not Equal To)
- Contains
- Does Not Contain
- Starts with
- Ends with
- Regular Express match

Once a comparison operation is selected, you can enter the **values** to compare to. There is a text field labeled “**Values”** below the comparisons select control that you use to enter values to compare the rule variable to. Multiple values can be entered by separating the values with a comma. Each value is checked, and if any are a match then the condition is considered fulfilled (or true).

Press the **+ New button** in the ‘Matches These Conditions’ box to add anew rule condition to the event rule.

Enter the condition parameters in the ‘Creating a New Condition’ box. Once you are done, press the ‘Add’ button to save. The new event condition will appear at the bottom of the Event Conditions section.

25.3.3.4 Deleting an Event Condition

You can delete an existing event condition by pressing the red X button next to the event condition.

25.3.4 Rule Actions

Are the operations the administrator wishes the server to carry out in response to server events that match their rule conditions. Event actions can be of two types:

1. Normal top-level actions that get executed sequentially, or
2. Failure actions that get executed whenever the event action they are associated with fails

Actions are normally executed one after the other, in sequential order. Failure actions are always associated with a top-level action, and only get executed if the action they are associated with fails. The failure action is executed right after the action it is associated with.

Each top-level action has a “**Stop on Failure**” option. If the “Stop on Failure” option is checked, no further actions will be executed for the event rule if the action fails (other than any failure action associated with the top-level action).

25.3.4.1 Adding Rule Actions

When an event matches all of the conditions of a rule then the rule actions are carried out. The current rule actions allow an administrator to

- Send an email message detailing the event that occurred
- Send an email session report of all user activity when a user logs off
- Launch an external process
- Perform a file copy, move, delete or directory create or delete operation
- Perform a user or group delete or disable
- Add a configurable delay before the next action is invoked
- Create a backup file of the server configuration

Each action can have optional parameters such as the email name and address to send a message to, or the ‘path from’ and ‘path to’ for a file move or copy operation. In addition, rule variables can be specified as parameters for the external processes command line or file operation parameters. You can use a rule variable as a parameter and when the rule is actually triggered, the variable's value will be substituted for the variable. You specify variables by enclosing the variable in double brackets, i.e. **{{U}}**.

25.3.4.2 To add a new Action to a Rule:

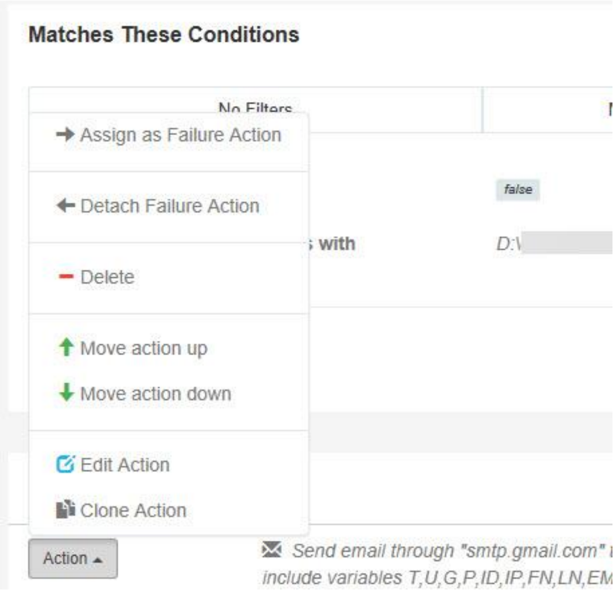
These instructions assume you have selected a rule for editing from the rules tables.

1. Go to the **ACTIONS** header section of the event rule.
2. Press the **Create** button in the ACTIONS header. The new action section will appear below the ACTIONs header.
3. Select an action from the Action drop-down list (i.e., Email someone)
4. Select any secondary actions associated with that action (i.e, an email server for emailing someone)
5. New fields will appear below the Actions drop-down lists based on the action and secondary action selected
6. Fill in the details for that action (i.e., an email address)
7. If you for the rule action list to stop executing if this action fails then select the "**Stop on Failure**" option for the action.
8. Press the **plus (+) button** to add the new action to the rule

The new event action will be added to the bottom of the Actions section. New actions will be added to the bottom of the list and will be executed in the order they appear in the list.

25.3.4.3 Editing an Existing Rule Action

You can edit an existing rule action by selecting the **Action** button to the left of the event action. Selecting the Action button will bring up a menu of available operations you can perform on the event action.



Send email through "smtp.gmail.com"
include variables T,U,G,P,ID,IP,FN,LN,EM

Select the Edit Action button from the menu that appears to have the action selected in the Actions section.

25.3.4.4 Deleting an Existing Rule Action

You can delete an existing rule action by selecting the **Action** button to the left of the event action.

Select the **Delete** button from the menu that appears to have the action deleted from the event rule.

25.3.4.5 Changing the order an action is executed

You change the existing execution order of event actions by selecting the **Action** button of the event action you wish to change.

Select the **Move Action Up** or **Move Action Down** to swap positions with the action above or below the selected action.

25.3.4.6 Creating a Failure Action

Each action can have a failure action associated with it. Failure actions are additional actions that only get executed whenever the action they are associated with fails. For example, you can add an "Email Someone" failure action to an action to email the administrator whenever the top-level action the failure actions is associated with fails. Or, you can try the action a second time as your failure action.

The same action options are available as failure actions as are available for top-level actions.

To create a failure action, create a new action as you normally would for a top-level action. Use the Move Up or Move Down action options to place the new failure action below the top-level action you wish it to be associated with.

Once the action you wish to associate as a failure action is below the top-level action, select the "Assign as Failure Action" option from the Actions button next to the failure action. You will now see the action become indented under the top-level action, and the text "if fail then" appear in front of the failure action.

25.3.4.7 Removing a failure action

Just requires pressing the Action button associated with the failure action's top-level action, and then selecting the "**Detach Failure Action**" option from the menu that appears.

Detaching a failure option from a top-level action will make the failure action a normal top-level action again. You can then move it around, re-assign it as a failure action of another top-level action, or delete it.

25.4 Scheduled Tasks

Please see the section on **adding and editing actions** in the **Event Rules help** section. The process is identical for scheduled tasks.

Scheduled Tasks are similar to event rules. However, rather than being triggered whenever an event like a file upload or directory creation occurs, Scheduled Tasks are time-based, and occur on an admin-defined schedule.

Administrators can configure Scheduled Tasks to occur once, or to repeat every minute, hour, day, week, weekday, month, or year.

You can create and edit Scheduled Tasks on the Scheduled Tasks page of the Event Manager.

25.4.1 Adding a schedule to a Scheduled Task

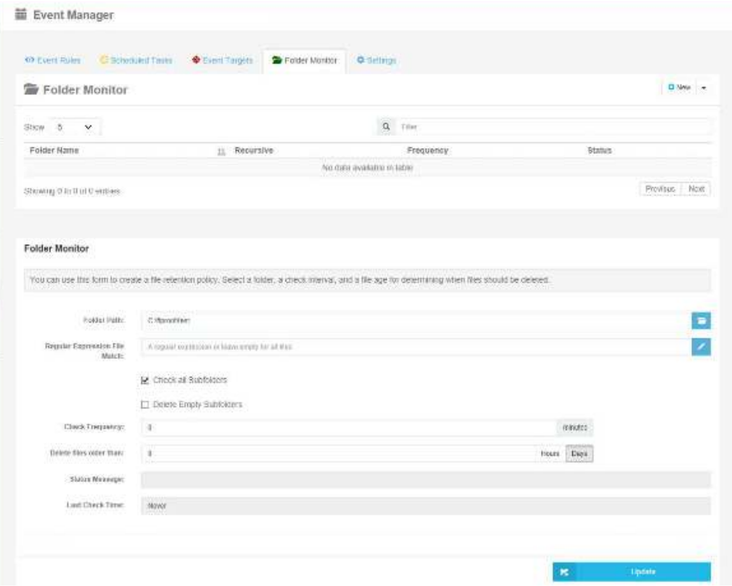
Schedules can be added to event tasks in a similar way that event conditions are added to event rules.

1. Specify a Start Date for the scheduled task. If you do not specify a Start Date then the task will be executed immediately.
2. Select how often you want the task to repeat. You can select a period and frequency. For example, every 5 hours.
3. Press the **plus (+) button** to add the schedule to the task.

25.5 Folder Monitor

The Folder Monitor page allows you to configure the server to monitor a top-level folder and subfolders for setting up a file retention policy.

You can configure a directory and subdirectories to be monitored for files older than a specified time period. The directory will be checked at an administrator-defined interval, and files older than the specified age will be deleted.



Folder Name	Recursive	Frequency	Status
No data available in table			

Showing 0 to 0 of 0 entries

Folder Monitor

You can use this form to create a file retention policy. Select a folder, a check interval, and a file age for determining when files should be deleted.

Folder Path: [illegible]

Regular Expression File Match: A regular expression or leave empty for all files

☑ Check all Subfolders

☐ Delete Empty Subfolders

Check Frequency: 1 minutes

Delete files older than: 1 Hours Days

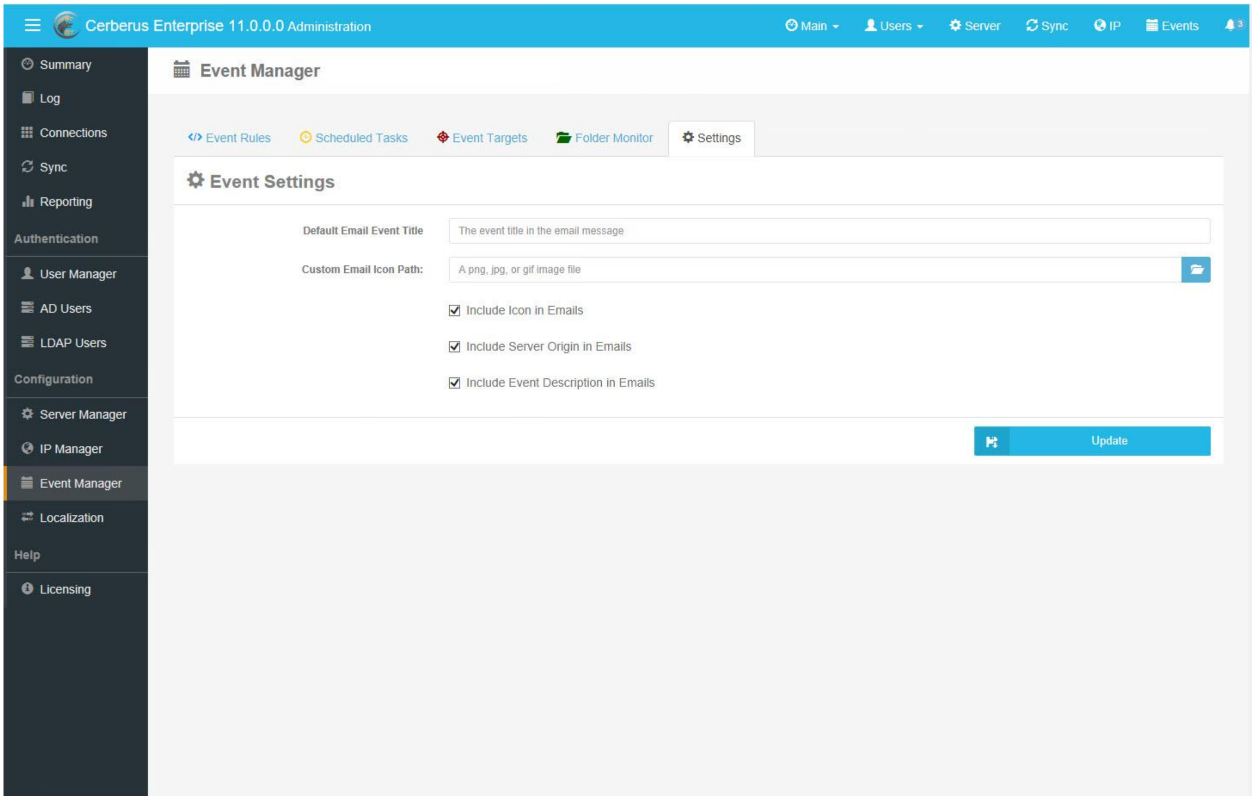
Status Message:

Last Check Time: Never

Folder Monitor for configuring file retention policies

25.6 Other Event Settings

The event settings page allows the administrator to configure settings like the email template logo, whether to include server information in event emails and other global event settings.



Event Manager

Event Settings

Default Email Event Title: The event title in the email message

Custom Email Icon Path: A png, jpg, or gif image file

☑ Include Icon in Emails

☑ Include Server Origin in Emails

☑ Include Event Description in Emails

Default Email Event Title	The email heading title is at the top of each event notification email.
Custom Email Icon Path	Allows the administrator to include their own icon logo with event notification emails, instead of the default logo icon.
Include Icon in Emails	Determines whether or not the default or customer email icon path is included with each event notification email.
Include Server Origin in Emails	Determines whether or not the server version and machine name are included with each event notification email.

Include Event Description in Emails	Determines whether or not the basic event description is included with each event notification email.

26.0 Localization

To access the localization tool, look for and select **Localization** on the left tool menu of the Cerberus user interface.

You can use the Localization tool to modify the HTTP/S web client user interface messages for English and translate them into other languages. There are translation tags for every string your end users may encounter, from the UI elements to the message notifications they receive.

To edit a translation of a language, find the language in the **Locale Name** list.

What if you don't see your language in the 'Locale Name' list?

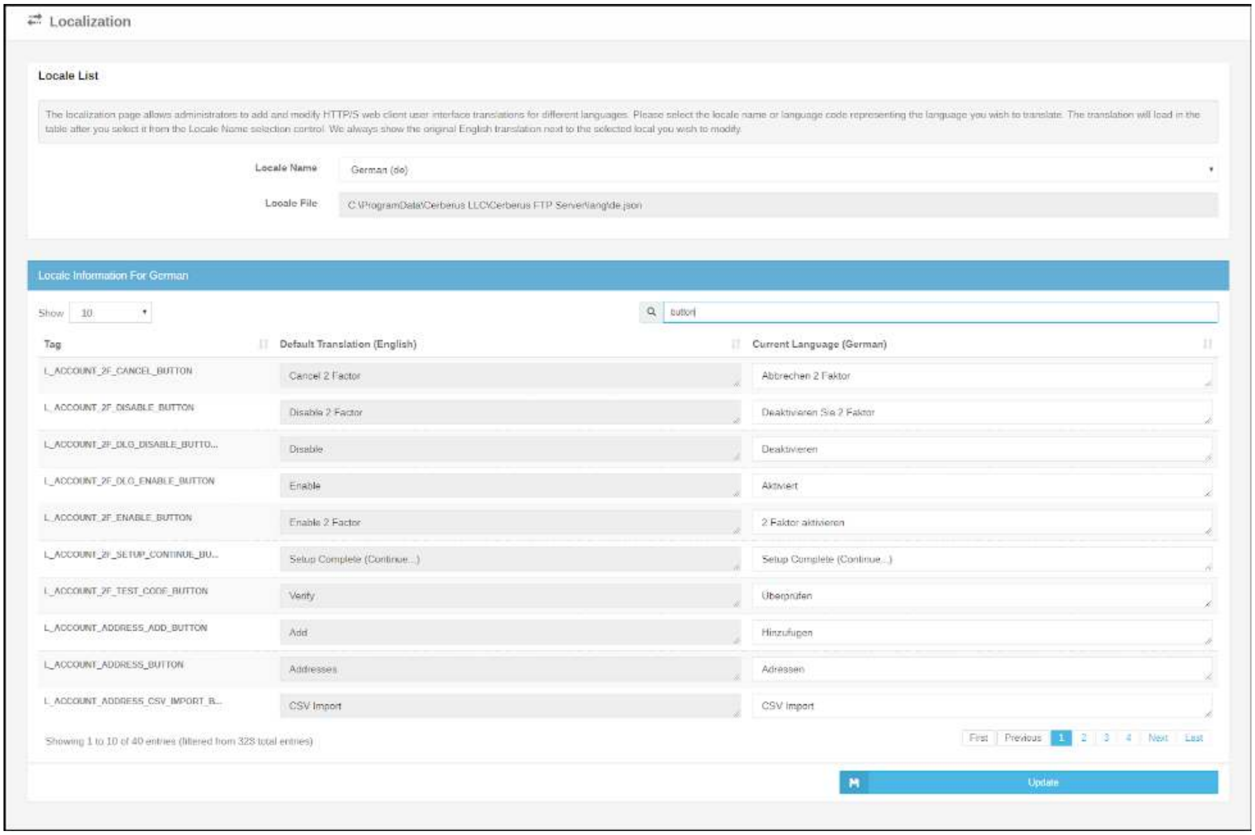
Cerberus doesn't create a language file until someone attempts to log in to the web client with their PC and/or browser settings set to another language. If the server you have installed Cerberus on is set to your desired language, you can try connecting to the Cerberus Web Client on that server to create your desired language translation file. If you do that and still don't see your desired language on the drop-down (after closing and reopening the user interface), you may want to go back and check the language settings on your server and browser if your language is missing from the drop-down list.

Alternative Method: If you wish to have more than one language available to your users and you don't have a way to have someone log into the web client from a browser/PC set to that language, you can also make a copy of the English *.json file in *C:\ProgramData\Cerberus LLC\Cerberus FTP Server\lang* (or whatever drive letter you have installed Cerberus on.) and give it the two-letter ISO language code for that language. Translations in the user interface are drawn from the <language_code>.json file for that language (see ISO 2 Letter Language Codes), where <language_code> is the two-letter ISO language code. For example, Spanish is 'es', French 'fr', and German 'de', using the two-letter ISO language codes.

You will note there are *.csv files in the language folder as well. They are there for convenience in case you wish to send it to someone to provide translations for you that you can later copy into the corresponding *.json.

26.1 Editing Translations

Once you have selected a language, you will see the language tags, default translation, and the translation for each available item. **(For a translation we don't have, you will see the default English values.)** Use the filter option at the top of the page to help find a specific language tag. You can filter by Tag or message string.



Localization

Locale List

The localization page allows administrators to add and modify HTTP/S web client user interface translations for different languages. Please select the locale name or language code representing the language you wish to translate. The translation will load in the table after you select it from the Locale Name selection control. We always show the original English translation next to the selected local you wish to modify.

Locale Name: German (de)

Locale File: C:\ProgramData\Cerberus LLC\Cerberus FTP Server\lang\de.json

Locale Information For German

Show 10

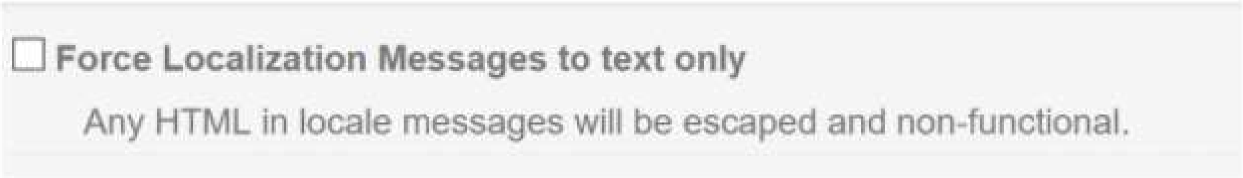
Search: button

Tag	Default Translation (English)	Current Language (German)
L_ACCOUNT_2F_CANCEL_BUTTON	Cancel 2 Factor	Abbrechen 2 Faktor
L_ACCOUNT_2F_DISABLE_BUTTON	Disable 2 Factor	Deaktivieren Sie 2 Faktor
L_ACCOUNT_2F_DLG_DISABLE_BUTTO...	Disable	Deaktivieren
L_ACCOUNT_2F_DLG_ENABLE_BUTTON	Enable	Aktiviert
L_ACCOUNT_2F_ENABLE_BUTTON	Enable 2 Factor	2 Faktor aktivieren
L_ACCOUNT_2F_SETUP_CONTINUE_BU...	Setup Complete (Continue...)	Setup Complete (Continue...)
L_ACCOUNT_2F_TEST_CODE_BUTTON	Verify	Überprüfen
L_ACCOUNT_ADDRESS_ADD_BUTTON	Add	Hinzufugen
L_ACCOUNT_ADDRESS_BUTTON	Addresses	Adressen
L_ACCOUNT_ADDRESS_CSV_IMPORT_B...	CSV Import	CSV Import

Showing 1 to 10 of 40 entries (filtered from 328 total entries)

Locale List and Translations

26.2 'Force Localization Messages to test only' Security Setting



□ Force Localization Messages to text only

Any HTML in locale messages will be escaped and non-functional.

Please be aware that if a translation manages to get corrupted and present a hyperlink to a malicious site, end-users could have security concerns. Cerberus FTP has the ability to turn off all HTML encoding in its translations if you know that you are not using the HTML UI feature. On the **Localization** page, select 'Force Localization Messages to text only' and then 'Update Settings'. This will ensure that all of your messages will be HTML escaped. Also, you will be notified if any of your messages did use HTML and which of these translation tags need to be updated.

26.3 Configuring Web Client Login Page Messages

Starting in version 12 of Cerberus, you have the ability to use HTML when customizing the Web Client login page. The login page for password-protected public shares can also be customized separately. For more information and configuration instructions, see Configuring Web Client Login Page Messages

27.0 Active Directory Authentication

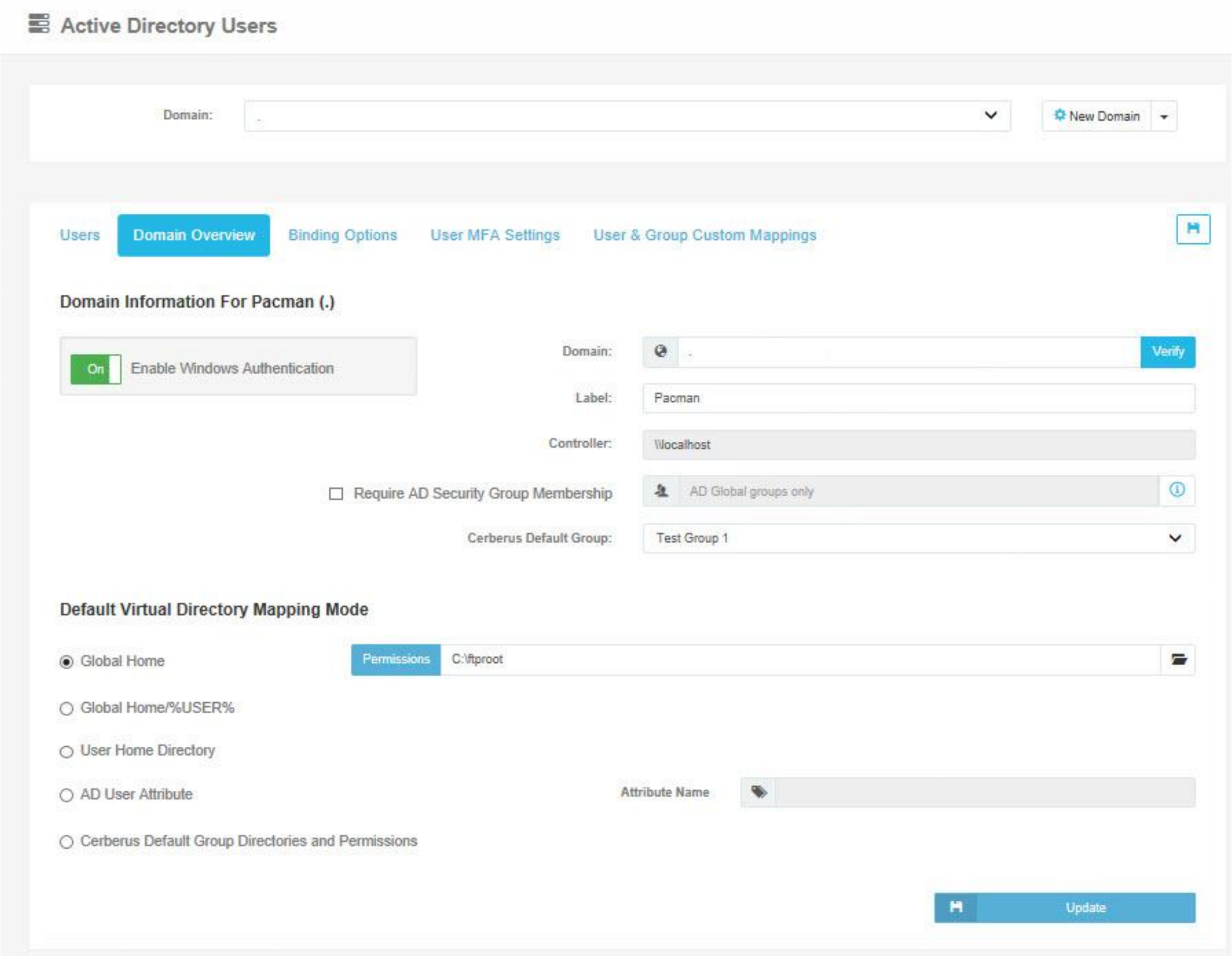
27.1 About Active Directory Integration

Cerberus FTP Server Professional and Enterprise editions are able to authenticate users on a Windows domain (or the local NT account database), even if the computer Cerberus FTP Server is installed on is not the domain controller. The domain may be a pre-Windows 2000 domain (NT4), a domain configured to use Active Directory, or the local system account database (use "." as the domain for authenticating against local machine accounts). However, the machine Cerberus FTP Server is running on must be a member of the domain you wish to authenticate users against.

Configuring Cerberus to use Active Directory authentication simply requires enabling Active Directory authentication and telling the server the name of the domain to authenticate against. The rest of the configuration is automatic. Users are able to log into the server using the same username and password they use to log into their workstations on the domain. For the purpose of access to files and folders, the user has the same access as the Active Directory user with the same name. All operations on the server by the user are carried out while impersonating the Active Directory user.

Important Security Consideration: There is an exception to impersonation for Active Directory authentication when using SFTP and **Public Key only** SSH authentication. The Active Directory user can still be authenticated with Public Key only authentication, but the Active Directory user cannot be impersonated. Only **Password** or **Public Key and Password** SSH authentication methods support AD user impersonation.

To allow Active Directory authentication, you will need to select the **Enable Windows Authentication for this Domain** slider in **AD Users**. Once selected, Cerberus will attempt to authenticate users from the domain listed in the **Domain** edit box.



Domain: .

Domain Information For Pacman (.)

On Enable Windows Authentication

Domain: .

Label: Pacman

Controller: \\localhost

□ Require AD Security Group Membership — AD Global groups only

Cerberus Default Group: Test Group 1

Default Virtual Directory Mapping Mode

● Global Home — C:\ftproot

○ Global Home/%USER%

○ User Home Directory

○ AD User Attribute — Attribute Name

○ Cerberus Default Group Directories and Permissions

Active Directory Authentication page

27.2 Default Virtual Directory Mapping for AD Users

Active Directory accounts are always configured for simple directory mode (See Adding a New User for more information about simple mode) if any mode other than **Cerberus Group** is selected for the *Default Virtual Directory Mapping* mode.

The *Default Virtual Directory Mapping* modes work as follows:

Global Home	Every AD account will use the directory specified under the "Global Home" edit box as the FTP root. This is the simplest option, and every AD user is assigned this one directory as their root folder. The Cerberus permissions on this folder can be restricted through the **Permissions** button to the right of the Global Home edit box. NTFS permissions for the AD user still apply.

<table>
<tr><td><b>Global Home\%USER%</b></td><td>Every AD account will use a subdirectory off of the "Global Home" directory that is the same as the account's name. This directory will be created automatically if it doesn't exist when the user logs in.
The Cerberus permissions on this folder can be restricted through the <b>Permissions</b> button to the right of the Global Home edit box. NTFS permissions for the AD user still apply.</td></tr>
<tr><td><b>AD User Home Directory</b></td><td>Every AD account will use that account's home directory as the FTP root.
The Cerberus <b>permissions</b> on this folder can be restricted through the Permissions button to the right of the Global Home edit box. NTFS permissions for the AD user still apply.</td></tr>
<tr><td><b>AD Directory Attribute</b></td><td>Every AD account will use the directory attribute defined here to determine what virtual directories to add to their account.
This attribute can have multiple values, and each value will be added as a separate virtual directory.
The default value will be a valid Windows directory path. By default, the last directory of the file path will be used for the virtual directory name, and the user will have full permissions to the directory path.
The value can be customized into 3 semicolon separate components to customize the added virtual directory path into a full directory path, a virtual directory name, and permissions set for the virtual directory.
For example, the value for the attribute could be:
<b>C:\ftproot\user\andrew;home;2047</b>
The first part is the directory path, the second is the directory name, and the third is a bit mask indicating the permissions the user has for that virtual directory.
The directory permissions field for a virtual directory is a simple bit mask. Permissions have the following values:
<table>
<tr><th>Permission</th><th>Value</th></tr>
<tr><td>DOWNLOAD</td><td>1</td></tr>
<tr><td>UPLOAD</td><td>2</td></tr>
<tr><td>RENAME</td><td>4</td></tr>
<tr><td>DELETE</td><td>8</td></tr>
<tr><td>CREATEDIR</td><td>16</td></tr>
<tr><td>LIST DIRECTORIES</td><td>32</td></tr>
<tr><td>LIST FILES</td><td>64</td></tr>
<tr><td>DISPLAY HIDDEN FILES</td><td>128</td></tr>
<tr><td>ZIP</td><td>256</td></tr>
<tr><td>UNZIP</td><td>512</td></tr>
<tr><td>SHARE</td><td>1024</td></tr>
</table>
</td></tr>
</table>

	Just add the values up to achieve the desired permissions. e.g., Download, Upload, Rename, and Delete permissions would be (1 + 2 + 4 + 8) = **15**. Granting all permissions would be **2047**.
Use Default Group Directories and Permissions	The specified Cerberus Group will be used to determine what directories and what settings to apply to the AD user when they log in, including any security requirements associated with the group.

27.2.1 Active Directory FTP Security Group

Optionally, you can also configure a Security Group for FTP users. This will cause Cerberus FTP Server to check that the Active Directory user is a member of the listed Active Directory Global security group before allowing login. If selected, only members of the security group will be allowed to log in.

27.3 Authenticating Against more than one Active Directory Domain

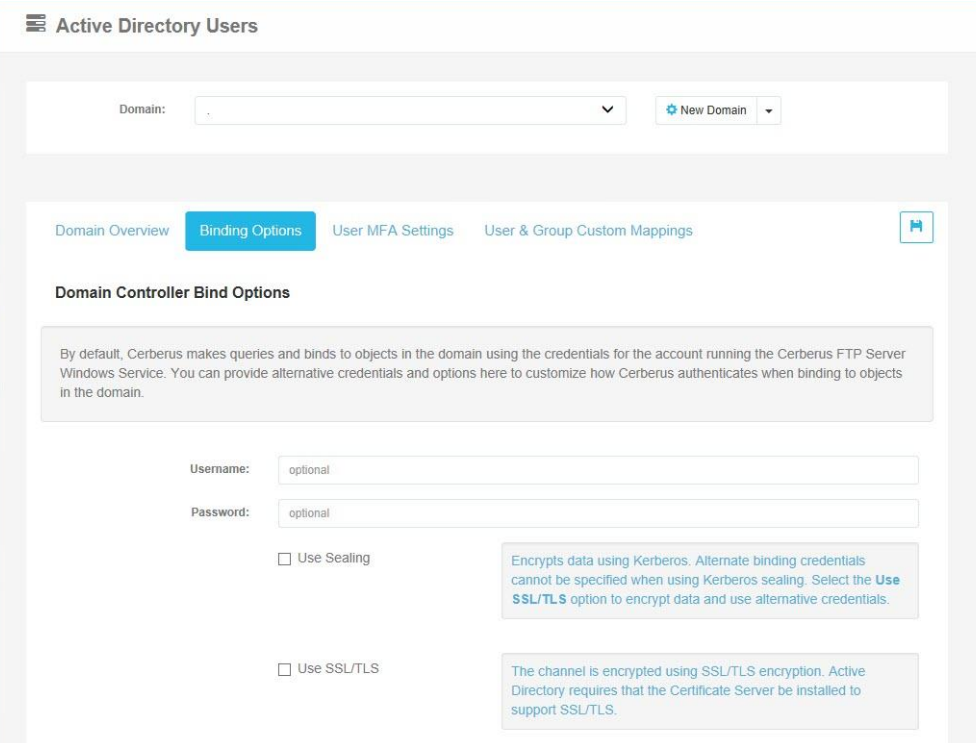
Cerberus FTP Server can be configured to authenticate against multiple domains. Select the **AD Users** page on the main menu and click the '**Domains**' drop down menu in the top right corner. Enter the domain name in the **Add A New Domains** form and click **Add**.This will add a new domain tab to the AD User **Domains** drop down. This new domain can now be configured.

27.4 Understanding Windows Authentication

Active Directory user authentication is intended for experienced system administrators that understand the NT security model. Novice users, or users wishing to avoid the details of Windows security, should leave Windows Authentication disabled and stick with native Cerberus FTP Server users.

27.5 Domain Controller Bind Options

By default, Cerberus makes queries and binds to objects in the domain using the credentials of the account running the Cerberus FTP Server Windows Service.You can provide alternative credentials and options here to customize how Cerberus authenticates when binding to objects in the domain.



Domain: .

Domain Controller Bind Options

By default, Cerberus makes queries and binds to objects in the domain using the credentials for the account running the Cerberus FTP Server Windows Service. You can provide alternative credentials and options here to customize how Cerberus authenticates when binding to objects in the domain.

Username: optional

Password: optional

□ Use Sealing

Encrypts data using Kerberos. Alternate binding credentials cannot be specified when using Kerberos sealing. Select the **Use SSL/TLS** option to encrypt data and use alternative credentials.

□ Use SSL/TLS

The channel is encrypted using SSL/TLS encryption. Active Directory requires that the Certificate Server be installed to support SSL/TLS.

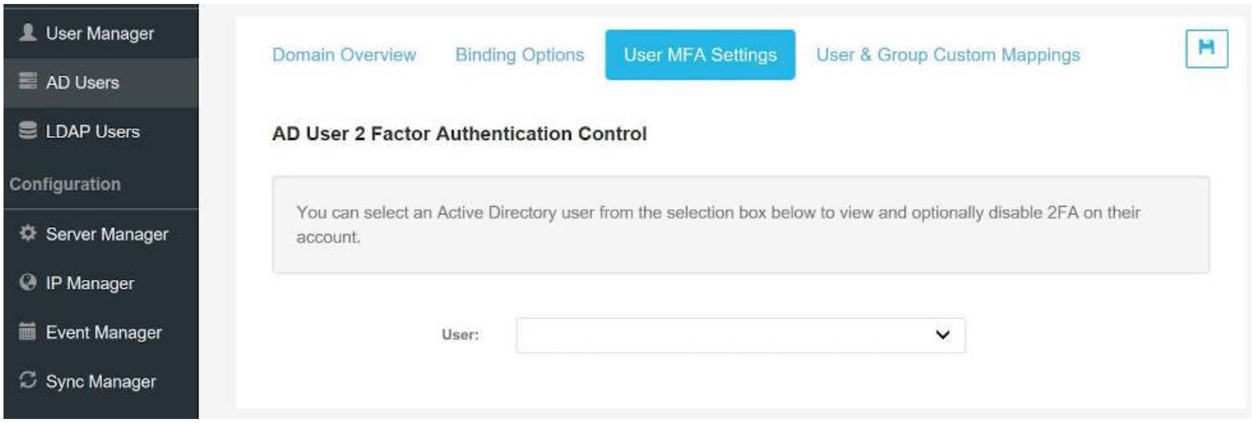
In the **AD Users** page, select the ‘**Binding Options**’ tab. Enter the Username and Password of the alternate account you wish to have Cerberus authenticate with when binding to the domain. There are also two other options:

- **Use Sealing:** If this option is selected, Cerberus encrypts data using Kerberos. **Alternate binding credentials cannot be specified when using Kerberos sealing.** Select the Use SSL/TLS option to encrypt data and use alternative credentials.
- **Use SSL/TLS**: If this option is selected, the channel is encrypted using SSL/TLS encryption. Active Directory requires that the Certificate Server be installed to support SSL/TLS.

If any changes are made to the settings on this page, ensure you click the diskette icon to save your changes.

27.6 User MFA Settings

If you're using Active Directory and the MFA requirement is set up on the default group the users are assigned to, you can do a one-time disablement of that user's 2FA requirement. The next time the user logs in, they will be required to set up 2FA again.

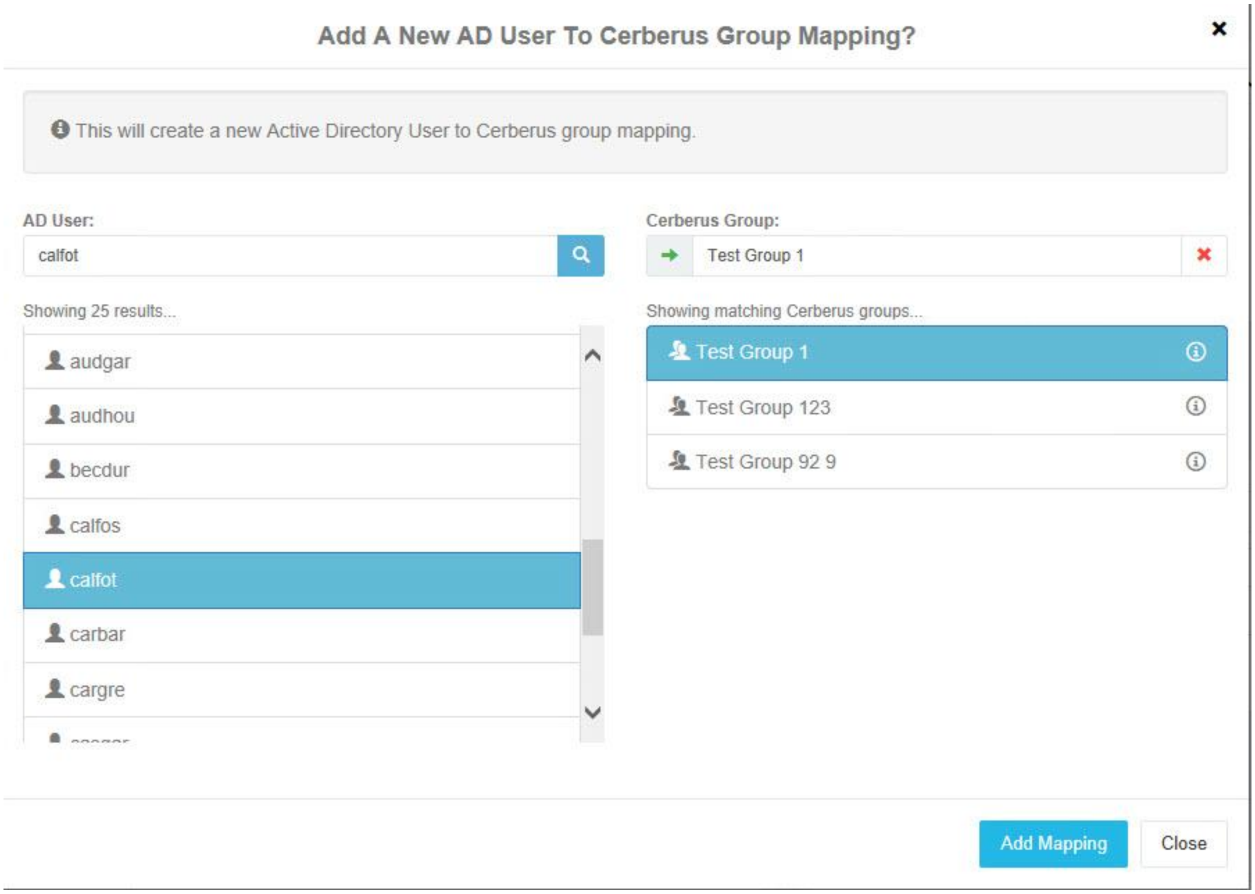


You can select an Active Directory user from the selection box below to view and optionally disable 2FA on their account.

1. In the Cerberus UI or Web Admin client, Click on 'AD Users
2. Select ‘User MFA Settings'
3. Select the affected user from the drop down
4. The user setting should say 'Enabled'. Click 'Disable 2FA' to disable their 2FA requirement
5. The user can now log in and be required to set up 2FA again. They should do it on their new device. Also advise the user to clear their browser cache before logging in, just in case there is any old session data lingering.
   Basically, this removes and re-adds 2FA for this user.

27.7 Active Directory User or Active Directory Group to Cerberus Group Mapping

By default, all AD users are assigned the same virtual directories and permissions. These defaults are configured on the Domain tab of the AD Users page. However, if you wish to customize the directory and permission mappings for individual AD users or AD Groups, you can do so using the **User Custom Mappings** button. You can select individual AD accounts and map them to Cerberus group accounts, or, you can map AD group accounts to Cerberus group accounts. Configuring an AD user to group mapping will override the default Cerberus Group and directory mapping specified for all AD users.



Add A New AD User To Cerberus Group Mapping?

This will create a new Active Directory User to Cerberus group mapping.

AD User: calfot

Cerberus Group: Test Group 1

Showing 25 results...

- audgar
- audhou
- becdur
- calfos
- calfot
- carbar
- cargre

Showing matching Cerberus groups...

- Test Group 1
- Test Group 123
- Test Group 92 9

Configuration page for AD User to Cerberus Group Mapping

27.7.1 Creating an AD User to Cerberus Group Mapping

Mappings between an AD User and a Cerberus Group can be achieved by clicking on **AD Users** on the main menu. Select an AD domain using the 'domain' drop down. Click on '**User Custom Mappings**' and Click the '**New**' button in the '**Active Directory User to Cerberus Group Mapping**' section. Select an AD user from the AD Users list box (or simply type the name of the AD user in the edit box) and then select a Cerberus Group. Click the **Add Mapping** button and a mapping entry will be placed in the '**Active Directory User to Cerberus Group Mapping**' section to indicate the AD user will now have the same constraints and virtual directory mappings as the Cerberus Group they are listed under.

27.7.2 Creating an AD Group to Cerberus Group Mapping

Customizing each individual AD User to a Cerberus group can be a time-consuming task if you have many users, especially if you can divide up large groups of users into just a few groups.

To make maintaining large numbers of users easier, you can use the AD group to Cerberus group mapping capability. On the **AD Users** page, you can map AD groups to Cerberus groups.

When an AD user logs into Cerberus, the server checks the **direct** AD group memberships for that AD user and sees if there are any AD group to Cerberus group mappings. If a mapping is found, the virtual directories for that Cerberus group will be added to the virtual root for the AD user. Only the virtual directories from the Cerberus group are added to the AD user. No other constraints are transferred.

Click on '**AD Users'** on the main menu. Select an AD domain using the 'domain' drop down. Click on '**User Custom Mappings**'. Then, click the '**New**' button in the '**Active Directory Group to Cerberus Group Mapping**' section. Select an AD group from the AD Groups list box (or simply type the name of the AD group in the edit box) and then select a Cerberus Group. Click the '**Add Mapping'** button and a mapping entry will be placed in the '**Active Directory Group to Cerberus Group Mapping**' section to indicate the AD group will now have the same virtual directory mappings as the Cerberus Group they are listed under.

Note: The Default Group and Default Virtual Directory mappings are still applied to the user when AD group to Cerberus group mappings are present, unlike AD user to Cerberus user mappings.

27.5.3 Removing an AD mapping

To remove a mapping, simply select the mapped entry by clicking the box on the left, select the drop down menu next to ,'**New**' and select **Delete Mapping**.

28.0 LDAP Authentication

Cerberus FTP Server Professional is able to authenticate users against LDAP directory services. The **Lightweight Directory Access Protocol**, or **LDAP**, is an application protocol for querying and modifying directory services running over TCP/IP.

Administrators can easily integrate Cerberus and LDAP or LDAPS (LDAP over SSL). All you need are a few parameters describing the LDAP service.

What do I need to use LDAP Authentication?

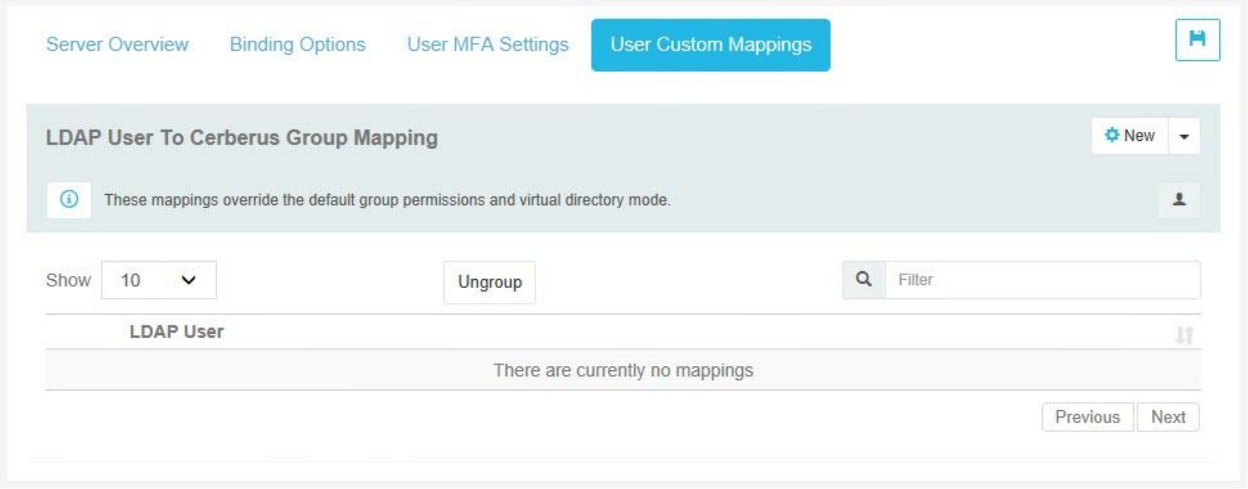
An LDAP service and some information about the server hosting the LDAP service:

Server	This parameter is the FQDN or IP address of the LDAP server to search.
Port	The network port of the LDAP server.
Enable SSL	This checkbox determines whether the connection to the LDAP server is encrypted. The LDAP server must support encryption for this to work. Port 389 is the default port for unencrypted LDAP and port 636 is the default LDAPS port.
Label	A label you can use to help identify the configuration you are setting up
Base DN	The distinguished name to use as the search base.
Search Scope	Base, One Level, Subtree
Username attribute	The name of the uid attribute for a user in the directory.

Search Filter	LDAP filter is used to limit results when searching the directory for users.This filter can be used to limit authentication to only certain object types or to members of certain groups.
	Search Filter Examples
	`(objectClass=User)`
	The above filter will include only search entities that have the object class **User**.
	`(memberof:1.2.840.113556.1.4.1941:=cn=FTPUsers,CN=Users,dc=corp,dc=cerberusllc,DC=local)`
	The above filter will include all users that are members of the group **FTPUsers**.
	Do not add a filter including the *Username Attribute* here, as this attribute is handled by Cerberus.
	I.e., if the *Username Attribute* is **sAMAccountName**, Cerberus will automatically create a string like
	`(&(objectClass=User)(sAMAccountName=ftpUser))`
	where *ftpUser* is the name of the user that attempted login.

User DN	The FDN of an account with read privileges to the LDAP server.
Password	The password for the User DN account. This password is encrypted when saved.

By default, all LDAP users are assigned the same virtual directories and permissions. These defaults are configured under the **Default Virtual Directory Mapping Mode** section of the LDAP Users page. However, if you wish to customize the directory and permission mappings for individual LDAP users then you can do so using the **User Custom Mappings** tab.



LDAP User To Cerberus Group Mapping

These mappings override the default group permissions and virtual directory mode.

There are currently no mappings

The **User Customer Mappings** section allows you to override the default settings for a user by mapping individual LDAP users to Cerberus groups. The mapped LDAP users will receive the settings and virtual directories from the mapped group, instead of the defaults.

28.1 Default Virtual Directory Mapping for LDAP Users

The *Default Virtual Directory Mapping* modes work as follows:

<table>
<tr><td><b>Global Home</b></td><td>Every LDAP account will use the directory specified under the "Global Home" edit box as the FTP root. This is the simplest option, and every LDAP user is assigned this one directory as their root folder.
The Cerberus permissions on this folder can be restricted through the <b>Permissions</b> button to the right of the Global Home edit box.</td></tr>
<tr><td><b>Global Home\%USER%</b></td><td>Every LDAP account will use a subdirectory off of the "Global Home" directory that is the same as the account's name. This directory will be created automatically, if it doesn't exist, when the user logs in.
The Cerberus permissions on this folder can be restricted through the <b>Permissions</b> button to the right of the Global Home edit box.</td></tr>
<tr><td><b>LDAP User Attribute</b></td><td>Every LDAP account will use the directory attribute defined here to determine what virtual directories to add to their account.
This attribute can have multiple values, and each value will be added as a separate virtual directory.
The default value will be a valid Windows directory path. By default, the last directory of the file path will be used for the virtual directory name, and the user will have full permissions to the directory path.
The value can be customized into 3 semicolon separate components to customize the added virtual directory path into a full directory path, a virtual directory name, and a permissions set for the virtual directory.
For example, the value for the attribute could be:
<b>C:\ftproot\user\andrew;home;2047</b>
The first part is the directory path, the second is the directory name, and the third is a bit mask indicating the permissions the user has for that virtual directory.
The directory permissions field for a virtual directory is a simple bit mask.
Permissions have the following values:
<table>
<tr><th>Permission</th><th>Value</th></tr>
<tr><td>DOWNLOAD</td><td>1</td></tr>
<tr><td>UPLOAD</td><td>2</td></tr>
<tr><td>RENAME</td><td>4</td></tr>
<tr><td>DELETE</td><td>8</td></tr>
<tr><td>CREATEDIR</td><td>16</td></tr>
<tr><td>LIST DIRECTORIES</td><td>32</td></tr>
<tr><td>LIST FILES</td><td>64</td></tr>
<tr><td>DISPLAY HIDDEN FILES</td><td>128</td></tr>
<tr><td>ZIP</td><td>256</td></tr>
<tr><td>UNZIP</td><td>512</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td></td><td><table><tr><td>SHARE</td><td>1024</td></tr></table>Just add the values up to achieve the desired permissions. e.g., Download, Upload, Rename, and Delete permissions would be (1 + 2 + 4 + 8) = <b>15</b>.
Granting all permissions would be <b>2047</b>.</td></tr>
<tr><td><b>Cerberus Default Group Directories and Permissions</b></td><td>The specified Cerberus Group will be used to determine what directories and what settings to apply to the LDAP user when they log in, including any security requirements associated with the group.</td></tr>
</table>

28.1 Setting up Active Directory Authentication using LDAP

The following steps detail the procedure for enabling LDAP Authentication to verify credentials against Active Directory. The steps are similar for connecting to other LDAP servers, such as OpenLDAP or ApacheDS.

1. Ensure you are on the '**Server Overview**' tab. Change the LDAP Port **Server** and **Port** attribute in the LDAP Users page to the hostname and port number of the Active Directory:

    - e.g., Server: hostname.domain.com **or** an IP address:192.168.0.100
    - Port: 389 is the default for unencrypted LDAP connections. Port 636 is the default for LDAPS encrypted connections.

2. Enter a **Label** to help you identify this configuration, for example 'HQ Domain"

3. Change the **Base DN** to the proper base for the Active Directory.

    Simply specifying the base suffix will not work in this attribute. For Active Directory, it would usually be the cn=Users plus base suffix. e.g.: for domain *corp.cerberusllc.com* :

    CN=Users,DC=corp,DC=cerberusllc,DC=com

    or for local domain *corp.cerberusllc.local* :

    CN=Users,DC=corp,DC=cerberusllc,DC=local

4. Select the **Search Scope** (Base, One Level, Two-Levels)

    This setting controls how deep into the directory to search for users. This setting combined with the Base DN and Search Filter determines which users are matched for authentication. **One Level** is usually the best setting for typical Active Directory configurations.

5. Change the **Username Attribute**.

    This attribute is the one that the LDAP module will search for in Active Directory and attempt to match against the supplied FTP username. It is often the UID attribute on many LDAP servers. For example, if users login using their Common Name, the value of this attribute would be **cn**. For Active Directory, the login name is usually mapped to **sAMAccountName** as it is the attribute in Active Directory most like UID. For Active Directory, it is usually best to specify **sAMAccountName**.

6. Change the **Search Filter**.

   This string is an LDAP search string used to locate and filter the account in Active Directory. This filter can be used to make sure only certain types of objects are checked for authentication.

   ```
   (objectClass=User)
   ```

   The above filter will include only search entities that have the object class **User**.

   ```
   (memberof:1.2.840.113556.1.4.1941:=cn=FTPUsers,CN=Users,dc=corp,dc=cerberusllc,DC=l
           ocal)
   ```

   The above filter will include all users that are members of the group **FTPUsers**. **Do not** attempt to add the uid search attribute here. Cerberus will automatically append an attribute filter to select the correct account based on the User DN Attribute, e.g., if the User DN Attribute is **sAMAccountName**, Cerberus will automatically create a string like

   ```
   (&(objectClass=User)(sAMAccountName=ftpUser))
   ```

   where *ftpUser* is the name of the user that attempted login.

7. Select a **Cerberus Default Directory**.

   The specified Cerberus Group will be used to determine what directories and what settings to apply to the LDAP user when they log in, including any security requirements associated with the group.

8. Click on the '**Bind Options**' tab. Change the DN for the **User DN** bind attribute to a user with the right to read the Active Directory.

   Anonymous access to Active Directory is not allowed, so a bind account is needed. This is simply an account for Active Directory that has read ability on the attribute to which the user will authenticate. An example might be **cn=administrator,CN=Users,DC=corp,DC=cerberusllc,DC=local**. Enter the password for the user account. Note: This password will be encrypted in memory before being saved to disk.

9. Enter the **User DN Password**. This is the password for the user with the right to read the Active Directory.

10. Once done, be sure to click 'Save' (The diskette icon)

11. Verify that the settings are correct by clicking the **Connect** button. You should see the user DNs from Active Directory that are able to log in to Cerberus FTP Server. Note: Unless "Use FQDN"

is checked, only the value of the **User DN Attribute** will be displayed in the LDAP user list. It is this value that will be compared against the FTP username to determine an account match.

12. Select a Cerberus FTP Group to represent the virtual directories and permissions for LDAP users. Note that the "isAnonymous" setting on the group is ignored. The group cannot be anonymous.

Cerberus FTP Server is now configured for authentication against an LDAP server (Active Directory, in this case).

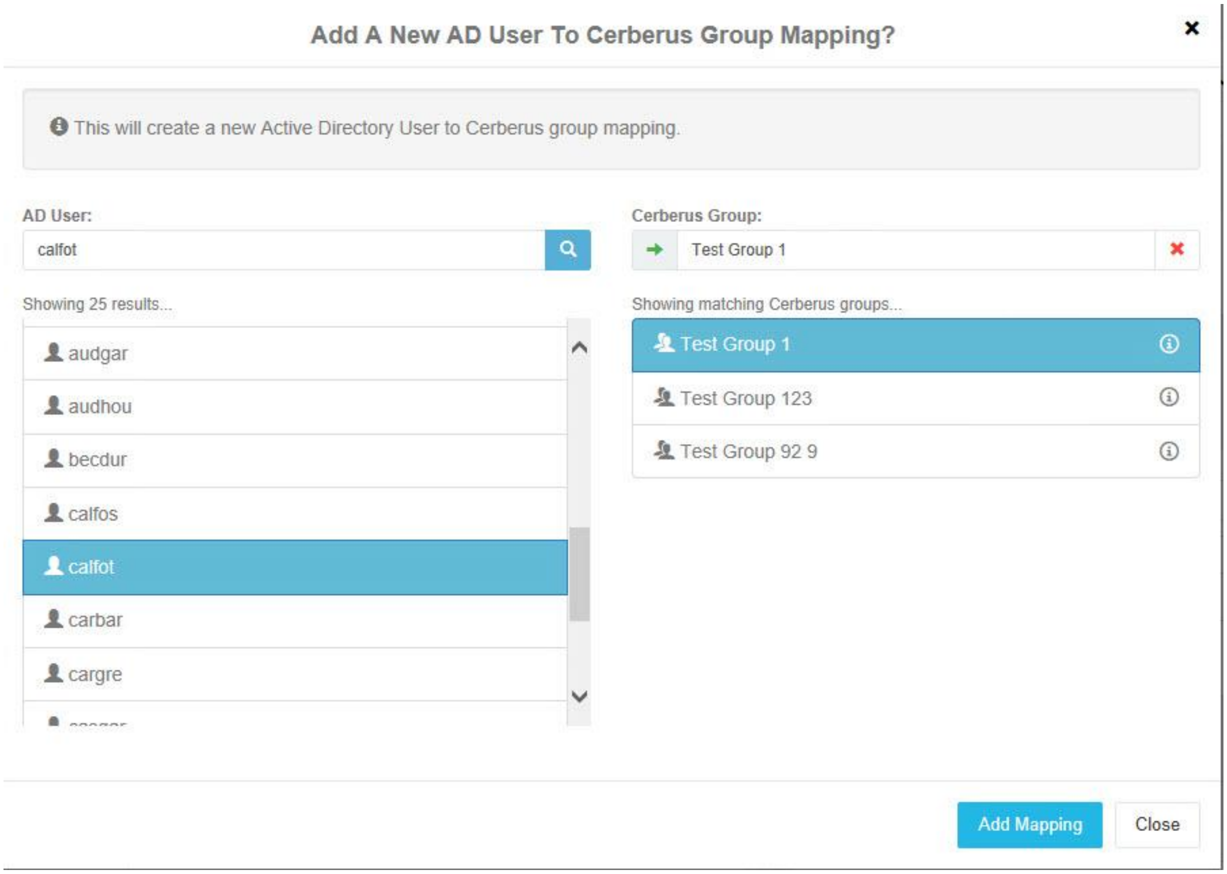
Other, optional LDAP settings are available in the ‘User MFA Settings’ and ‘User Custom Mappings’ sections. See the relevant sections of this document for details.

28.2 LDAP User to Cerberus Group Mapping

You can customize the directory and permission mappings for individual LDAP users through the **LDAP Directory Mapping** tab. Customizing an LDAP account is accomplished by mapping an LDAP user account to a Cerberus group account. This mapping will override the default Cerberus Group and directory mapping, specified on the LDAP Users page, for the mapped LDAP account.

28.2.1 Creating an LDAP User to Cerberus Group Mapping

Mappings between an LDAP User and a Cerberus Group can be achieved by first selecting an LDAP user. Then, select an LDAP user (or simply type the name of the LDAP user in the edit box) and then select a Cerberus Group. Select the **Assign** button and a mapping entry will be placed in the mapping list box to indicate the LDAP user will now have the same constraints and virtual directory mappings as the selected Cerberus Group.



Add A New AD User To Cerberus Group Mapping?

This will create a new Active Directory User to Cerberus group mapping.

AD User: calfot

Cerberus Group: Test Group 1

Showing 25 results...

- audgar
- audhou
- becdur
- calfos
- calfot
- carbar
- cargre

Showing matching Cerberus groups...

- Test Group 1
- Test Group 123
- Test Group 92 9

Configuration page for LDAP User to Cerberus Group Mapping

28.2.2 Removing an LDAP mapping

To remove a mapping, simply select the mapped entry and press the **Remove** button.

28.3 LDAP User Two Factor Authentication Control

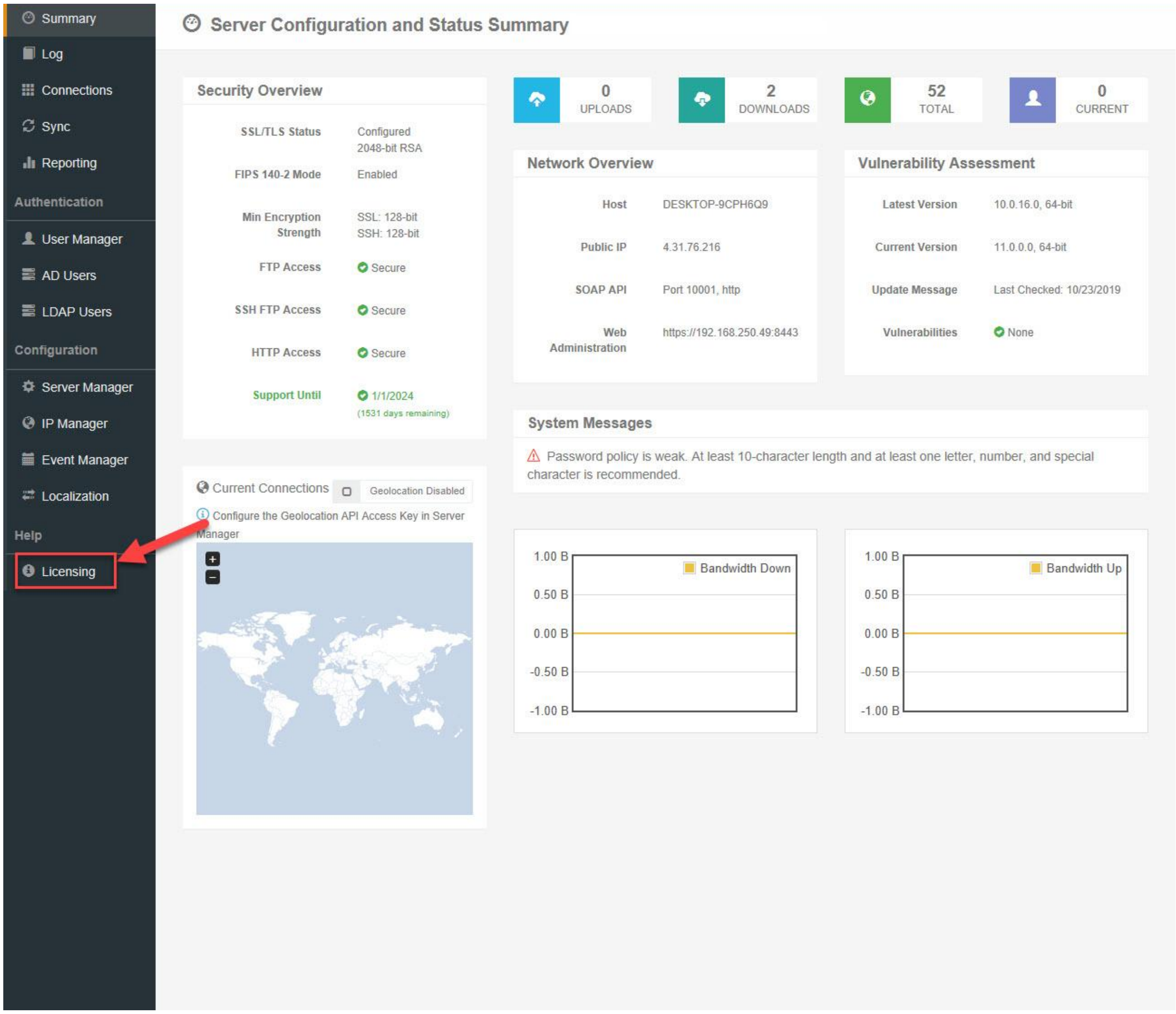
If you wish to disable two-factor authentication (2FA) for an LDAP user that has 2FA enabled, you can select an LDAP user from the selection box in this section to view and optionally disable 2FA on their account. The user can then log into the web client without having to do the additional 2FA authentication step. They can re-enable 2FA if they wish by logging in and viewing their account settings.

32.0 Entering a license for Cerberus FTP Server

32.1 The Registration dialog box

Using Cerberus FTP Server for commercial use past the 25-day evaluation period requires a license key. Once you have purchased and received a license key, you need to enter the license key details in the registration dialog box.

To open the registration dialog box, go to the **Licensing** menu item.



Server Configuration and Status Summary

Security Overview

SSL/TLS Status	Configured 2048-bit RSA
FIPS 140-2 Mode	Enabled
Min Encryption Strength	SSL: 128-bit SSH: 128-bit
FTP Access	✓ Secure
SSH FTP Access	✓ Secure
HTTP Access	✓ Secure
Support Until	✓ 1/1/2024 (1531 days remaining)

Network Overview

Host	DESKTOP-9CPH6Q9
Public IP	4.31.76.216
SOAP API	Port 10001, http
Web Administration	https://192.168.250.49:8443

Vulnerability Assessment

Latest Version	10.0.16.0, 64-bit
Current Version	11.0.0.0, 64-bit
Update Message	Last Checked: 10/23/2019
Vulnerabilities	✓ None

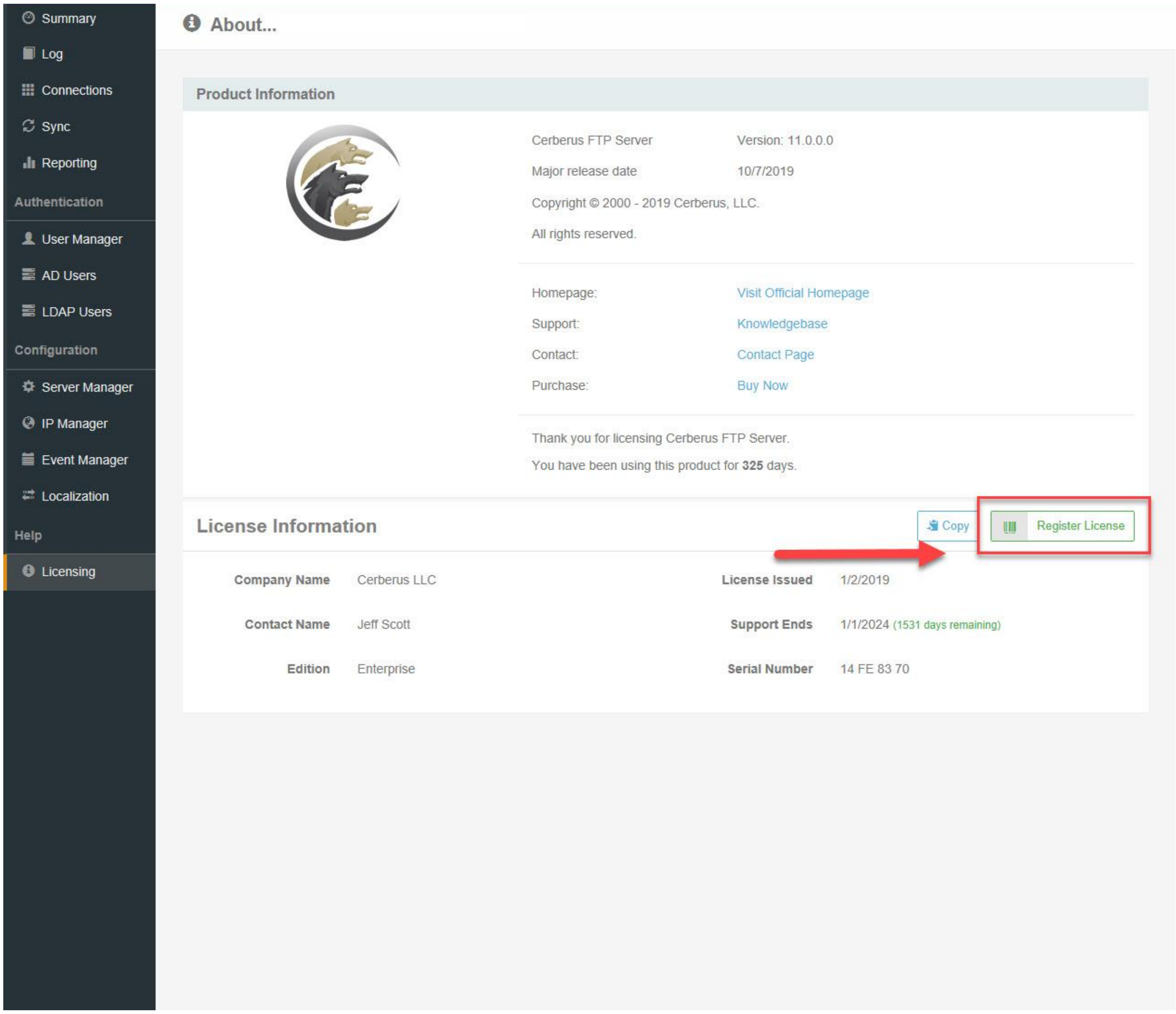
System Messages

⚠ Password policy is weak. At least 10-character length and at least one letter, number, and special character is recommended.

Current Connections □ Geolocation Disabled

ⓘ Configure the Geolocation API Access Key in Server Manager

Click the **Register License** button.



Product Information

Cerberus FTP Server	Version: 11.0.0.0
Major release date	10/7/2019

Copyright © 2000 - 2019 Cerberus, LLC.

All rights reserved.

Homepage:	Visit Official Homepage
Support:	Knowledgebase
Contact:	Contact Page
Purchase:	Buy Now

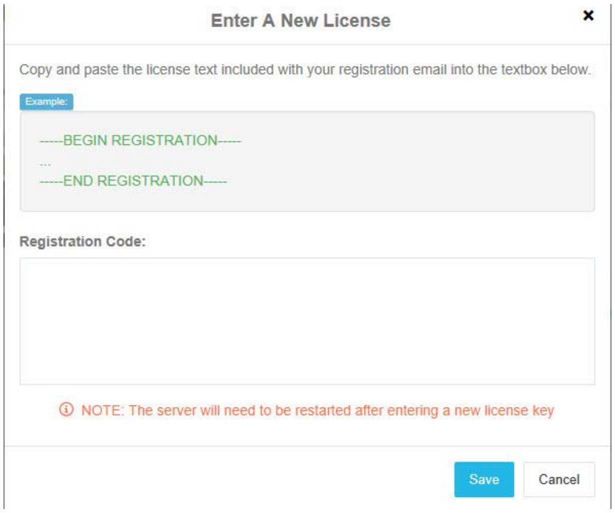
Thank you for licensing Cerberus FTP Server.

You have been using this product for **325** days.

License Information

Company Name	Cerberus LLC	**License Issued**	1/2/2019
Contact Name	Jeff Scott	**Support Ends**	1/1/2024 (1531 days remaining)
Edition	Enterprise	**Serial Number**	14 FE 83 70

Open your license email and copy everything starting at and including "-----BEGIN REGISTRATION-----" all the way until and including "-----END REGISTRATION-----". Paste the copied text into the **Registration Code** box.


Enter A New License

Copy and paste the license text included with your registration email into the textbox below.

```
-----BEGIN REGISTRATION-----
...
-----END REGISTRATION-----
```

Registration Code:

NOTE: The server will need to be restarted after entering a new license key

Press the **Save** button. Another dialog box will appear, after you press enter, to inform you of correct or incorrect registration information. Please note that a service restart is required after entering a new license key. Cerberus will prompt you to restart after successfully entering a new license key.

Once you have successfully registered Cerberus FTP Server, the "About" page in **'Licensing'** will display the registration contact name, company name, purchase date, and for how long the license entitles the user to free updates.

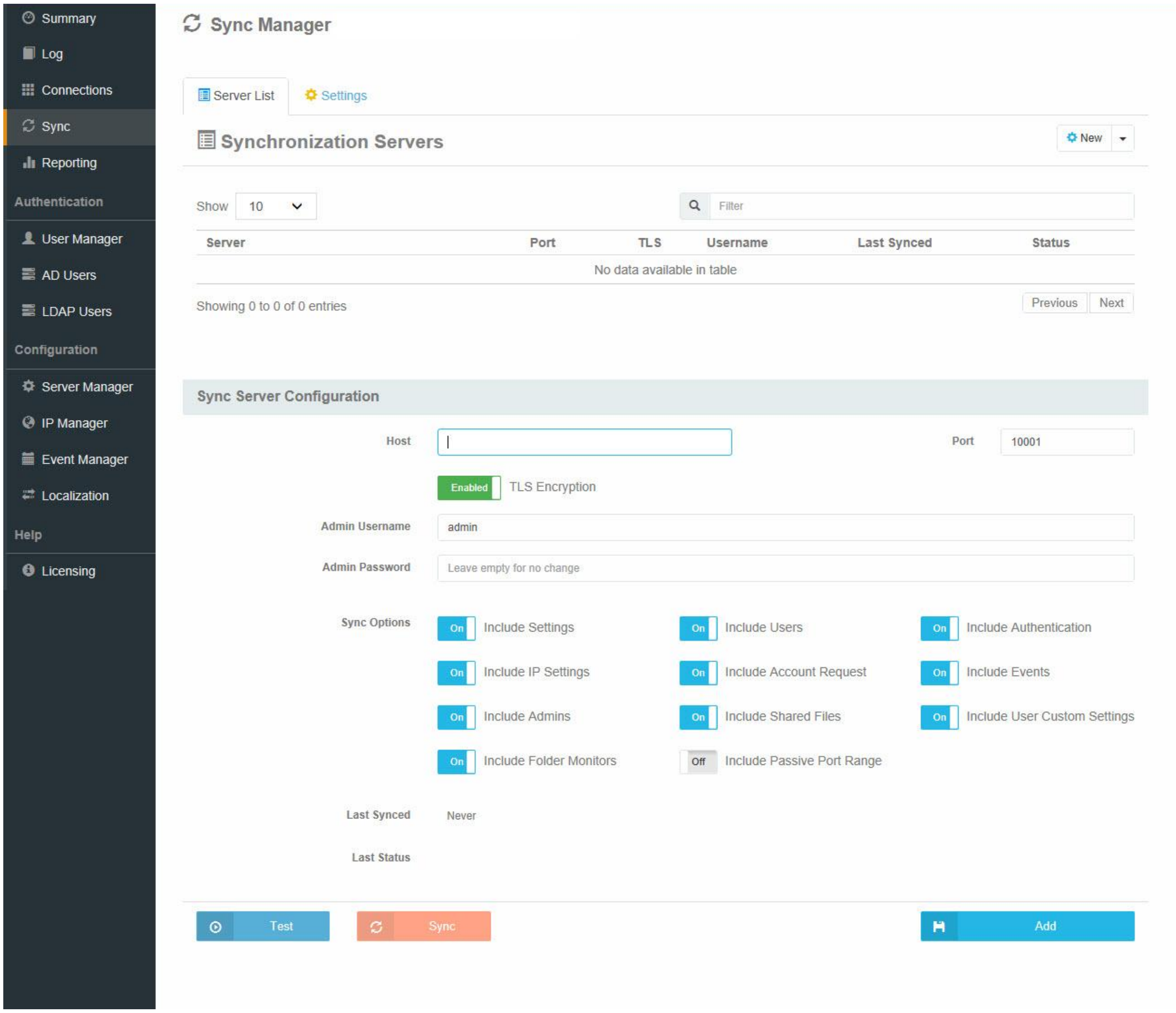
33.0 The Synchronization Manager

Cerberus FTP Server Professional and Enterprise editions support automatically replicating users and settings from a primary or master server to other running Cerberus FTP Server machines. This capability allows administrators to maintain active backups of the main server in case of failure, or to ensure a cluster of servers contains identical configurations while only having to manage one machine.

The Synchronization Manager is used from the machine you want to use as the primary server. The Manager allows an administrator to designate one or more running Cerberus instances for syncing. With the exception of machine-specific configuration information, the other servers become exact copies of the primary server. Each server that is being synced to will have its users and settings replaced by the users and settings on the primary server.

The replication process can be configured to occur at regular intervals to ensure that all of your synced servers are kept current with the primary server.

The server instances must all be running the same version, and have unique license keys.



Synchronization Servers

Show 10

Server	Port	TLS	Username	Last Synced	Status
No data available in table					

Showing 0 to 0 of 0 entries

Sync Server Configuration

Host:
Port: 10001
Enabled TLS Encryption
Admin Username: admin
Admin Password: Leave empty for no change

Sync Options:

On Include Settings	On Include Users	On Include Authentication
On Include IP Settings	On Include Account Request	On Include Events
On Include Admins	On Include Shared Files	On Include User Custom Settings
On Include Folder Monitors	Off Include Passive Port Range	

Last Synced: Never
Last Status:

Cerberus Synchronization Manager

33.1 Backup Server Requirements

To add a backup server to the synchronization list, that backup server must be running the same version of Cerberus FTP Server as the primary server and have a valid, unique license key. All users, groups, and other settings will be synchronized to the backup servers, **except**:

- License keys
- SOAP and remote/web administration settings
- Server certificate, private key, CA, and CRL security settings
- Client SSH public key files

33.2 Available Settings

33.2.1 Backup Server

Server	The hostname or IP address of a backup server
Port	The remote administration port of the backup server to connect to.
Secure Connection	Instructs this server to connect using TLS/SSL security to the backup server. This setting must always be enabled.
Username	The remote administration account username on the remote server.
Password	The remote administration account password on the remote server. This value will be encrypted before being saved to disk.

33.2.2 Synchronization Settings

These are basic server synchronization settings. You can enable and set server synchronization intervals using these settings.

Enable Server Synchronization	Checking this setting will enable automatically replicating this server's users and settings to the added backup servers. This replication will occur at the sync interval, in minutes.
Sync Interval	How often, in minutes, to synchronize this server's setting to the backup servers.

34.0 Server Certificates

34.1 What is a Server Certificate?

The most common use of a digital certificate is to verify that a user (or server) sending a message is who he or she claims to be and to provide the receiver with the means to encode a reply.

There are generally two options for obtaining a digital certificate (and the accompanying private key).

1. You can generate your own self-signed certificate using the Cerberus FTP Server Getting Started Wizard.

2. You can obtain a certificate from a recognized Certificate Authority

Which is more appropriate really depending upon your goals. If you just want to make sure that client and server connections are securely encrypted then a self-signed certificate is all you need. It has the benefit of being easily created through Cerberus and completely free.

If your goal is to make sure that your clients can verify that the server they are connecting to is legitimate and to ensure they don't see any warning messages about being "unable to verify the server" then using a certificate signed by a trusted certificate authority is required. You will have to contact one of the recognized Certificate Authorities such as Comodo, Thawte, Verisign, or one of the many other recognized Certificate Authorities and request a server certificate (for a price).

34.2 Can I just use a Self-Signed Certificate?

Yes, but your users will not be able to easily verify your server's identity. If you are using Cerberus FTP Server exclusively on your own private network, or are just looking to test Cerberus FTP Server out before deploying it on the Internet, a self-signed certificate is more than adequate. You can always change your certificate later to one signed by a recognized Certificate Authority.

34.3 More Information

- Creating a Certificate Signing Request

- Creating a Self-Signed Certificate

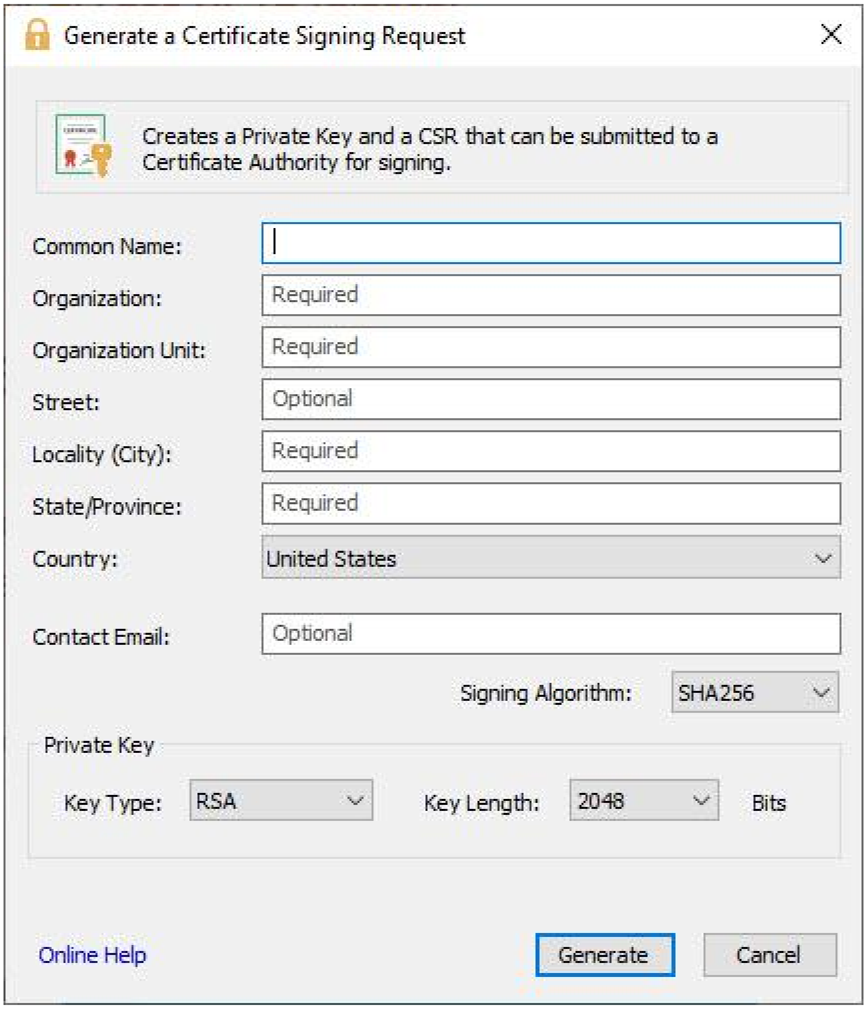
- Importing a 3rd Party Certificate

- Exporting a certificate from the Windows Certificate Store for use by Cerberus FTP Server

35.0 Certificate Signing Request

35.1 Creating a Certificate Signing Request

The first step in requesting a certificate from a Certificate Authority (CA) is usually creating what is called a Certificate Signing Request (CSR).



Generate a Certificate Signing Request

Creates a Private Key and a CSR that can be submitted to a Certificate Authority for signing.

Common Name:	
Organization:	Required
Organization Unit:	Required
Street:	Optional
Locality (City):	Required
State/Province:	Required
Country:	United States
Contact Email:	Optional

Signing Algorithm: SHA256

Private Key

Key Type: RSA   Key Length: 2048 Bits

Certificate Signing Request Wizard

Access the CSR form by going to ‘Tools’, ‘Generate a CSR’. Fill in all of the required fields for the CSR and then press the **Generate** button. After you select the Generate button a directory selection dialog will appear to allow you to specify a directory to save the private key and certificate signing request.

Make sure you save both the private key file, and the CSR file. You will need both of these files.

35.2 Submitting your CSR to a Certificate Authority

You will submit the CSR file to your CA and keep the private key file. Once your CA has approved your CSR they will issue you a signed public certificate file. This signed public certificate file from your CA and the private key file, created during your certificate signing request, together represent your server public and private key pair.

The CA will usually provide several different format options for the signed public certificate. The preferred format is a PEM-formatted certificate (the same format Apache web server uses). PEM is also called a Base64 encoded DER certificate. You can tell if a certificate is in this format by opening it in a text editor, and looking for the beginning and ending lines "-----BEGIN CERTIFICATE-----" and "-----END CERTIFICATE-----".

36.0 Installing a Digital Certificate

35.1 Digital Certificate Support

There are generally two options for obtaining a digital certificate (and a private key).

1. You can generate your own certificate using the Cerberus **Create Cert** button.

2. You can obtain a certificate from a recognized Certificate Authority

Which option is more appropriate really depends upon your goals. If you just want to make sure that client and server connections are securely encrypted then a self-signed certificate is all you need. It has the benefit of being easily created through Cerberus and completely free.

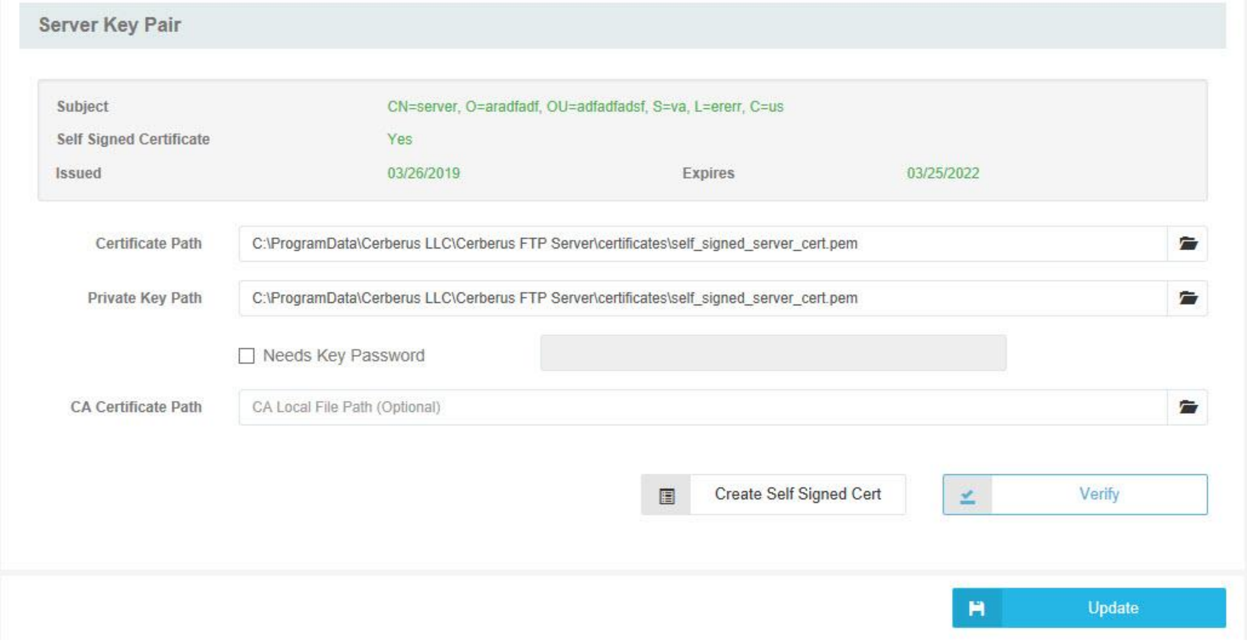
36.2 Creating a Self-Signed Certificate

If you just want to be sure that connections are security encrypted then a self-signed certificate is sufficient for your organization.

36.2.1 Steps to Create a Self-Signed Certificate:

1. Open the Server Manager by selecting the **Server Manager** item from the main menu.

2. Select the **Security** tab.



Server Key Pair

Subject	CN=server, O=aradfadf, OU=adfadfadsf, S=va, L=ererr, C=us		
Self Signed Certificate	Yes		
Issued	03/26/2019	Expires	03/25/2022

Certificate Path: C:\ProgramData\Cerberus LLC\Cerberus FTP Server\certificates\self_signed_server_cert.pem

Private Key Path: C:\ProgramData\Cerberus LLC\Cerberus FTP Server\certificates\self_signed_server_cert.pem

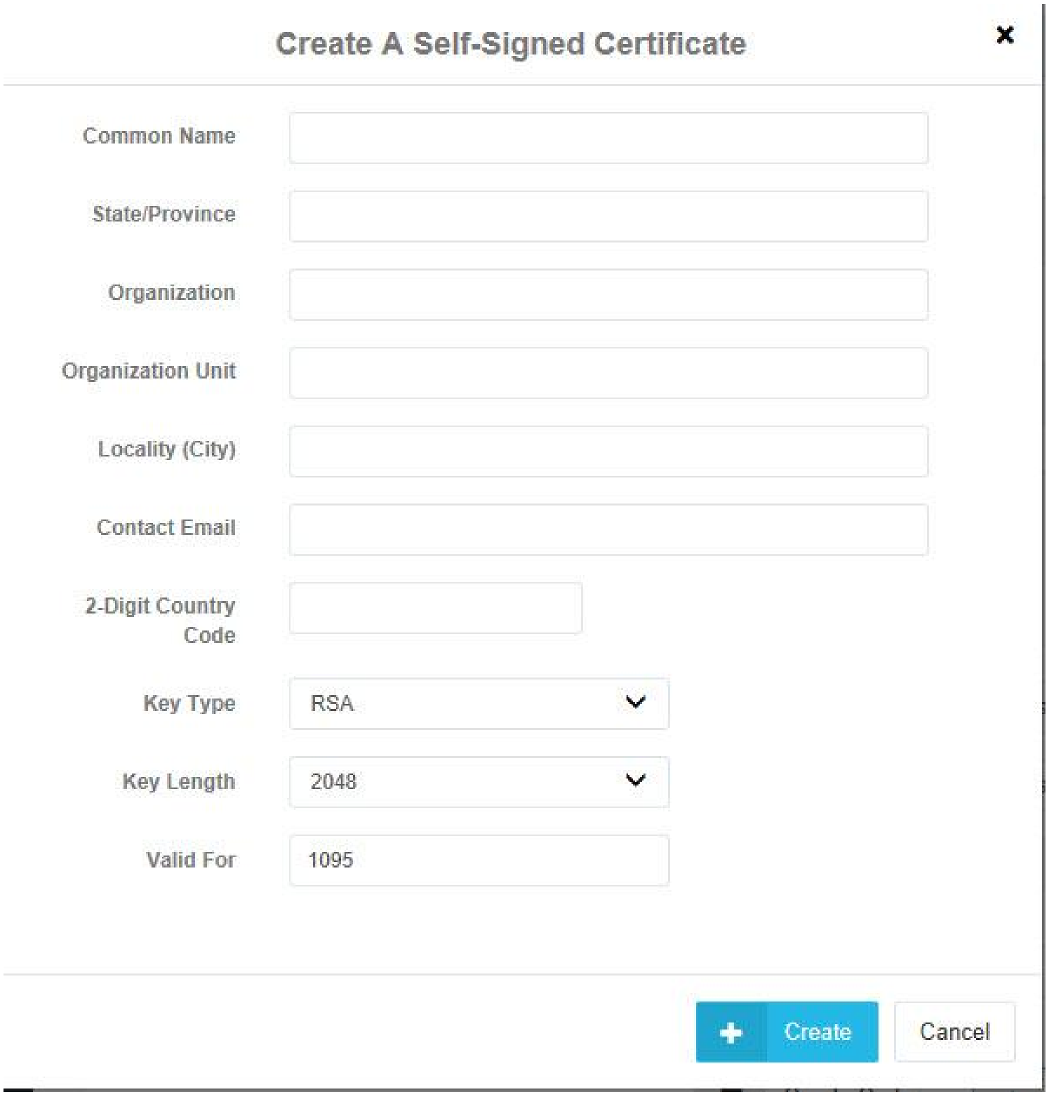
☐ Needs Key Password

CA Certificate Path: CA Local File Path (Optional)

Security settings page of the Server Manager

3. Click the **Create Self Signed Cert** button

4. A ”**Create a Self-Signed Certificate”** dialog will appear that asks for certificate details. The organization details that you use will be displayed to the FTP client user when they securely connect to your server. The Key Type should normally be RSA for maximum client compatibility. They Key Length value controls how strong the generated keys are and should normally be set to 2048. The default validity period for the certificate is 1095 days (3 years). Press the **Generate** button to create the certificate.



Create A Self-Signed Certificate

Common Name

State/Province

Organization

Organization Unit

Locality (City)

Contact Email

2-Digit Country Code

Key Type: RSA

Key Length: 2048

Valid For: 1095

Create a Self-Signed Certificate dialog

5. A self-signed certificate will be created and Cerberus will be automatically configured to use it.

6. Click **Ok** to close the Server Manager. If no certificate was previously being used then Cerberus will configure itself immediately to use the new certificate. You may need to restart the FTP server service if you were overwriting a previous certificate.

36.3 Using a Certificate created by a 3rd Party Certificate Authority

If your goal is to make sure that your clients can verify that the server they are connecting to is legitimate, and to ensure users don't see any warning messages about being "unable to verify the server", then you must use a certificate signed by a trusted certificate authority. You will have to contact one of the recognized Certificate Authorities such as Comodo, Thawte, Verisign, or one of the many other recognized Certificate Authorities and request a server certificate (for a price).

36.3.1 Steps to Import a 3rd Party Certificate:

1. Ensure that you have a digital certificate and private key in a format that Cerberus FTP Server understands. First, you will need to generate a new certificate (either by purchasing one from a public Certificate Authority, or you can install a Certificate Authority in your domain). You need to have a public certificate and a private key along with the passphrase for the private key.

2. Open the Server Manager by selecting the **Server Manager** item from the main menu.

3. Select the **Security** tab.

4. Under the Server Key Pair group, Click the **Certificate ...** button.

5. A file open dialog will appear that will allow you to select the public certificate provided by your certificate authority.

6. Under the Server Key Pair group, Click the **Private Key...** button.

7. A file open dialog will appear that will allow you to select the server's private key. If your public and private key are in the same file then set this path to be the same as the Certificate file path. *NOTE*: Cerberus understands both DER and PEM encoded certificate formats.

8. **Needs Key Password** - Check this option if the digital certificate is encrypted.

9. **Password** - If the digital certificate is encrypted then this is the password used to decrypt your digital certificate. The password is the same password you used to create the certificate request with your 3rd party certificate authority.

10. Click the **Verify** button to verify that Cerberus FTP Server can read the certificate and private key. If there are no errors then the certificate is valid and can be used by Cerberus.

11. Click **Ok** to close the Server Manager. If no certificate was previously being used or the certificate file path changed then Cerberus will configure itself immediately to use the new certificate.

12. Most CAs also provide a CA bundle/intermediate file that contains all of the intermediate CA certificates leading up to your signed certificate. If you plan to use the Cerberus Web Client, you will want to download and assign that file to the **CA File** field.

37.0 Clustering

37.1 Failover Clustering and Load Balancing

Cerberus FTP Server does not natively support clustering. However, using Active Directory or LDAP authentication, and a hardware or software load balancer (such as Microsoft NLB), you can achieve simple load balancing and failover with Active Directory or LDAP authenticated accounts.

To achieve Active Directory or LDAP-based load balancing, each Cerberus FTP Server machine is configured to point to the same AD or LDAP database, and requests can be load balanced to any of the available servers in that fashion. Many of our customers use such an arrangement for achieving simple failover and load balancing support.

Cerberus FTP Server Professional and Enterprise editions can now be configured to automatically synchronize all user accounts and settings to one or more other Cerberus servers. This new capability allows native Cerberus accounts, as well as customizations to Active Directory and LDAP authentication, to be easily synchronized across several Cerberus instances. Combining the new synchronization manager with shared storage between Cerberus FTP Server machines allows for multiple active backup and failover servers.

37.2 Load Balancing Exceptions

HTTP/S web client traffic cannot be load balanced using a simple connection balancer. The HTTP/S session database is local to each Cerberus machine, and any load balancer will have to ensure that all of the connections coming from a single IP are routed to the same Cerberus machine. We are working on a solution that will bring full clustering support to Cerberus FTP Server in the near future.

38.0 Web Services API Support

The Cerberus FTP Server Graphical User Interface (GUI) and underlying Windows Service uses a distributed remote protocol called SOAP for communication. The primary function of the SOAP API is to allow communication between these two services. However, we've made the API available so that anyone can use it to programmatically control the server.

Please note: The SOAP API can change between releases. We do try to maintain backwards compatibility, but sometimes we have to make breaking changes in the interest of improving the API. Always refer to the actual WSDL included with the Cerberus distribution you are using for the latest definitions.

38.1 What is SOAP and How Does Cerberus Use It?

38.1.1 What is SOAP?

SOAP is an acronym for Simple Object Access Protocol. SOAP is a method of describing operations that can be performed by a service. Many programming languages have tools that support SOAP, allowing developers to easily write programs and scripts to utilize services exposed through SOAP.

38.1.2 How Cerberus uses SOAP

Cerberus uses SOAP to define commands that can be issued to Cerberus FTP Server. Nearly everything that can be done from the Cerberus Administration GUI is described in Cerberus' SOAP API: adding, removing, and modifying users, groups, and interfaces, retrieving server statistics, and managing public shares, to name a few.

The complete API is described by two files, Cerberus.wsdl and ns1.xsd, both of which can be downloaded from your Cerberus FTP Server on the HTTPS Admin listener port. By default, the URLs are https://localhost:8443/wsdl/Cerberus.wsdl and https://localhost:8443/wsdl/ns1.xsd.

Together, these files define 87 operations and 65 object types. The list of supported operations (as of version 10.0.10.0) is listed at the bottom of the Understanding Cerberus SOAP API guide.

38.1.3 What you can do with SOAP

Cerberus SOAP API allows you to integrate Cerberus into your existing IT solutions. Using the API, Cerberus FTP Server can exchange data and events with the rest of the applications serving your users.

With the API you might:

- Write specialized tools for administering Cerberus FTP Server
- Perform mass changes to Cerberus users and groups
- Synchronize users and groups between Cerberus and other stores (e.g. Active Directory)

38.2 Available Features

Programmers can now access most of Cerberus FTP Server's common functions through a new Web Services interface. These services use SOAP 1.2 over HTTP or HTTPS and do not require a separate HTTP server. Cerberus FTP Server's implementation of Web Services includes a built-in, lightweight HTTP stack.

The following functionality is available through the Web Services API:

- Listing the current Cerberus FTP Server user and group accounts
- Adding new users or groups and modifying existing users and groups
- Deleting users or groups
- Retrieving user or group information
- Adding new virtual directories or modifying existing directories for a given user or group
- Deleting a virtual directory for a given user or group
- Getting the server's current started or stopped status
- Stopping or Starting the server
- Retrieving server statistics
- Retrieving and modifying interface details
- List, terminating, and blocking active connections
- Retrieving and saving configuration information

Refer to the included **Cerberus.wsdl** file for specifics on the Web Services interface to these functions. You can view an example Cerberus.wsdl online here.

38.3 Example SOAP applications

We have two example applications available that use the SOAP API. There is an example .NET project available here:

.NET SOAP Example Application

A newer, simpler WCF-based client application is available for download here:

WCF SOAP Client Example

38.4 Access URL

Make sure you enable SOAP access from the Remote settings page on the Server Manager. You can access the SOAP service WSDL on your local machine at the URL http://localhost:10001/wsdl/Cerberus.wsdl.

Make sure you have **Enable Web Administration** selected to view the actual WSDL. If Web Administration is not enabled you will still be able to use the WSDL to develop SOAP services but you won't be able to use the built-in web server to view the WSDL using the URL link. The WSDL is located in the installation directory where Cerberus is installed.

38.5 Security Considerations

By default, Cerberus FTP Server's Web Services access is turned off. Before allowing Web Services access to Cerberus FTP Server, you should be well aware of the security implication that this entails. While it is the user's responsibility to be knowledgeable of Web Services and the risks associated with using them, here are some reminders:

- Make sure the port you are running the Web service on is properly locked down. If you are only using Web Services to communicate between programs on the same machine, the port Cerberus is running the Web Services on shouldn't be accessible from outside of the local machine.

- When using Web Services, remember that anyone with access to the port that the Web Services is running on can send service requests to Cerberus FTP Server. This can represent a serious security risk. Make sure you set a strong Remote access password.

- HTTP, the backbone of Cerberus FTP Server's Web Services, transmits information as unencrypted text. Anything you send over HTTP has the potential to be intercepted and read. Cerberus also has the option of using SSL/TLS support for Web Services over HTTPS. Using HTTPS instead of HTTP significantly increased the security of any data transmitted.

Cerberus FTP Server uses the gSOAP toolkit to implement Web Services. You can find out more about gSOAP at the gSOAP home page.

38.6 Calling Cerberus SOAP API from PowerShell

38.6.1 Introduction

In this example, we use PowerShell to demonstrate calling Cerberus SOAP API. PowerShell’s inclusion in Windows and relatively simple syntax make it a natural starting point for experimentation and prototyping.

PowerShell expertise isn’t required to follow this guide. Nor is any experience with SOAP or XML. However, previous experience writing scripts in some shell language is recommended. Review Microsoft’s PowerShell documentation or a beginner’s guide if necessary.

Note: The example code has been tested with PowerShell version 5.1 and it may not run correctly on older versions.

38.6.2 Quick Start - HelloCerberus.ps1

The example script `HelloCerberus.ps1` calls **ServerInformation**, which requests basic information from Cerberus FTP Server. While the results are very simple, the code and concepts introduced are relevant to all Cerberus SOAP API operations.

To begin:

1. Open a PowerShell console on the same system hosting Cerberus FTP Server
2. Copy HelloCerberus.ps1 to the local hard drive of the same system
3. Paste the command below into the PowerShell console, hit Enter, and confirm by typing ‘Y’ and hitting Enter.

   ```
   Set-ExecutionPolicy -ExecutionPolicy Bypass -Scope Process
   ```

   See *Execution Policy* below for more information on this command

4. Type the ampersand character (&) into the console, then drag and drop HelloCerberus.ps1 from File Explorer the PowerShell console.
5. Hit Enter in the PowerShell console to execute the script.
6. Provide the username and password of the Cerberus primary administrator account when requested.

If successful, the console will contain basic information about the running Cerberus FTP Server. Hostname, status, and version information will be displayed:

```
PS C:\> Set-ExecutionPolicy -ExecutionPolicy Bypass -Scope Process
Execution Policy Change

The execution policy helps protect you from scripts that you do not trust.
Changing the execution policy might expose
you to the security risks described in the about_Execution_Policies help topic
at
https:/go.microsoft.com/fwlink/?LinkID=135170. Do you want to change the
execution policy?
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend [?] Help (default
is "N"): Y

PS C:\> &C:\Cerberus\Scripts\HelloCerberus.ps1


version            : CerberusFtp.Version
hostname           : DESKTOP-QFAOC1H
isStarted          : True
```

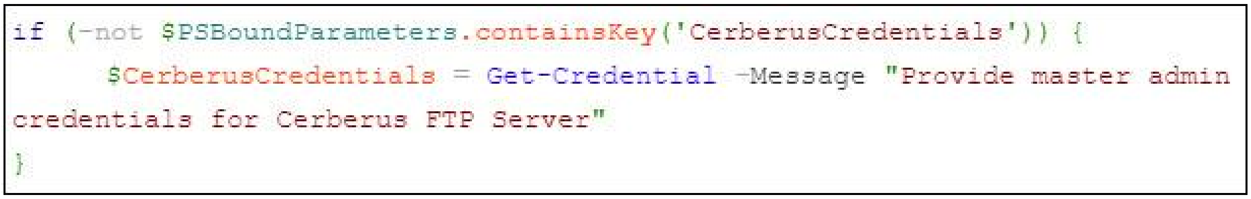
```
isSuccess          : True
isSuccessSpecified : True
message            :


maj   : 10
min   : 0
maint : 10
build : 0
```

38.6.3 Stepping Through the Code

What took place in HelloCerberus.ps1 can be summarized in five parts. Let’s take the most important lines of code from and examine them in detail.

1: Get Credentials

Every Cerberus SOAP operation requires credentials to authenticate the request. Since it is bad practice to store credentials directly within a script, HelloCerberus.ps1 either takes the credentials from the command-line or requests them interactively from the user.



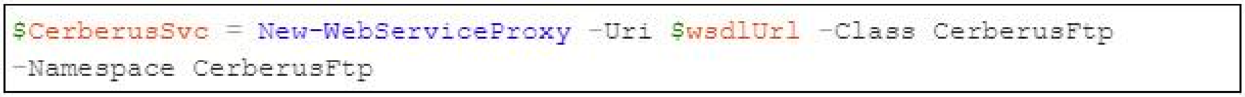
```
if (-not $PSBoundParameters.containsKey('CerberusCredentials')) {
      $CerberusCredentials = Get-Credential -Message "Provide master admin
credentials for Cerberus FTP Server"
}
```

The lines above check if credentials were provided on the command-line. If not, the Get-Credential command is called to request them from the user. Depending on the shell environment, the user may be presented with a pop-up window or a text input prompt. The result is stored in a variable named *$CerberusCredentials* to be used to authenticate later requests to Cerberus FTP Server.

PowerShell offers many options for storing credentials securely when authentication is required but user interaction is not possible. More complicated automation scenarios will need to use some of these techniques.

2: Create a SOAP Proxy

New-WebServiceProxy is used to read the SOAP API definition provided by Cerberus. It returns an object used by the script to make subsequent requests of Cerberus FTP Server.



```
$CerberusSvc = New-WebServiceProxy -Uri $wsdlUrl -Class CerberusFtp
-Namespace CerberusFtp
```

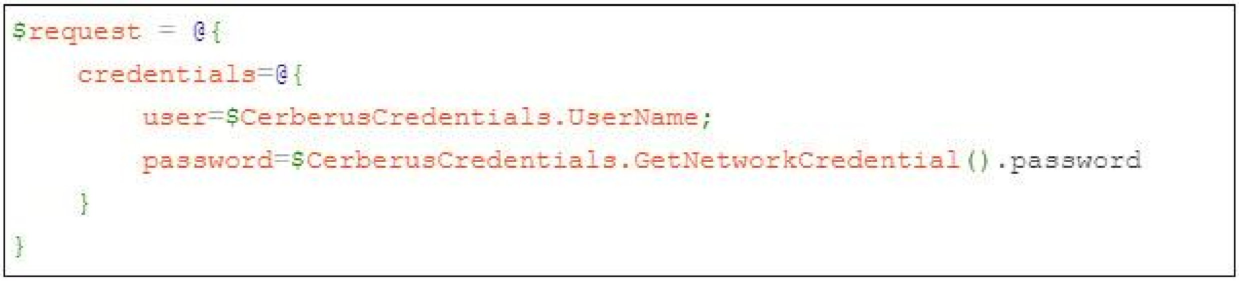
This seemingly simple command triggers a cascade of activity. The Cerberus SOAP API definition is retrieved from your Cerberus FTP Server. The definition is translated into dependent types which are placed in the CerberusFtp namespace of the current shell environment.

The *$CerberusSvc* variable stores a newly-created object with methods representing every API operation supported by Cerberus FTP Server. *$CerberusSvc* object is later used to issue requests to your running Cerberus FTP Server.

3: Prepare the Request

All Cerberus SOAP API operations follow the same pattern: a *request* object is sent to the server and a *response* object is received in reply. All operations require credentials, so all *request* objects will contain at least a username and password.

Most operations require additional information, like the name of a user or group when retrieving such objects, or a complete user or group object when making modifications.



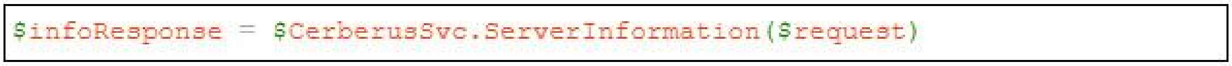
```
$request = @{
    credentials=@{
        user=$CerberusCredentials.UserName;
        password=$CerberusCredentials.GetNetworkCredential().password
    }
}
```

The lines above prepare a request object to call **ServerInformation**. Technically, this syntax creates a hash table. Because the hash table's names and values are consistent with a **ServerInformationRequest** object, the conversion is made automatically when we call **ServerInformation**.

The inner hash table, named credentials, is populated with the primary administrator account username and password. At this time, only the primary administrator account is allowed to make SOAP requests.

4: Send the Request

The *$CerberusSvc* object contains every operation available in Cerberus SOAP API, so making requests of Cerberus FTP Server is just a matter of calling methods on the object.

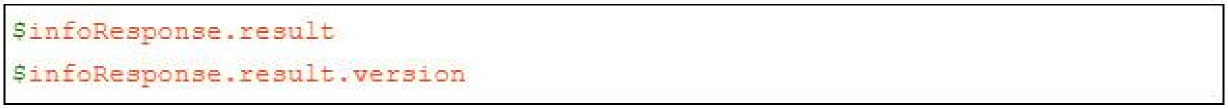


```
$infoResponse = $CerberusSvc.ServerInformation($request)
```

In the above line, we call **ServerInformation**, passing the *$request* object to the method. We store the result in a variable named *$infoResponse*.

5: Interpret the Response

Response objects vary from one operation to another, but generally, they contain a “result” value indicating success or failure of the operation and a “message” value containing details of the success or failure. If successful, additional data will be contained in the response.



```
$infoResponse.result
$infoResponse.result.version
```

In these lines, PowerShell emits the content our response object’s “result” member and the “version” information contained within the result.

The main logic of the script is complete. In relatively few lines of code, we’ve opened a connection to, authenticated with, and retrieved information from Cerberus FTP Server. The bulk of the work was handled by the .NET SOAP tools. See **Understanding Cerberus SOAP API** below for greater detail on how SOAP definitions relate to PowerShell code.

38.6.4 Security Considerations

A shrewd reader will note that many lines in HelloCerberus.ps1 were not detailed in the previous section. This code is used to ensure a successful experience when running this script against a new Cerberus FTP Server installation. This section and the comments throughout HelloCerberus.ps1 outline the reasons for this code.

PowerShell Execution Policy

Execution Policy is a Windows security feature that restricts PowerShell code based on its origin. The default settings restrict all script execution, so a change must be made before HelloCerberus.ps1 is allowed to run. The Quickstart instructions achieved this by running this line before executing the script:

```
Set-ExecutionPolicy -ExecutionPolicy Bypass -Scope Process
```

This command disables execution policy checking for the duration of the PowerShell process, but leaves the existing local machine and user policies unmodified. See Microsoft’s Set-ExecutionPolicy documentation for greater detail.

NOTE: In Windows Domain environments, Execution Policy may be controlled by system administrators via Group Policy. Consult with your domain administrators if Set-ExecutionPolicy is ineffective.

Securing Code

Ensuring the integrity of executable code is critical to system security. Scripted code is no different. Administrative scripts may run with elevated privileges, utilize sensitive credentials, and access critical resources. If the content of a script is compromised, then any credentials and resources used by the script are also compromised.

Bare minimum, use NTFS permissions to restrict write-access to scripts. Write-access should be as restricted as possible in production.

Ideally, a system to cryptographically sign scripts should be employed, but this is a significant undertaking. This may require coordination with your IT department to issue, deploy, and trust code-signing certificates. Implementing such a system is outside the scope of this guide.

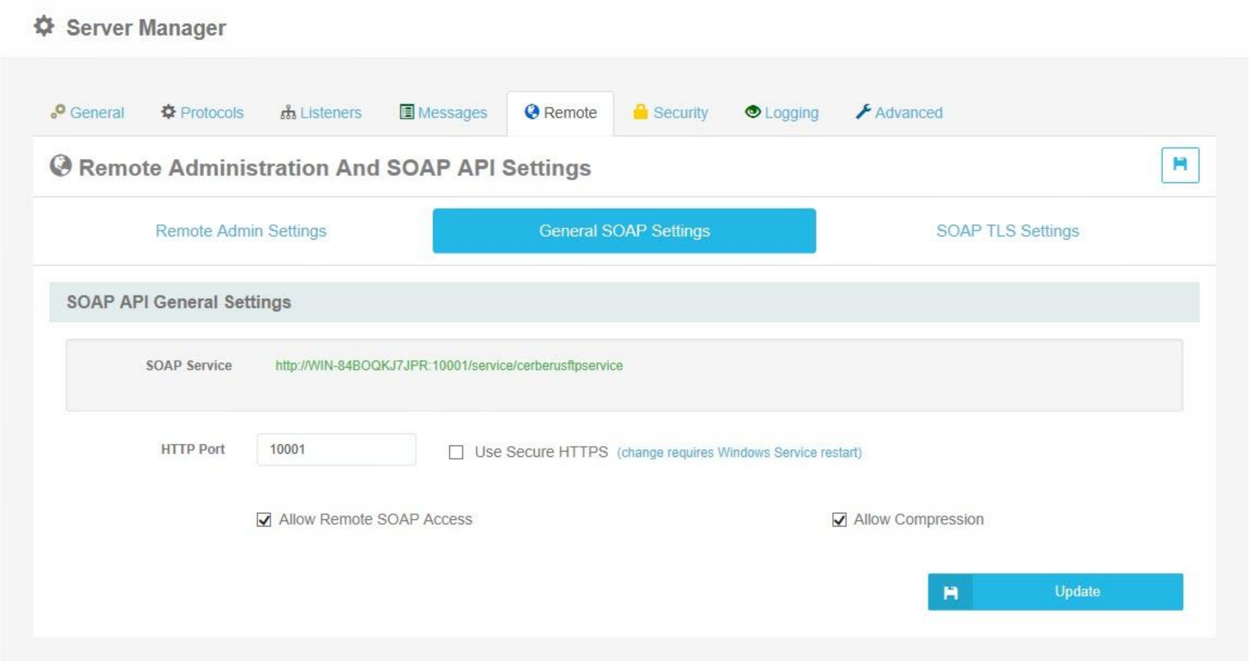
Self-Signed Certificate

PowerShell's default settings reject self-signed, expired, or otherwise misconfigured certificates when establishing an HTTPS connection. HelloCerberus.ps1 includes a function **Disable-CertificateValidation** to work around this restriction.

In production, however, this must not be used. Cerberus FTP Server should be configured with a legitimate certificate issued by a trusted certificate authority. Once in place, default certificate validation will succeed, eliminating the need for a workaround.

38.6.5 Securing Cerberus SOAP API Service Endpoint

Cerberus FTP Server configuration may be used to restrict or relax access to the SOAP service endpoint, according to needs. The configuration options are found under Configure, Remote tab, and SOAP Administration Settings.



SOAP Service http://WIN-84BOQKJ7JPR:10001/service/cerberusftpservice

HTTP Port 10001

□ Use Secure HTTPS (change requires Windows Service restart)

☑ Allow Remote SOAP Access

☑ Allow Compression

Use Secure HTTPS

Determines whether SOAP requests are accepted on HTTP or HTTPS.

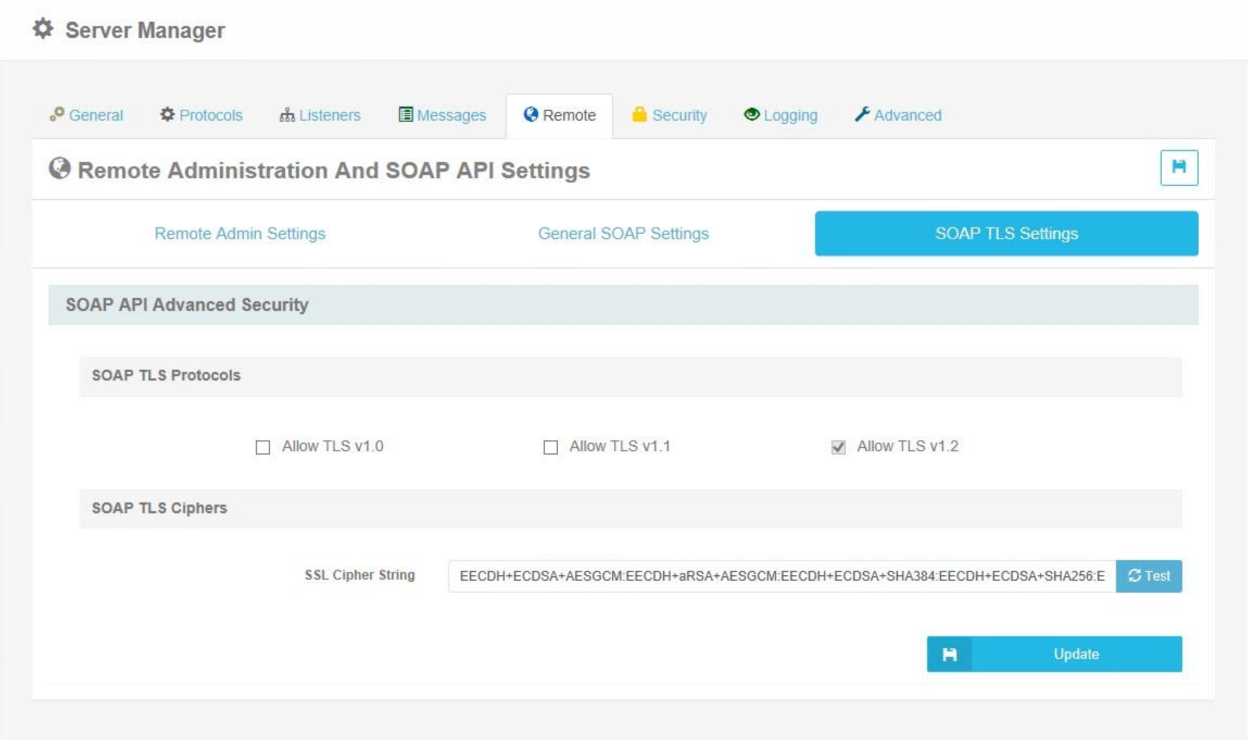
Recommended: HTTPS should be used whenever possible.

Allow Remote SOAP Access

When unchecked, SOAP requests must originate from the localhost (thus PowerShell requests must be run locally); Remote connections are refused.

Recommended: Unchecked unless remote SOAP clients are a business requirement.

SOAP TLS Protocols



SOAP API Advanced Security

SOAP TLS Protocols

☐ Allow TLS v1.0 ☐ Allow TLS v1.1 ☑ Allow TLS v1.2

SOAP TLS Ciphers

SSL Cipher String: EECDH+ECDSA+AESGCM:EECDH+aRSA+AESGCM:EECDH+ECDSA+SHA384:EECDH+ECDSA+SHA256:E

TLS 1.2 is the most secure option. Others are provided for compatibility with SOAP clients incapable of 1.2.

Recommended: TLS v1.2 enabled, others disabled.

38.6.6 Conclusion

The Cerberus SOAP API offers great potential for automating Cerberus FTP Server administration to those who need it. In the next installment, we'll perform operations more interesting than ServerInformation. Look forward to example code that adds, removes, and modifies Cerberus users.

38.7 Understanding Cerberus SOAP API

38.7.1 Introduction

In **Cerberus SOAP API with PowerShell** above, we used a small script to issue a simple command to Cerberus FTP Server. It is not necessary to completely understand SOAP to make use of Cerberus SOAP API. However, being casually aware of the infrastructure behind your code is a good idea. This document pulls back the curtain a bit, providing insight into how SOAP bridges the gap between PowerShell and your Cerberus FTP Server.

From WSDL to PowerShell

Cerberus.wsdl and *ns1.xsd* are in two XML formats, Web Service Definition Language and Xml Schema Definition. Generally, you will not need to read these files directly to know how to call SOAP APIs; The .NET toolchain automatically creates PowerShell object types according to the definitions in these files. As an exercise, however, we will trace the definitions for the **ServerInformation** operation used by HelloCerberus.ps1.

Here is the excerpt from *Cerberus.wsdl* which first defines the **ServerInformation** operation:

```
<operation name="ServerInformation">
  <documentation>
    Service definition of function tns__ServerInformation
  </documentation>
  <input message="tns:ServerInformationRequestMessage" />
  <output message="tns:ServerInformationResponseMessage" />
</operation>
```

The above fragment describes an operation named **ServerInformation** supported by Cerberus FTP Server. The operation takes a **ServerInformationRequestMessage** object and replies with a **ServerInformationResponseMessage**. To see what these message objects contain, the references must be followed within *Cerberus.wsdl*:

```
<message name="ServerInformationRequestMessage">
  <part name="in" element="tns:ServerInformationRequest" />
</message>
<message name="ServerInformationResponseMessage">
  <part name="out" element="tns:ServerInformationResponse" />
</message>
```

The messages are composed of **ServerInformationRequest** and **ServerInformationResponse** parts, respectively. These are further defined within *Cerberus.wsdl*:

```
<!-- operation request xsd:element -->
<xsd:element name="ServerInformationRequest">
```

```
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="ns1:AuthenticatedRequest">
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>



  <!-- operation response xsd:element -->
  <xsd:element name="ServerInformationResponse">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="result" type="ns1:ServerInformation" />
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
```

These lines describe the request and response objects as complex types, defined in the *ns1* namespace. The *ns1* namespace is defined near the top of *Cerberus.wsdl*:

```
  <xsd:schema
      targetNamespace="http://cerberusllc.com/service/cerberusftpservice"
      xmlns:ns1="http://cerberusllc.com/common"
      attributeFormDefault="qualified"
      elementFormDefault="qualified">
  <xsd:import
      namespace="http://cerberusllc.com/common"
      schemaLocation="./ns1.xsd" />
```

This fragment says further schema definitions can be found in external file, **ns1.xsd**. Within *ns1.xsd*, the definitions of the **AuthenticatedRequest** and **ServerInformation** types can be found:

```
  <xsd:complexType name="Credentials">
   <xsd:sequence>
     <xsd:element name="user" type="xsd:string" />
     <xsd:element name="password" type="xsd:string" />
   </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="AuthenticatedRequest">
   <xsd:sequence>
     <xsd:element name="credentials" type="ns1:Credentials" />
   </xsd:sequence>
  </xsd:complexType>



  <xsd:complexType name="ServerInformation">
    <xsd:sequence>
      <xsd:element name="version" type="ns1:Version" />
      <xsd:element name="hostname" type="xsd:string" />
      <xsd:element name="isStarted" type="xsd:boolean" />
    </xsd:sequence>
    <xsd:attribute name="isSuccess" type="xsd:boolean" use="optional" />
    <xsd:attribute name="message" type="xsd:string" use="optional" />
  </xsd:complexType>



  <xsd:complexType name="Version">
    <xsd:sequence>
    </xsd:sequence>
    <xsd:attribute name="maj" type="xsd:int" use="required" />
    <xsd:attribute name="min" type="xsd:int" use="required" />
    <xsd:attribute name="maint" type="xsd:int" use="required" />
    <xsd:attribute name="build" type="xsd:int" use="required" />
  </xsd:complexType>
```

With the fragment above, the definition is complete. The **AuthenticatedRequest** and **ServerInformation** types contain nested objects of **Credentials** and **Version types**, respectively.

When *New-WebServiceProxy* consumes the WSDL and XSD files, it generates corresponding .NET types, usable to PowerShell. You can run the *New-WebServiceProxy* command interactively to examine these types.

Here is an example:

```
PS C:\> $CerberusFtpSvc = New-WebServiceProxy -Uri
"https://localhost:8443/wsdl/cerberus.wsdl" -Namespace CerberusFtp -Class
CerberusFtp

PS C:\> [CerberusFtp.AuthenticatedRequest].DeclaredProperties | Select-Object
Name,PropertyType

Name        PropertyType
----        ------------
credentials CerberusFtp.Credentials

PS C:\> [CerberusFtp.ServerInformation].DeclaredProperties | Select-Object
Name,PropertyType

Name               PropertyType
----               ------------
version            CerberusFtp.Version
hostname           System.String
isStarted          System.Boolean
isSuccess          System.Boolean
isSuccessSpecified System.Boolean
message            System.String
```

The above console transcript first creates a new web service proxy object using *New-WebServiceProxy*. Passing the *-Namespace* parameter instructs the command to create necessary types within the “CerberusFtp” namespace.

The bracket syntax allows us to view and query type information of any type visible to PowerShell. Above, we examine declared properties of **AuthenticatedRequest** and **ServerInformation**. **ServerInformation** is nearly identical to its corresponding XML in ns1.xsd.

38.7.2 Where to Go From Here

Once *New-WebServiceProxy* is finished processing the WSDL file, everything you need to make SOAP requests is available within PowerShell. The *$CerberusFtpSvc* object can be examined using the built-in PowerShell command *Get-Member*.

Here is an example that yields the 87 basic Cerberus API operations on the proxy object:

```
       PS C:\>$CerberusFtpSvc |Get-Member -Type Method |Where-Object {$_.definition
       -like "CerberusFtp*" -and $_.name -notlike "Begin*" -and $_.name -notlike
       "End*"}
```

Pick and interesting one and start examining its associated request and response objects:

```
       PS C:\>$CerberusFtpSvc |Get-Member -Type Method -Name AddUser |fl *

       TypeName   : CerberusFtp.CerberusFTPService

       Name       : AddUser

       MemberType : Method

       Definition : CerberusFtp.AddUserResponse AddUser(CerberusFtp.AddUserRequest
       AddUserRequest)
```

Here, for instance, we have the *AddUser* method, which returns a **CerberusFtp.AddUserResponse** object and takes a **CerberusFtp.AddUserRequest** object. We can drill-down on these types to get an idea of what the *AddUser* will do and how to correctly call it:

```
       PS C:\> [CerberusFtp.AddUserRequest].DeclaredProperties |Select-Object
       Name,PropertyType

       Name                                  PropertyType
       ----                                  ------------
       User                                  CerberusFtp.User
       saveToDisk                            System.Nullable`1[System.Boolean]
       saveToDiskSpecified                   System.Boolean
       createNonExistentDirectories          System.Nullable`1[System.Boolean]
       createNonExistentDirectoriesSpecified System.Boolean
```

We see that AddUser requires a CerberusFtp.User object, so we can repeat the above to get insight into what kind of information a User object contains:

```
PS C:\> [CerberusFtp.User].DeclaredProperties |Select-Object Name,PropertyType

Name                              PropertyType
----                              ------------
password                          CerberusFtp.Password
```

```
isAllowPasswordChange             CerberusFtp.UserPropertyBool
isAnonymous                       CerberusFtp.UserPropertyBool
isSimpleDirectoryMode             CerberusFtp.UserPropertyBool
isDisabled                        CerberusFtp.UserPropertyBool
maxLoginsAllowed                  CerberusFtp.UserPropertyInt
requireSecureControl              CerberusFtp.UserPropertyBool
requireSecureData                 CerberusFtp.UserPropertyBool
disableAfterTime                  CerberusFtp.UserPropertyDateTime
authMethod                        CerberusFtp.UserPropertyAuthentication
protocols                         CerberusFtp.ProtocolsAllowed
maxUploadFilesize                 CerberusFtp.UserPropertyULong
ipAllowedList                     CerberusFtp.UserPropertyString
groupList                         CerberusFtp.groupMember[]
rootList                          CerberusFtp.VirtualDirectory[]
lastLogin                         System.DateTime
lastLoginSpecified                System.Boolean
createDate                        System.DateTime
createDateSpecified               System.Boolean
notifiedExpiringPassword          System.Boolean
notifiedExpiringPasswordSpecified System.Boolean
requirePasswordChange             System.Boolean
requirePasswordChangeSpecified    System.Boolean
email                             System.String
tel                               System.String
mobile                            System.String
desc                              System.String
fname                             System.String
sname                             System.String
name                              System.String
```

38.7.3 Conclusion

We've covered how the WSDL and XSD files describe Cerberus FTP Server's SOAP API, traced from those definitions to live .NET objects and types, and demonstrated how they are self-describing using PowerShell's Get-Member command. We've only scratched the surface on the *AddUser* method, as there will be other code examples and tutorials demonstrating its use.

Cerberus SOAP API Operations (as of version 12.x)

```
Name                                Definition
----                                ----------
AddDirectoryToGroup                 CerberusFtp.AddDirectoryToGroupResponse
AddDirectoryToGroup(CerberusFtp.AddDirectoryToGroupRequest AddDirectoryToGroupRequest)
AddDirectoryToUser                  CerberusFtp.AddDirectoryToUserResponse
AddDirectoryToUser(CerberusFtp.AddDirectoryToUserRequest AddDirectoryToUserRequest)
AddGroup                            CerberusFtp.AddGroupResponse
AddGroup(CerberusFtp.AddGroupRequest AddGroupRequest)
AddIp                               CerberusFtp.AddIpResponse AddIp(CerberusFtp.AddIpRequest
AddIpRequest)
AddUser                             CerberusFtp.AddUserResponse
AddUser(CerberusFtp.AddUserRequest AddUserRequest)
BackupServerConfiguration           CerberusFtp.BackupServerConfigurationResponse
BackupServerConfiguration(CerberusFtp.BackupServerConfigurationRequest
BackupServerConfigurationRequest)
```

```
BackupStatisticsDatabase            CerberusFtp.BackupStatisticsDatabaseResponse
BackupStatisticsDatabase(CerberusFtp.BackupStatisticsDatabaseRequest
BackupStatisticsDatabaseRequest)
BlockAddress                        CerberusFtp.BlockAddressResponse
BlockAddress(CerberusFtp.BlockAddressRequest BlockAddressRequest)
ChangePassword                      CerberusFtp.ChangePasswordResponse
ChangePassword(CerberusFtp.ChangePasswordRequest ChangePasswordRequest)
CommitSettings                      CerberusFtp.CommitSettingsResponse
CommitSettings(CerberusFtp.CommitSettingsRequest CommitSettingsRequest)
CreateDirectory                     CerberusFtp.CreateDirectoryResponse
CreateDirectory(CerberusFtp.CreateDirectoryRequest CreateDirectoryRequest)
CreateStatisticsDatabase            CerberusFtp.CreateStatisticsDatabaseResponse
CreateStatisticsDatabase(CerberusFtp.CreateStatisticsDatabaseRequest
CreateStatisticsDatabaseRequest)
CurrentStatus                       CerberusFtp.CurrentStatusResponse
CurrentStatus(CerberusFtp.CurrentStatusRequest CurrentStatusRequest)
DeleteDirectory                     CerberusFtp.DeleteDirectoryResponse
DeleteDirectory(CerberusFtp.DeleteDirectoryRequest DeleteDirectoryRequest)
DeleteDirectoryFromGroup            CerberusFtp.DeleteDirectoryFromGroupResponse
DeleteDirectoryFromGroup(CerberusFtp.DeleteDirectoryFromGroupRequest
DeleteDirectoryFromGroupRequest)
DeleteDirectoryFromUser             CerberusFtp.DeleteDirectoryFromUserResponse
DeleteDirectoryFromUser(CerberusFtp.DeleteDirectoryFromUserRequest
DeleteDirectoryFromUserRequest)
DeleteGroup                         CerberusFtp.DeleteGroupResponse
DeleteGroup(CerberusFtp.DeleteGroupRequest DeleteGroupRequest)
DeleteIp                            CerberusFtp.DeleteIpResponse
DeleteIp(CerberusFtp.DeleteIpRequest DeleteIpRequest)
DeletePublicShares                  CerberusFtp.DeletePublicSharesResponse
DeletePublicShares(CerberusFtp.DeletePublicSharesRequest DeletePublicSharesRequest)
DeleteRequestedAccounts             CerberusFtp.DeleteRequestedAccountsResponse
DeleteRequestedAccounts(CerberusFtp.DeleteRequestedAccountsRequest
DeleteRequestedAccountsRequest)
DeleteUser                          CerberusFtp.DeleteUserResponse
DeleteUser(CerberusFtp.DeleteUserRequest DeleteUserRequest)
DropStatisticsDatabase              CerberusFtp.DropStatisticsDatabaseResponse
DropStatisticsDatabase(CerberusFtp.DropStatisticsDatabaseRequest DropStatisticsDatabaseRequest)
EndAddDirectoryToGroup              CerberusFtp.AddDirectoryToGroupResponse
EndAddDirectoryToGroup(System.IAsyncResult asyncResult)
EndAddDirectoryToUser               CerberusFtp.AddDirectoryToUserResponse
EndAddDirectoryToUser(System.IAsyncResult asyncResult)
EndAddGroup                         CerberusFtp.AddGroupResponse EndAddGroup(System.IAsyncResult
asyncResult)
EndAddIp                            CerberusFtp.AddIpResponse EndAddIp(System.IAsyncResult
asyncResult)
EndAddUser                          CerberusFtp.AddUserResponse EndAddUser(System.IAsyncResult
asyncResult)
EndBackupServerConfiguration        CerberusFtp.BackupServerConfigurationResponse
EndBackupServerConfiguration(System.IAsyncResult asyncResult)
EndBackupStatisticsDatabase         CerberusFtp.BackupStatisticsDatabaseResponse
EndBackupStatisticsDatabase(System.IAsyncResult asyncResult)
EndBlockAddress                     CerberusFtp.BlockAddressResponse
EndBlockAddress(System.IAsyncResult asyncResult)
EndChangePassword                   CerberusFtp.ChangePasswordResponse
EndChangePassword(System.IAsyncResult asyncResult)
EndCommitSettings                   CerberusFtp.CommitSettingsResponse
EndCommitSettings(System.IAsyncResult asyncResult)
EndCreateDirectory                  CerberusFtp.CreateDirectoryResponse
EndCreateDirectory(System.IAsyncResult asyncResult)
EndCreateStatisticsDatabase         CerberusFtp.CreateStatisticsDatabaseResponse
EndCreateStatisticsDatabase(System.IAsyncResult asyncResult)
```

```
EndCurrentStatus                    CerberusFtp.CurrentStatusResponse
EndCurrentStatus(System.IAsyncResult asyncResult)
EndDeleteDirectory                  CerberusFtp.DeleteDirectoryResponse
EndDeleteDirectory(System.IAsyncResult asyncResult)
EndDeleteDirectoryFromGroup         CerberusFtp.DeleteDirectoryFromGroupResponse
EndDeleteDirectoryFromGroup(System.IAsyncResult asyncResult)
EndDeleteDirectoryFromUser          CerberusFtp.DeleteDirectoryFromUserResponse
EndDeleteDirectoryFromUser(System.IAsyncResult asyncResult)
EndDeleteGroup                      CerberusFtp.DeleteGroupResponse
EndDeleteGroup(System.IAsyncResult asyncResult)
EndDeleteIp                         CerberusFtp.DeleteIpResponse EndDeleteIp(System.IAsyncResult
asyncResult)
EndDeletePublicShares               CerberusFtp.DeletePublicSharesResponse
EndDeletePublicShares(System.IAsyncResult asyncResult)
EndDeleteRequestedAccounts          CerberusFtp.DeleteRequestedAccountsResponse
EndDeleteRequestedAccounts(System.IAsyncResult asyncResult)
EndDeleteUser                       CerberusFtp.DeleteUserResponse
EndDeleteUser(System.IAsyncResult asyncResult)
EndDropStatisticsDatabase           CerberusFtp.DropStatisticsDatabaseResponse
EndDropStatisticsDatabase(System.IAsyncResult asyncResult)
EndGenerateStatistics               CerberusFtp.GenerateStatisticsResponse
EndGenerateStatistics(System.IAsyncResult asyncResult)
EndGetAdminAccounts                 CerberusFtp.GetAdminAccountsResponse
EndGetAdminAccounts(System.IAsyncResult asyncResult)
EndGetAdminCustomSettings           CerberusFtp.GetAdminCustomSettingsResponse
EndGetAdminCustomSettings(System.IAsyncResult asyncResult)
EndGetAllCurrentConnectionCount     CerberusFtp.GetAllCurrentConnectionCountResponse
EndGetAllCurrentConnectionCount(System.IAsyncResult asyncResult)
EndGetAppPaths                      CerberusFtp.GetAppPathsResponse
EndGetAppPaths(System.IAsyncResult asyncResult)
EndGetAuthenticationList            CerberusFtp.GetAuthenticationListResponse
EndGetAuthenticationList(System.IAsyncResult asyncResult)
EndGetAutoBlockList                 CerberusFtp.GetAutoBlockListResponse
EndGetAutoBlockList(System.IAsyncResult asyncResult)
EndGetBackupServers                 CerberusFtp.GetBackupServersResponse
EndGetBackupServers(System.IAsyncResult asyncResult)
EndGetConfiguration                 CerberusFtp.GetConfigurationResponse
EndGetConfiguration(System.IAsyncResult asyncResult)
EndGetConnectedUserList             CerberusFtp.GetConnectedUserListResponse
EndGetConnectedUserList(System.IAsyncResult asyncResult)
EndGetCurrentBandwidth              CerberusFtp.GetCurrentBandwidthResponse
EndGetCurrentBandwidth(System.IAsyncResult asyncResult)
EndGetCurrentConnectionCount        CerberusFtp.GetCurrentConnectionCountResponse
EndGetCurrentConnectionCount(System.IAsyncResult asyncResult)
EndGetEventRules                    CerberusFtp.GetEventRulesResponse
EndGetEventRules(System.IAsyncResult asyncResult)
EndGetFeatures                      CerberusFtp.GetFeaturesResponse
EndGetFeatures(System.IAsyncResult asyncResult)
EndGetFileTransfers                 CerberusFtp.GetFileTransfersResponse
EndGetFileTransfers(System.IAsyncResult asyncResult)
EndGetFolderMonitors                CerberusFtp.GetFolderMonitorsResponse
EndGetFolderMonitors(System.IAsyncResult asyncResult)
EndGetGroupInformation              CerberusFtp.GetGroupInformationResponse
EndGetGroupInformation(System.IAsyncResult asyncResult)
EndGetGroupList                     CerberusFtp.GetGroupListResponse
EndGetGroupList(System.IAsyncResult asyncResult)
EndGetGroups                        CerberusFtp.GetGroupsResponse
EndGetGroups(System.IAsyncResult asyncResult)
EndGetHostname                      CerberusFtp.GetHostnameResponse
EndGetHostname(System.IAsyncResult asyncResult)
```

```
EndGetInterfaceByID                 CerberusFtp.GetInterfaceResponse
EndGetInterfaceByID(System.IAsyncResult asyncResult)
EndGetInterfaceList                 CerberusFtp.GetInterfaceListResponse
EndGetInterfaceList(System.IAsyncResult asyncResult)
EndGetInterfaces                    CerberusFtp.GetInterfacesResponse
EndGetInterfaces(System.IAsyncResult asyncResult)
EndGetIPBlockList                   CerberusFtp.GetIPBlockListResponse
EndGetIPBlockList(System.IAsyncResult asyncResult)
EndGetLicenseInfo                   CerberusFtp.GetLicenseInfoResponse
EndGetLicenseInfo(System.IAsyncResult asyncResult)
EndGetLogMessages                   CerberusFtp.GetLogMessagesResponse
EndGetLogMessages(System.IAsyncResult asyncResult)
EndGetMimeMappings                  CerberusFtp.GetMimeMappingsResponse
EndGetMimeMappings(System.IAsyncResult asyncResult)
EndGetProfiles                      CerberusFtp.GetProfilesResponse
EndGetProfiles(System.IAsyncResult asyncResult)
EndGetPublicShares                  CerberusFtp.GetPublicSharesResponse
EndGetPublicShares(System.IAsyncResult asyncResult)
EndGetRequestedAccounts             CerberusFtp.GetRequestedAccountsResponse
EndGetRequestedAccounts(System.IAsyncResult asyncResult)
EndGetSavedReports                  CerberusFtp.GetSavedReportsResponse
EndGetSavedReports(System.IAsyncResult asyncResult)
EndGetStatistics                    CerberusFtp.GetStatisticsResponse
EndGetStatistics(System.IAsyncResult asyncResult)
EndGetUserCustomSettings            CerberusFtp.GetUserCustomSettingsResponse
EndGetUserCustomSettings(System.IAsyncResult asyncResult)
EndGetUserInformation               CerberusFtp.GetUserInformationResponse
EndGetUserInformation(System.IAsyncResult asyncResult)
EndGetUserList                      CerberusFtp.GetUserListResponse
EndGetUserList(System.IAsyncResult asyncResult)
EndInitializeInterface              CerberusFtp.InitializeInterfaceResponse
EndInitializeInterface(System.IAsyncResult asyncResult)
EndInitializeServer                 CerberusFtp.InitializeServerResponse
EndInitializeServer(System.IAsyncResult asyncResult)
EndModifyInterface                  CerberusFtp.ModifyInterfaceResponse
EndModifyInterface(System.IAsyncResult asyncResult)
EndRenameGroup                      CerberusFtp.RenameGroupResponse
EndRenameGroup(System.IAsyncResult asyncResult)
EndRenameUser                       CerberusFtp.RenameUserResponse
EndRenameUser(System.IAsyncResult asyncResult)
EndRestoreServerConfiguration       CerberusFtp.RestoreServerConfigurationResponse
EndRestoreServerConfiguration(System.IAsyncResult asyncResult)
EndRestoreStatisticsDatabase        CerberusFtp.RestoreStatisticsDatabaseResponse
EndRestoreStatisticsDatabase(System.IAsyncResult asyncResult)
EndSaveBackupServers                CerberusFtp.SaveBackupServersResponse
EndSaveBackupServers(System.IAsyncResult asyncResult)
EndSaveBlockList                    CerberusFtp.SaveBlockListResponse
EndSaveBlockList(System.IAsyncResult asyncResult)
EndSaveConfiguration                CerberusFtp.SaveConfigurationResponse
EndSaveConfiguration(System.IAsyncResult asyncResult)
EndSaveMimeMappings                 CerberusFtp.SaveMimeMappingsResponse
EndSaveMimeMappings(System.IAsyncResult asyncResult)
EndSaveProfiles                     CerberusFtp.SaveProfilesResponse
EndSaveProfiles(System.IAsyncResult asyncResult)
EndServerInformation                CerberusFtp.ServerInformationResponse
EndServerInformation(System.IAsyncResult asyncResult)
EndServerStarted                    CerberusFtp.ServerStartedResponse
EndServerStarted(System.IAsyncResult asyncResult)
EndServerSummaryStatus              CerberusFtp.ServerSummaryStatusResponse
EndServerSummaryStatus(System.IAsyncResult asyncResult)
```

```
EndSetAdminAccounts                 CerberusFtp.SetAdminAccountsResponse
EndSetAdminAccounts(System.IAsyncResult asyncResult)
EndSetAdminCustomSettings           CerberusFtp.SetAdminCustomSettingsResponse
EndSetAdminCustomSettings(System.IAsyncResult asyncResult)
EndSetAuthenticationList            CerberusFtp.SetAuthenticationListResponse
EndSetAuthenticationList(System.IAsyncResult asyncResult)
EndSetEventRules                    CerberusFtp.SetEventRulesResponse
EndSetEventRules(System.IAsyncResult asyncResult)
EndSetFolderMonitors                CerberusFtp.SetFolderMonitorsResponse
EndSetFolderMonitors(System.IAsyncResult asyncResult)
EndSetPublicShares                  CerberusFtp.SetPublicSharesResponse
EndSetPublicShares(System.IAsyncResult asyncResult)
EndSetRequestedAccounts             CerberusFtp.SetRequestedAccountsResponse
EndSetRequestedAccounts(System.IAsyncResult asyncResult)
EndSetSavedReports                  CerberusFtp.SetSavedReportsResponse
EndSetSavedReports(System.IAsyncResult asyncResult)
EndSetUserCustomSettings            CerberusFtp.SetUserCustomSettingsResponse
EndSetUserCustomSettings(System.IAsyncResult asyncResult)
EndSetWANIP                         CerberusFtp.SetWANIPResponse EndSetWANIP(System.IAsyncResult
asyncResult)
EndSharePublicFile                  CerberusFtp.SharePublicFileResponse
EndSharePublicFile(System.IAsyncResult asyncResult)
EndShutdownConnectionsOnInterface   CerberusFtp.ShutdownConnectionsOnInterfaceResponse
EndShutdownConnectionsOnInterface(System.IAsyncResult asyncResult)
EndShutdownInterface                CerberusFtp.ShutdownInterfaceResponse
EndShutdownInterface(System.IAsyncResult asyncResult)
EndShutdownServer                   CerberusFtp.ShutdownServerResponse
EndShutdownServer(System.IAsyncResult asyncResult)
EndStartServer                      CerberusFtp.StartServerResponse
EndStartServer(System.IAsyncResult asyncResult)
EndStopServer                       CerberusFtp.StopServerResponse
EndStopServer(System.IAsyncResult asyncResult)
EndTerminateConnection              CerberusFtp.TerminateConnectionResponse
EndTerminateConnection(System.IAsyncResult asyncResult)
EndTestAndVerifyDatabase            CerberusFtp.TestAndVerifyDatabaseResponse
EndTestAndVerifyDatabase(System.IAsyncResult asyncResult)
EndVerifyLicense                    CerberusFtp.VerifyLicenseResponse
EndVerifyLicense(System.IAsyncResult asyncResult)
GenerateStatistics                  CerberusFtp.GenerateStatisticsResponse
GenerateStatistics(CerberusFtp.GenerateStatisticsRequest GenerateStatisticsRequest)
GetAdminAccounts                    CerberusFtp.GetAdminAccountsResponse
GetAdminAccounts(CerberusFtp.GetAdminAccountsRequest GetAdminAccountsRequest)
GetAdminCustomSettings              CerberusFtp.GetAdminCustomSettingsResponse
GetAdminCustomSettings(CerberusFtp.GetAdminCustomSettingsRequest GetAdminCustomSettingsRequest)
GetAllCurrentConnectionCount        CerberusFtp.GetAllCurrentConnectionCountResponse
GetAllCurrentConnectionCount(CerberusFtp.GetAllCurrentConnectionCountRequest
GetAllCurrentConnectionCountRequest)
GetAppPaths                         CerberusFtp.GetAppPathsResponse
GetAppPaths(CerberusFtp.GetAppPathsRequest GetAppPathsRequest)
GetAuthenticationList               CerberusFtp.GetAuthenticationListResponse
GetAuthenticationList(CerberusFtp.GetAuthenticationListRequest GetAuthenticationListRequest)
GetAutoBlockList                    CerberusFtp.GetAutoBlockListResponse
GetAutoBlockList(CerberusFtp.GetAutoBlockListRequest GetAutoBlockListRequest)
GetBackupServers                    CerberusFtp.GetBackupServersResponse
GetBackupServers(CerberusFtp.GetBackupServersRequest GetBackupServersRequest)
GetConfiguration                    CerberusFtp.GetConfigurationResponse
GetConfiguration(CerberusFtp.GetConfigurationRequest GetConfigurationRequest)
GetConnectedUserList                CerberusFtp.GetConnectedUserListResponse
GetConnectedUserList(CerberusFtp.GetConnectedUserListRequest GetConnectedUserListRequest)
GetCurrentBandwidth                 CerberusFtp.GetCurrentBandwidthResponse
GetCurrentBandwidth(CerberusFtp.GetCurrentBandwidthRequest GetCurrentBandwidthRequest)
```

```
GetCurrentConnectionCount           CerberusFtp.GetCurrentConnectionCountResponse
GetCurrentConnectionCount(CerberusFtp.GetCurrentConnectionCountRequest
GetCurrentConnectionCountRequest)
GetEventRules                       CerberusFtp.GetEventRulesResponse
GetEventRules(CerberusFtp.GetEventRulesRequest GetEventRulesRequest)
GetFeatures                         CerberusFtp.GetFeaturesResponse
GetFeatures(CerberusFtp.GetFeaturesRequest GetFeaturesRequest)
GetFileTransfers                    CerberusFtp.GetFileTransfersResponse
GetFileTransfers(CerberusFtp.GetFileTransfersRequest GetFileTransfersRequest)
GetFolderMonitors                   CerberusFtp.GetFolderMonitorsResponse
GetFolderMonitors(CerberusFtp.GetFolderMonitorsRequest GetFolderMonitorsRequest)
GetGroupInformation                 CerberusFtp.GetGroupInformationResponse
GetGroupInformation(CerberusFtp.GetGroupInformationRequest GetGroupInformationRequest)
GetGroupList                        CerberusFtp.GetGroupListResponse
GetGroupList(CerberusFtp.GetGroupListRequest GetGroupListRequest)
GetGroups                           CerberusFtp.GetGroupsResponse
GetGroups(CerberusFtp.GetGroupsRequest GetGroupsRequest)
GetHostname                         CerberusFtp.GetHostnameResponse
GetHostname(CerberusFtp.GetHostnameRequest GetHostnameRequest)
GetInterfaceByID                    CerberusFtp.GetInterfaceResponse
GetInterfaceByID(CerberusFtp.GetInterfaceByIDRequest GetInterfaceByIDRequest)
GetInterfaceList                    CerberusFtp.GetInterfaceListResponse
GetInterfaceList(CerberusFtp.GetInterfaceListRequest GetInterfaceListRequest)
GetInterfaces                       CerberusFtp.GetInterfacesResponse
GetInterfaces(CerberusFtp.GetInterfacesRequest GetInterfacesRequest)
GetIPBlockList                      CerberusFtp.GetIPBlockListResponse
GetIPBlockList(CerberusFtp.GetIPBlockListRequest GetIPBlockListRequest)
GetLicenseInfo                      CerberusFtp.GetLicenseInfoResponse
GetLicenseInfo(CerberusFtp.GetLicenseInfoRequest GetLicenseInfoRequest)
GetLogMessages                      CerberusFtp.GetLogMessagesResponse
GetLogMessages(CerberusFtp.GetLogMessagesRequest GetLogMessagesRequest)
GetMimeMappings                     CerberusFtp.GetMimeMappingsResponse
GetMimeMappings(CerberusFtp.GetMimeMappingsRequest GetMimeMappingsRequest)
GetProfiles                         CerberusFtp.GetProfilesResponse
GetProfiles(CerberusFtp.GetProfilesRequest GetProfilesRequest)
GetPublicShares                     CerberusFtp.GetPublicSharesResponse
GetPublicShares(CerberusFtp.GetPublicSharesRequest GetPublicSharesRequest)
GetRequestedAccounts                CerberusFtp.GetRequestedAccountsResponse
GetRequestedAccounts(CerberusFtp.GetRequestedAccountsRequest GetRequestedAccountsRequest)
GetSavedReports                     CerberusFtp.GetSavedReportsResponse
GetSavedReports(CerberusFtp.GetSavedReportsRequest GetSavedReportsRequest)
GetStatistics                       CerberusFtp.GetStatisticsResponse
GetStatistics(CerberusFtp.GetStatisticsRequest GetStatisticsRequest)
GetUserCustomSettings               CerberusFtp.GetUserCustomSettingsResponse
GetUserCustomSettings(CerberusFtp.GetUserCustomSettingsRequest GetUserCustomSettingsRequest)
GetUserInformation                  CerberusFtp.GetUserInformationResponse
GetUserInformation(CerberusFtp.GetUserInformationRequest GetUserInformationRequest)
GetUserList                         CerberusFtp.GetUserListResponse
GetUserList(CerberusFtp.GetUserListRequest GetUserListRequest)
InitializeInterface                 CerberusFtp.InitializeInterfaceResponse
InitializeInterface(CerberusFtp.InitializeInterfaceRequest InitializeInterfaceRequest)
InitializeServer                    CerberusFtp.InitializeServerResponse
InitializeServer(CerberusFtp.InitializeServerRequest InitializeServerRequest)
ModifyInterface                     CerberusFtp.ModifyInterfaceResponse
ModifyInterface(CerberusFtp.ModifyInterfaceRequest ModifyInterfaceRequest)
RenameGroup                         CerberusFtp.RenameGroupResponse
RenameGroup(CerberusFtp.RenameGroupRequest RenameGroupRequest)
RenameUser                          CerberusFtp.RenameUserResponse
RenameUser(CerberusFtp.RenameUserRequest RenameUserRequest)
```

```
RestoreServerConfiguration          CerberusFtp.RestoreServerConfigurationResponse
RestoreServerConfiguration(CerberusFtp.RestoreServerConfigurationRequest
RestoreServerConfigurationRequest)
RestoreStatisticsDatabase           CerberusFtp.RestoreStatisticsDatabaseResponse
RestoreStatisticsDatabase(CerberusFtp.RestoreStatisticsDatabaseRequest
RestoreStatisticsDatabaseRequest)
SaveBackupServers                   CerberusFtp.SaveBackupServersResponse
SaveBackupServers(CerberusFtp.SaveBackupServersRequest SaveBackupServersRequest)
SaveBlockList                       CerberusFtp.SaveBlockListResponse
SaveBlockList(CerberusFtp.SaveBlockListRequest SaveBlockListRequest)
SaveConfiguration                   CerberusFtp.SaveConfigurationResponse
SaveConfiguration(CerberusFtp.SaveConfigurationRequest SaveConfigurationRequest)
SaveMimeMappings                    CerberusFtp.SaveMimeMappingsResponse
SaveMimeMappings(CerberusFtp.SaveMimeMappingsRequest SaveMimeMappingsRequest)
SaveProfiles                        CerberusFtp.SaveProfilesResponse
SaveProfiles(CerberusFtp.SaveProfilesRequest SaveProfilesRequest)
ServerInformation                   CerberusFtp.ServerInformationResponse
ServerInformation(CerberusFtp.ServerInformationRequest ServerInformationRequest)
ServerStarted                       CerberusFtp.ServerStartedResponse
ServerStarted(CerberusFtp.ServerStartedRequest ServerStartedRequest)
ServerSummaryStatus                 CerberusFtp.ServerSummaryStatusResponse
ServerSummaryStatus(CerberusFtp.ServerSummaryStatusRequest ServerSummaryStatusRequest)
SetAdminAccounts                    CerberusFtp.SetAdminAccountsResponse
SetAdminAccounts(CerberusFtp.SetAdminAccountsRequest SetAdminAccountsRequest)
SetAdminCustomSettings              CerberusFtp.SetAdminCustomSettingsResponse
SetAdminCustomSettings(CerberusFtp.SetAdminCustomSettingsRequest SetAdminCustomSettingsRequest)
SetAuthenticationList               CerberusFtp.SetAuthenticationListResponse
SetAuthenticationList(CerberusFtp.SetAuthenticationListRequest SetAuthenticationListRequest)
SetEventRules                       CerberusFtp.SetEventRulesResponse
SetEventRules(CerberusFtp.SetEventRulesRequest SetEventRulesRequest)
SetFolderMonitors                   CerberusFtp.SetFolderMonitorsResponse
SetFolderMonitors(CerberusFtp.SetFolderMonitorsRequest SetFolderMonitorsRequest)
SetPublicShares                     CerberusFtp.SetPublicSharesResponse
SetPublicShares(CerberusFtp.SetPublicSharesRequest SetPublicSharesRequest)
SetRequestedAccounts                CerberusFtp.SetRequestedAccountsResponse
SetRequestedAccounts(CerberusFtp.SetRequestedAccountsRequest SetRequestedAccountsRequest)
SetSavedReports                     CerberusFtp.SetSavedReportsResponse
SetSavedReports(CerberusFtp.SetSavedReportsRequest SetSavedReportsRequest)
SetUserCustomSettings               CerberusFtp.SetUserCustomSettingsResponse
SetUserCustomSettings(CerberusFtp.SetUserCustomSettingsRequest SetUserCustomSettingsRequest)
SetWANIP                            CerberusFtp.SetWANIPResponse
SetWANIP(CerberusFtp.SetWANIPRequest SetWANIPRequest)
SharePublicFile                     CerberusFtp.SharePublicFileResponse
SharePublicFile(CerberusFtp.SharePublicFileRequest SharePublicFileRequest)
ShutdownConnectionsOnInterface      CerberusFtp.ShutdownConnectionsOnInterfaceResponse
ShutdownConnectionsOnInterface(CerberusFtp.ShutdownConnectionsOnInterfaceRequest
ShutdownConnectionsOnInterfaceRequest)
ShutdownInterface                   CerberusFtp.ShutdownInterfaceResponse
ShutdownInterface(CerberusFtp.ShutdownInterfaceRequest ShutdownInterfaceRequest)
ShutdownServer                      CerberusFtp.ShutdownServerResponse
ShutdownServer(CerberusFtp.ShutdownServerRequest ShutdownServerRequest)
StartServer                         CerberusFtp.StartServerResponse
StartServer(CerberusFtp.StartServerRequest StartServerRequest)
StopServer                          CerberusFtp.StopServerResponse
StopServer(CerberusFtp.StopServerRequest StopServerRequest)
TerminateConnection                 CerberusFtp.TerminateConnectionResponse
TerminateConnection(CerberusFtp.TerminateConnectionRequest TerminateConnectionRequest)
TestAndVerifyDatabase               CerberusFtp.TestAndVerifyDatabaseResponse
TestAndVerifyDatabase(CerberusFtp.TestAndVerifyDatabaseRequest TestAndVerifyDatabaseRequest)
VerifyLicense                       CerberusFtp.VerifyLicenseResponse
VerifyLicense(CerberusFtp.VerifyLicenseRequest VerifyLicenseRequest)
```

38.8 Cerberus User Modifications with PowerShell

38.8.1 Introduction

So far, we've demonstrated how to connect to Cerberus FTP Server and make SOAP API calls using PowerShell. We've also explored how PowerShell interfaces with the SOAP API via WSDL.

Now we demonstrate making modifications to Cerberus' native user repository. Example-UserManipulation.ps1 creates a user, lists all Cerberus users, changes our user's email address and password, adds a virtual directory to our user, and finally delete them.

We assume you've reviewed previous guides in this series and have successfully run HelloCerberus.ps1. You should already know how to run PowerShell scripts and know when to change PS execution policy. You'll once again need the URL to *Cerberus.wsdl*, served by your Cerberus FTP Server.

As before, we'll start by running the script, then step through the most significant parts of the script.

38.8.2 Running Example-UserManipulation.ps1

Since our example script makes modifications to the Cerberus User store, it is best not to run it against your production Cerberus environment; we **strongly** recommend that you use a separate instance of Cerberus for testing.

1. Download the script
2. Open a PowerShell console to the downloaded location
3. Run the script

```
PS C:\> & .\Example-UserManipulation.ps1 -EnableTls12 -DisableCertValidation
```

If all went well, you'll see something like this in the PowerShell console:

```
PS C:\> & .\Example-UserManipulation.ps1 -EnableTls12 -DisableCertValidation
Windows PowerShell credential request.
Provide master admin credentials for Cerberus FTP Server
User: Admin
Password for user Admin: *****************
Successfully created user PsSOAPTestUser
Successfully retrieved list of users
PsSOAPTestUser
PsSOAPTestUser exists in the list of users
Successfully updated email address of PsSOAPTestUser
Successfully changed password for PsSOAPTestUser
Successfully added NewRoot to PsSOAPTestUser
Successfully deleted PsSOAPTestUser
```

Code Walk-Through

Let's review each section of this script.

Note that the style of this script differs from HelloCerberus.ps1. Objects are explicitly created and their storage variables are type-constrained with the bracket syntax. This results more verbose expressions like:

```
[CerberusFtp.User] $newUser = New-Object -TypeName CerberusFtp.User
```

We've found, though, that this syntax seems to work better with PowerShell's code-completion features. Hopefully this makes it easier to integrate snippets of this code into your own scripts.

38.8.3 Setup SOAP Connection

This is the same code used in HelloCerberus.ps1. Cerberus credentials are requested if not provided. TLS 1.2 and certificate validation are enabled or disabled according to parameters passed to the script. The Web Service Proxy object is created. The only significant difference is the addition of the `$EnableTls12` and `$DisableCertValidation` switches:

```
# Collect credentials if not provided in parameters
if (-not $PSBoundParameters.containsKey('CerberusCredentials')) {
    $CerberusCredentials = Get-Credential -Message "Provide master admin
credentials for Cerberus FTP Server"
}

if ($EnableTls12) {
    [Net.ServicePointManager]::SecurityProtocol =
[Net.SecurityProtocolType]::Tls12
}

if ($DisableCertValidation) {
    if (-not("dummy" -as [type])) {
        add-type -TypeDefinition @"
    using System;
    using System.Net;
    using System.Net.Security;
    using System.Security.Cryptography.X509Certificates;

    public static class Dummy {
        public static bool ReturnTrue(object sender,
            X509Certificate certificate,
            X509Chain chain,
            SslPolicyErrors sslPolicyErrors) { return true; }

        public static RemoteCertificateValidationCallback GetDelegate() {
            return new RemoteCertificateValidationCallback(Dummy.ReturnTrue);
        }
    }
"@
    }
    [System.Net.ServicePointManager]::ServerCertificateValidationCallback =
[dummy]::GetDelegate()
}

# Create Web Service Proxy object and CerberusFtp data-types
$CerberusSvc = New-WebServiceProxy -Uri $WSDLUrl -Class CerberusFtp -Namespace
CerberusFtp

# Override default SOAP endpoint if provided in parameters
if ($PSBoundParameters.ContainsKey('CerberusServiceUrl')){
    $CerberusSvc.Url = $CerberusServiceUrl
}
```

38.8.4 Create a New Test User

To create a user, invoke the *AddUser* operation passing a *CerberusFtp.AddUserRequest* object containing a *CerberusFtp.User* object. The User object must be populated with all of the properties you'd like for the user entry.

As always, create a request object corresponding to the operation we're about to invoke:

```
# Create new AddUserRequest object
[CerberusFtp.AddUserRequest] $addUserRequest = New-Object -TypeName
CerberusFtp.AddUserRequest
```

Every request needs credentials to authenticate:

```
# Populate request object with Cerberus Admin credentials
$addUserRequest.credentials = New-Object -TypeName CerberusFtp.Credentials
$addUserRequest.credentials.user = $CerberusCredentials.UserName
$addUserRequest.credentials.password =
$CerberusCredentials.GetNetworkCredential().Password
```

The last bit of information the *AddUser* operation requires is a user. Create an object of type *CerberusFtp.User* and populate relevant properties:

```
# Create new User object
[CerberusFtp.User] $newUser = New-Object -TypeName CerberusFtp.User

# Populate user object with user details
$newUser.name = $newTestUserName
$newUser.password = New-Object -TypeName CerberusFtp.Password
$newUser.password.value = "TestPasswordChangeImmediately1234!@#$"
$newUser.requirePasswordChange = $true
$newUser.fname = "NewUserFrom"
$newUser.sname = "PowerShell"
$newUser.email = "NewTestUser@powershellExample.net"
$newUser.desc = "This user was created from PowerShell using SOAP"

# Test account not allowed to change its own password
$newUser.isAllowPasswordChange = New-Object -TypeName
CerberusFtp.UserPropertyBool
```

CerberusFtp.UserPropertyBool belongs to a family of types that deal with how user properties interact with the group membership. We'll cover this in more detail when demonstrating group operations. For now, just set the *.value* to *$false* and *.valueSpecified* properties to *$true*:

```
$newUser.isAllowPasswordChange.value = $false
$newUser.isAllowPasswordChange.valueSpecified = $true
```

Repeat with the *.isDisabled* attribute to ensure no one may login to our new test account:

```
# Test account disabled
$newUser.isDisabled = New-Object -TypeName CerberusFtp.UserPropertyBool
$newUser.isDisabled.value = $true
$newUser.isDisabled.valueSpecified = $true
```

Now that the user object is created, we copy it to the request object and invoke the AddUser operation:

```
# Populate request object with new user object
$addUserRequest.User = $newUser

# Issue the AddUser request
[CerberusFtp.AddUserResponse] $addUserResponse =
$CerberusSvc.AddUser($addUserRequest)
```

Finally, we test the result of the operation and display feedback accordingly:

```
# Check response for success or failure
if (-not $addUserResponse.result){
    Write-Error "Failed to create user: $($addUserResponse.message)"
} else {
    Write-Host "Successfully created user $newTestUserName"
}
```

38.8.5 Get a list of All Cerberus Users

Now that we've created a new user, we can request a list of users from Cerberus FTP Server and confirm that our new user exists in the list.

Once again, every operation must have a corresponding request object populated with admin credentials. Going forward, we'll skip details for concepts we've already covered.

```
# Create new GetUserListRequest object
[CerberusFtp.GetUserListRequest] $getUserListRequest = New-Object
CerberusFtp.GetUserListRequest

# Populate request object with Cerberus Admin credentials
$getUserListRequest.credentials = New-Object -TypeName CerberusFtp.Credentials
$getUserListRequest.credentials.user = $CerberusCredentials.UserName
$getUserListRequest.credentials.password =
$CerberusCredentials.GetNetworkCredential().Password
```

The list of usernames is sent in the *$getUserListResponse.UserList* property. In PowerShell, it appears as an array of strings. We can use the *-contains* operator to test the contents of the array for the name of our new user:

```
[CerberusFtp.GetUserListResponse] $getUserListResponse =
$CerberusSvc.GetUserList($getUserListRequest)

# Check response for success or failure
if (-not $getUserListResponse.result){
    Write-Error "Failed to retrieve user list: $($getUserListResponse.message)"
} else {
    Write-Host "Successfully created retrieved list of users"
    Write-Host $getUserListResponse.UserList
    if ($getUserListResponse.UserList -contains $newTestUserName){
        Write-Host "$newTestUsername exists in the list of users"
    } else {
        Write-Error "$newTestUsername was not found in the list of users"
    }
}
```

38.8.6 Modify Email Address of a User

Modifying a user is a compound operation.

1. Retrieve the user with *getUserInformation*
2. Modify the local copy of the user
3. Invoke *AddUser* to overwrite the user

```
# Create new GetUserInformationRequest object
[CerberusFtp.GetUserInformationRequest] $getUserInformationRequest =
New-Object CerberusFtp.GetUserInformationRequest

# Populate request object with Cerberus Admin credentials
$getUserInformationRequest.credentials = New-Object -TypeName
CerberusFtp.Credentials
$getUserInformationRequest.credentials.user = $CerberusCredentials.UserName
$getUserInformationRequest.credentials.password =
$CerberusCredentials.GetNetworkCredential().Password

# Populate request object with the username to retrieve
$getUserInformationRequest.userName = $newTestUserName

# Issue the getUserInformation request
[CerberusFtp.GetUserInformationResponse] $getUserInformationResponse =
$CerberusSvc.getUserInformation($getUserInformationRequest)
```

We check the result of the response to make sure we found an existing user account:

```
# Check response for success or failure
if (-not $getUserInformationResponse.result){
    Write-Error "Failed to retrieve user:
$($getUserInformationResponse.message)"
```

```
        } else {
```

Then we change the email address for this user object:

```
        [CerberusFtp.User] $userToModify = $getUserInformationResponse.UserInformation
        $userToModify.email = "NewEmailAddress@powershellExample.net"
```

We use *AddUser* and its corresponding request type *AddUserRequest* to both create users and modify existing users. If the userName matches that of an existing user, the existing user is overwritten. We've named the request object *$modifyUserRequest* to express our intentions clearly:

```
        # Populate request with modified user object
        [CerberusFtp.AddUserRequest] $modifyUserRequest = New-Object
        CerberusFtp.AddUserRequest

        # Populate an AddUserRequest object with the modified user object
        $modifyUserRequest.credentials = New-Object CerberusFtp.Credentials
        $modifyUserRequest.credentials.user = $CerberusCredentials.UserName
        $modifyUserRequest.credentials.password =
        $CerberusCredentials.GetNetworkCredential().Password

        # Copy the newly-modified user object to the $modifyUserRequest object
        $modifyUserRequest.User = $userToModify

        # Issue AddUser request to modify existing user
        [CerberusFtp.AddUserResponse] $modifyUserResponse =
        $CerberusSvc.AddUser($modifyUserRequest)

        # Check response for success or failure
        if (-not $modifyUserResponse.result){
            Write-Error "Failed to update user: $($modifyUserResponse.message)"
        } else {
            Write-Host "Successfully updated email address of $($userToModify.name)"
        }
}
```

38.8.7 Change Password of a User

The password is just another property of the *User* object. You could modify it in the same fashion we modified the *.email* property. However, password resets are frequent enough that a dedicated operation is provided. This reduces complexity of copying the whole user object from server to client and back again.

Use the *ChangePassword* operation along with *ChangePasswordRequest*:

```
        # Create new ChangePasswordRequest object
        [CerberusFtp.ChangePasswordRequest] $changePasswordRequest = New-Object
        CerberusFtp.ChangePasswordRequest

        # Populate request object with Cerberus Admin credentials
        $changePasswordRequest.credentials = New-Object CerberusFtp.Credentials
        $changePasswordRequest.credentials.user = $CerberusCredentials.UserName
```

```
$changePasswordRequest.credentials.password =
$CerberusCredentials.GetNetworkCredential().Password

# Populate with the user whose password we wish to change
$changePasswordRequest.userName = $newTestUserName

# Setting adminPasswordReset to true allows us to change the password without
knowing the existing password
$changePasswordRequest.adminPasswordReset = $true
$changePasswordRequest.adminPasswordResetSpecified = $true

# Populate request with the desired password
$changePasswordRequest.newPassword = "ThisIsANewPassword1234!@#$"
```

Note that the password is sent in plain-text. It is for this reason that we always recommend using HTTPS for SOAP communication.

Note also that Cerberus FTP Server never stores passwords in plain-text. As soon as the plain-text is received, Cerberus salts and hashes the value before saving it to the user.

```
# Issue the ChangePassword request
[CerberusFtp.ChangePasswordResponse] $changePasswordResponse =
$CerberusSvc.ChangePassword($changePasswordRequest)

# Check response for success or failure
if (-not $changePasswordResponse.result){
   Write-Error "Failed to change password: $($changePasswordResponse.message)"
} else {
   Write-Host "Successfully changed password for $newTestUserName"
}
```

38.8.8 Add Virtual Directory to a User

As with to password reset, dedicated APIs are provided for adding and removing virtual directories. Provide the *userName* whose directories will be modified and a *CerberusFtp.VirtualDirectory* object. If the *name* of the virtual directory matches an existing one, the existing one is overwritten:

```
# Create a new AddDirectoryToUserRequest object
[CerberusFtp.AddDirectoryToUserRequest] $addDirectoryRequest = New-Object
-TypeName CerberusFtp.AddDirectoryToUserRequest

# Populate request object with Cerberus Admin credentials
$addDirectoryRequest.credentials = New-Object -TypeName
CerberusFtp.Credentials
$addDirectoryRequest.credentials.user = $CerberusCredentials.UserName
$addDirectoryRequest.credentials.password =
$CerberusCredentials.GetNetworkCredential().Password

# Populate request object with the target username
$addDirectoryRequest.userName = $newTestUserName
```

```
# Create new VirtualDirectory object
$addDirectoryRequest.directory = New-Object -TypeName
CerberusFtp.VirtualDirectory

# Populate virtual directory object with name, path, and permissions
$addDirectoryRequest.directory.name = "NewRoot"
$addDirectoryRequest.directory.path = "c:\testroot"
```

With twelve defined permissions, this section can be a little verbose:

```
$addDirectoryRequest.directory.permissions = New-Object -TypeName
CerberusFtp.DirectoryPermissions

# Grant download, upload, list files, list directories, rename, create, and
delete
$addDirectoryRequest.directory.permissions.allowDownload = $true
$addDirectoryRequest.directory.permissions.allowUpload = $true
$addDirectoryRequest.directory.permissions.allowListDir = $true
$addDirectoryRequest.directory.permissions.allowListFile = $true
$addDirectoryRequest.directory.permissions.allowRename = $true
$addDirectoryRequest.directory.permissions.allowDirectoryCreation = $true
$addDirectoryRequest.directory.permissions.allowDelete = $true

# Issue the AddDirectoryToUser request
[CerberusFtp.AddDirectoryToUserResponse] $addDirectoryResponse =
$CerberusSvc.AddDirectoryToUser($addDirectoryRequest)

# Check response for success or failure
if (-not $addDirectoryResponse.result){
    Write-Error "Failed to add virtual directory to user:
$($addDirectoryResponse.message)"
} else {
    Write-Host "Successfully added $($addDirectoryRequest.directory.name) to
$newTestUserName"
}
```

38.8.9 Delete a User

The last operation we demonstrate is perhaps the simplest. Provide a valid username and Cerberus FTP Server will delete the user.

```
# Create a new DeleteUserRequest object
[CerberusFtp.DeleteUserRequest] $deleteUserRequest = New-Object -TypeName
CerberusFtp.DeleteUserRequest

# Populate request object with Cerberus Admin credentials
$deleteUserRequest.credentials = New-Object CerberusFtp.Credentials
$deleteUserRequest.credentials.user = $CerberusCredentials.UserName
$deleteUserRequest.credentials.password =
$CerberusCredentials.GetNetworkCredential().Password

# Populate request object with username to be deleted
$deleteUserRequest.name = $newTestUserName
```

```
# Issue the DeleteUser request
[CerberusFtp.DeleteUserResponse] $deleteUserResponse =
$CerberusSvc.DeleteUser($deleteUserRequest)

# Check response for success or failure
if (-not $deleteUserResponse.result){
    Write-Error "Failed to delete user $newTestUserName :
$($deleteUserResponse.message)"
} else {
    Write-Host "Successfully deleted $newTestUserName"
}
```

38.8.10 Conclusion

That covers some of the most common operations involving Cerberus native users. In the next guide we'll cover group manipulation. Adding, modifying, and removing groups as well as adding members to groups. We'll also revisit the UserPropertyBool type and how constraints get applied to users.

38.9 Cerberus Group Modifications with PowerShell

38.9.1 Introduction

In the last article we made real modifications to Cerberus with the SOAP API. It may not seem like much to create a single user, but it was an important step in the path to understanding the API.

This time we'll make modifications to Cerberus Group objects. We will create a group, modify its members, and modify the virtual directories it grants to members. We also explain interactions between groups and users that determine a user's effective constraints.

Running Example-GroupManipulation.ps1

Once again, we've provided sample code to demonstrate the changes we'll be making. If you've followed the previous guides closely, you'll recognize similar patterns and even some duplicate code.

We **strongly** recommend running this script in a test environment, **not** on a production Cerberus instance.

- Download Example-GroupManipulation.ps1
- Open a PowerShell console and change directory to the script location
- Invoke Unblock-File and/or modify execution policy
- Run the script

  Use -*EnableTls12* and -*DisableCertValidation* flags, if necessary
  Supply -*WSDLUrl* parameter if Cerberus is on a different host
  Supply -*CerberusServiceUrl* if running Cerberus 10.0.9 or earlier

  For example:

```
& .\Example-GroupManipulation.ps1 -EnableTls12 -DisableCertValidation
```

- Provide Cerberus primary admin credentials when prompted

If successful, something like this will appear in the console:

```
PS C:> & .\Example-GroupManipulation.ps1 -EnableTls12 -DisableCertValidation
Windows PowerShell credential request.
Provide master admin credentials for Cerberus FTP Server
User: Admin
Password for user Admin: *****************
Successfully created group PsSOAPTestGroup
Successfully added groupRoot to PsSOAPTestGroup
Retrieved PsSOAPTestGroup
Successfully modified PsSOAPTestGroup
Successfully retrieved list of groups
PsSOAPTestGroup
PsSOAPTestGroup exists in the list of groups
Successfully created user PsSOAPTestUser
Successfully found PsSOAPTestUser
Successfully made PsSOAPTestUser a member of PsSOAPTestGroup
Successfully retrieved PsSOAPTestUser
Group-allowed protocols now overridden by user-allowed protocols
Successfully retrieved PsSOAPTestUser
Successfully removed PsSOAPTestUser from PsSOAPTestGroup
Successfully deleted PsSOAPTestUser
Successfully deleted PsSOAPTestGroup
```

As you can see, this script creates a group, user, modifies a few of their properties, shuffles the user in and out of the group, and deletes them both.

38.9.2 Code Walk-Through

Group manipulation looks similar to User manipulation since, after all, Groups and Users share many of the same properties (allowed protocols, permissible IPs, virtual directory lists, for example). *AddGroup* is invoked for creating and updating groups and *AddDirectoryToGroup* is invoked for modifying virtual directories on groups.

This script uses a few PowerShell idioms to reduce verbosity in areas we've already covered. We'll call these out during the walk-through.

Setup SOAP Connection

This section should look quite familiar by now; it is quite similar to the sections in HelloCerberus.ps1 and Example-UserManipulation.ps1.

The one change is the creation of a hash table containing Cerberus credentials. We'll reuse this object to initialize request objects later in the script. We make thirteen requests to the Cerberus SOAP API, so individually initializing the credentials on every request begins to add up.

```
# Hashtable containing credentials for later request object population
$requestWithCreds = @{
  credentials = @{
    user = $CerberusCredentials.UserName
    password = $CerberusCredentials.GetNetworkCredential().Password
  }
}
```

Create a New Group

Similar to dealing with users, we create a new object, populate the relevant properties and invoke an Add operation to create it within Cerberus FTP Server.

```
# The name of the test group
$newTestGroupName = "PsSOAPTestGroup"

# Create new Group object
[CerberusFtp.Group] $newGroup = New-Object -TypeName CerberusFtp.Group
$newGroup.name = $newTestGroupName
$newGroup.desc = "New Test Cerberus Native Group from PowerShell"
```

We set *isSimpleDirectoryMode* and *protocols* to allow only *https*. Later we'll demonstrate how to override these on a per-user basis.

```
$newGroup.isSimpleDirectoryMode = $true
$newGroup.isSimpleDirectoryModeSpecified = $true
$newGroup.protocols = New-Object -TypeName CerberusFtp.ProtocolsAllowed
$newGroup.protocols.https = $true
```

Finally, create the *addGroupRequest* object, invoke the *AddGroup* operation and check the result. This time, when we create the *CerberusFtp.AddGroupRequest* object, we initialize its credentials with the *$requestWithCreds* variable we declared earlier:

```
# Create addGroupRequest object
[CerberusFtp.AddGroupRequest] $addGroupRequest = New-Object -TypeName
CerberusFtp.AddGroupRequest
$addGroupRequest = $requestWithCreds
$addGroupRequest.Group = $newGroup

# Issue the AddGroup request
[CerberusFtp.AddGroupResponse] $addGroupResponse =
$CerberusSvc.AddGroup($addGroupRequest)

# Check response for success or failure
if (-not $addGroupResponse.result){
  Write-Error "Failed to create group: $($addGroupResponse.message)"
} else {
  Write-Host "Successfully created group $newTestGroupName"
}
```

Add Virtual Directory to a Group

Again, similar to adding virtual directories to users, but a different operation is provided for groups:

```
# Create a new AddDirectoryToGroupRequest object
[CerberusFtp.AddDirectoryToGroupRequest] $addDirectoryRequest =
$requestWithCreds
$addDirectoryRequest.groupName = $newTestGroupName
$addDirectoryRequest.directory = New-Object -TypeName
CerberusFtp.VirtualDirectory

# Populate virtual directory object with name, path, and permissions
$addDirectoryRequest.directory.name = "groupRoot"
$addDirectoryRequest.directory.path = "c:\groupRoot"
$addDirectoryRequest.directory.permissions = New-Object -TypeName
CerberusFtp.DirectoryPermissions

# Grant download, upload, list files, list directories, rename, create, and
delete
$addDirectoryRequest.directory.permissions.allowDownload = $true
$addDirectoryRequest.directory.permissions.allowUpload = $true
$addDirectoryRequest.directory.permissions.allowListDir = $true
$addDirectoryRequest.directory.permissions.allowListFile = $true
$addDirectoryRequest.directory.permissions.allowRename = $true
$addDirectoryRequest.directory.permissions.allowDirectoryCreation= $true
$addDirectoryRequest.directory.permissions.allowDelete= $true

# Issue the AddDirectoryToGroup request
[CerberusFtp.AddDirectoryToGroupResponse] $addDirectoryResponse =
$CerberusSvc.AddDirectoryToGroup($addDirectoryRequest)

# Check response for success or failure
if (-not $addDirectoryResponse.result){
  Write-Error "Failed to add virtual directory to group:
$($addDirectoryResponse.message)"
} else {
  Write-Host "Successfully added $($addDirectoryRequest.directory.name) to
$newTestGroupName"
}
```

Modify Group Description

Modifying a Group object is a compound operation. Retrieve the existing group object with *GetGroupInformation*, make modifications, then overwrite the existing group with *AddGroup*:

```
[CerberusFtp.GetGroupInformationRequest] $getGroupRequest = $requestWithCreds
$getGroupRequest.name = $newTestGroupName

[CerberusFtp.GetGroupInformationResponse] $getGroupResponse =
$CerberusSvc.GetGroupInformation($getGroupRequest)
```

```
if (-not $getGroupResponse.result) {
  Write-Error "Failed to retrieve group: $($getGroupResponse.message)"
} else {
  Write-Host "Retrieved $newTestGroupName"

  $existingGroup = $getGroupResponse.group
  $existingGroup.desc = "This group was created for demonstration purposes in
PowerShell"

  [CerberusFtp.AddGroupRequest] $modifyGroupRequest = $requestWithCreds
  $modifyGroupRequest.Group = $existingGroup

  [CerberusFtp.AddGroupResponse] $modifyGroupResponse =
$CerberusSvc.AddGroup($modifyGroupRequest)
  if(-not $modifyGroupResponse.result){
    Write-Error "Failed to modify group: $($modifyGroupResponse.message)"
  } else {
    Write-Host "Successfully modified $($existingGroup.name)"
  }
}
```

List Current Groups

A list of all current group names can be retrieved with *GetGroupList*.

Because the *GetGroupListRequest* object contains only credentials and no other request data, we can get away with passing the *$requestWithCreds* hash table as a short-cut. PowerShell manages the conversion of the hash table to a *GetGroupListRequest* object transparently.

```
[CerberusFtp.GetGroupListResponse] $getGroupListResponse =
$CerberusSvc.GetGroupList($requestWithCreds)

if (-not $getGroupListResponse.result){
  Write-Error "Failed to retrieve group list:
$($getGroupListResponse.message)"
} else {
  Write-Host "Successfully retrieved list of groups"
  Write-Output $getGroupListResponse.GroupList
  if ($getGroupListResponse.GroupList -contains $newTestGroupName){
    Write-Host "$newTestGroupName exists in the list of groups"
  } else {
    Write-Host "$newTestGroupName was not found in the list of groups"
  }
}
```

Create New User

This time we create the User object with a nested hash table. This does exactly what you'd expect; names and values in the hash table populate the *CerberusFtp.User* object. PowerShell will emit an error message if required properties are missing and if unexpected properties are found.

```
$newTestUserName = "PsSOAPTestUser"

[CerberusFtp.User] $newUser = @{
  name = $newTestUserName
  password = @{value = "TestPasswordChangeImmediately1234!@#$"}
  desc = "This user is for testing group membership modifications"
  isDisabled = @{value = $true; valueSpecified = $true}
}

[CerberusFtp.AddUserRequest] $addUserRequest = $requestWithCreds
$addUserRequest.User = $newUser

[CerberusFtp.AddUserResponse] $addUserResponse =
$CerberusSvc.AddUser($addUserRequest)

if (-not $addUserResponse.result){
  Write-Error "Failed to create user: $($addUserResponse.message)"
} else {
  Write-Host "Successfully created user $newTestUserName"
}
```

Add User to Group

There are some surprises in Cerberus' model of the group-user relationship:

Cerberus Users carry a reference to the Group they're a member of.

This is a departure from typical identity systems, where group objects contain references to their members.

Cerberus does not currently support multi-group membership for native users.

The user property named named "groupList" stores the single group reference.

It is an array, accepting many group names, but only the first is evaluated for user constraints and virtual directories.

When using SOAP API to add users to groups, group properties are not automatically inherited by the user.

The GUI Admin tools perform this step for you, whereas in SOAP API, your script must perform this step, if desired.

As we add the user to the group, we'll demonstrate how all of the above surprises affect our script.

Since we are modifying an existing user, we first retrieve the current user object:

```
[CerberusFtp.GetUserInformationRequest] $userInfoRequest = $requestWithCreds
$userInfoRequest.userName = $newTestUserName
```

```
[CerberusFtp.GetUserInformationResponse] $existingUserResponse =
$CerberusSvc.GetUserInformation($userInfoRequest)

if (-not $existingUserResponse.result){
  Write-Error "Failed to find user $newTestUserName :
$($existingUserResponse.message)"
} else {
  Write-Host "Successfully found $newTestUserName"

  $existingUser = $existingUserResponse.UserInformation
```

In PowerShell, @() signifies an array and @{} signifies a hash table. So the next line creates an array containing a single hash table containing a '*name*' property, whose value is the group name:

```
$existingUser.groupList = @(@{name=$newTestGroupName})
```

There are twelve user properties that may be inherited through group membership. This bit of code sets them all to defer to the group's value. We'll cover the details of this "priority" attribute in the next section.

```
foreach ($propName in @( "authMethod"
                          "disableAfterTime"
                          "ipAllowedList"
                          "isAllowPasswordChange"
                          "isAnonymous"
                          "isDisabled"
                          "isSimpleDirectoryMode"
                          "maxLoginsAllowed"
                          "maxUploadFilesize"
                          "protocols"
                          "requireSecureControl"
                          "requireSecureData")
 ) {
   $existingUser.$propName = @{priority = "group"; prioritySpecified = $true}
 }
```

The rest is a typical user update with an invocation of AddUser:

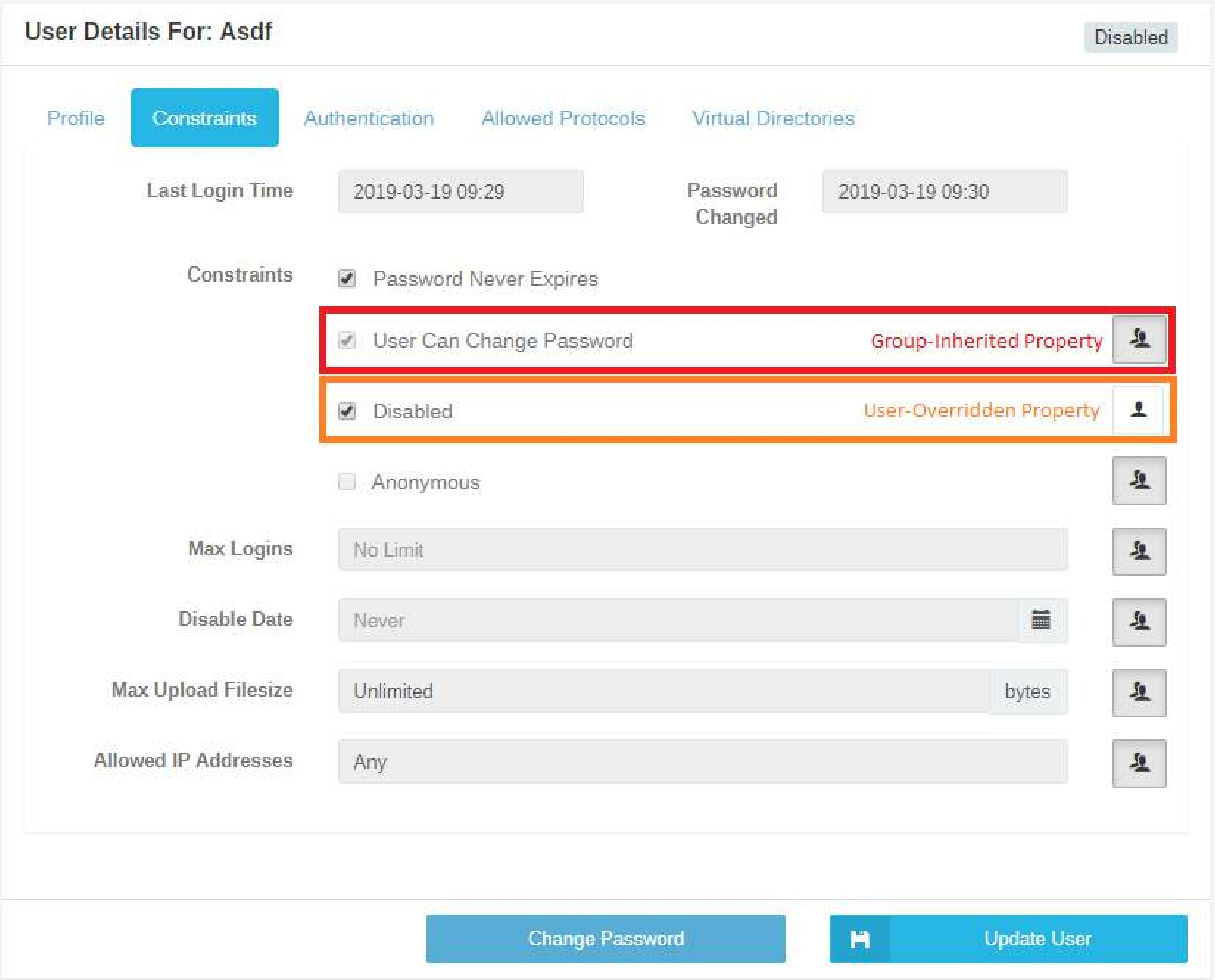
```
  [CerberusFtp.AddUserRequest] $modifyUserRequest = $requestWithCreds
  $modifyUserRequest.User = $existingUser

  [CerberusFtp.AddUserResponse] $modifyUserResponse =
$CerberusSvc.AddUser($modifyUserRequest)

  if (-not $modifyUserResponse.result){
    Write-Error "Failed to update exiting user:
$($modifyUserResposne.message)"
  } else {
    Write-Host "Successfully made $newTestUserName a member of
$newTestGroupName"
  }
}
```

Override Allowed Protocols for Group Member

In the last section, we added the test user to a group and set their properties to "*group*" priority. We'll explain a little more about what this means and then walk through the code that changes the priority of a user property to override the group value.

Our existing Group Documentation briefly explains this feature of Cerberus FTP Server.

Essentially, Cerberus tags some user properties as *group-inherited* or *user-overridden* values. The switch is is expressed intuitively in the User Manager GUI with a toggle button and symbols for each state:



User Details For: Asdf

Disabled

Last Login Time: 2019-03-19 09:29

Password Changed: 2019-03-19 09:30

Constraints

☑ Password Never Expires

☑ User Can Change Password — Group-Inherited Property

☑ Disabled — User-Overridden Property

☐ Anonymous

Max Logins: No Limit

Disable Date: Never

Max Upload Filesize: Unlimited bytes

Allowed IP Addresses: Any

In SOAP API, this is expressed as a *priority* attribute on each property, which may be set to "*group*" or "*user*" accordingly. The attribute defaults to "*user*" for newly created users.

For instance, the *CerberusFtp.ProtocolsAllowed* type appears like this in PowerShell. Note the various protocols which may be allowed, plus the a priority attribute:

```
PS C:\> CerberusFtp.ProtocolsAllowed].declaredProperties |Select-Object name,
propertyType

Name              PropertyType
----              ------------
priority          CerberusFtp.UserPropertyPriority
prioritySpecified System.Boolean
ftp               System.Boolean
ftps              System.Boolean
sftp              System.Boolean
http              System.Boolean
https             System.Boolean
```

The override takes place on the *User* object, so we first retrieve our user from Cerberus FTP Server:

```
[CerberusFtp.GetUserInformationRequest] $getUserRequest = $requestWithCreds
$getUserRequest.userName = $newTestUserName

[CerberusFtp.GetUserInformationResponse] $getUserResponse =
$CerberusSvc.GetUserInformation($getUserRequest)

if (-not $getUserResponse.result){
  Write-Error "Unable to retrieve user: $($getUserResponse.message)"
} else {
  Write-Host "Successfully retrieved $($getUserResponse.UserInformation.name)"
```

We want this user to have both FTPS and HTTPS access, regardless of the constraints set by their group. We allow these protocols by setting them to *$true* on the user *protocols* property, then enable the override by setting priority to "*user*" and *prioritySpecified* to *$true*:

```
  $existingUser = $getUserResponse.UserInformation
  $existingUser.protocols.ftps = $true
  $existingUser.protocols.https = $true
  $existingUser.protocols.priority = "user"
  $existingUser.protocols.prioritySpecified = $true
```

And now we push the modified user object with *AddUser*:

```
  [CerberusFtp.AddUserRequest] $modifyUserRequest = $requestWithCreds
  $modifyUserRequest.User = $existingUser

  [CerberusFtp.AddUserResponse] $modifyUserResponse =
$CerberusSvc.AddUser($modifyUserRequest )

  if (-not $modifyUserResponse.result){
    Write-Error "Unable to update user: $($modifyUserResponse.message)"
  } else {
    Write-Host "Group-allowed protocols now overridden by user-allowed
protocols"
  }
}
```

Remove User from Group

As previously mentioned, group membership is stored with the *User* object, so removing a user from their group means making a change to their *groupList* property.

```
[CerberusFtp.GetUserInformationRequest] $getUserRequest = $requestWithCreds
$getUserRequest.userName = $newTestUserName

[CerberusFtp.GetUserInformationResponse] $getUserResponse =
$CerberusSvc.GetUserInformation($getUserRequest)

if (-not $getUserResponse.result){
 Write-Error "Failed to retrieve user: $(getUserResponse.message)"
} else {
  Write-Host "Successfully retrieved $($getUserResponse.UserInformation.name)"

  $existingUser = $getUserResponse.UserInformation

  if ($existingUser.groupList.Count -lt 1){
    Write-Error "Cannot remove user from group; user is not a member of any
group"
  } else {
```

We empty the groupList property by assigning it an empty array:

```
    $previousMembership = $existingUser.groupList
    $existingUser.groupList = @()
```

This bit of code sets all of the twelve user properties that may be inherited through group membership to defer to the user's value.

```
foreach ($propName in @( "authMethod"
                         "disableAfterTime"
                         "ipAllowedList"
                         "isAllowPasswordChange"
                         "isAnonymous"
                         "isDisabled"
                         "isSimpleDirectoryMode"
                         "maxLoginsAllowed"
                         "maxUploadFilesize"
                         "protocols"
                         "requireSecureControl"
                         "requireSecureData")
 ) {
   $existingUser.$propName = @{priority = "user"; prioritySpecified = $true}
 }
```

Then we push the modified user with *AddUser* and display feedback to the host console:

```
    [CerberusFtp.AddUserRequest] $modifyUserRequest = $requestWithCreds
    $modifyUserRequest.User = $existingUser

    [CerberusFtp.AddUserResponse] $modifyUserResponse =
$CerberusSvc.AddUser($modifyUserRequest)
```

```
    if (-not $modifyUserResponse.result){
      Write-Error "Failed to update exiting user:
$($modifyUserResponse.message)"
    } else {
      Write-Host "Successfully removed $newTestUserName from
$($previousMembership.name -join ', ')"
    }
  }
}
```

Delete User

Clean up our test user before we're done.

```
[CerberusFtp.DeleteUserRequest] $deleteUserRequest = $requestWithCreds
$deleteUserRequest.user = $newTestUserName

[CerberusFtp.DeleteUserResponse] $deleteUserResponse =
$CerberusSvc.DeleteUser($deleteUserRequest)

if (-not $deleteUserResponse.result){
  Write-Error "Failed to delete user: $($deleteUserResponse.message)"
} else {
  Write-Host "Successfully deleted $newTestUserName"
}
```

Delete Group

Finally, we delete the group.

```
[CerberusFtp.DeleteGroupRequest] $deleteGroupRequest = $requestWithCreds
$deleteGroupRequest.name = $NewTestGroupName

[CerberusFtp.DeleteGroupResponse] $deleteGroupResponse =
$CerberusSvc.DeleteGroup($deleteGroupRequest)

if (-not $deleteGroupResponse.result){
  Write-Error "Failed to delete group: $($deleteUserResponse.message)"
} else {
  Write-Host "Successfully deleted $NewTestGroupName"
}
```

A group cannot be deleted if it still has members. You'll receive an error message like this one if you try:

```
result message
------ -------
 False The following accounts are still members of this group: PsSOAPTestUser
```

Conclusion

There are many more Cerberus API operations available beyond user and group management. They deal with server configuration, listener management, and backup/restore, to name a few. At this point, you should have all the tools and techniques necessary to explore these operations on your own.

To round out this introduction, the last article will provide an overview of entire Cerberus SOAP API grouped by functional domain. We'll also identify operations we recommend avoiding, as they're primarily for internal use.

38.10 Cerberus SOAP API Reference

Below is a list of all operations supported by the Cerberus SOAP API as of 10.0.10, grouped by functional domain, along with a short description of the operation.

Internal Operations, colored in red, should not be used. They are used internally by the Cerberus GUI. Misuse may result in server instability and loss of data.

Advanced Operations, colored in orange , are low-level operations whose use is discouraged. They are difficult to use correctly, as they exchange blocks of XML, rather than well-defined objects. Sending malformed XML to these APIs may result in loss of data and services. Safer alternatives are noted where available.

Index

User Management
Operations to manage Cerberus native users.

AddUser
Add a user to Cerberus. If the user already exists, it is overwritten.

ChangePassword
Change the password of an existing user.

DeleteUser
Remove a user account.

GetUserInformation
Get all properties of a specified user.

GetUserList
Retrieves the list of all native Cerberus user names.

RenameUser
Renames an existing user.

GetUserCustomSettings
Retrieves a complete XML representation of users' custom settings. This includes multi-factor authentication settings and security questions.

SetUserCustomSettings
Sets users' custom settings given an XML representation of all settings.

Group Management
Operations to manage Cerberus native groups.

AddGroup
Add new group. If the group name already exists, it is overwritten.

DeleteGroup
Remove group.

GetGroupList
Returns the list of all group names.

GetGroupInformation
Get all properties of a specified group.

GetGroups
Returns an XML representation of all group information. A safer alternative is GetGroupList coupled with GetGroupInformation.

RenameGroup
Change the name of an existing group.

Virtual Directory Management
Operations to manage virtual directories for both users and groups.

AddDirectoryToGroup
Add a virtual directory to a group.

AddDirectoryToUser
Add a virtual directory to a user account.

DeleteDirectoryFromGroup
Remove a virtual directory from a group.

DeleteDirectoryFromUser
Remove a virtual directory from the user account.

Interface/Listener Management
Operations to manage interfaces and connections.

GetInterfaceByID
Retrieve the interface definition for the given interface ID.

GetInterfaceList
Retrieve all interface definitions.

GetConnectedUserList
Retrieve list of currently connected users. Results contain both connection ID and interface ID.

GetCurrentConnectionCount
Retrieve count of current connections to given interface ID.

GetInterfaces
Retrieve XML block representing all listeners. Safer alternatives are GetInterfaceList and GetInterfacesByID.

InitializeInterface
Start an interface. Returns 'false' if the interface is already started and listening for connections.

ModifyInterface
Modify properties of a given interface.

ShutdownConnectionsOnInterface
Shutdown all connections to the given interface ID.

ShutdownInterface
Shutdown interface. Existing connections are closed.

TerminateConnection
Terminate the given connection ID.

Event Management
Advanced operations to manage event rules.

GetEventRules
Retrieve XML block representing configured Event rules.

SetEventRules
Set event rules with properly structured XML block.

IP Manager
Operations to manage IP allow/deny functionality.

GetAutoBlockList
Retrieves XML block representing auto-blocking settings.

SaveBlockList
Set the IP Manager list with XML block.

AddIp
Add an IP or IP range to the IP Manager.

BlockAddress
Block the given address. Removes from IP manager if in white-list mode, adds if in black-list mode.

DeleteIp
Remove an IP address/range from the IP Manager.

GetIPBlockList
Retrieves the current list of tracked IP addresses/ranges from IP Manager.

Folder Monitor
Advanced operations for managing folder monitoring functionality.

GetFolderMonitors
Retrieves an XML block of all currently-monitored folders.

SetFolderMonitors
Overwrites the current list of monitored folders with supplied XML block.

Reporting Database
Operations related to reporting and the Cerberus statistics database.

BackupStatisticsDatabase
Backup reporting database.

CreateStatisticsDatabase

Create tables on the currently-configured statistics database.

DropStatisticsDatabase
Drop tables from the configured reporting database.

GenerateStatistics
Generate statistics report. Returns path to report on Cerberus' host.

RestoreStatisticsDatabase
Restore reporting database from backup.

TestAndVerifyDatabase
Connect and verify configured reporting database.

Public Shares
Advanced operations for managing publicly-shared files.

DeletePublicShares
Removes public shares from the server, represented by a list of GUID strings.

GetPublicShares
Retrieves an XML block of all current public shares.

SetPublicShares
Overwrites public shares with supplied XML block.

SharePublicFile
Create a new public-shared file. Requires login and password of a standard user who has access to the file/folder.

Account Requests
Operations for managing the list of requested accounts.

DeleteRequestedAccounts
Delete the specified account requests, identified by list of GUID strings.

GetRequestedAccounts
Retrieve XML block of current account requests.

SetRequestedAccounts
Set XML block of account requests.

Server Information and Status
These operations provide information about server configuration and retrieve current statistics on the server's run-time state.

CurrentStatus
Retrieve basic status of Cerberus FTP Server including bandwidth, connections, and start date.

GetAllCurrentConnectionCount
Retrieve current number of active connections.

GetAppPaths
Retrieve Cerberus FTP Server’s working directories.

GetCurrentBandwidth
Retrieve current bandwidth utilization.

GetFeatures
Retrieves list of enabled features and allowed connections.

GetFileTransfers
Retrieve list of files currently in transit.

GetHostname
Get the hostname of the server running Cerberus FTP Server.

GetLicenseInfo
Retrieve detailed license information.

GetLogMessages
Retrieve log messages from the logging queue.

GetStatistics
Retrieve file and connection counts since the last restart. Includes number of files uploaded/downloaded, total/current connections, and failed up/downloads.

ServerInformation
Retrieve basic information about Cerberus FTP Server.

ServerSummaryStatus
Retrieve overview of Cerberus FTP Server status and configuration. Includes all information from Cerberus' "Server Configuration and Status Summary" page displayed in the main GUI.

VerifyLicense
Validates a given license string.

Server Configuration
Low-level operations for configuration

CommitSettings
Commit changes to configuration.

CreateDirectory
Create a directory on the Cerberus server filesystem.

DeleteDirectory
Delete a directory from the Cerberus server filesystem.

GetConfiguration
Retrieve XML block of all server configuration settings.

GetProfiles
Retrieves a complete XML block of all user profiles. GetUserInformation is a safer alternative.

SaveConfiguration
Save server configuration settings in XML format.

SaveProfiles
Write an XML block of all user profiles. AddUser is a safer alternative.

GetAdminAccounts
Retrieve the list of administrator accounts.

GetMimeMappings
Retrieve the file extension to mime-type map.

SaveMimeMappings
Set the file extension to mime-type map.

SetAdminAccounts
Set the list of administrator accounts.

SetWANIP
Set the publicly available IP address.

GetAuthenticationList
Retrieve XML block of authentication providers in order of priority.

SetAuthenticationList
Set the list of authentication providers in XML format.

Backup and Sync
Operation of Cerberus FTP Server's backup and restore facilities.

GetBackupServers

Retrieve XML block representing Cerberus' configured Sync servers.

SaveBackupServers
Set XML block of Cerberus' Sync servers.

BackupServerConfiguration
Creates a backup of the Cerberus FTP Server configuration at the specified location.

RestoreServerConfiguration
Restores Cerberus configuration from backup.

Startup and Shutdown
These operations are for internal use. Do not use these operations. Incorrect use may cause server instability and loss of service. Use operating system facilities to start and stop the Cerberus service.

InitializeServer
Initializes the server.

ShutdownServer
Shuts down the Cerberus service.

StartServer
Starts Cerberus listeners.

StopServer
Stops Cerberus listeners.

ServerStarted
Checks whether the Cerberus Server has started.

39.0 Command Support

39.1 FTP Commands Supported

The following FTP commands are supported by Cerberus FTP Server:

- ABOR
- ACCT
- ADAT
- ALLO
- APPE
- AUTH
- CCC
- CDUP
- CLNT
- CSID
- CWD
- DELE
- EPSV
- EPRT
- FEAT
- HASH
- HELP
- LANG
- LIST
- MDTM
- MFMT
- MFCT
- MKD
- MODE
- MLSD
- MLST
- MLSD
- NLST
- NOOP
- OPTS
- P@SV
- PASS
- PASV

- PBSZ
- PWD
- PORT
- PROT
- QUIT
- REIN
- RETR
- REST
- RMD
- RMDA
- RNFR
- RNFT
- SITE
- SIZE
- STOR
- STOU
- STRU
- SYST
- TYPE
- USER
- XCRC
- XCUP
- XPWD
- XMD5
- XMKD
- XSHA1
- XSHA256
- XSHA512
- XRMD