IBM Aspera Proxy Admin Guide 1.4.4

Linux

Revision:ffd421 Generated:03/19/2020 17:56

Contents

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.

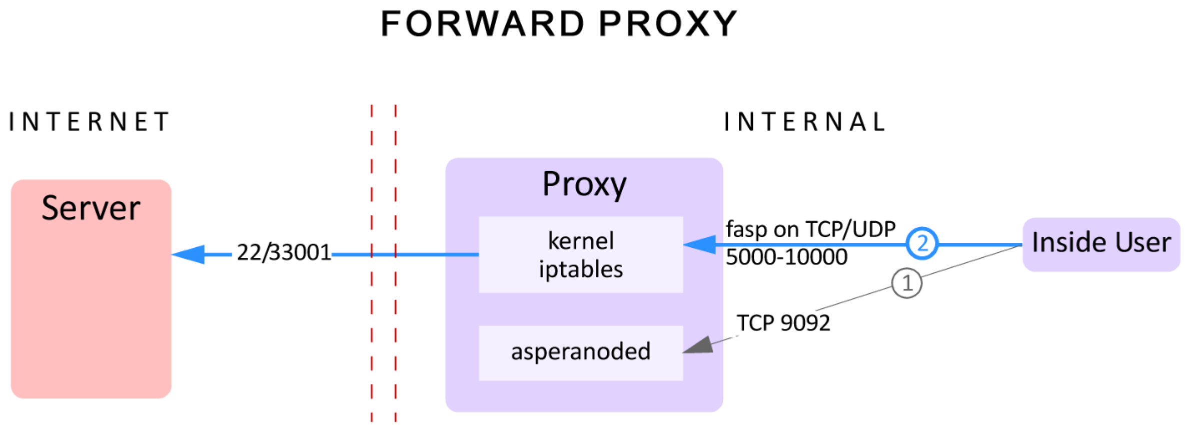
IBM Aspera Proxy also supports load balancing, high availability, and flexible security policies. It consolidates FASP transfers in and out of a corporate network and enables precise control over which users can initiate transfers with remote Aspera transfer servers. With Proxy support built into all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

IBM Aspera 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. IBM Aspera 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, IBM Aspera 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

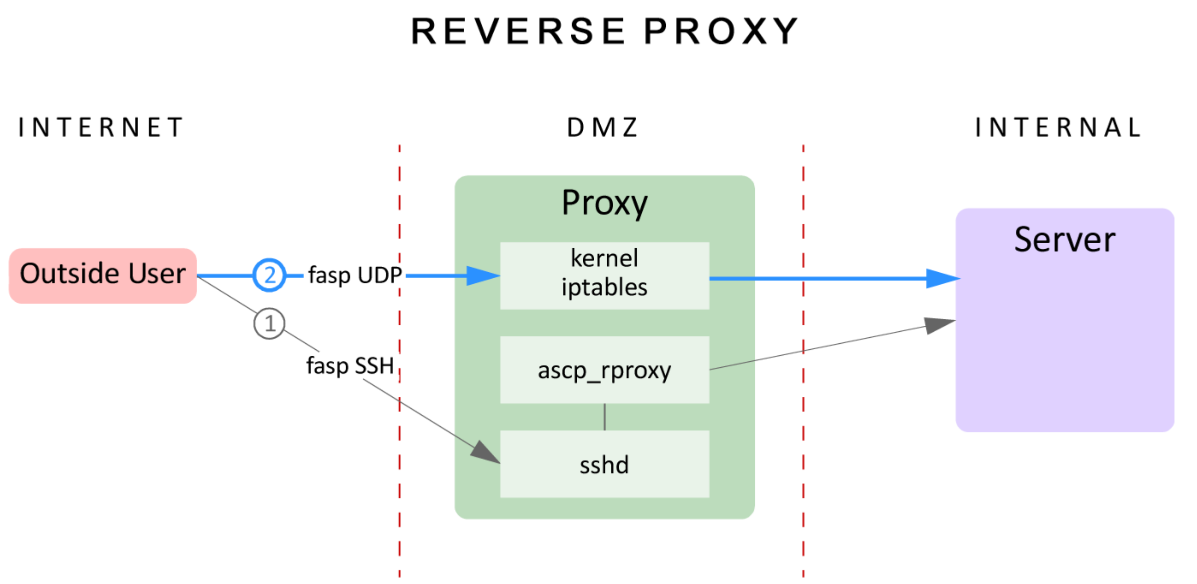
Reverse proxy provides a secure way for users outside company network firewalls to initiate requests for FASP transfers from servers inside the firewall. It addresses the following customer use case:

- 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 launched 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 invokes 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 invoked 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 pre-installed SSH key. It then bridges the two SSH connections, by forwarding incoming data from one connection to the other, in both directions. In order to forward FASP (UDP) traffic, the `ascp_rproxy` program proxy server sets up a dynamic network address translation (DNAT) rule using the Linux `iptables` kernel module. Since UDP traffic forwarding is done using the Linux `iptables` kernel module, high-speed packet forwarding can be achieved without any reduction in speed.



Installation

System Requirements

CAUTION: Do not install IBM Aspera Proxy on a machine where IBM Aspera Enterprise Server or IBM Aspera Connect Server is installed. If these products are already installed, be sure to remove them before installing Proxy.

Your IBM Aspera Proxy server requires the following:

- A Linux system (See the release notes for a list of supported Linux systems and versions.)
- iptables v1.3.0+ installed and not blocking TCP/UDP 33001
- Exclusive control of the following iptables/chains:

  nat/prerouting
  nat/postrouting
  filter/forwarding

Note: Before enabling and starting `iptables`, disable `firewalld`, if present on your system.

Installing IBM Aspera Proxy

To install IBM Aspera Proxy, log into your computer as root, and follow the steps below.

1. Download the Proxy installer from the Aspera download site.

   Use the credentials Aspera has provided to your organization to access: https://downloads.asperasoft.com/en/downloads/42

   If you need help determining your access credentials, contact your Aspera account manager.
2. Launch the installer by running the following commands with root privileges:

   RPM

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

   For upgrades, use `-U` instead of `-i`. `-U` is the same except that it removes all other versions of the package after the new one is installed.

   DEB

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

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

   This starts the IBM Aspera Proxy daemon and makes adjustments to the **iptables** system settings.
3. Install the license.

   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:

   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 the system uses `init`. (Although the above command for `init` searched for `init`, a return of `init` may be misleading if it's actually a symlink to `systemd`.)

`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 (and/or IP address), otherwise the transfers are refused. The default, self-signed certificate 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 Containing an IP Address on page 37.

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

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

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

   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 above `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 command below that corresponds to your type of Linux system, replacing *pkg_name* with the package name derived in the previous step.

RPM

```
$ rpm -e pkg_name
```

DEB

```
$ dpkg -r pkg_name
```

Forward Proxy

Configuring the Proxy Server for Forward Proxy

The configuration steps below 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 are described below.

The `asconfigurator` command is located in `/opt/aspera/proxy/bin`. The examples below 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 execute it.

1. Enable HTTP and/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's required to begin using forward proxy. However, you may need to change other `<proxy>` settings based on your unique network configuration.

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 table below shows the parameters and options 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.

asconfigurator parameter aspera.conf option	**Description**	**Default Value**
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.	false
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 `proxy_bind_ip_address`.	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

asconfigurator parameter aspera.conf option	**Description**	**Default Value**
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

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

```
<server>
  ...
  <proxy>
    <enabled>true</enabled>
    <authentication>false</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
```

4. Restart the proxy node service.

`systemd`	`# systemctl restart asperaproxy`
`init`	`# service asperaproxy restart`

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

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

For more information, see

5. Check log entries for startup.

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

Red Hat Linux:	`/var/log/messages`
Debian-based Linux:	`/var/log/syslog`

The only proxy entries that should 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.

6. 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

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 the client uses for SSH and FASP data transfer.

External Firewall

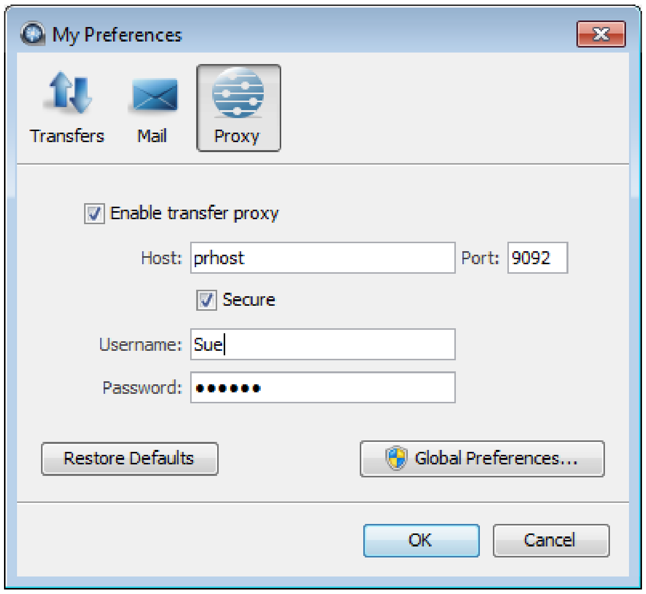
If outbound connections are restricted by the external firewall, the external firewall must allow **outbound TCP and UDP/33001** for SSH and FASP data transfer.

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


My Preferences

[✓] Enable transfer proxy

Host: prhost Port: 9092

[✓] Secure

Username: Sue

Password: ••••••

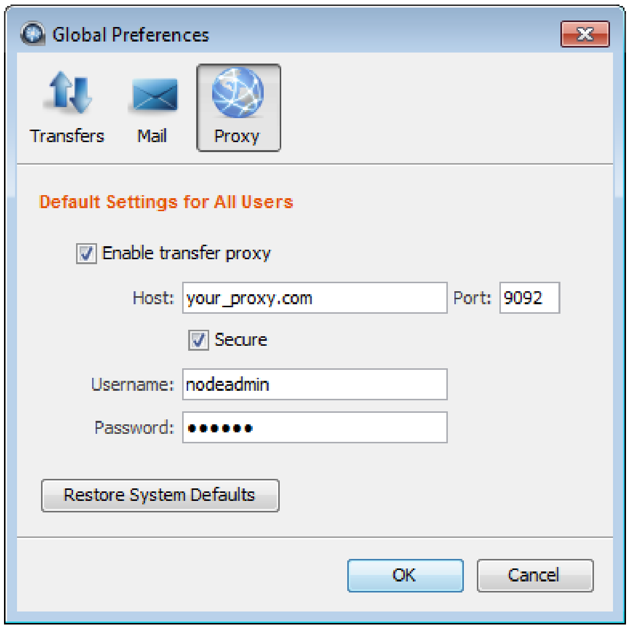
Case-by-Case Settings

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 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, fill out 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 you created when you were configuring the proxy server. (See Configuring the Proxy Server for Forward Proxy on page 7.)


Global Preferences

Transfers Mail Proxy

Default Settings for All Users

[✓] Enable transfer proxy

Host: your_proxy.com Port: 9092

[✓] Secure

Username: nodeadmin

Password: ••••••

Restore System Defaults

OK Cancel

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:
```

- 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@des
```

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 above, 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

Configuring the Proxy Server for Reverse Proxy

Reverse proxy is used to route incoming transfers from the proxy server to the internal server destination. In order 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 instructions below 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 destinatoin, 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 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 into 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. Generate an SSH key pair for each user on the proxy server:

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

   By default, `ssh-keygen` generates and copies the private key (usually `id_rsa`) and public key (usually `id_rsa.pub`) 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 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.
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 the group was created.

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

   The new group should 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 execute the `/sbin/iptables-restore` command as `root` 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 above 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 using the `visudo` command. In this case, make sure that the above 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 configuration steps below 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 are described below.

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

For more information about the `aspera.conf` settings and corresponding `asconfigurator` settings, see Reverse Proxy Configuration Options on page 16.

1. Enable reverse proxy:

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

   This 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 only setting 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 keyed by the IP address or host name used for connecting to the Proxy server. For example, using multiple rules allows you to 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_rsa`, 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_rsa</keyfile>
               </rule>
           </rules>
       </rproxy>
   </server>
   ```

To set an additional rule such that transfers destined for 7.7.7.8 should 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_rsa</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 above 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

Single rule: If you are configuring a single rule, you can use `asconfigurator` to set most options, 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 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 example below.

Reverse Proxy Configuration Options

aspera.conf option **asconfigurator parameter**	**Description**	**Default Value**
`<enabled>` rproxy_enabled	Turn reverse proxy on/off (true/false).	false
`<log_level>` rproxy_log_level	Set the logging level. 0 - normal 1 - normal plus debug 2 - level 1 plus additional debugging	0
`<rule>`	Rule with no conditional attributes.	N/A
`<rule host_domain="`*`hostname:port`*`">` (no asconfigurator option)	Set a rule for requests directed to the specified host name (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 `ascp` client version 3.1 or later.	(none)
`<rule host_ip="`*`ipaddr`*`">` (no asconfigurator option)	Rule specifying the IP address of the proxy server.	(none)
`<rule host_domain="`*`hostname:port`*`" host_ip="`*`ipaddr`*`">` (no asconfigurator option)	Combined version of the above.	(none)
`<host>` rproxy_rules_rule_host	IP address and optional SSH port of the internal server (destination), with the syntax *ip_address*[:*port*]. 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 *ip_address*:*port*. See Load Balancing 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. **Important**: For Proxy servers with more than one NIC, `<bind_source_address>` should be specified to	blank (null)

aspera.conf option asconfigurator parameter	**Description**	**Default Value**
	identify the interface for the connection with Enterprise Server.	
`<balancing>` (no asconfigurator option)	Enables load balancing. The internal servers specified in the hosts sections are used for load balancing. Round-robin selection is currently the only supported method. For details, see Load Balancing.	load balancing not enabled
`<squash_user>` rproxy_rules_rule_squash_user	Squash account name 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 Source-Port Filtering.	`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. **Note:** Must be set to false when internal servers are running Windows.	`true`

Sample aspera.conf for Reverse Proxy

The sample below 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_rsa</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, only transfers sent by users with valid SSH key authentication are allowed through the reverse proxy. Since three hosts have been 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_rsa</keyfile>
      <udp_port_reuse>true</udp_port_reuse>
      <balancing>round_robin</balancing>
    </rule>
```

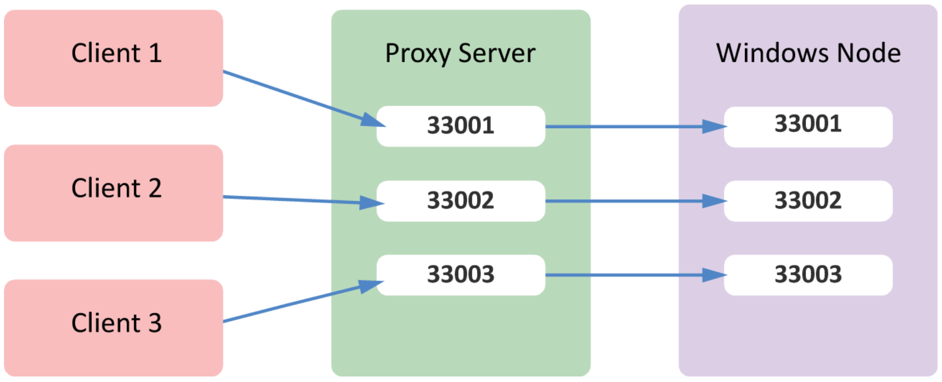
3. **UDP port reuse and squash user account rule.** Incoming transfers to the proxy server with the IP address of 10.20.101.151 that are destined to an internal Windows server are received and forwarded on incremental UDP ports beginning with 5555. Transfers are forwarded as the squash user once the clients have been authenticated on the proxy server. The incremental UDP ports beginning 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_rsa</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 will bind 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 IBM Aspera Proxy, this configuration could be achieved by setting `<proxy_port>` to 0. However, setting it to zero is no longer supported and now results in an error.

For information about how to set up load balancing when `<udp_port_reuse>` = `false`, see UDP Port Reuse and Load Balancing on Windows on page 20.

Note: When setting `<udp_port_reuse>` to `false`, and thereby enabling the use of incremented UDP port numbers, make sure the UDP ports 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 will not forward transfers to it. After two 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 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_rsa</keyfile>
</rule>
```

UDP Port Requirements for Load Balancing

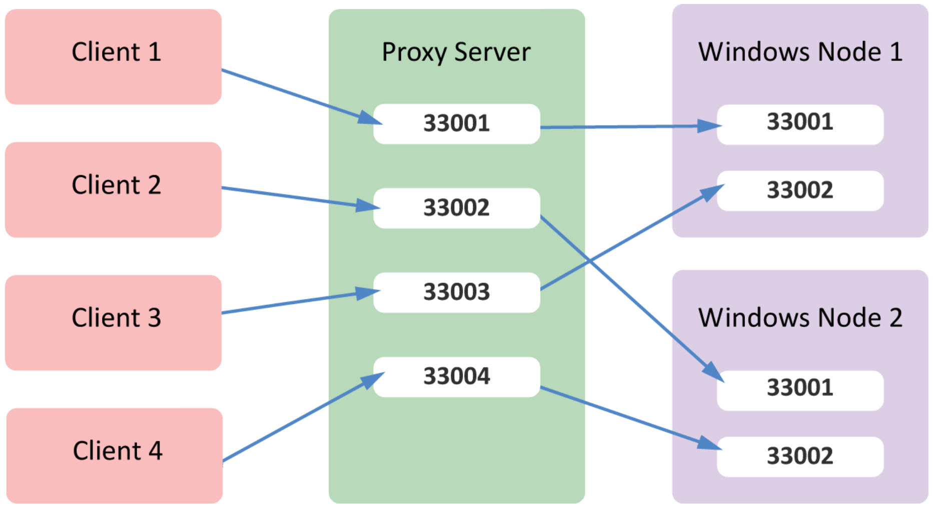
Load balancing requires exposing multiple UDP ports on reverse proxy. Ports are opened based on the value specified by `<proxy_port>` (33001 if not specified). The formula for port exposure is *proxy port base + the number of the host* in the load-balance configuration. For example, if the load is to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

The reverse proxy server in the above example determines port numbers by incrementing from proxy port 33001, the default port number. However, users may 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 above 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 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 setup below, concurrent transfers from four clients are being load-balanced between two Windows destinations.



Source-Port Filtering

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` may 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 specified by `<proxy_port>` are allowed on the external firewall.

One indication that source-port filtering may 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 35.

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

Reverse Proxy Firewall Configuration

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 originating from the proxy server. By default, the ports are TCP/22 and UDP/33001.

Note: If `proxy_port` has been set to a value other than 33001, or has been 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 19. For more information on port configuration for UDP port reuse, see UDP Port Reuse on page 19.

Configuring Internal Servers

The internal server must be running an Aspera Transfer server (a.k.a. Enterprise Server or Connect Server).

1. Log into 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.

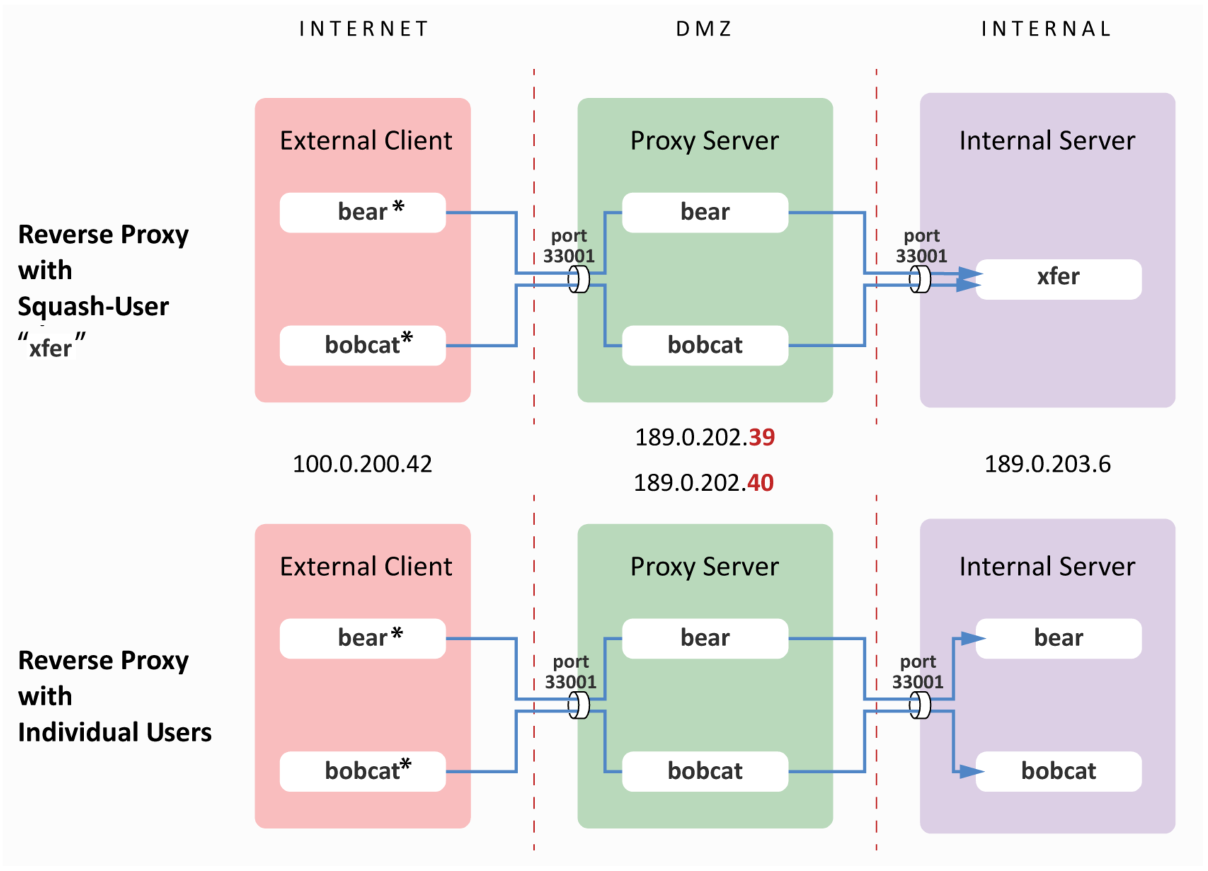
Transferring Files with Reverse Proxy

Once the configuration tasks have been completed for the proxy server, internal server, and external clients, file transfers from external users are completely transparent. To make transfers to the internal server, users need only specify the following:

- 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 related to a transfer session

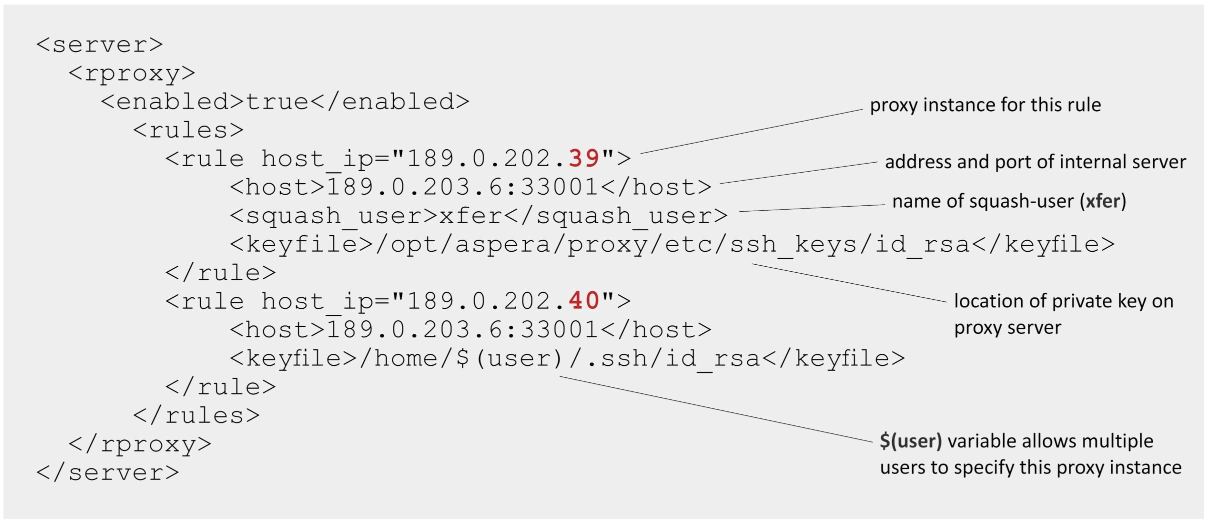
From the Command Line

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



* Although the user names on the external client are `bear` and `bobcat` in the above example, they do not need to correspond to user names on the Proxy server and internal server.

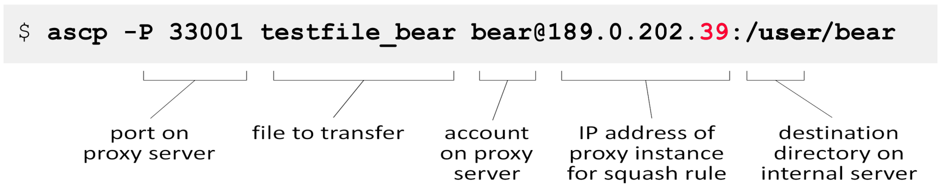
The reverse-proxy rules for each configuration are defined on the proxy server in `aspera.conf`:



```
<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_rsa</keyfile>
        </rule>
        <rule host_ip="189.0.202.40">
            <host>189.0.203.6:33001</host>
            <keyfile>/home/$(user)/.ssh/id_rsa</keyfile>
        </rule>
      </rules>
  </rproxy>
</server>
```

proxy instance for this rule
address and port of internal server
name of squash-user (xfer)
location of private key on proxy server
$(user) variable allows multiple users to specify this proxy instance

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 governed by the squash rule:



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

port on proxy server
file to transfer
account on proxy server
IP address of proxy instance for squash rule
destination directory on internal server

Since the rule for proxy instance 189.0.202.39 specifies a squash user (xfer), the file belonging to bear, bobcat, or anyone using 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:

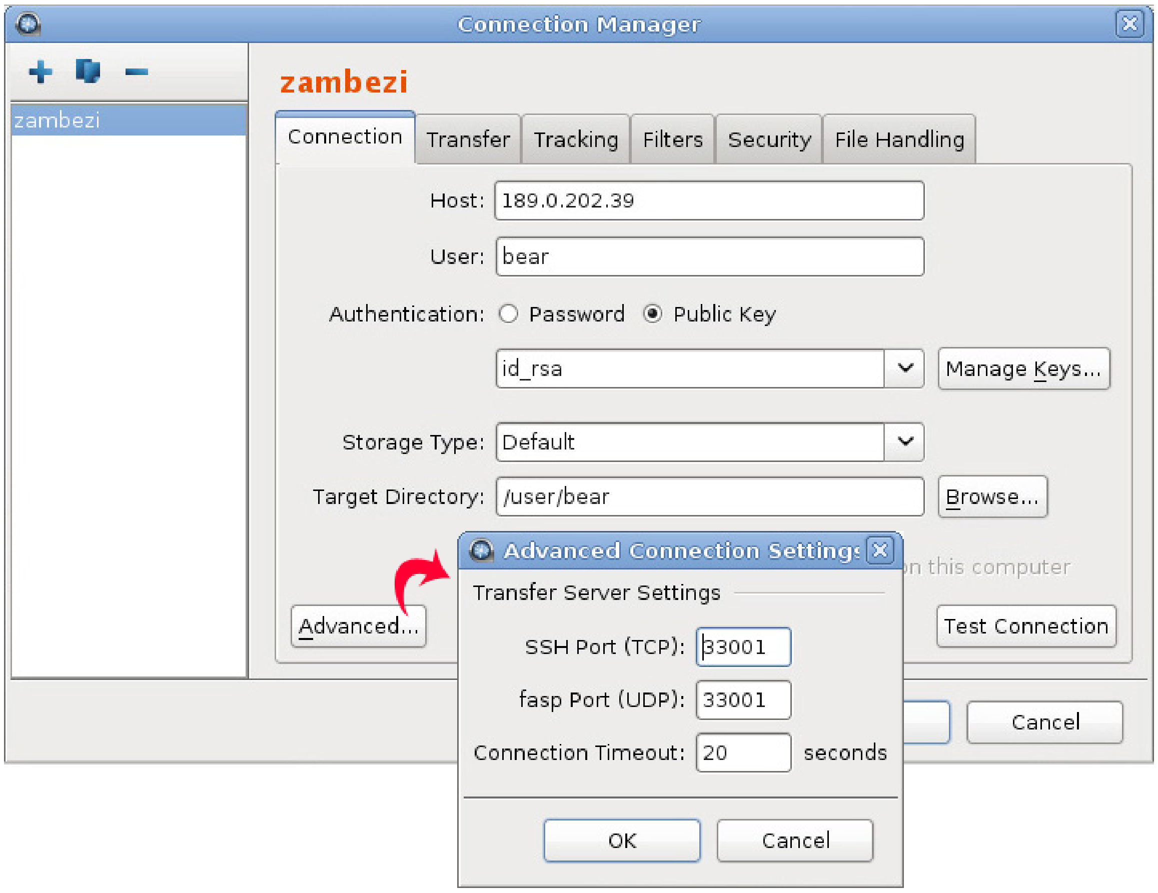
```
$ ascp -P 33001 testfile_bobcat bobcat@189.0.202.40:/user/bobcat
```

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

From the Enterprise Server GUI

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

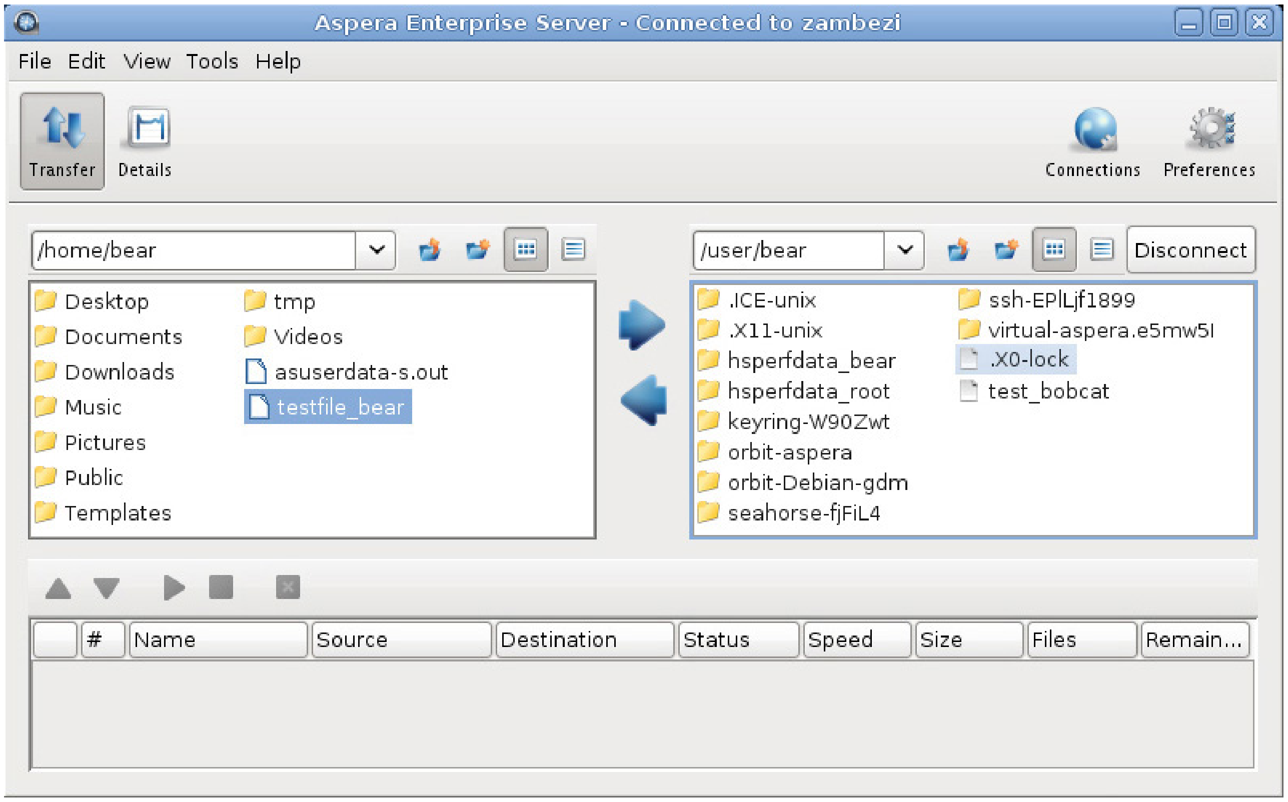
For example, user bear could also have made the above transfer with the Enterprise Server GUI. In the following display, bear has set up a connection called “zambezi” using the same parameters as above. The IP address of the proxy instance 189.0.202.39 (squash-user rule) is specified as the host. The filename 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 button.



Host: 189.0.202.39
User: bear
Authentication: Password [✓ Public Key]
id_rsa
Storage Type: Default
Target Directory: /user/bear

Transfer Server Settings
SSH Port (TCP): 33001
fasp Port (UDP): 33001
Connection Timeout: 20 seconds

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-hand panel in the display below, and ready for `bear` to make the transfer.



/home/bear

Desktop
Documents
Downloads
Music
Pictures
Public
Templates
tmp
Videos
asuserdata-s.out
testfile_bear

/user/bear

.ICE-unix
.X11-unix
hsperfdata_bear
hsperfdata_root
keyring-W90Zwt
orbit-aspera
orbit-Debian-gdm
seahorse-fjFiL4
ssh-EPlLjf1899
virtual-aspera.e5mw5l
.X0-lock
test_bobcat

Using IBM Aspera Sync and IBM Aspera Drive with Reverse Proxy

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

- Allows large Sync transfers to make use of Proxy's round-robin load-balancing feature, thereby spreading the load, increasing efficiency, and providing a form of high availability. For details, see Load Balancing.
- Allows Drive 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 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 database must be stored in a remote NFS mount.

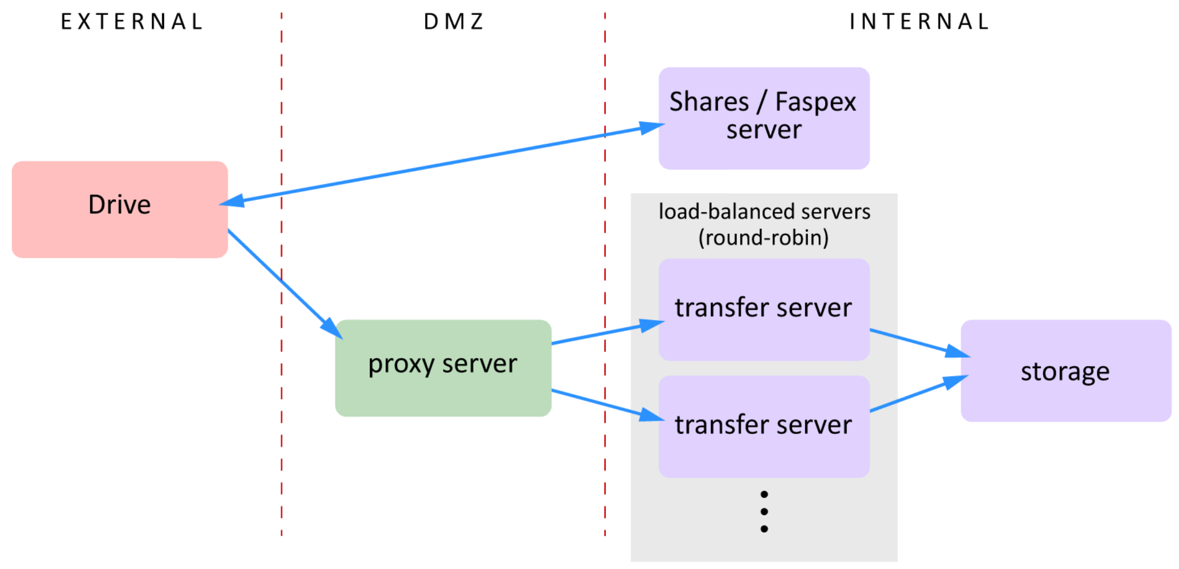
To configure `async` to store the database in a remote storage, run the following command on each of the round-robin servers:

```
# asconfigurator -x "set_node_data;async_db_dir,remote_dir"
```

Note that `async` proxy sessions are always logged to syslog.

Configuring Drive for Reverse Proxy with Round-Robin Load Balancing

External Aspera Drive users can view and transfer files and directories on internal Aspera servers (usually IBM Aspera Shares or IBM Aspera Faspex, although IBM Aspera Enterprise Server, Connect Server, or Point-to-Point Client can also be used) through Reverse Proxy with additional configuration. In the following diagram, the external Drive 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_rsa.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 either RSA (`rsa`) or ED25519 (`ed25519`). At the prompt for the key-pair's filename, press ENTER to use the default 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 -t key_type
```

   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 (Connect Server, Enterprise Server, or Point-to-Point) that are used by Faspex and Shares and are running behind a reverse proxy, and the proxy server.

Configuring the Transfer Nodes

1. Set the transfer user's default shell to `aspshell`.

   The transfer user's username is typically **faspex** for Faspex and **asp1** or **shares** for Shares. The method to set the default shell to `aspshell` 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 Mac OS X:** Change the transfer user's default shell by running the following commands as root. For example, using user `faspex` as the user):

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

     If a warning message appears saying `/bin/aspshell` is not listed in `/etc/shells`, it can be safely ignored.
   - **Mac OS X:**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/aspshell`.

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

2. 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 righthand 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 Mac OS X, open a Terminal window as root. For Windows, launch Command Prompt as an administrator (click **Start**, 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"
   ```

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

   Log in as the transfer user to ensure that the user will own 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:

Windows	`C:\Users\faspex\.ssh\authorized_keys`
Linux	`/home/faspex/.ssh/authorized_keys`
Mac OS X	`/Users/faspex/.ssh/authorized_keys`

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

Windows	`C:\Program Files[ (x86)]\Aspera\Enterprise Server\var \aspera_tokenauth_id_rsa.pub`
Linux	`/opt/aspera/var/aspera_tokenauth_id_rsa.pub`
Mac OS X	`/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 Mac OS X, 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*.

Configuring the Proxy Server

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

   For instructions on creating the transfer user, see Configuring the Server for Reverse Proxy. Set the default shell to `aspshell`, as described above in *Configuring the Transfer Nodes*.
2. Confirm 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_rsa
   ```
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 above section.

   Note: You may 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

IBM 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.

1. Create the Aspera-specific Nginx configuration file.

   The Proxy installer includes an Aspera-specific example Nginx configuration file `/opt/aspera/nginx/nginx.conf.example`. Copy this file to create the file that Nginx uses, `/opt/aspera/nginx/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 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 have been 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 using 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 IBM Aspera 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 should 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 using a `require` statement.

Running Your Scripts

To launch 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 IBM Aspera 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

Securing your SSH Server

Keeping your data secure is critically important. Aspera strongly recommends 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 into 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 well 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 using the default port 33001. However, if the connection fails, the application attempts the connection using port 22.

The following explains 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
   ```

```
...
```

3. Disable non-admin SSH tunneling

   Note: The instructions below 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; Aspera recommends updating 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 non-admin 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 already exist):

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

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

   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 instructions below 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, Aspera recommends 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 utilize the `su` command if root privileges are needed.

6. Restart the SSH server to apply new settings

   When you have 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	Instructions
RedHat, zLinux (restart)	`$ sudo service sshd restart`
RedHat (reload)	`$ sudo service sshd reload`
Debian (restart)	`$ sudo /etc/init.d/ssh restart`
Debian (reload)	`$ sudo /etc/init.d/ssh reload`

7. Review your logs periodically for attacks.

Aspera recommends reviewing 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 may vary.

Look for invalid users in the log, especially a series of login attempts with common user names 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`).

Restricting User Access

Restricting user access is a critical component of securing your server. By default, all user accounts are allowed to browse and read all files on the server. To limit a user's access to a portion of the system, set the account's shell to the Aspera secured shell (`aspshell`) and create a document root (docroot) for that user. The `aspshell` permits only the following operations:

- Run Aspera uploads and downloads to or from this computer.
- Establish connections in the application and browse, create, delete, rename, or list contents.

1. Set the user's login shell to `aspshell`.

Open the following file with a text editor:

```
/etc/passwd
```

Add or replace the user's shell with `/bin/aspshell`. For example, to apply `aspshell` to the user `aspera_user_1`, use the following settings in the `passwd` file:

```
...
aspera_user_1:x:501:501:...:/home/aspera_user_1:/bin/aspshell
...
```

2. **Set a user's docroot.**

   When a user's docroot is empty (i.e. blank), that user has full access to your server's directories and files. To restrict the user, you must set a non-empty docroot.

   Edit the `aspera.conf` file (`/opt/aspera/etc/aspera.conf`). The following template displays access options:

```
<file_system>
   <access>
   <paths>
   <path>
   <absolute>/sandbox/aspera_user_1</absolute>      <!-- Absolute Path -->
   <read_allowed>true</read_allowed>          <!-- Read Allowed -->
   <write_allowed>true</write_allowed>        <!-- Write Allowed -->
   <dir_allowed>true</dir_allowed>            <!-- Browse Allowed -->
   </path>
   </paths>
   </access>
...
</file_system>
```

3. **Restrict read, write, and browse privileges.**

   Once you have set the user's shell and docroot, you can further restrict access by disabling read, write, and/or browse using `<path>` settings in `aspera.conf`, as shown in the example above.

Field	**Description**	**Values**
Absolute Path	The area of the file system (path) that is accessible to the Aspera user. The default empty value gives a user access to the entire file system.	Path or blank
Read Allowed	Setting this to `true` allows users to transfer from the designated area of the file system as specified by the Absolute Path value.	`true` `false`
Write Allowed	Setting this to `true` allows users to transfer to the designated area of the file system as specified by the Absolute Path value.	`true` `false`
Browse Allowed	Setting this to `true` allows users to browse the directory.	`true` `false`

4. **Run the `asp-check` tool to check for potential user-security issues**

   The `asp-check` tool performs the following secure checks:

   - Searches for full-access users and reports how many exist on the system. Note that the existence of full-access users does not necessarily indicate that your system is vulnerable; however, it is being brought to the attention of the system administrator to ensure that the existence of full-access users is intentional.
   - Searches for restricted users and potential misconfigurations, including incorrect login shell (one that is not restricted via `aspshell`); SSH tunnel access (which can be used to work around the restricted shell); and docroot settings that allow the users to access the home directory.

   Note: A docroot setting that allows access to the home directory does not necessarily indicate that your system is vulnerable; however, a user with this docroot can download or upload keys in `.ssh`, as well as upload `.login` scripts. These capabilities may be used to circumvent the intended, restricted-nature of the user. Aspera highly recommends setting the docroot under the user's home folder (such as `/home/jane/data`) or in an alternate location (for example, `/data`).

To run the `asp-check` tool, run the following on the command line:

```
$ sudo /opt/aspera/bin/asp-check.sh
```

Search results are displayed as in the following example. If potential issues are identified, review your users' settings before proceeding.

```
Users with full access: 22 (not considered insecure)
Restricted users: 0
Insecure users: 0
 - no restricted shell (aspshell): 0
 - docroot above home directory: 0
 - ssh tunneling enabled: 0
```

Troubleshooting

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 is an example of a proxy transfer log entry 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 above:

- 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 if `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_rproxy.log`, it means 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 used by the client matching a rule defined in `aspera.conf`.

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

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 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 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:

1. Ensure that IP forwarding is enabled. IP forwarding must be enabled and is enabled automatically when IBM Aspera 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;
   rproxy_rules_rule_src_port_filtering,false"
   ```

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

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

   For more information about source-port filtering, see Source-Port Filtering on page 21.

   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 21.

Iptables Rules Left on the Proxy Server

On rare occasions, `iptables` rules are left on the proxy server for sessions that have 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 Containing an IP Address

As described in Installing IBM Aspera Proxy on page 5, forward proxy transfers require that Proxy server SSL certificates include the hostname or IP address. For this reason, the default, self-signed certificate created by the Proxy installer must be replaced. Although Aspera recommends using 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.

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:

   (1) 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
   ```

   (2) Locate the `v3_req` section and add the lines shown in bold below, 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
   ```

   (3) 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. Fill in the requested fields as desired. However, for this purpose, no values are required in any of the fields, including the proxy server hostname.

   Note: Aspera recommends not creating a challenge password, because waiting for a password to be entered could interfere in setups where the proxy server is booted 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 desired.
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.

`systemd`	`# systemctl restart asperaproxy`
`init`	`# service asperaproxy restart`

Running Proxy as a Non-Root User

The steps below explain how to set up a Proxy service account that is not root. This is the account under which Proxy services will run. All setup steps require root privileges. The example user in these steps is `aspsvc`.

1. Create the user `aspsvc` and the group `aspadmins`:

   ```
   # groupadd -r aspadmins
   # useradd -rm -g aspadmins -c "Aspera Service User" aspsvc
   ```
2. Update the configuration file `/etc/sudoers` (or a file you create under the directory `/etc/sudoers.d/` for Aspera-related changes). You will add an entry for `aspsvc`, in addition to the changes for reverse proxy transfer users (in this example, for `xfer`, `faspex`, and `shares` in the user alias `ASPXFER`):

   ```
   User_Alias ASPXFER = xfer, faspex, shares

   Defaults:ASPXFER !requiretty
   ```

```
Defaults:ASPXFER secure_path = /sbin:/bin:/usr/sbin:/usr/bin
ASPXFER ALL = NOPASSWD: /sbin/iptables-restore
```

Add the following lines for the user `aspsvc`:

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

3. Stop `asperaproxy` and `asperanginx` services (if running) and update permissions for the files in the directories `/opt/aspera/proxy/var` and `/opt/aspera/nginx`:

```
# systemctl stop asperanginx.service
# systemctl stop asperaproxy.service
# /opt/aspera/proxy/bin/asnodeadmin --db-shutdown

# chown -R aspsvc:aspadmins /opt/aspera/proxy/var/
# chown -R aspsvc:aspadmins /opt/aspera/nginx/
```

4. Modify your system startup scripts as follows, depending on whether your system uses `systemd` or SysV scripts:

- **Systemd startup scripts** (RHEL 7.x and CentOS 7.x, for example):

  Update `systemd` startup scripts for `asperaproxy` and `asperanginx` services, and add the line `User=aspsvc`.

  For `asperaproxy`:

  ```
  # systemctl show asperaproxy | grep Path=
  FragmentPath=/usr/lib/systemd/system/asperaproxy.service
  ```

  Open `/usr/lib/systemd/system/asperaproxy.service` with a text editor; locate the line `[Service]`; and just below it, add the line `User=aspsvc`:

  ```
  [Service]
  User=aspsvc
  ```

  For `asperanginx`:

  ```
  # systemctl show asperanginx | grep Path=
  FragmentPath=/usr/lib/systemd/system/asperanginx.service
  ```

  Open `/usr/lib/systemd/system/asperanginx.service` with a text editor; locate the line `[Service]`; and just below it, add the line `User=aspsvc`:

  ```
  [Service]
  User=aspsvc
  ```

  ```
  # systemctl daemon-reload
  ```

- **SysV startup scripts** (RHEL 6.x and CentOS 6.x, for example):

  Follow the steps below for the `/etc/init.d/asperaproxy` and `/etc/init.d/asperanginx` startup scripts:

  a. Open the startup file in a text editor, and locate the following line:

     ```
     DAEMON=$ASPERAROOT/sbin/$NAME
     ```

b. Just below it, add the following line:

```
DAEMON_USER=aspsvc
```

c. For the `/etc/init.d/asperaproxy` script:

Locate the section and change as indicated:

Locate this section

```
redhat_start() {
    $DAEMON -t
    RETVAL=$?

    if [ $RETVAL -eq 0 ]; then
        daemon $DAEMON
        RETVAL=$?
    fi
}
```

Change it to

```
redhat_start() {
#   $DAEMON -t
    daemon --user=$DAEMON_USER
 $DAEMON
    RETVAL=$?

#   if [ $RETVAL -eq 0 ]; then
#       daemon $DAEMON
#       RETVAL=$?
#   fi
}
```

d. For the `/etc/init.d/asperanginx` script:

Locate the following section:

```
# RedHat, Mandrake and Yellowdog
elif test -f /etc/redhat-release -o -f /etc/yellowdog-release -o -f /
etc/mandrake-release; then
    # Source function library
    . /etc/rc.d/init.d/functions
    # Source networking configuration.
    . /etc/sysconfig/network
    RETVAL=0
    redhat_start() {
        daemon $DAEMON
        RETVAL=$?
    }
    case "$1" in
    start)
        echo -n $"$STARTING $DAEMON: "
        make_dirs
        daemon $DAEMON -c $NGINX_CONF_FILE
        RETVAL=$?
        echo
        [ $RETVAL -eq 0 ] && touch $lockfile
        ;;
```

In the above, look for this line:

```
daemon $DAEMON -c $NGINX_CONF_FILE
```

Change the line to the following:

```
daemon --user=$DAEMON_USER $DAEMON -c $NGINX_CONF_FILE
```

5. Start services normally:

- Using `systemctl`:

```
# systemctl start asperaproxy.service
# systemctl start asperanginx.service
```

- Using SysV scripts:

```
# service asperaproxy start
# service asperanginx start
```

6. Check that services are now running under the user `aspsvc`:

```
# ps xau | grep aspera | grep -v grep
aspsvc  120032  0.1  0.7 1294992 7740 ?   Ssl  18:46   0:01 /opt/aspera/
proxy/sbin/asperanoded
aspsvc  120038  0.1  0.2 133648  2172 ?   Ssl  18:46   0:01 /opt/aspera/
proxy/sbin/asperaredisd 127.0.0.1:31415
aspsvc  120163  0.0  0.0  22164   956 ?   Ss   18:46   0:00 nginx: master
 process /opt/aspera/nginx/nginx
```