

*IBM Aspera Proxy*



---

# Contents

- IBM Aspera Proxy Admin Guide.....1**
- Introduction..... 1
- Installation..... 3
  - System requirements..... 3
  - If you are upgrading..... 4
  - Installing Aspera Proxy..... 4
  - Uninstalling Aspera Proxy..... 6
- Forward proxy..... 7
  - Configuring the proxy server for forward proxy ..... 7
  - Forward proxy firewall configuration..... 10
  - Configuring the client..... 10
  - Transferring from the command line through forward proxy..... 12
- Reverse proxy..... 13
  - Configuring the proxy server for reverse proxy..... 13
  - Reverse Proxy configuration options..... 17
    - UDP port reuse..... 19
    - Load balancing..... 20
    - Source-Port filtering..... 22
  - Reverse Proxy firewall configuration..... 23
  - Configuring internal servers..... 23
  - Transferring files with Reverse Proxy..... 24
  - Using IBM Aspera Sync and IBM Aspera Connect with reverse proxy..... 27
  - Configuring reverse proxy for use with Faspex and Shares..... 28
  - Configuring reverse proxy for HTTP fallback with Nginx..... 30
  - Lua scripting with reverse proxy..... 31
- Appendices..... 32
  - Securing your SSH Server..... 32
  - Troubleshooting..... 35
  - Generating an SSL certificate that contains an IP address..... 38

# IBM Aspera Proxy Admin Guide

---

Welcome to the Proxy documentation, where you can find information about how to install, maintain, and use Proxy.

## Introduction

---

IBM Aspera Proxy protects your organization's network while enabling secure, high-speed FASP transfers to and from highly restrictive network environments. Built on top of the Linux kernel, it allows transparent pass-through of FASP transfer sessions across secure DMZs without impeding transfer speeds or compromising the security of your internal network.

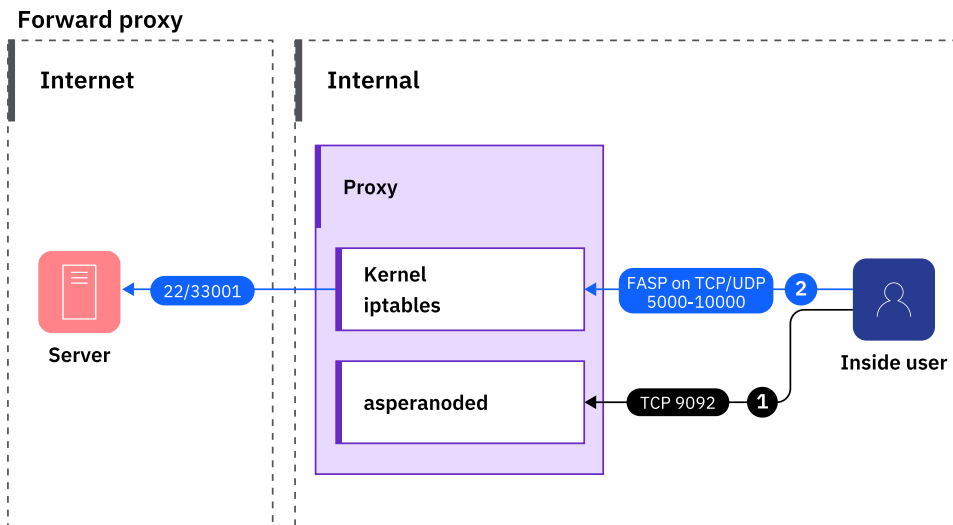
Proxy also supports load balancing, high availability, and flexible security policies. It consolidates FASP transfers in and out of a corporate network and enables precise control over which users can initiate transfers with remote Aspera transfer servers. With Proxy support built into all Aspera desktop and browser-based transfer clients, its configuration and use is straightforward for all your users.

Proxy supports both forward (outbound) and reverse (inbound) proxy modes, allowing FASP transfers to be initiated by users who are either inside or outside the corporate network.

### Forward proxy

Forward proxy provides a secure way for users behind company network firewalls to initiate requests for FASP transfers of files that are on servers outside the firewall. It addresses the following customer use cases:

- **Limited-use Internet access:** Your enterprise has security requirements that prevent you from deploying IBM Aspera Enterprise Server (or IBM Aspera Connect Server) inside your DMZ. Organizations often limit general Internet access for their employees, which can affect the FASP protocol even if used for legitimate business needs. Proxy provides secure access to the Aspera transfer servers residing outside of your corporate network without exposing users' IP addresses. It also enforces strict user authentication for Aspera clients that initiate connections to the outside servers.
- **Consolidation and control of FASP transfers:** If you are an IT systems manager and want to establish better control and security around FASP transfers that your internal users initiate, Proxy can fulfill your requirements without impeding the users' experience. It provides a single point through which all FASP transfers flow in and out of your corporate network, hiding internal clients' IP addresses and allowing you to control which users can initiate FASP transfers, without slowing down the speed of the transfers.



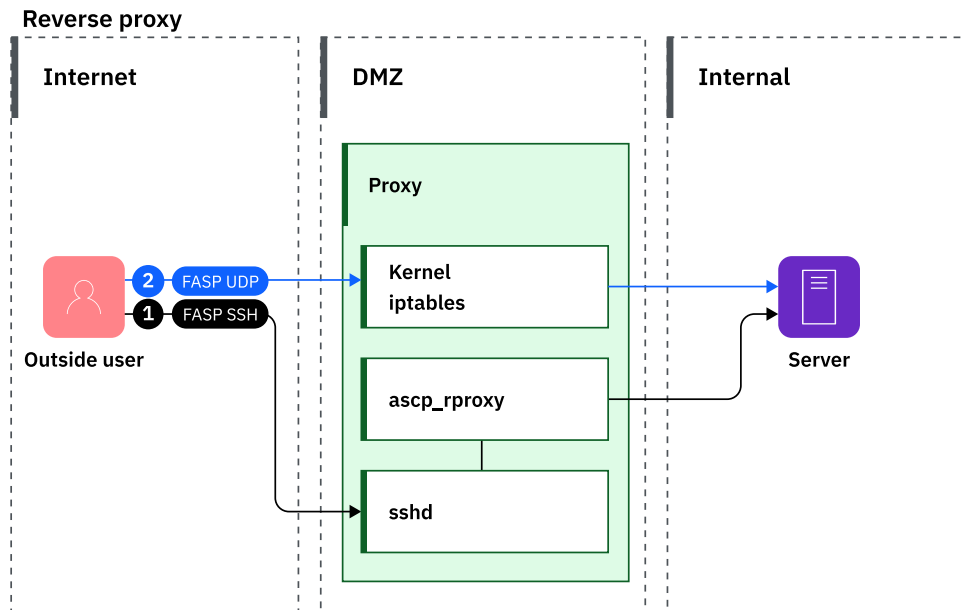
## Reverse proxy

Reverse proxy provides a secure way for users outside company network firewalls to initiate requests for FASP transfers from servers inside the firewall. It addresses the following customer use case:

- **Trusted partners need access to files on your servers:** Customers want to allow users outside their company firewall to initiate FASP transfers to and from servers inside the company network.

Reverse proxy is usually deployed inside a DMZ, on top of a Linux-based server. Multiple proxy instances can also be started on a server cluster, behind an enterprise-grade load balancer, forming a high-availability solution. Reverse proxy currently employs the same security model as IBM Aspera Enterprise Server and Connect Server, based on the SSHD service. As a result, no changes are needed on the client side. Once authenticated, the proxy server calls one program: **ascp\_rproxy**, which is in charge of bidirectional forwarding of SSH control traffic and FASP (UDP) traffic between the client and the internal server.

The **ascp\_rproxy** program maintains an SSH connection with the **ascp** client when it's called by the SSHD service. A second SSH connection is set up between the proxy server and the internal Enterprise Server instance by virtue of a preinstalled SSH key. It then bridges the two SSH connections, by forwarding incoming data from one connection to the other, in both directions. To forward FASP (UDP) traffic, the **ascp\_rproxy** program proxy server sets up a dynamic network address conversion (DNAT) rule by using the Linux **iptables** kernel module. Since UDP traffic forwarding is done by using the Linux **iptables** kernel module, high-speed packet forwarding can be achieved without any reduction in speed.



## Installation

To install Aspera Proxy, log in to your computer with root permissions.

### System requirements

Find the system requirements for Aspera Proxy.



**CAUTION:** Do not install IBM Aspera Proxy on a machine where IBM Aspera High-Speed Transfer Server, IBM Aspera High-Speed Transfer Endpoint, or Desktop Client is installed. If these products are already on your machine, be sure to remove them before installing Proxy.

Your Proxy server requires the following:

- A Linux system. (For a list of supported Linux systems and versions, see the release notes).
- **iptables** v1.3.0 or later.

#### Configuring iptables

Proxy server makes use of the Linux iptables module as well as the **iptables** and **iptables-restore** commands (reverse proxy). Nevertheless, the local firewall software is not necessary.

To operate correctly, the Proxy server must have exclusive control of these **iptables** chains:

---

#### iptables chains

nat/prerouting

nat/postrouting

filter/forwarding

Typically, the local firewall should be disabled. This does not degrade the security of the system if deployed in a DMZ and protected by an external firewall.

If the local firewall software needs to be activated, it must be configured to allow appropriate FASP traffic (TCP/UDP 33001, range if Windows), as well as receive HTTP/HTTPS requests on the Node API port (default TCP 9091/9092).

## If you are upgrading

Follow these steps before you upgrade to a newer version of Aspera Proxy.



**Attention:** Upgrading from Aspera Proxy 1.4.4 is a special case. To upgrade from 1.4.4, you must first uninstall it. See [“Uninstalling Aspera Proxy” on page 6](#). However, before uninstalling, you must still complete the following the steps to back up configuration files, license information, database files, and other content.

If you plan to upgrade to a new release of Aspera Proxy, the following steps will help you prepare. When you're ready to install, follow the installation instructions in [“Installing Aspera Proxy” on page 4](#). The instructions include an example of the **xpm** command for upgrading.

1. Review the Aspera Proxy release notes for versions of Proxy that were released since your current version. In particular, the **Breaking Changes** section highlights changes that might require you to adjust your workflow, configuration, or usage.
2. Stop or allow to complete any FASP transfers that were initiated by the computer that you are upgrading. FASP transfers cannot proceed during your Aspera product upgrade.
3. Back up configuration and settings files. These files are found in the `etc` and `var` folders:

Folder path	Description
<code>/opt/aspera/etc/</code>	Contains server configuration, web configuration, user settings, and license information.
<code>/opt/aspera/var/</code>	Contains pre and post-processing scripts.

4. Back up the Redis database. The Redis database is backed up as part of backing up the `var` directory, but also backing it up separately, particularly if it's stored on a different machine:

```
# sudo /opt/aspera/bin/asnodeadmin -b /path/database.backup
```

## Installing Aspera Proxy

To install Aspera Proxy, log in to your computer as root, and run the following steps.

### Procedure

1. Download the Proxy installer from the Aspera download site.  
Use the credentials Aspera provided to your organization to access [Fix Central](#).  
If you need help with determining your access credentials, contact your Aspera account manager.
2. Start the installer by running the following commands with root privileges.

#### Pack Commands

##### RPM

```
# rpm -ivh ibm-aspera-proxy-version.rpm
```

For upgrades, use **-U** instead of **-i**.

**Note:** If upgrading from Proxy 1.4.4, do not use **-U**. Proxy 1.4.4 must first be uninstalled and the upgrade treated as a fresh install with (**-i**). For details, see [“If you are upgrading” on page 4](#).

```
# rpm -Uvh ibm-aspera-proxy-version.rpm
```

**-U** is the same except that it removes all other versions of the package after the new one is installed.

## Pack Commands age

DEB

```
# dpkg -i ibm-aspera-proxy-version.deb
```

For upgrades, see your system's man page or other documentation for **dpkg**.

This starts the Proxy daemon and makes adjustments to the **iptables** system settings.

### 3. Install the license.

**Note:** For purchasers of IBM Aspera Enterprise, a license enabling IBM Aspera Proxy as part of Enterprise can be downloaded from [Fix Central](#).

In a terminal window, create the following file. Open the file with a text editor and paste your license key string into it:

```
/opt/aspera/proxy/etc/aspera-license
```

- If you're updating an existing license, open the file and replace the existing license string with a new one.
- Save and close the file, then run the following command to verify the installed version is correct:

Package	Commands
---------	----------

RPM

```
# rpm -q aspera-proxy
```

DEB

```
# dpkg-query -l aspera-proxy
```

### 4. Review or update OpenSSH authentication methods.

Open your SSH server configuration file with a text editor:

```
/etc/ssh/sshd_config
```

To allow public key authentication, set `PubkeyAuthentication` to **yes**. If you also plan to allow password authentication, which is less secure than keys, set `PasswordAuthentication` to **yes**:

```
...  
PubkeyAuthentication yes  
PasswordAuthentication yes  
...
```

**Note:** For information about security options with Aspera products, see [Appendix: Securing Your SSH Server](#).

Save and close the file, then run one of the following commands to restart SSH.

**Note:** Depending on the Linux type and version, your system's restart procedure uses either `init` or `systemd`. To determine which of these your system uses, you can run either or both the following:

```
# ps -C systemd  
# ps -C init
```

If the output reports that there are `systemd` processes, use the **systemctl** command. If no `systemd` processes are reported, you can generally assume that the system uses `init`. (Although the previous command for `init` searched for `init`, a return of `init` might be misleading if it's a symlink to `systemd`).

## Daemon Commands

systemd

```
# systemctl restart sshd
```

init

```
# service ssh restart
```

5. Generate a new self-signed certificate that includes a hostname.

Forward proxy transfers now require the Proxy server's SSL certificate to include the hostname (or IP address), otherwise the transfers are refused. The default, self-signed certificate that is created by the Proxy installer must be replaced. Although certificates that include only an IP address can be used, Aspera recommends including the hostname.

To replace the existing `aspera_server_cert.pem` in the Proxy installation with a new certificate that contains the hostname, run the following command:

```
# /opt/aspera/proxy/bin/generate-cert.sh proxy_server_hostname
```

If you instead want the replacement certificate to contain the IP address, see [“Generating an SSL certificate that contains an IP address”](#) on page 38.

## Results

**Note:** The hostname or IP address that is specified in the `dnat` connection to the Proxy server must match the name that is contained in the certificate. For example, if the subject CN in the certificate is `aspera.proxy.us`, the address that is specified with the `ascp --proxy` option must match:

```
$ ascp --proxy=dnats://aspera.proxy.us myfile user@host:/
```

Likewise, if you instead choose to use the IP address for the Proxy server, the certificate must contain the matching address.

## Uninstalling Aspera Proxy

The following steps explain how to uninstall Proxy from your Linux system.

1. Obtain the name of the Proxy package that is installed on your system by running the following command that corresponds to the type of Linux system you're using:

### Package Commands

RPM

```
$ rpm -q aspera-proxy
```

DEB

```
$ dpkg -l aspera-proxy
```

Determine the Aspera Proxy package name from the command output. For example, on Debian, the previous `dpkg` command returns something similar to the following:

```
||/ Name          Version          Description
+++-----
ii  ibm-aspera-proxy 1.2.2.100179-2  IBM Aspera Proxy Server
```

In this example, the package name is **ibm-aspera-proxy-1.2.2.100179**.

2. Uninstall this package by running the following command that corresponds to your type of Linux system, replacing `pkg_name` with the package name derived in the previous step.



## Package Commands

RPM

```
$ rpm -e pkg_name
```

DEB

```
$ dpkg -r pkg_name
```

## Forward proxy

---

The following references describe the proxy settings that can be configured for forward proxy.

### Configuring the proxy server for forward proxy

When you are using a proxy server for a forward proxy, you need to configure the proxy server.

#### About this task

The following configuration steps require setting values in the proxy server's `aspera.conf` file, which is found in the following location:

```
/opt/aspera/proxy/etc/aspera.conf
```

You can edit the file manually or by using the **asconfigurator** utility. Both methods that are described in the *Procedure* section.

The **asconfigurator** command is located in `/opt/aspera/proxy/bin`. The following examples assume that the command is already on the path. You can either add it to root's path or prefix the command with the path each time you run it.

#### Procedure

1. Enable HTTP or HTTPS.

Run the following commands:

```
# asconfigurator -x "set_server_data;enable_http,true"
# asconfigurator -x "set_server_data;enable_https,true"
```

These commands create the following lines in `aspera.conf`, which can also be added to the `<server>` section manually:

```
<server>
...
  <enable_http>true</enable_http>      <!-- true | false -->
  <enable_https>true</enable_https>    <!-- true | false -->
...
</server>
```

2. Enable the proxy server.

Run the following command:

```
# asconfigurator -x "set_server_data;proxy_enabled,true"
```

This command creates the following lines in `aspera.conf`, which can also be added to the `<server>` section manually:

```
<server>
...
  <proxy>
    <enabled>true</enabled>          <!-- Proxy server is enabled -->
  </proxy>
```

```
...
</server>
```

This is the only setting on the proxy server that is required to begin using forward proxy. However, you might need to change other <proxy> settings based on your unique network configuration. You can disable the proxy by using `asconfigurator`, change the value `true` with the `false`.

### 3. Update additional forward proxy settings, as needed.

To use **asconfigurator** to set proxy options, use the following syntax:

```
# asconfigurator -x "set_server_data;parameter,value"
```

The following table shows the parameters and options that are used in the <server> / <proxy> section of `aspera.conf`. To also view all forward-proxy configuration options and the **asconfigurator** command to set them, run the **asuserdata** command as follows:

```
# /opt/aspera/proxy/bin/asuserdata -+
```

**Note:** The **asuserdata -+** command displays the default values for the server setup, not the currently set values.

<b>asconfigurator parameter aspera.conf option</b>	<b>Description</b>	<b>Default Value</b>
proxy_enabled <enabled>	Disable or enable the proxy server. Must be set to true to turn on the service.	false
proxy_authentication <authentication>	Disable or enable the authentication requirement for the proxy server.	true
proxy_bind_ip_address <bind_ip_address>	The IP address that the proxy server binds to (also the IP address that the client connects to). The default value, 0.0.0.0, allows the proxy server to bind to all available interfaces.	0.0.0.0
proxy_bind_ip_netmask <bind_ip_netmask>	The netmask for the <b>proxy_bind_ip_address</b> .	blank (null)
proxy_port_range_low <port_range_low>	The lower bound of the port range. Ensure that the firewall allows the port range you specify.	5000
proxy_port_range_high <port_range_high>	The upper bound of the port range. Ensure that the firewall allows the port range you specify.	10000
proxy_cleanup_interval <cleanup_interval>	The interval, in seconds, at which the proxy server scans and cleans up expired sessions.	0
proxy_session_timeout <session_timeout>	The interval, in seconds, after which a session times out if no keep-alive updates have been received.	0
proxy_keepalive_interval <keepalive_interval>	The interval, in seconds, at which an ascp client sends keep-alive requests. This option is propagated to the client.	0

The following is an example of the <proxy> portion of the <server> section in `aspera.conf`:

```
<server>
...
  <proxy>
    <enabled>true</enabled>
    <authentication>true</authentication>
    <bind_ip_address>0.0.0.0</bind_ip_address>
    <bind_ip_netmask></bind_ip_netmask>
    <port_range_low>5000</port_range_low>
    <port_range_high>10000</port_range_high>
    <cleanup_interval>0</cleanup_interval>
    <keepalive_interval>0</keepalive_interval>
    <session_timeout>0</session_timeout>
  </proxy>
...
</server>
```

If you have manually edited `aspera.conf`, save your changes and validate the syntax and XML tags by running:

```
# /opt/aspera/proxy/bin/asuserdata -v
```

- Restart the proxy node service.

Daemon	Commands
--------	----------

systemd	
---------	--

```
# systemctl restart asperaproxy
```

init	
------	--

```
# service asperaproxy restart
```

If you receive the following error when attempting to start the node service, **iptables** might not be installed on your machine:

```
ERR Failed to initialize proxy service
```

For more information, see

- Check log entries for startup.

After starting up the **asperaproxy** service, check the system log-file entries:

Operating system	Path
------------------	------

Red Hat Linux:	/var/log/messages
----------------	-------------------

Debian-based Linux:	/var/log/syslog
---------------------	-----------------

The only proxy entries that must be displayed are similar to the following:

```
LOG proxy service ready (port range 5000-10000)
LOG Started on port(s) 9091,9092s ...
```

The port range (lower and upper bounds) can be modified by changing the <port\_range\_low> and <port\_range\_high> options in the <proxy> section of `aspera.conf`; whereas, the default node service ports (9091 and 9092) can be modified by changing the <http\_port> and <https\_port> options in the <server> section.

- Create a node API user. (Only necessary if authentication is required).

On the proxy machine, create a node API user by running **asnodeadmin** command:

```
$ sudo /opt/aspera/proxy/bin/asnodeadmin -au node_api_user -p password -x transfer_user
```

The transfer user must be an existing user on the proxy server.

## Forward proxy firewall configuration

Restricting outbound connections is possible when using an internal or external firewall.

### Internal firewall

If outbound connections are restricted by an internal firewall, the firewall must be open to the following ports:

- **Outbound TCP/9091 and 9092** (or whatever ports are configured for HTTP and HTTPS the client transfer application). These are the ports through which a client on the internal network establishes communication with the proxy server.
- **Outbound TCP and UDP/5000-10000** (or whatever range of ports are set in `aspera.conf` using `port_range_low` and `port_range_high`). These are the ports that the client uses for SSH and FASP data transfer.

### External firewall

If outbound connections are restricted by the external firewall, the external firewall must allow **outbound TCP and UDP/33001** for SSH and FASP data transfer.

If the destination server has a Windows, FreeBSD, or Isilon operating system that does not allow concurrent transfers to bind to the same UDP port, the external firewall must allow a range of UDP ports, for example **outbound UDP/33001-33100**.

## Configuring the client

You must configure your client transfer application by specifying your proxy host, port number, username, and password.

In the Enterprise Server GUI, go to **Preferences > Proxy**. Note that the transfer proxy feature is disabled by default.

On this screen, you can do the following:

- Configure connections on a case-by-case basis by using this screen.
- Configure proxy settings for all transfers by clicking **Global Preferences**. This requires root privileges.

**My Preferences** ×

↑↓ Transfers   
 ✉ Mail   
 🌐 Proxy

Enable transfer proxy  
 Host:  Port:   
 Secure  
 Username:   
 Password:   
 Enable HTTP proxy  
 Host:  Port:   
 Authenticated  
 Username:   
 Password:

### Case-by-Case Settings

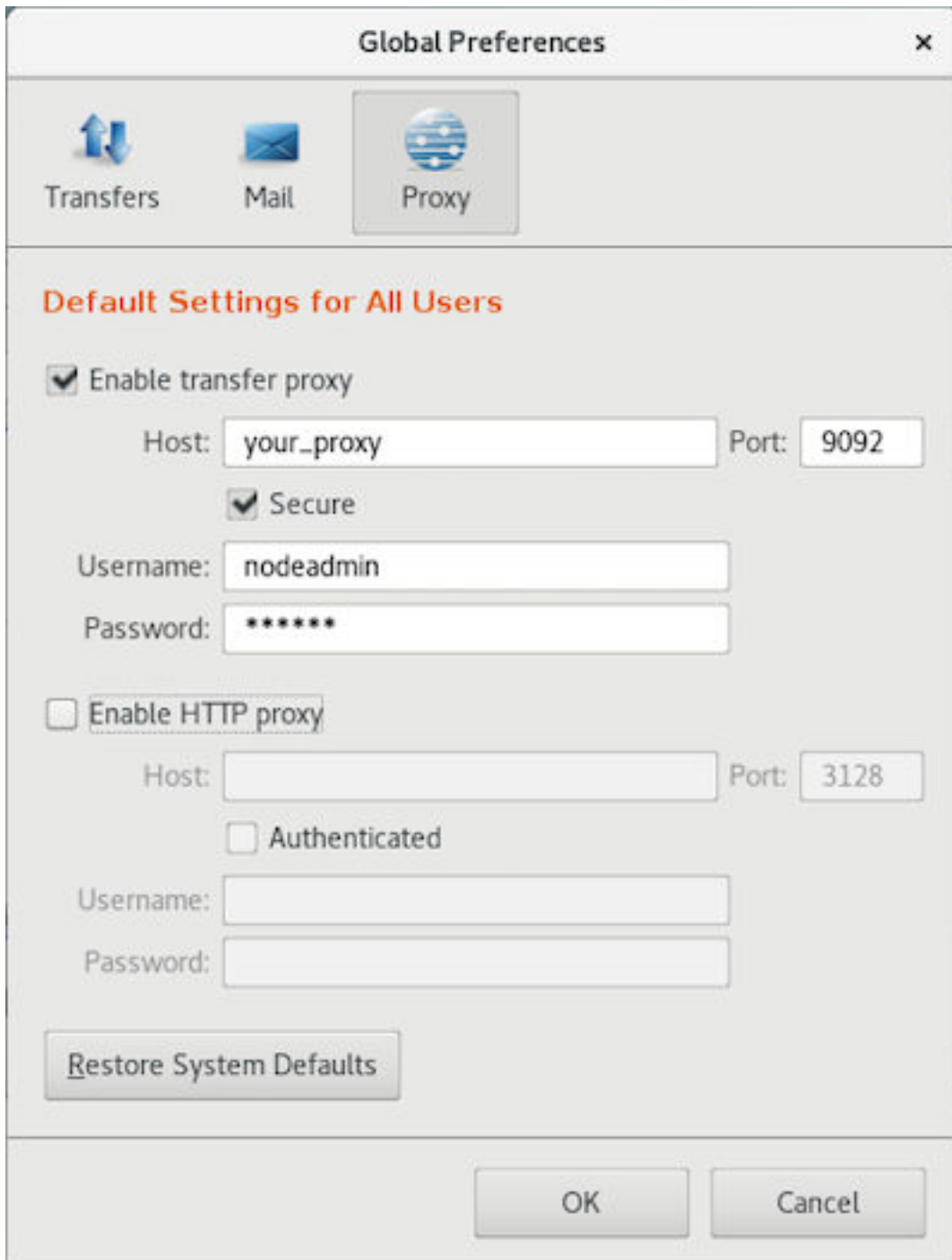
1. Check the **Enable transfer proxy** checkbox if you want to turn on transfer proxy and override global settings for connecting to your proxy server.
2. Enter the proxy server’s hostname or IP address, and enter the port number.
3. Enable the **Secure** checkbox if your proxy server allows secure connections.
4. Enter the proxy server’s Node API username and password. This is the Node API user that you created when you were configuring the proxy server. (See [“Configuring the proxy server for forward proxy”](#) on page 7).

### Global Settings

Setting global preferences for proxy transfers requires root/admin privileges. To configure your global proxy settings, click the **Global Preferences** button. In the **Global Preferences** window, enter the choices as follows:

1. Check the **Enable transfer proxy** checkbox. Note that the transfer proxy facility is disabled by default.

2. Enter the proxy server's hostname or IP address, and enter the port number.
3. Enable the **Secure** checkbox if your proxy server requires secure connections (recommended).
4. Enter the proxy server's Node API username and password. This is the Node API user that you created when you were configuring the proxy server. (See [“Configuring the proxy server for forward proxy”](#) on page 7).



The screenshot shows the 'Global Preferences' dialog box with the 'Proxy' tab selected. The 'Default Settings for All Users' section is visible. The 'Enable transfer proxy' checkbox is checked. The 'Host' field contains 'your\_proxy' and the 'Port' field contains '9092'. The 'Secure' checkbox is also checked. The 'Username' field contains 'nodeadmin' and the 'Password' field contains six asterisks. The 'Enable HTTP proxy' checkbox is unchecked. The 'Host' field is empty and the 'Port' field contains '3128'. The 'Authenticated' checkbox is unchecked. The 'Username' and 'Password' fields are empty. A 'Restore System Defaults' button is located at the bottom left. 'OK' and 'Cancel' buttons are at the bottom right.

## Transferring from the command line through forward proxy

Command line **ascp** and **ascp4** transfers can be run through a forward proxy.

The syntax depends on whether proxy authentication is required, that is, whether `<server>/<proxy>/<authentication>` is set to `false`. In either case, you are prompted for your password unless the

environment variable `ASPERA_SCP_PASS=password` has already been set. Specify the Proxy port if you are using a port other than the default ports for DNAT and DNATS protocols (ports 9091 and 9092).

**Note:** If you are transferring with `ascp4`, the version of `ascp4` on the server and on the client must be the same.

- With no proxy authentication required (<authentication> is set to `false` in `aspera.conf`):

```
$ ascp options --proxy dnat[s]://proxy_user@proxy_hostname[:port] file1[,file2,...]
username@dest_hostname:target_path
```

- With proxy authentication (<authentication> is set to `true` in `aspera.conf`):

```
$ ascp options --proxy dnat[s]://proxy_username:passwd@proxy_hostname[:port]
file1[,file2,...] username@dest_hostname:target_path
```

If the environment variable `ASPERA_PROXY_PASS=proxy_server_passwd` has been set, you do not need to specify the proxy server password.

## Examples

- No proxy authentication required: In the following command the user Sue transfers the file `/data/file1` to `/Sue_data/` on `ihost.com`, through the proxy server at `phost.com`. The username `user` is necessary but only as a placeholder; it can be anything. No password is required. After running the command, Sue is prompted for the **ascp** password.

```
$ ascp --proxy dnat://user@phost.com /data/file1 Sue@ihost.com:/Sue_data/
```

- Proxy authentication required: The following example is the same as the previous one, except that authentication is required. In this case, the proxy user (`aspera_proxy`) and password (`pa33w0rd`) are required. After running the command, Sue is prompted for the **ascp** password.

```
$ ascp --proxy dnats://aspera_proxy:pa33w0rd@phost.com /data/file1 Sue@ihost.com:/Sue_data/
```

## Reverse proxy

The following references describe the proxy settings that can be configured for reverse proxy.

### Configuring the proxy server for reverse proxy

Reverse proxy is used to route incoming transfers from the proxy server to the internal server destination. To do so, user accounts must be set up on the proxy server and rules that dictate how transfers are routed must be configured. The following instructions describe the steps to set up user accounts, grant them sudo access, and configure settings and logging.

#### Creating and authorizing users on the Proxy Server

Proxy user accounts can be set up in two ways:

- **Squashed user account:** Multiple users make transfers to a single “squashed” user account on the internal server. No individual accounts are required on the destination server, but individual accounts are still required on the proxy server. The `squash-user` account is required only on the destination server, not on the proxy server. At the destination, the transferred files are owned by the `squash-user`. The `squash-user` approach is generally considered the best choice for IBM Aspera Faspex.
- **Individual user accounts:** Each user makes transfers to their own account on the destination server. The individual user accounts must exist at the destination, as well as on the proxy server. When transferred files arrive at the destination, they are still owned by the user who is specified when the transfer was initiated.

Proxy supports a mix of these two approaches. The following steps cover the setup of both squashed and individual accounts:

1. Log in to the Proxy server as root and create an account for each user.

You do not need to set up a squashed user account on the Proxy server, but you do need individual accounts for each user that will use the squashed account.

2. a. Generate an SSH key pair for each user on the proxy server:

```
$ su - username -c ssh-keygen
```

**Note:** You can specify the key type by using `-t key_type`. Consider that the key type depends on your configuration and workload, so suggesting a specific type is not within the scope of this guide.

The **ssh-keygen** generates and copies the private key and public key to the `.ssh` directory in the user's home directory, typically `/home/username/.ssh`.

If you are using a squashed user account on the Proxy server, generate an SSH key pair on the Proxy server by using the same command.

3. Add the public keys for individual or squashed user accounts to the appropriate server.

For each user, create the file `authorized_keys` in `/home/username/.ssh` on the Proxy server. Copy and paste the text of each user's public key into their corresponding `authorized_keys` file.

For a squashed user account, create the file `/home/squash_username/.ssh/authorized_keys` on the internal servers and copy and paste the text of the squashed user's public key into their `authorized_keys` file.

If you created the `.ssh` directories and `authorized_keys` files as root, be sure to change the ownership from root to the transfer user.

4. On the Proxy server, set the default shell to `/bin/aspshell` for each user by running the following command:

```
# chsh -s /bin/aspshell username
```

For example,

```
# chsh -s /bin/aspshell bear
Changing shell for bear.
Warning: "/bin/aspshell" is not listed in /etc/shells.
Shell changed.
```

The warning message can be safely ignored.

## Creating a group of proxy users (optional)

Managing permissions for Proxy users is often easier if they are part of a system group.

1. Create a group for Proxy users.

```
# groupadd groupname
```

2. Confirm that the group was created.

```
# cat /etc/group
```

The new group must appear at the end of list.

3. To add users to the group, run the following command:

```
# usermod -a -G groupname username
```



## Granting sudo access to proxy users or groups

To use reverse proxy, the transfer user must be able to run the `/sbin/iptables-restore` command as **root** by using **sudo** and without a terminal (tty).

In a default Linux configuration, as **root**, create a file in `/etc/sudoers.d/` (for example, `aspera_rproxy`) containing the following three lines for each user:

```
Defaults:username !requiretty
Defaults:username secure_path = /sbin:/bin:/usr/sbin:/usr/bin
username ALL = NOPASSWD: /sbin/iptables-restore
```

You can specify multiple users in the same file or in different files. To specify a group instead of individual users, replace `username` in the previous example with `%groupname`.

Notes on **sudo**:

- The configuration file is `/etc/sudoers`.
- Groups must be specified with a leading `%`.
- By default, the last entry in the `sudoers` file is usually `#includedir /etc/sudoers.d`. This evaluates all files in the `sudoers.d` folder in lexical order.
- Configuration stanzas are evaluated in order, and the last evaluation takes precedence.

For more information, see the man page for **sudo**.

Based on these notes, it is also possible to edit the main `sudoers` files by using the **visudo** command. In this case, make sure that the previous stanzas are evaluated *after* the corresponding default generic stanzas:

```
Defaults requiretty
```

```
%wheel ALL=(ALL) ALL or %sudo ALL=(ALL) ALL (depending on your system configuration).
```

## Configure reverse proxy settings

The following configuration steps require setting values in the proxy server's `aspera.conf` file, which is found in the following location:

```
/opt/aspera/proxy/etc/aspera.conf
```

You can edit the file manually, or by using the **asconfigurator** utility. The following steps describe both methods.

The **asconfigurator** command is located in `/opt/aspera/proxy/bin`. The following examples assume that the command is already on the path. You can either add it to root's path or add a prefix to the path to the command each time you run it.

For more information about the `aspera.conf` settings and corresponding **asconfigurator** settings, see [“Reverse Proxy configuration options” on page 17](#).

1. Enable reverse proxy:

```
# asconfigurator -x "set_server_data;rproxy_enabled,true"
```

This command adds the following to the `<server>` section of `/opt/aspera/proxy/etc/aspera.conf`:

```
<server>
  <rproxy>
    <enabled>true</enabled>
  </rproxy>
</server>
```

2. Create forwarding rules.

**Single Rule:** If you are setting only one rule, you can use **asconfigurator** commands. Rules must specify a host, which is the IP address of the internal server. You can also specify the port to use, rather than the default 22. To set a rule and specify the file to use for SSH authentication, run the following commands:

```
# asconfigurator -x "set_server_data;rproxy_rules_rule_host,host_ip_address[:port]"
# asconfigurator -x "set_server_data;rproxy_rules_rule_keyfile,filepath"
```

**Multiple Rules:** You can specify different rules that are keyed by the IP address or hostname that is used for connecting to the Proxy server. For example, by using multiple rules you can set one rule block for transfers to `faspex.asperasoft.com` and set another for transfers to `shares.asperasoft.com`.

**Authentication:** Each rule requires a `<keyfile>` setting of `$(user)/.ssh/id_ed25519`, which specifies the location of the SSH private keyfile. If no `<squash_user>` is specified, the proxy server uses the proxy user's account to authenticate with the internal server.

For example, to set a rule such that transfers destined for Proxy host `7.7.7.7` are forwarded to internal server `10.0.0.10`, add the following:

```
<server>
  <rproxy>
    <enabled>true</enabled>
    <rules>
      <rule host_ip="7.7.7.7">
        <host>10.0.0.10:22</host>
        <keyfile>/home/$(user)/.ssh/id_ed25519</keyfile>
      </rule>
    </rules>
  </rproxy>
</server>
```

To set an additional rule such that transfers destined for `7.7.7.8` must be forwarded to the squashed user account `xfer` on internal server `10.0.0.30`, add the following:

```
<server>
  <rproxy>
    <enabled>true</enabled>
    <rules>
      <!-- Incoming SSH connections to 7.7.7.7 -->
      <rule host_ip="7.7.7.7">
        <host>10.0.0.10:22</host>
      </rule>

      <!-- Incoming SSH connections to 7.7.7.8 -->
      <rule host_ip="7.7.7.8">
        <host>10.0.0.30:22</host>
        <squash_user>xfer</squash_user>
        <keyfile>/opt/aspera/proxy/etc/ssh_keys/id_ed25519</keyfile>
      </rule>
    </rules>
  </rproxy>
</server>
```

### 3. Set up logging for reverse proxy.

a. In `/etc/rsyslog.d`, create the file `aspera.conf`. Enter the following in the file:

```
local2.*    -/var/log/aspera.log
& stop
```

The previous example is for CentOS 7. In other Linux platforms, the equivalent commands are similar.

b. Restart the logger:

```
# systemctl restart rsyslog
```

c. Create the file `/etc/logrotate.d/aspera` containing the following:

```
/var/log/aspera.log {
daily
rotate 15
copytruncate
postrotate
chmod 644 /var/log/aspera || true
endscript
compress
}
```

## Reverse Proxy configuration options

Reverse Proxy requires that you specify the configuration options in the `aspera.conf`.

**Single rule:** If you are configuring a single rule, you can use **asconfigurator** to set most options, by using the syntax:

```
# asconfigurator -x "set_server_data;parameter,value"
```

**Multiple rules:** If you are configuring multiple rules or setting options that do not have **asconfigurator** parameters, you must manually edit the Proxy configuration file that is found in:

```
/opt/aspera/proxy/etc/aspera.conf
```

To display a list all reverse-proxy configuration options in well-formed XML, run:

```
# /opt/aspera/proxy/bin/asuserdata -s
```

Configuration options and default values for the reverse proxy server are contained in the `<rproxy>` subsection of the `<server>` section. The following table describes these options. For an example of an `aspera.conf` file that incorporates these options, see the following example.

### Reverse Proxy configuration options

aspera.conf option asconfigurator parameter	Description	Default Value
<code>&lt;enabled&gt;</code> <code>rproxy_enabled</code>	Turn reverse proxy on/off (true/false).	false
<code>&lt;log_level&gt;</code> <code>rproxy_log_level</code>	Set the logging level. 0 - normal 1 - normal plus debug 2 - level 1 plus additional debugging	0
<code>&lt;rule&gt;</code>	Rule with no conditional attributes.	N/A
<code>&lt;rule host_domain="hostname:port"&gt;</code> (no asconfigurator option)	Set a rule for requests that are directed to the specified hostname (and optionally the SSH port) of the proxy server. This setting can be used to set rules for different hosts and different ports. Requires an <b>ascp</b> client version 3.1 or later.	(none)
<code>&lt;rule host_ip="ipaddr"&gt;</code> (no asconfigurator option)	Rule specifying the IP address of the proxy server.	(none)

aspera.conf option asconfigurator parameter	Description	Default Value
<rule host_domain="hostname:port" host_ip="ipaddr"> (no asconfigurator option)	Combined version of the previous entry.	(none)
<host> rproxy_rules_rule_host	IP address and optional SSH port of the internal server (destination), with the syntax <i>ip_address[:port]</i> . The default port (if unspecified) is 22.	blank (null)
<hosts> (no asconfigurator option)	Specifies a list of hosts for load balancing. Each host is listed as <i>ip_address:port</i> . See <a href="#">Load Balancing</a> and “UDP port reuse” on page 19.	(none)
<proxy_port> rproxy_rules_rule_proxy_port	Proxy server port that receives FASP traffic. This port must be allowed through any firewall protecting Proxy.	33001
<bind_source_address> rproxy_rules_rule_bind_source_address	Bind the outgoing TCP channel to a specified IP address. By default the system assigns the source IP address to use for its SSH session.  <b>Important:</b> For Proxy servers with more than one NIC, <bind_source_address> must be specified to identify the interface for the connection with Enterprise Server.	blank (null)
<balancing> (no asconfigurator option)	Enables load balancing. The internal servers that are specified in the hosts sections are used for load balancing. Round-robin selection is the only supported method. For details, see <a href="#">Load Balancing</a> .	load balancing not enabled
<squash_user> rproxy_rules_rule_squash_user	Squash account name that is used for authenticating with the internal server.	blank (null)
<keyfile> rproxy_rules_rule_keyfile	Path of the SSH private key for authenticating with the internal server.	blank (null)
<src_port_filtering> rproxy_rules_rule_src_port_filtering	Enable/disable (true/false) FASP source-port filtering on or off (true/false). For details, see <a href="#">Source-Port Filtering</a> .	false
<udp_port_reuse> rproxy_rules_rule_udp_port_reuse	Setting this option to false enables reverse proxy to create iptables rules that increment the UDP port number to which clients send each concurrent transfer and the internal server's UDP port to which the transfer is routed. For details, see “UDP port reuse” on page 19. <b>Note:</b> Must be set to false when internal servers are running Windows.	true

### Sample aspera.conf for Reverse Proxy

The following sample includes three example rules.

1. **Minimal rule:** Incoming transfers are forwarded to the specified internal server (10.0.0.10:22) if they have valid SSH key authentication. Other configuration options take default values (or in the case of load balancing, are not enabled).

```
<rule>
  <host>10.0.0.10:22</host>
  <keyfile>/home/$(user)/.ssh/id_ed25519</keyfile>
</rule>
```

2. **Load-balancing rule:** Incoming transfers are forwarded to the specified internal servers (10.20.103.133-135:33001). As in rule 1, transfers only sent by users with valid SSH key authentication are allowed through the reverse proxy. Since three hosts were specified and the default UDP port is 33001, the three incremental ports UDP/33001-33003 must be open on the external firewall.

```
<rule>
  <hosts>
    <host>10.20.103.133:33001</host>
    <host>10.20.103.134:33001</host>
    <host>10.20.103.135:33001</host>
  </hosts>
  <keyfile>/home/$(user)/.ssh/id_ed25519</keyfile>
  <udp_port_reuse>true</udp_port_reuse>
  <balancing>round_robin</balancing>
</rule>
```

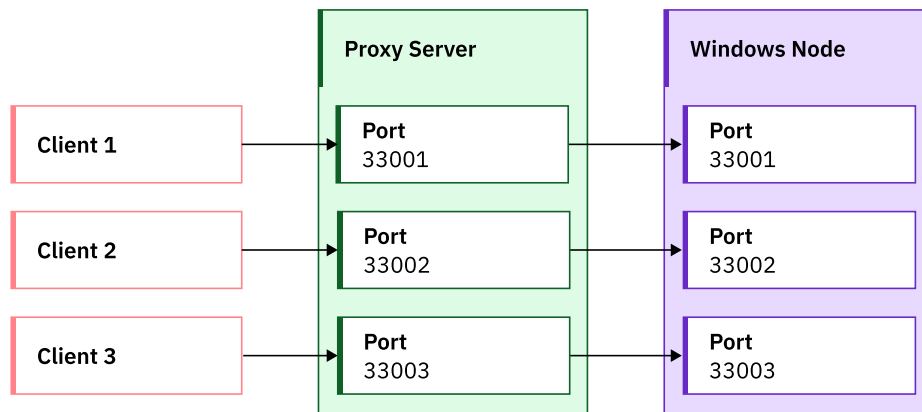
3. **UDP port reuse and squash user account rule:** Incoming transfers to the proxy server with the IP address of 10.20.101.151 that are destined to an internal Windows server are received and forwarded on incremental UDP ports beginning with 5555. Transfers are forwarded as the squash user once the clients were authenticated on the proxy server. The incremental UDP ports that begin with 5555 and up to the number of concurrent UDP transfers allowed must be open on the external firewall. For example, if 10 concurrent UDP streams are allowed, UDP ports 5555-5564 must be open.

```
<rule host_ip="10.20.101.151">
  <proxy_port>5555</proxy_port>
  <squash_user>sender</squash_user>
  <keyfile>/opt/aspera/proxy/etc/ssh_keys/id_ed25519</keyfile>
  <udp_port_reuse>>false</udp_port_reuse>
</rule>
```

## UDP port reuse

When a transfer server is running on Windows, concurrent transfers cannot bind to the same FASP (UDP) port. Each concurrent transfer binds to the next open port starting from the default port of 33001.

When the nodes behind the reverse proxy server are Windows hosts, the iptables rules that are created must account for this and create a rule that routes traffic from a UDP port on the proxy server to a UDP port on the destination host. To enable this feature set the <udp\_port\_reuse> option to false. When set to false, reverse proxy creates rules where each concurrent transfer through the proxy server gets an incremented UDP port number for the clients to send to. Then, the iptables rule routes the traffic from that port to the **ascp** server's port. In this scenario, the UDP ports are incremented by 1 with each concurrent connection. The following diagram shows how the UDP ports on the proxy server and the Windows target are updated with each concurrent transfer.



In prior versions of Proxy, this configuration might be achieved by setting `<proxy_port>` to 0. However, setting it to zero is no longer supported and now results in an error.

Whether you want to set `<udp_port_reuse>` to true or false, add the `udp` tags under the `<rules>` tag, and between the `<rule>` tag. For example,

```
<server>
  <iproxy>
    <enabled>true</enabled>
    <rules>
      <rule host_ip="1.1.1.1">
        <hosts>
          <host>2.2.2.2:33001</host>
        </hosts>
        <keyfile>/home/$(user)/.ssh/id_ed25519</keyfile>
        <udp_port_reuse>>false</udp_port_reuse>
      </rule>
      <log_level>2</log_level>
    </rules>
  </iproxy>
</server>
```

For more information about how to set up load balancing when `<udp_port_reuse> = false`, see [“UDP port reuse and load balancing on Windows”](#) on page 21.

**Note:** When setting `<udp_port_reuse>` to false, and thereby enabling the use of incremented UDP port numbers, make sure the UDP ports that are specified by `<proxy_port>` are open on the external firewall.

## Load balancing

The host rule for a single proxy instance can include a list of multiple destination servers that reverse proxy will use to distribute transfer sessions.

If a destination does not respond, reverse proxy marks it as faulty, logs this information, and tries the next destination in the list. While the server is marked as faulty, reverse proxy won't forward transfers to it. After 2 minutes, the faulty mark is removed, and the server is returned to the list of available servers.

### Enabling load balancing

To turn on the balancing feature in the proxy server's `aspera.conf`, set `<balancing>` to `round_robin` (currently the only supported balancing method). Each destination is specified with a `<host>` tag and the

host tags list is defined by using <hosts>. Each <host> specifies the destination host's IP address and the SSH port to which it connects. For example,

```
<rule host_ip="10.20.101.151">
  <balancing>round_robin</balancing>
  <hosts>
    <host>10.20.103.133:33001</host>
    <host>10.20.103.134:33001</host>
    <host>10.20.103.135:33001</host>
  </hosts>
  <keyfile>/home/$(user)/.ssh/id_ed25519</keyfile>
</rule>
```

## UDP port requirements for load balancing

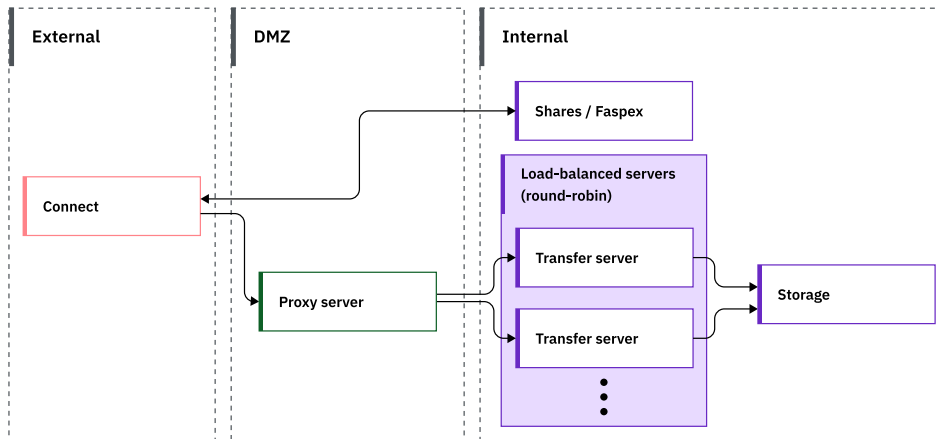
Load balancing requires exposing multiple UDP ports on reverse proxy. Ports are opened based on the value that is specified by <proxy\_port> (33001 if not specified). The formula for port exposure is *proxy port base + the number of the host* in the load-balance configuration. For example, if the load is to be balanced among three host destinations and the `aspera.conf` configuration uses the default UDP proxy port value (33001), the three UDP ports for reverse proxy are 33001, 33002, 33003. In this case, the port numbers that are used on the destination hosts do not matter.

The reverse proxy server in the previous example determines port numbers by incrementing from proxy port 33001, the default port number. However, users might want to expose different port numbers to the outside world. To do this, set <proxy\_port> to a different value, and reverse proxy will use this as the starting point for opening the series of consecutive port numbers for load balancing, turning off UDP port reuse, or combinations of both. For example, if <proxy\_port> is set to 5555 in the previous configuration, the four proxy ports that would be opened are 5555, 5556, 5557, and 5558.

**Note:** When setting up load balancing, make sure the UDP ports that are specified by <proxy\_port> are allowed on the firewall.

## UDP port reuse and load balancing on Windows

If load balancing is used because the destination hosts are Windows, <udp\_port\_reuse> must be set to `false`. In the following setup, concurrent transfers from four clients are being load-balanced between two Windows destinations.



## Source-Port filtering

The `source-port` filtering can be enabled or disabled.

The `<src_port_filtering>` option in `aspera.conf` enables or disables source-port filtering (`true` or `false`). By default, source-port filtering is disabled (`false`).

### When source-port filtering is enabled (`true`)

When source-port filtering is enabled, reverse proxy restricts client connections to only those UDP source ports opened internally by each transfer session. Enabling source-port filtering allows the reverse proxy to use UDP ports as dictated by network connections between clients and servers. Use this option only if there are network address translation (NAT) devices between the client and the reverse proxy that require the ports set up by the UDP sockets remain intact and unchanged. Setting this option to `true` might require changes to any firewalls in front of your reverse proxy to allow for the different UDP ports.

### When source-port filtering is disabled (`false`)

In cases where client-side firewalls change the specified source port in transit, source-port filtering must be disabled to allow the connection to be established. When disabling source-port filtering, make sure the UDP ports that are specified by `<proxy_port>` are allowed on the external firewall.

One indication that source-port filtering might need to be disabled is when client connections fail with a timeout such as “Error establishing UDP connection (check UDP port and firewall)”. Aspera transfer logs on either the client or server side will also show “Client unable to connect to server (check UDP port and firewall)” or “Server unable to hear from client (check UDP port and firewall)”. If the same timeout errors still occur when source-port filtering is disabled, this generally indicates that traffic is being blocked at a firewall. For related information, see [“UDP port and firewall timeout errors” on page 36](#).

**Note:** Disabling source-port filtering relaxes reverse proxy security and therefore must be used only when necessary.



## Reverse Proxy firewall configuration

In the following sections, you can find the information to configure the Reverse Proxy firewall.

### External Firewall

The TCP and UDP ports on which the internal server is listening must be allowed. By default, these are TCP/22 and UDP/33001.

### Internal Firewall

The TCP and UDP ports on which the internal server is listening must be allowed, but only for connections that originate from the proxy server. By default, the ports are TCP/22 and UDP/33001.

**Note:** If `proxy_port` was set to a value other than 33001, or was set to a range to accommodate load balancing or UDP port reuse, then that port or range of ports must be allowed through the external and internal firewalls. For more information on port configuration for load balancing, see [“Load balancing” on page 20](#). For more information on port configuration for UDP port reuse, see [“UDP port reuse” on page 19](#).

## Configuring internal servers

To configure internal servers, the internal server must be running an Aspera Transfer server.

1. Log in to the internal server as root.
2. Create an account for each user who is not using the squash-user account. If you are using a squash-user account, create an account for the squash user.
3. For each user, including the squash user, create the file `/home/user/.ssh/authorized_keys`. Copy and paste the text from the user's public key (generated for that user when you ran **ssh-keygen** on the proxy server) into the `authorized_keys` file.

## Transferring files with Reverse Proxy

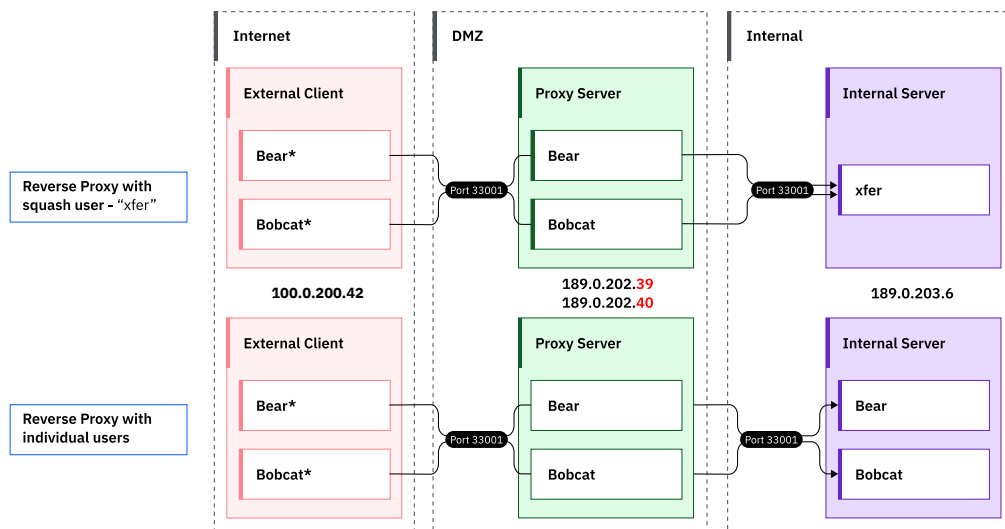
Once the configuration tasks were completed for the proxy server, internal server, and external clients, file transfers from external users are transparent.

To make transfers to the internal server, users need to specify only the following parameters:

- The IP address or hostname on the proxy server that corresponds to the internal destination.
- The correct SSH port for the connection to the proxy server.
- The target directory on the internal server.
- Any optional parameters that are related to a transfer session.

### From the command line

The following configuration examples show the squash-user and individual-account approaches in the same system:



\* Although the usernames on the external client are **bear** and **bobcat** in the previous example, they do not need to correspond to usernames on the Proxy server and internal server.

The reverse-proxy rules for each configuration are defined on the proxy server in `aspera.conf`:

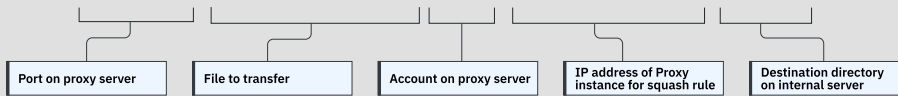
```

<server>
  <rproxy>
    <enabled>true</enabled>
    <rules>
      <rule host_ip="189.0.202.39">
        <host>189.0.203.6:33001</host>
        <squash_user>xfer</squash_user>
        <keyfile>/opt/aspera/proxy/etc/ssh_keys/id_ed25519</keyfile>
      </rule>
      <rule host_ip="189.0.202.40">
        <host>189.0.203.6:33001</host>
        <keyfile>/home/$(user)/.ssh/id_ed25519</keyfile>
      </rule>
    </rules>
  </rproxy>
</server>

```

Users **bear** and **bobcat** have valid SSH key pairs and accounts on the proxy server. From the command line, **bear** runs the following **ascp** command specifying the proxy instance that is governed by the squash rule:

```
$ ascp -P 33001 testfile_bear bear@189.0.202.39:/user/bear
```



Since the rule for proxy instance 189.0.202.39 specifies a squash user (`xfer`), the file that belongs to `bear`, `bobcat`, or anyone that uses that proxy instance, will be owned by `xfer` when it arrives on the internal server.

The `-P 33001` flag specifies the SSH port on the proxy server (not the port on the internal server, which is specified in the rule). The port must be specified on the command line if port 22 is disabled in `/etc/ssh/sshd_config`.

Users `bear` and `bobcat` have valid SSH key pairs and accounts on both the proxy server and the internal server. From the command line, `bobcat` runs the following `ascp` command specifying the proxy instance for the individual-user approach:

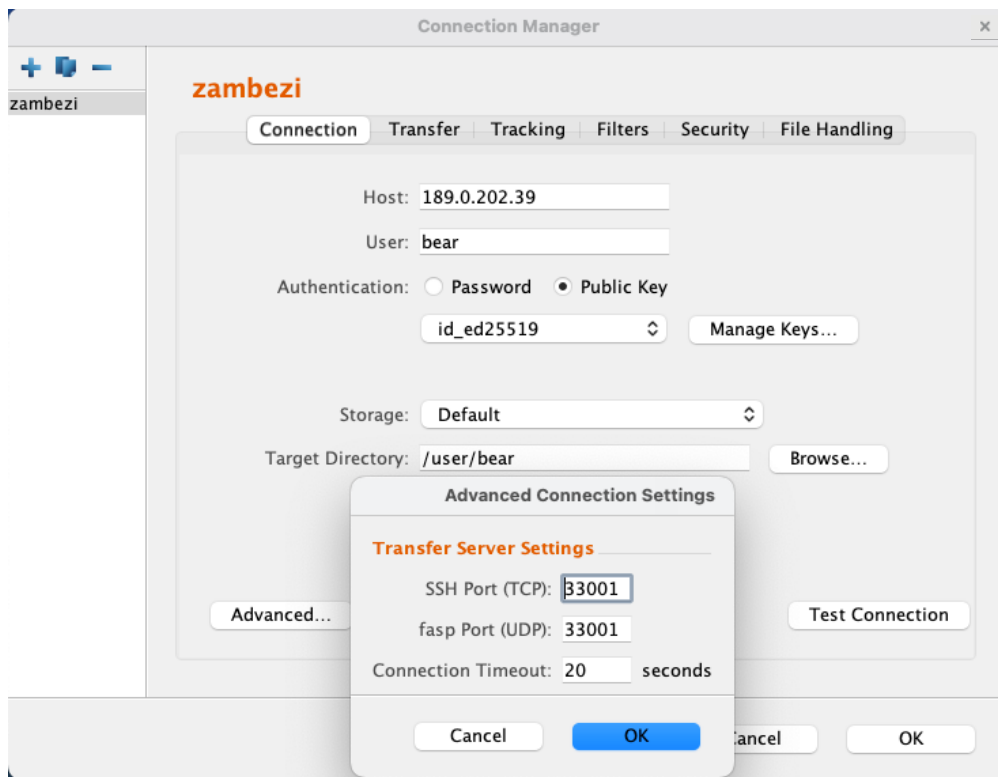
```
$ ascp -P 33001 testfile_bobcat bobcat@189.0.202.40:/user/bobcat
```

Since the rule for proxy instance 189.0.202.40 does not specify a squash user, the file is still owned by `bobcat` when it arrives on the internal server.

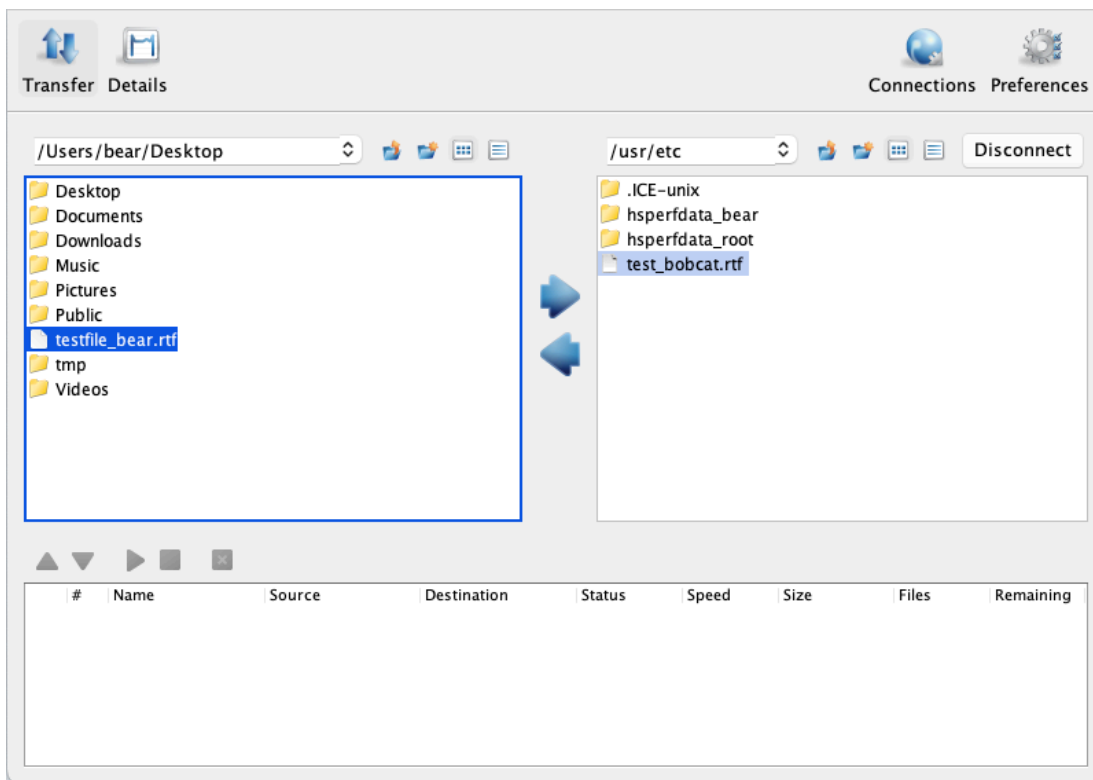
## From High-Speed Transfer Server GUI

All GUI-based Aspera transfer products can be used with Proxy, as well.

For example, user `bear` also made the previous transfer with the HSTS GUI. In the following display, `bear` set up a connection called “`zambezi`” using the same parameters that were previously described. The IP address of the proxy instance 189.0.202.39 (squash-user rule) is specified as the host. The file name for `bear`’s private SSH key is specified under Authentication/Public Key. The target directory on the internal server is specified as `/user/bear`. The ports are specified as 33001 on the Advanced Connection Settings menu accessed from the **Advanced** option.



When bear's connection to the proxy server is established, the /user/bear target directory on the internal server is visible as in the right panel in the following image, and ready for bear to make the transfer.



## Using IBM Aspera Sync and IBM Aspera Connect with reverse proxy

Sync and Connect can be use with reverser proxy by configuring with round-robin load balancing.

Using Sync and Connect with reverse proxy offers the following advantages:

- Enables the use of the round-robin load-balancing function of the proxy to distribute the load, boost efficiency, and provide a semblance of high availability for larger Sync transfers. For more information, see [Load Balancing](#).
- Allows Connect users outside a corporate network firewall to sync with internal corporate storage.

### Configuring Sync for reverse proxy with round-robin load balancing

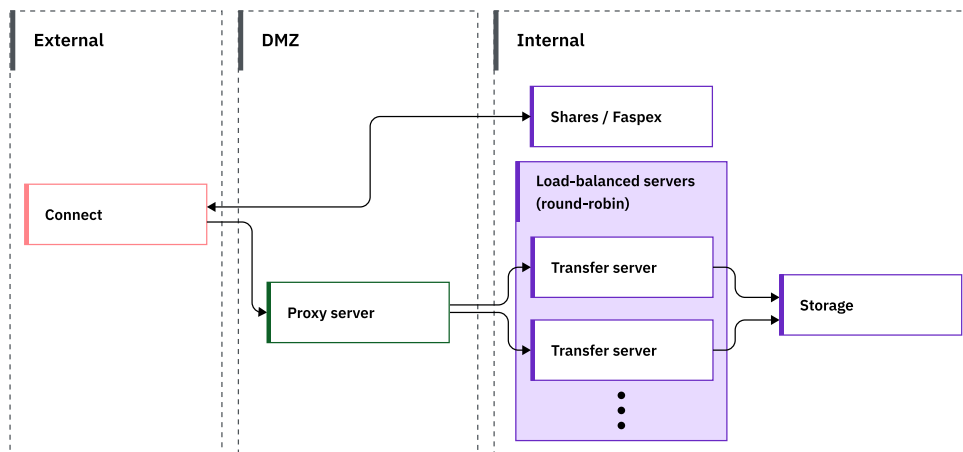
Normally, the SQL database that is used by **async** is located on the transfer server. However, when using reverse proxy in a round-robin load-balancing configuration, **async** can connect to multiple servers.

The storage has to be mounted in the same location on each transfer server for the round-robin setup to work. By default, the databases are kept in the directory that is being synced. To save in alternate locations, see the *Aspera Sync* section in the [IBM Aspera High-Speed Transfer Server Admin Guide](#) for details on `async_db_dir`, `async_db_store_dir`, `async_lock`.

**Note:** **Async** proxy sessions are always logged to syslog.

### Configuring Connect for reverse proxy with round-robin load balancing

External Aspera Connect users can view and transfer files and directories on internal Aspera servers (usually IBM Aspera Shares or IBM Aspera Faspex, although IBM Aspera High-Speed Transfer Server, or High-Speed Transfer Endpoint can also be used) through Reverse Proxy with additional configuration. In the following diagram, the external Connect user connects to the internal Shares or Faspex server (through Reverse Proxy) through which they can access content on the load-balanced internal servers.



1. On the internal server, set the value for `<server_name>` in `aspera.conf` as the IP address or hostname of the Proxy server by running the following command:

```
# asconfigurator -x "set_server_data;server_name,proxy_address"
```

2. On the internal server, choose or create an Aspera transfer user (a system user that has a docroot configured in `aspera.conf`) and then associate the transfer user with a Node API username and password. For more information, see the admin guide for your Aspera server. To create a system user, assign a docroot, and associate the transfer user with a Node API user, run the following commands:

```
# useradd username
# asconfigurator -x "set_user_data;user_name,username;absolute,docroot_path"
# /opt/aspera/bin/asnodeadmin -a -u node_api_username -p node_api_passwd -x username
```

3. On the load-balanced internal servers, configure the database storage as described in the previous section for Aspera Sync.
4. On the Proxy server, create a user account that has the same username as the transfer user on the internal server.
5. On the Proxy server, configure the transfer user with the Aspera public key by running the following commands:

```
# mkdir /home/username/.ssh
# cat /opt/aspera/proxy/var/aspera_tokenauth_id_ed25519.pub >> /home/username/.ssh/authorized_keys
# chown -R username:username /home/username/.ssh
# chmod 700 /home/username
# chmod 700 /home/username/.ssh
# chmod 600 /home/username/.ssh/authorized_keys
```

6. On the Proxy server, create a new SSH key pair for the transfer user to authenticate to the internal server. Run the following command in the `.ssh` folder to create a key pair. For `key_type`, specify RSA (`rsa`). At the prompt for the key-pair's file name, press ENTER to use the name `id_rsa` or `id_ed25519`, or enter a different name, such as your username. For a passphrase, you can either enter a password or press return twice to leave it blank:

```
# ssh-keygen
```

**Note:** You can specify the key type by using `-t key_type`. Consider that the key type depends on your configuration and workload, so suggesting a specific type is not within the scope of this guide.

Retrieve the public key from `/home/username/.ssh/key_name.pub` and append it to the transfer user's `authorized_keys` file on the internal server by using a process similar to the previous step.

## Configuring reverse proxy for use with Faspex and Shares

To use IBM Aspera Faspex and IBM Aspera Shares from behind a reverse proxy, you must configure both the transfer nodes (IBM Aspera High-Speed Transfer Server or IBM Aspera High-Speed Transfer Endpoint) that are used by Faspex and Shares and are running behind a reverse proxy, and the proxy server.

### Configuring the transfer nodes

#### Procedure

1. Authorize a public SSH key for use by the transfer user.

Log in as the transfer user to ensure that the user owns any files that are created. Create the directory `/.ssh` in the transfer user's home directory and create the file `authorized_keys` (with no `.txt` extension) in the new `/.ssh` directory:

Operating system	Path
Windows	C:\Users\faspex\.ssh\authorized_keys
Linux	/home/faspex/.ssh/authorized_keys
MacOS	/Users/faspex/.ssh/authorized_keys

Aspera provides a public key in the file `aspera_tokenauth_id_rsa.pub` stored in the following locations:

Operating system	Path
Windows	C:\Program Files[ (x86)]\Aspera\Enterprise Server\var\aspera_tokenauth_id_rsa.pub
Linux	/opt/aspera/var/aspera_tokenauth_id_rsa.pub

Operating system	Path
MacOS	/Library/Aspera/var/aspera_tokenauth_id_rsa.pub

Copy and paste the public key into the transfer user's `authorized_keys` file. Save the file and confirm that `.ssh` and `authorized_keys` are owned by the user.

**Note:** On Linux and MacOS, permissions on these files must be set as specified in the admin guide for your server product. See *Configuring for Faspex* or *Configuring for Shares*.

2. Set the transfer user's default shell to `aspsshell`.

The transfer user's username is typically **faspex** for Faspex and **asp1** or **shares** for Shares. The method to set the default shell to `aspsshell` varies by operating system:

- **Windows:** Set the transfer user's docroot in the GUI to automatically set up the user with the Aspera shell.
- **Linux and MacOS:** Change the transfer user's default shell by running the following commands as root. For example, by using user `faspex` as the user):

```
$ chsh -s /bin/aspsshell faspex
```

If a warning message appears saying `/bin/aspsshell` is not listed in `/etc/shells`, it can be safely ignored.

- **MacOS:** You can also change the transfer user's default shell from the GUI. In **System Preferences**, click **Accounts** or **Users & Groups** (depending on your version). Click **Click the lock to make changes** and enter admin credentials. Right-click the transfer-user account name and click **Advanced Options**. Look for the "Login shell" field and replace the default value `/bin/bash` with `/bin/aspsshell`.

For further information about `aspsshell`, see the guide for your Aspera server product (Connect Server, Enterprise Server, or Point-to-Point).

3. Configure token authentication is set for the transfer user.

Token authentication can be set in the GUI or using the **asconfigurator** command.

**Note:** If the transfer nodes are in a cluster, use the same token encryption key on all nodes in the cluster.

**From the GUI:**

- Click the **Configuration** button to open the **Server Configuration** dialog.
- Click the **Users** tab and click the transfer user. In the right pane, click the **Authorization** tab.
- For **Incoming Transfers** and **Outgoing Transfers**, select **Override** and select **token** from the dropdown menu.
- For **Token Encryption Key**, select **Override** and set the value to your encryption key.

**Using asconfigurator:**

- For Linux and MacOS, open a Terminal window as root. For Windows, start Command Prompt as an administrator (go to Search from the taskbar and type **Command Prompt**, right-click **Command Prompt**, and click **Run as administrator**).

- To require a valid token for transfers *to* this computer, run the following command:

```
# asconfigurator -x
"set_user_data;user_name,transfer_username;authorization_transfer_in_value,token"
```

- To require a valid token for transfers *from* this computer, run the following command:

```
# asconfigurator -x
"set_user_data;user_name,transfer_username;authorization_transfer_out_value,token"
```

- To specify the token encryption key, run the following command:

```
# asconfigurator -x
"set_user_data;user_name,transfer_username;token_encryption_key,my_secret_key"
```

## Configuring the proxy server

### Procedure

1. Create the `faspex` or `shares` transfer user on the proxy server.

For instructions on creating the transfer user, see [“Configuring the proxy server for reverse proxy” on page 13](#). Set the default shell to `aspshe11`, as described previously in *Configuring the Transfer Nodes*.

2. Confirm that the transfer user is using the correct SSH private key.

File permissions for the private key must be set as follows for Faspex (transfer user **faspex**), similarly for Shares:

```
# cd /home/faspex
# chown faspex:faspex .ssh
# chmod 700 .ssh
# chmod 600 .ssh/id_ed25519
```

3. Authorize the public SSH key for use by the transfer user.

Cut and paste the public key text from `/opt/aspera/var/aspera_tokenauth_id_rsa.pub` into the file `/home/transfer_username/.ssh/authorized_keys`, as described in step 3 in the previous section.

**Note:** You might use a newly generated set of SSH private/public keys for authenticating the transfer sessions coming from the reverse proxy to the transfer node, rather than those provided by Aspera. This ensures that no FASP transfer session can be established without going through the reverse proxy.

## Configuring reverse proxy for HTTP fallback with Nginx

Aspera Proxy can be configured to handle HTTP Fallback transfers so that the internal Aspera transfer is not accessible directly from the Internet. Proxy does this using the Nginx™ web server, which is packaged with Proxy but must be configured as described in the following steps.

### Procedure

1. Create the Aspera-specific Nginx configuration file.

The Proxy installer includes an Aspera-specific example Nginx configuration file `/opt/aspera/nginx/conf/nginx.conf.example`. Copy this file to create the file that Nginx uses, `/opt/aspera/nginx/conf/nginx.conf`.

2. Edit `nginx.conf` with a text editor.

Comments within the file show you where to enter the IP addresses of your internal servers. If your internal Aspera servers use a different port other than 8080 for HTTP or 8443 for HTTPS connections, change the values in `nginx.conf` to match.

3. Restart the Nginx server.

4. Verify your configuration.

You can test your configuration from the command line or by using a web browser.

**Command line:** Run the following command:

```
# netstat -anp | grep nginx
```

The output is similar to the following if the IP addresses and ports were configured correctly:

```
tcp  0  0  0.0.0.0:8443  0.0.0.0:* LISTEN  6438/nginx: master
tcp  0  0  0.0.0.0:8080  0.0.0.0:* LISTEN  6438/nginx: master
```



**Web browser:** Enter the following address in your web browser, for HTTP or HTTPS, and using the correct port if different from the default:

```
http://proxy_ip_address:8080/aspera/http/server_status
https://proxy_ip_address:8443/aspera/https/server_status
```

## Lua scripting with reverse proxy

Reverse proxy supports dynamic evaluation of HTTP fallback requests that use Lua scripting.

Lua scripts can be used to allow, reject, or modify HTTP fallback transfer requests in real time as they are received by the Proxy server. Lua scripts have access to the HTTP request information such as body, headers, and URL. The Lua scripting facility is included in the Nginx™ software that is distributed with Proxy.

### Configuring Lua

The primary configuration file for Lua is `/etc/nginx/nginx.conf`. To enable the importing of external Lua libraries, either compiled or written in Lua, set `lua_package_path` and `lua_package_cpath` in the `nginx.conf` file. These settings must be inside the `http` directive and precede any server directives.

```
http {
    ...
    # set search paths for pure Lua external libraries (';' is the default path):
    lua_package_path '/opt/aspera/nginx/lua/?.lua;';
    # set search paths for Lua external libraries written in C (can also use ';;'):
    lua_package_cpath '/opt/aspera/nginx/lua/?.so;';
    server {
        ...
    }
}
```

This allows sources in `/opt/aspera/nginx/lua` to be imported by using a `require` statement.

### Running Your Scripts

To start your Lua scripts, use a `content_by_lua` or `access_by_lua` block in a location directive. For example,

```
server {
    ...
    location /url/path/to/resource {
        content_by_lua 'ngx.say("Hello world")'
    }
}
```

For a list of Lua directives available in Nginx, see:

```
https://github.com/openresty/lua-nginx-module#readme
```

### Errors

If errors are encountered during the evaluation of a Lua script, they are logged to the Nginx error log located here:

```
/opt/aspera/nginx/logs/error.log
```

### Limitations

The Nginx component of Proxy was not compiled with luaJIT. For this reason, the `*_block` directives cannot be used. For example, `content_by_lua_block` and `access_by_lua_block` are not available; use `content_by_lua` or `access_by_lua` instead.

## Appendices

---

The following sections cover additional information about Aspera proxy.

### Securing your SSH Server

Keeping your data secure is critically important. Aspera strongly encourages taking additional steps to set up and configure your SSH server so that it's protected against common attacks.

These steps include:

1. Changing the TCP port.
2. Restricting user access.

#### Changing the TCP Port

Most automated robots try to log in to your SSH server on Port 22 as root with various brute force and dictionary combinations in order to gain access to your data. Furthermore, automated robots can put enormous loads on your server as they perform thousands of retries to break into your system. This topic addresses steps to secure your SSH server against potential threats, including changing the default port for SSH connections from TCP/22 to TCP/33001.

It is known that SSH servers listen for incoming connections on TCP Port 22. As such, Port 22 is subject to countless, unauthorized login attempts by hackers who are attempting to access unsecured servers. A highly effective deterrent is to simply turn off Port 22 and run the service on a seemingly random port above 1024 (and up to 65535). To standardize the port for use in Aspera transfers, we recommend using TCP/33001.

**Note:** Remote Aspera application connections attempt to establish an SSH connection by using the default port 33001. However, if the connection fails, the application attempts the connection using port 22.

The following steps explain how to change the SSH port to 33001 and take additional steps to secure your SSH server. The steps all require root access privileges.

1. Locate and open your system's SSH configuration file.

The SSH configuration file can be found in the following location:

```
/etc/ssh/sshd_config
```

2. Add a new SSH port.

**Note:** Before changing the default port for SSH connections, verify with your network administrators that TCP/33001 is open.

The OpenSSH suite included in the installer uses TCP/22 as the default port for SSH connections. Aspera recommends opening TCP/33001 and disabling TCP/22 to prevent security breaches of your SSH server.

To enable TCP/33001 while your organization is migrating from TCP/22, open port 33001 from your `sshd_config` file (where SSHD is listening on both ports). As demonstrated by this exercise, SSHD is capable of listening on multiple ports.

```
...  
Port 22  
Port 33001  
...
```

Once your client users have been notified of the port change (from TCP/22 to TCP/33001), you can disable port 22 in your `sshd_config` file. To disable TCP/22 and use only TCP/33001, comment out "Port 22" in your `sshd_config` file.

```
...  
#Port 22
```

```
Port 33001
...
```

```
$ ascp -P 33001 ...
```

### 3. Disable nonadmin SSH tunneling.

**Note:** The following instructions assume that OpenSSH 4.4 or newer is installed on your system. For OpenSSH 4.4 and newer versions, the **Match** directive allows some configuration options to be selectively overridden if specific criteria (based on user, group, hostname and/or address) are met. If you are running an OpenSSH version older than 4.4, the **Match** directive is not available; update to the latest version.

In OpenSSH versions 4.4 and newer, disable SSH tunneling to avoid potential attacks; thereby only allowing tunneling from root users. To disable nonadmin SSH tunneling, open your SSH Server configuration file, `sshd_config`, with a text editor.

Add the following lines to the end of the file (or modify them if they exist):

```
...
AllowTcpForwarding no
Match Group root
AllowTcpForwarding yes
```

Depending on your `sshd_config` file, you might have additional instances of `AllowTCPForwarding` that are set to the default `Yes`. Review your `sshd_config` file for other instances and disable.

Note that disabling TCP forwarding does not improve security unless users are also denied shell access, as they can always install their own forwarders. Review your user and file permissions, and see the following instructions on modifying shell access.

### 4. Update authentication methods.

Public key authentication can prevent brute-force SSH attacks if all password-based authentication methods are disabled. For this reason, disabling password authentication in the `sshd_config` file and enabling private/public key authentication. To do so, add or uncomment `PubkeyAuthentication yes` and comment out `PasswordAuthentication yes`.

```
...
PubkeyAuthentication yes
#PasswordAuthentication yes
PasswordAuthentication no
...
```

**Note:** If you choose leave password authentication enabled, be sure to advise account creators to use strong passwords. Be sure also to set `PermitEmptyPasswords` to "no".

```
PermitEmptyPasswords no
```

### 5. Disable root login.

OpenSSH defaults to allowing root logins; however, disabling root access helps you to maintain a more secure server. Aspera recommends commenting out `PermitRootLogin yes` in the `sshd_config` file and adding `PermitRootLogin No`.

```
...
#PermitRootLogin yes
PermitRootLogin no
...
```

Administrators can then use the **su** command if root privileges are needed.

### 6. Restart the SSH server to apply new settings.

When you finished updating your SSH server configuration, you must restart or reload the SSH service to apply your new settings. Note that restarting or reloading SSH does not impact currently connected users.

To restart or reload your SSH server, run the following commands:

OS Version	Commands
Red Hat, zLinux (restart)	<code>\$ sudo service sshd restart</code>
Red Hat (reload)	<code>\$ sudo service sshd reload</code>
Debian (restart)	<code>\$ sudo /etc/init.d/ssh restart</code>
Debian (reload)	<code>\$ sudo /etc/init.d/ssh reload</code>

#### 7. Review your logs periodically for attacks.

Review your SSH log periodically for signs of a potential attack. Locate and open your syslog—for example, `/var/log/auth.log` or `/var/log/secure`. Depending on your system configuration, syslog's path and file name might vary.

Look for invalid users in the log, especially a series of login attempts with common usernames from the same address, usually in alphabetical order. For example,

```
...
Mar 10 18:48:02 sku sshd[1496]: Failed password for invalid user alex from 1.2.3.4 port 1585
ssh2
...
Mar 14 23:25:52 sku sshd[1496]: Failed password for invalid user alice from 1.2.3.4 port
1585 ssh2
...
```

If you identify attacks, do the following:

- Double-check the SSH security settings in this topic.
- Report attackers to your ISP's email address for abuse reports (often `abuse@your_isp.com`).

## Configuring Transfer Server Authentication

### 1. Open the transfer server's `aspera.conf` file.

The file is found in the following location:

```
/opt/aspera/etc/aspera.conf
```

### 2. Locate the `<server>` section, and add the `<ssh_host_key_fingerprint>` or the `<ssh_host_key_path>` option.

**Note:** When the HTTP fallback feature is enabled and the client "falls back" to HTTP, a fingerprint specification in `aspera.conf` enforces server SSL certificate validation (HTTPS). Validation fails if the server has a self-signed certificate; a properly signed certificate is required.

#### • `<ssh_host_key_fingerprint>`

Use this option to specify the fingerprint itself:

```
<ssh_host_key_fingerprint>fingerprint</ssh_host_key_fingerprint>
```

To retrieve the SSH fingerprint, locate the transfers server's public or private key, and run the following command on a Linux, Mac, Ison, or other UNIX computer:

```
# cd /etc/ssh
# cat ssh_host_rsa_key.pub | cut -d' ' -f2 | base64 -d | sha1sum | cut -d' ' -f1
```

The following is an example SSH fingerprint:

```
43:51:43:a1:b5:fc:8b:b7:0a:3a:a9:b1:0f:66:73:a8
```

By convention, Aspera uses a hex string without the colons ( : ""). For example,

```
435143a1b5fc8bb70a3aa9b10f6673a8
```

The `aspera.conf` setting for this key would then be as follows:

```
<ssh_host_key_fingerprint>435143a1b5fc8bb70a3aa9b10f6673a8
</ssh_host_key_fingerprint>
```

#### • **<ssh\_host\_key\_path>**

Use the key-path option to specify the transfer server's public or private key file and its location. The fingerprint is extracted automatically.

```
<ssh_host_key_path>key_file</ssh_host_key_path>
```

On most Linux systems, the SSH keys are located in `/etc/ssh`. On OSX, the SSH keys are located in `/etc`. The following example uses a Linux server's public RSA key:

```
<ssh_host_key_path>/etc/ssh/ssh_host_rsa_key.pub</ssh_host_key_path>
```

3. After modifying `aspera.conf`, restart the node service by running **asperanoded**:

```
# /etc/init.d/asperanoded restart
```

## Troubleshooting

Solutions to common problems.

### Tracking connection status with proxy logs

The connection status for both forward and reverse proxy transfers is subject to regular logging in the system log file: `/var/log/messages` on Red Hat Linux and `/var/log/syslog` on Debian-based Linux. Root access is required for viewing the syslog file. The following output is an example of a proxy transfer log entry that is triggered at the start of a transfer:

```
Dec  5 17:32:11 test1 ascp_rproxy[26250]: LOG Received connection request from 10.0.31.133
Dec  5 17:32:11 test1 ascp_rproxy[26250]: LOG Established SSH connection with server
10.0.30.6:22
Dec  5 17:32:11 test1 ascp_rproxy[26250]: LOG Setup UDP forwarding between 10.0.31.133:60953
and 10.0.30.6:33001
```

In the previous output:

- `10.0.31.133:60953` – client IP address and UDP port.
- `10.0.30.6:22` – server IP and SSH port.
- `10.0.30.6:33001` – IP and UDP port.

The following is an example of a log entry when the connection is closed:

```
Dec  5 18:38:22 test1 ascp_rproxy[27238]: LOG Connection closed (EOF)
```

In the event of errors, individual error scenarios are logged separately.

To activate verbose debug logging, edit the value for `<log_level>` in `aspera.conf` to set or increase the log-level value.

## Error when trying to start node service

If you receive the following error when attempting to start the node service, check to see whether **iptables** is installed on your machine:

```
ERR Failed to initialize proxy service
```

If **iptables** is not installed, install it. See the documentation for your Linux distribution.

## Error when no matching rule for host

If you receive the following error in `as_nginx.log`, it means that Proxy did not find a rule in `aspera.conf` that matches the incoming SSH connection information:

```
ERR No matching rule found for host.
```

Check the IP address or hostname and SSH port that is used by the client matching a rule that is defined in `aspera.conf`.

**Note:** The proxy IP address must be used in the rule when its IP address is NAT.

## Using iptables to track forwarding rules

Proxy server administrators can also take advantage of the **iptables** tool to inspect the traffic forwarding rules that are in place. For example, the following example shows six **iptables** rules (in the `nat` and `filter` tables), corresponding to two different **ascp** connections (reverse proxy creates three rules for each connection). The comment field of each rule contains the UUID of the **ascp** session. The **iptables** command requires root privileges.

```
# iptables -t nat -L
Chain PREROUTING (policy ACCEPT)
target    prot opt source                destination
DNAT      udp  --  10.0.35.37             anywhere          udp dpt:33001 /* 8de6121e-
c6a4-4384-8b67-123f6bf453a2 */ to:10.0.143.102:33001
DNAT      udp  --  10.0.35.36             anywhere          udp dpt:33001 /* 813d334f-b47f-46ea-83ed-
e13779f9b9c8 */ to:10.0.143.102:33001

Chain POSTROUTING (policy ACCEPT)
target    prot opt source                destination
SNAT      udp  --  10.0.35.37             anywhere          udp dpt:33001 /* 8de6121e-
c6a4-4384-8b67-123f6bf453a2 */ to:10.0.143.110
SNAT      udp  --  10.0.35.36             anywhere          udp dpt:33001 /* 813d334f-b47f-46ea-83ed-
e13779f9b9c8 */ to:10.0.143.110

Chain OUTPUT (policy ACCEPT)
target    prot opt source                destination

# iptables -L
Chain INPUT (policy ACCEPT)
target    prot opt source                destination

Chain FORWARD (policy DROP)
target    prot opt source                destination
ACCEPT   udp  --  10.0.35.36             10.0.143.102     udp dpt:33001 /* 813d334f-b47f-46ea-83ed-
e13779f9b9c8 */
ACCEPT   udp  --  10.0.35.37             10.0.143.102     udp dpt:33001 /* 8de6121e-
c6a4-4384-8b67-123f6bf453a2 */
ACCEPT   udp  --  anywhere               anywhere          state ESTABLISHED

Chain OUTPUT (policy ACCEPT)
target    prot opt source                destination
```

## UDP port and firewall timeout errors

The following example is a common timeout error:

```
Session Stop (Error: Client unable to connect to server -- check UDP port and firewall.)
```

If you get this error, check the following settings:

1. Ensure that IP forwarding is enabled. IP forwarding must be enabled and is enabled automatically when Proxy is installed. To confirm, run the following command:

```
# cat /proc/sys/net/ipv4/ip_forward
```

If the command returns 1, IP forwarding is enabled. If it returns 0, it is not. IP forwarding can be enabled manually by setting the `net.ipv4.ip_forward` line in `/etc/sysctl.conf` as follows:

```
# Controls IP packet forwarding
net.ipv4.ip_forward=1
```

To activate changes to `/etc/sysctl.conf`, run the following:

```
# /sbin/sysctl -p /etc/sysctl.conf
```

2. If the error still occurs when IP forwarding is on, turn off source-port filtering. By default, source-port filtering is disabled (`false`). To reset, run the following command:

```
# asconfigurator -x "set_server_data;
iproxy_rules_rule_src_port_filtering,false"
```

This results in the following text in `aspera.conf`:

```
...
<iproxy>
  ...
  <rules>
    <rule>
      ...
      <src_port_filtering>false</src_port_filtering>
    </rule>
  </rules>
</iproxy>
...
```

For more information about source-port filtering, see [“Source-Port filtering” on page 22](#).

If the same timeout errors still occur when source-port filtering is disabled, this generally indicates that traffic is being blocked at a firewall. For information on configuring firewalls for forward proxy, see [“Forward proxy firewall configuration” on page 10](#). For information on configuring firewalls for reverse proxy, see [“Reverse Proxy firewall configuration” on page 23](#).

## Iptables rules left on the proxy server

On rare occasions, **iptables** rules are left on the proxy server for sessions that completed. To purge the rules, issue a stop and then a start (or restart) to the proxy service:

```
# /etc/init.d/asperaproxy stop
# /etc/init.d/asperaproxy start
```

Or:

```
# /etc/init.d/asperaproxy restart
```

## Clearing tracked NAT connection flows when the proxy service is stopped or restarted

If the **conntrack** tool is installed, the proxy service clears tracked NAT connection flows when the proxy service is stopped. This ensures that connections through the proxy are terminated when the proxy service is stopped or restarted. To enable this capability, your system must have the **conntrack** package for your distribution installed.

## Generating an SSL certificate that contains an IP address

As described in Installing Aspera Proxy, forward proxy transfers require that Proxy server SSL certificates include the hostname or IP address. For this reason, the default, self-signed certificate that is created by the Proxy installer must be replaced. Although use the hostname, certificates that include only an IP address can also be used. The following procedure describes how to generate a self-signed certificate that uses an IP address and replace the default certificate with it.

### Procedure

1. Back up the following files, but leave the originals in place:

```
/opt/aspera/proxy/etc/aspera_server_cert.pem
```

```
/opt/aspera/proxy/etc/openssl.cnf
```

2. Modify the Aspera openssl.cnf file.

Open /opt/aspera/proxy/etc/openssl.cnf and make the following changes:

- a. Uncomment the following line if it's commented. (Remove "#" from the start of the line).

```
req_extensions = v3_req # The extensions to add to a certificate request
```

- b. Locate the v3\_req section and add the following lines that are shown in bold, replacing *proxy\_server\_ip\_address* with your IP address:

```
[ v3_req ]
subjectAltName = @alt_names

# Extensions to add to a certificate request
basicConstraints = CA:FALSE
keyUsage = nonRepudiation, digitalSignature, keyEncipherment

[alt_names]
IP.1 = proxy_server_ip_address
```

- c. Save your changes.

3. Create the certificate signing request.

```
# cd /opt/aspera/proxy/bin
# ./openssl req -new -config ../etc/openssl.cnf -key ../etc/aspera_server_key.pem
-out ../etc/my.csr
```

In the output that appears, ignore the warning. Complete the requested fields as wanted. However, for this purpose, no values are required in any of the fields, including the proxy server hostname.

**Note:** Do not create a challenging password because waiting for a password to be entered might interfere in setups where the proxy server is started automatically.

```
WARNING: can't open config file: /opt/aspera/ssl/openssl.cnf
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:US
State or Province Name (full name) [Some-State]:California
Locality Name (eg, city) []:Emeryville
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Aspera
Organizational Unit Name (eg, section) []:Engineering
Common Name (e.g. server FQDN or YOUR name) []:proxy_server_hostname
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```



4. Generate the certificate.

```
# ./openssl x509 -req -days 365 -in ../etc/my.csr -signkey ../etc/aspera_server_key.pem  
-out ../etc/aspera_server_cert.pem -extensions v3_req -extfile ../etc/openssl.cnf
```

This creates a certificate valid for 365 days. Specify the number of days as wanted.

5. Check the values in the resulting certificate.

```
# ./openssl x509 -text -noout -in /opt/aspera/proxy/etc/aspera_server_cert.pem  
WARNING: can't open config file: /opt/aspera/ssl/openssl.cnf  
.....  
Subject: C=US, ST=California, L=Emeryville, O=Aspera, OU=Engineering,  
CN=proxy_server_hostname  
.....  
X509v3 extensions:  
X509v3 Subject Alternative Name:  
IP Address:proxy_server_ip_address  
.....
```

6. Append the key file to the certificate file.

```
# cd ../etc  
# cat aspera_server_key.pem >> aspera_server_cert.pem
```

7. Restart asperaproxy.

**Daemon**

**Commands**

**systemd**

```
# systemctl restart asperaproxy
```

**init**

```
# service asperaproxy restart
```

