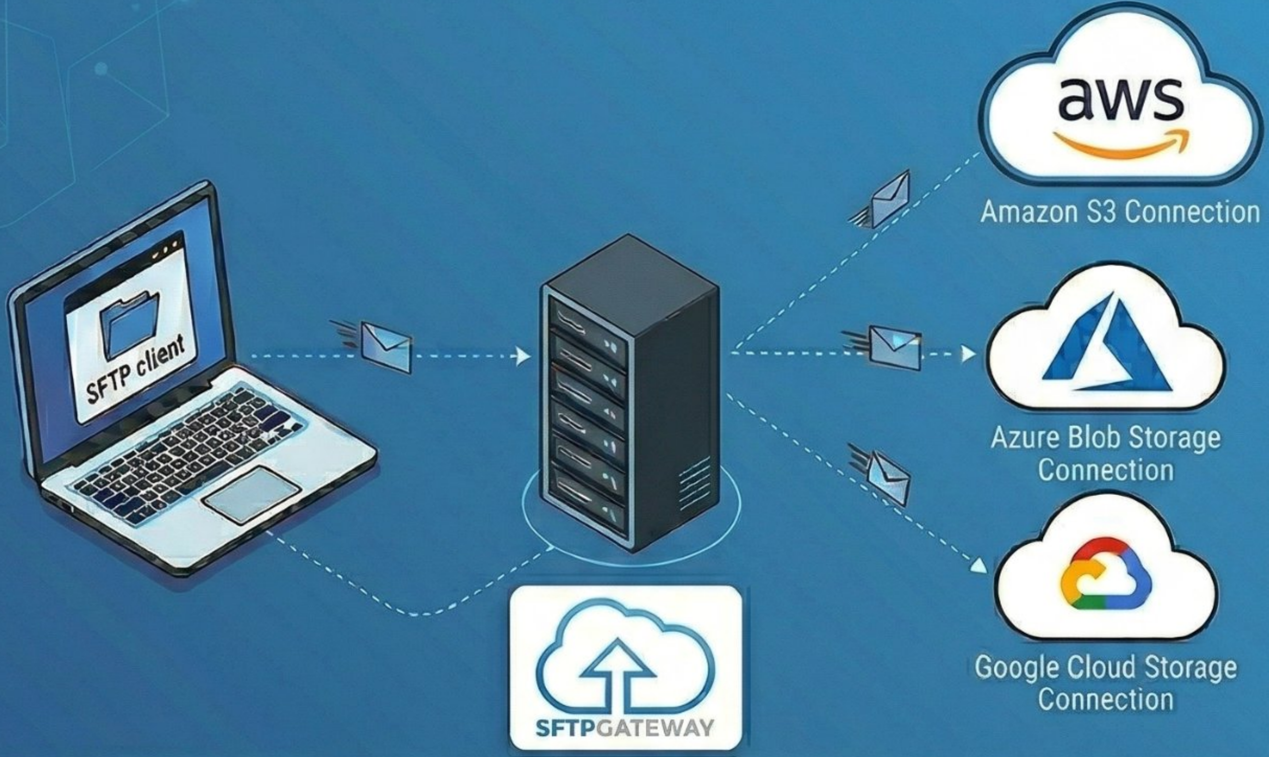
SFTP client
aws
Amazon S3 Connection
Azure Blob Storage Connection
Google Cloud Storage Connection
SFTPGATEWAY

SFTP GATEWAY 3.0

IMPLEMENTATION & ADMINISTRATION GUIDE

Secure Cloud Data Transfer Management and Multi-Cloud Connectivity

- User Provisioning & SSH Key Management
- Virtual File System Configuration
- Multi-Cloud Connectivity (S3, Azure, GCS)
- High Availability (HA) & Database Mgmt (PostgreSQL)
- API Reference & Security Best Practices

Document Version: 3.0.x | Technical Support: help.thorntech.com | developed by ThornTech

SFTP Gateway Administrator Guide

Table of Contents

This guide is intended for administrators who deploy and manage SFTP Gateway. It covers the Admin Web Interface, API, and application properties for managing users, cloud connections, folders, and other settings. It also includes instructions you can share with your SFTP users.

Introduction

SFTP Gateway is a managed SFTP server that enables secure file transfers directly to cloud storage. When users upload files via SFTP, those files are written immediately to your configured cloud storage backend (Amazon S3, Azure Blob Storage, or Google Cloud Storage)—there is no intermediate staging area or sync process.

Key Benefits

- **Direct cloud writes**: Files are written directly to cloud storage as users upload them
- **Familiar protocol**: Users connect with standard SFTP clients they already know
- **Centralized management**: Manage users, folders, and cloud connections from a single Admin Web Interface

- **API automation**: Programmatically manage users and configurations via REST API
- **Enterprise security**: SSH key authentication, encryption in transit, and configurable security levels

Architecture Overview

```
SFTP Client  →  SFTP Gateway  →  Cloud Storage (S3/Azure Blob/GCS)
                      ↑
              Admin Web Interface
```

SFTP Gateway acts as a bridge between traditional SFTP clients and modern cloud storage. Administrators configure cloud connections and folders, then create user accounts with access to specific folder paths.

First Launch Experience

The first time you launch SFTP Gateway, you'll go through a one-time configuration to create an admin user and configure a default cloud storage bucket. These steps are performed in the web interface—no SSH access to the server is required.

Accessing the Web Interface

Open a web browser and navigate to the public IP address of your SFTP Gateway instance:

```
https://<your-sftp-gateway-ip>
```

If you deployed via CloudFormation, find the IP address in the **Outputs** tab under the **Hostname** key.

You'll see a self-signed certificate warning. Click **Advanced** and then **Proceed** to continue. (You can replace the certificate with your own later.)

Creating an Admin Account

On the welcome page, create your admin account:

1. **Username**: Enter a username (e.g., `admin`)
2. **Password**: Enter a password that meets the strength requirements (hover over the icon to see requirements)
3. Click **Create Account**

Configuring Default Storage

After creating your admin account, you'll be prompted to configure your default cloud storage connection. This is required before you can create folders and users.

See the Cloud Connections section for detailed configuration instructions.

Admin Web Interface

The Admin Web Interface provides a convenient way to manage users, folders, and system settings.

Navigation

After logging in, use the top navigation menu to access:

Section	Purpose
Users	Create and manage SFTP user accounts
Folders	Configure folder structure and cloud storage mappings
Settings	Manage admin users, cloud connections, and backup/recovery

Users Page

The Users page displays a paginated list of all SFTP users. You can:

- **Search** for users by username
- **Add User** to create a new SFTP user
- **Edit or Delete** users by hovering over a row or clicking on it
- Adjust **rows per page** for the list display

Folders Page

The Folders page displays the virtual file system that SFTP users navigate when they connect. This is not a local Linux file system—it's a virtual structure stored in the database. Each folder can map to a Cloud Connection, allowing different folders to point to different buckets or even different cloud providers.

- **Search** for folders
- **Add** new folders (optionally mapping to a Cloud Connection)
- **Edit** the current folder (shown in the breadcrumb)
- **Navigate** by clicking on folder rows

Settings Page

The Settings page allows you to:

- Manage **Admin Users** (add, edit, delete, enable/disable)

- Configure **Cloud Connections** (S3, Azure Blob, etc.)
- **Backup & Recovery** (export/import configuration)

Managing Users

Users are the SFTP accounts that connect to transfer files. Each user has authentication credentials and a home directory that determines which folders they can access.

Creating a User

1. Navigate to **Users** in the top menu
2. Click **Add User** in the upper right corner
3. Configure the user settings (see below)
4. Click **Save**

User Configuration Options

Username (Required)

Must be unique among all users.

Notes (Optional)

Add details or context about the user for administrative reference.

Home Directory

Specify where the user lands when they connect:

- **Create Username Home Directory** (Default): Creates a folder with the user's name inside the `users` folder (e.g., `/users/alice`). This is the recommended approach for most deployments because it automatically isolates each user's data—users are placed in sibling directories and cannot traverse into each other's folders.

- **Select Home Directory**: Opens a folder selector to choose an existing folder. Use this when you need users to share a common directory or access a specific folder structure.

    - Use the folder navigator to browse available folders
    - Set **permissions** (read, write) for the selected folder

Data Isolation: By using the default behavior, you get automatic separation of user data. Each user's home directory is a sibling under `/users/`, so `alice` cannot see or access files belonging to `bob`. Users are chrooted to their home directory and cannot navigate outside of it.

SFTP SSH Keys (Recommended)

SSH key authentication is the preferred method because no secrets are transmitted over the wire during authentication. The user proves their identity using a cryptographic challenge-response—the private key never leaves their machine.

Click **Add SSH Key** to open the SSH key modal:

1. **Key Name**: Identifier for this key in the list
2. **Enabled**: Toggle to enable/disable this key
3. **SSH Public Key**: Paste the user's public key (text format)
4. **Upload Public Key**: Upload a public key file (key name auto-populates if empty)
5. **Generate Key**: Generate a new key pair
    - The public key populates in the text area
    - The private key downloads automatically (send this to the user)
    - **Bit Selector**: Choose key size (larger = more secure but slower to generate)

Ideal vs. Convenient: The most secure workflow is for the SFTP user to generate their own key pair and send you only the public key (which you paste into the SSH Public Key field). This way, the private key is created on the user's machine and never travels over email or other channels. However, not all users are comfortable generating keys at the command line. The **Generate Key** option provides a convenient alternative—just be sure to transmit the downloaded private key to the user through a secure channel.

Advanced Options

Password authentication and IP restrictions are placed under Advanced Options because SSH keys are the recommended default. Expand this section to access:

- **SFTP Password**: Set a password for password authentication. This is less secure than SSH keys because users tend to choose weak passwords, and passwords are transmitted (encrypted) during each login. Use only when SSH keys are not feasible for your users. Passwords must meet the following requirements:
    - At least 8 characters
    - Mixed case (uppercase and lowercase)
    - At least 1 special character
- **IP Address Allowlist**: Restrict SFTP access for this user to specific IP addresses or ranges. Enter addresses in CIDR notation (e.g., `192.168.1.100/32` for a single IP, or `10.0.0.0/8` for a range). When configured, connection attempts from non-allowed IPs are rejected.

  When to use this: Most users won't need IP restrictions, but this feature is valuable in mixed environments. Consider a scenario where some SFTP users connect from a consistent IP (e.g., a server running automated upload jobs) while others connect from varying IPs (working from home, office, or a coffee shop). The moment any user has a changing IP, you can't practically restrict access at the firewall or security group level—you'd have to constantly update the rules to accommodate mobile users. As a result, many administrators end up opening the SFTP port to `0.0.0.0/0` to remain practical.

The per-user IP Address Allowlist solves this by letting you protect the users you *can* protect. Configure allowlists for users with predictable IPs (like automated systems), while leaving the allowlist empty for users with dynamic IPs. This way, even with an open firewall, your server-to-server integrations have an extra layer of protection.

Authentication Methods

SSH Key Authentication (Recommended)

SSH key authentication is more secure than passwords because:

- **No secrets transmitted**: Authentication uses a cryptographic challenge-response. The private key never leaves the user's machine and is never sent over the network.
- **Cryptographically strong**: Keys are virtually impossible to brute-force, unlike passwords.
- **No human weakness**: Users don't choose weak keys—they're generated randomly.
- **Granular control**: Each key can be individually enabled/disabled without affecting other keys.

Workflow for SSH key authentication:

1. The SFTP user generates a key pair on their machine (see Generating SSH Key Pairs)
2. The user sends their **public key** to you (never the private key)
3. You add the public key to the user's account in SFTP Gateway
4. The user connects using their private key

Alternatively, use **Generate Key** in the Admin UI to create a key pair, then send the downloaded private key to the user securely.

Password Authentication

Password authentication is available but discouraged for production use. Passwords are:

- Often weak or reused across services

- Transmitted during each login (encrypted, but still a secret crossing the wire)

- Susceptible to phishing and social engineering

If you must use password authentication, enforce strong password policies and consider combining with IP allowlisting for defense-in-depth.

Editing and Deleting Users

- **Edit**: Click on a user row or hover and click the edit button
- **Delete**: Hover over a user row and click the delete button

Note: Deleting a user removes only the user account from SFTP Gateway. The user's folder and any files in cloud storage remain intact—SFTP Gateway does not delete data from cloud storage. To remove the data, delete it directly from your cloud storage provider.

Cloud Connections

A **Cloud Connection** defines a cloud storage destination and its credentials. The root of the virtual file system must have a Cloud Connection, but child folders inherit from their parent by default.

For example, if the root `/` points to `s3://my-bucket/`, then:

- `/users/` (inherited) → `s3://my-bucket/users/`

- `/users/alice/` (inherited) → `s3://my-bucket/users/alice/`

You can override inheritance by assigning a different Cloud Connection to any folder. You can also point multiple folders to a single Cloud Connection, making it a single point of change (e.g., when rotating credentials).

Amazon S3

Navigate to **Settings** > **Cloud Connections** > **Create AWS Connection**

Configure the following fields:

Connection Name

A friendly name to identify this Cloud Connection.

Cloud Connection Notes (Optional)

Additional context about this connection.

S3 URL

Enter the S3 bucket and optional path in the format: `s3://bucket-name` or `s3://bucket-name/path/`

Examples:

- `s3://my-bucket` — points to the root of the bucket

- `s3://my-bucket/sftp-data/` — points to a subfolder within the bucket

This allows you to use a specific folder within a bucket rather than the entire bucket.

S3 bucket naming requirements:

- Globally unique (cannot conflict with buckets in other AWS accounts)

- Lowercase letters, numbers, and hyphens only

- Fewer than 63 characters

Region (Optional)

The AWS region where the S3 bucket is located. If left blank, the SFTP Gateway server's region is used. If you plan on cross-region failover, make sure you populate the bucket's region.

S3 Encryption Option

- **SSE-S3**: S3 manages encryption. Objects are encrypted at rest and automatically decrypted when you have read access.
- **KMS**: KMS encryption offers more security because KMS key permissions are decoupled from S3 access permissions.
- **No Encryption**: Use the S3 bucket's default encryption setting.

Cloud Connection Credentials

- **Use instance profile credentials** (Recommended for AWS deployments): Leverages IAM permissions on the EC2 instance. Credentials are handled transparently and rotated automatically.
- **Use unique credentials**: Enter AWS Access Key and Secret Key. Required when SFTP Gateway is running on a different cloud (e.g., Azure or GCP) and cannot use an AWS instance profile.

Azure Blob Storage

Navigate to **Settings** > **Cloud Connections** > **Create Azure Connection**

Configure the following fields:

Connection Name

A friendly name to identify this Cloud Connection.

Cloud Connection Notes (Optional)

Additional context about this connection.

Storage Account Name

The name of your Azure Storage Account (e.g., `mystorageaccount`).

Container Name

The name of the Blob Storage container. Container names must be lowercase.

You can append a path to point to a subfolder within the container:

- `mycontainer` — points to the container root

- `mycontainer/sftp-data/` — points to a subfolder within the container

Cloud Connection Credentials

- **Instance Identity**: Uses the VM's managed identity permissions. No credentials needed.
- **Connection String**: Get this from **Azure Portal** > **Storage Account** > **Access keys** > **Connection String**. This is the more common choice because it is scoped to a single storage account.

Google Cloud Storage

Navigate to **Settings** > **Cloud Connections** > **Create GCP Connection**

Configure the following fields:

Connection Name

A friendly name to identify this Cloud Connection.

Cloud Connection Notes (Optional)

Additional context about this connection.

GCS URL

Enter the GCS bucket name in the format: `gs://bucket-name`

You can append a path to write files to a subdirectory:

`gs://bucket-name/path/to/folder`

Cloud Connection Credentials

Provide a **Service Account Key** (JSON format). To create one:

1. Go to **Google Cloud Console** > **IAM & Admin** > **Service Accounts**
2. Create a service account or select an existing one
3. Click **Keys** > **Add Key** > **Create new key** > **JSON**
4. Download the JSON file
5. Paste the contents into the Service Account Key field

The service account needs these permissions on the bucket:

- `storage.objects.create`
- `storage.objects.delete`
- `storage.objects.get`
- `storage.objects.list`

Or assign the **Storage Object Admin** role for simplicity.

Testing Connections

Always click **Test Connection** before saving. A successful test shows three green checkmarks. If the test fails, verify your settings before proceeding.

Folders

Folders define a virtual file system that SFTP users see when they connect. This virtual file system exists in the database—not on the local Linux file system—and maps paths to cloud storage locations via Cloud Connections.

How the Virtual File System Works

The virtual file system allows you to create a unified directory structure that spans multiple cloud storage locations:

- **Root mapping**: The root of the virtual file system maps to a Cloud Connection, which defines a bucket and path (e.g., `s3://my-bucket/data`)
- **Per-folder mapping**: Any folder within the virtual file system can map to a different Cloud Connection. This means a subfolder could point to a different S3 bucket, or even to a completely different cloud provider like Azure Blob Storage
- **User home directories**: Each SFTP user's home directory maps to a folder in the virtual file system. Users are chrooted to this path and can access everything downstream
- **Lazy creation**: Folders created in the UI don't appear in cloud storage until a user logs in via SFTP and accesses them

Example: You could configure `/uploads` to point to a location only visible to one user, and `/finance` to point to a shared location accessible by multiple users—both within the same SFTP session.

Cloud Connection Inheritance

Folders inherit their Cloud Connection from their parent folder unless explicitly overridden:

- If you create `/data/reports` and only `/data` has a Cloud Connection, then `/data/reports` inherits that connection
- Files uploaded to `/data/reports` are written to the same bucket as `/data`, under the `reports/` prefix

Important: The Folders Tab Is Not a Live View

The Folders tab does not show files that exist in cloud storage. It only defines mappings between virtual paths and cloud storage locations—similar to `/etc/fstab` in Linux, where you define mount points rather than browse files.

If you navigate through the Folders tab and see no files, this is expected. To see actual files, connect via SFTP or use your cloud provider's console.

When to Create Folders

Do not recreate your cloud storage structure in the Folders tab. For example, if your S3 bucket contains `s3://my-bucket/marketing/` and `s3://my-bucket/finance/`, you do not need to create `/marketing` and `/finance` folders in SFTP Gateway.

Why this matters: If you manually mirror the cloud structure, you're duplicating state. If someone renames `/marketing` to `/sales` in S3, SFTP Gateway would still show `/marketing` (now empty) because the database hasn't changed. This creates confusion and a maintenance burden.

Only create folders when you have a specific reason:

1. **Mapping to a different cloud location** — The folder needs to point to a different bucket or path than its parent
2. **Creating a user home directory** — You need a folder like `/users/alice/` to assign as a user's home directory
3. **Assigning permissions** — You need to set specific permissions on a folder (e.g., `/users/alice/dropoff/` needs to be write-only)

If none of these apply, let the folder structure emerge naturally from cloud storage—SFTP users will see whatever exists in the mapped cloud location.

Folder Permissions

Permissions control what SFTP users can do within each folder:

Permission	Description
Read	Download files and list folder contents
Write	Upload files and create folders

Inheritance: Permissions are inherited from parent folders by default. A user with write access to `/data` automatically has write access to `/data/reports`.

Override: You can override inherited permissions on any subfolder. For example, grant a user full access to `/data` but set `/data/downloads` to read-only for that user.

Creating a Folder

1. Navigate to **Folders**
2. Click **Add Folder**
3. Enter the folder name
4. Optionally select a **Cloud Connection** (inherits from parent if not specified)
5. Click **Save**

Navigating the Folder Hierarchy

- Click on folder rows to navigate deeper
- Use the breadcrumb to navigate up
- Use search to find specific folders

System Settings

Access system settings via **Settings** in the top navigation. The Settings page contains four tabs: Admin Users, Cloud Connections, OIDC Connections, and Backup & Recovery.

Admin Users

Admin users are accounts that can log into the Admin Web Interface to manage SFTP users, folders, and cloud connections. Each administrator should have their own account so access can be revoked individually if needed.

To create an admin user, click **Add User** and fill in the username (required) and optional email address. Set a **One Time Password**—the user will be required to reset this password on first login. You can also use this field to reset a forgotten password.

Admin users can be enabled or disabled. A disabled user cannot log into the Admin Web Interface but their account remains in the system for re-enabling later.

Cloud Connections

See the Cloud Connections section for details on configuring S3, Azure Blob Storage, and GCS connections.

OIDC Connections

OIDC (OpenID Connect) connections allow you to integrate SFTP Gateway with identity providers like Microsoft Entra ID (Azure AD), Okta, Ping, or other OIDC-compliant providers. This enables single sign-on (SSO) for the Admin Web Interface. OIDC is also how you enable multi-factor authentication (MFA)—configure MFA policies in your identity provider, and they apply when admins log into SFTP Gateway.

Backup & Recovery

The Backup & Recovery feature exports and imports your SFTP Gateway configuration, including users, folders, admin users, and cloud connections.

Export downloads all configuration to a YAML file named `sftpgw-backup-<date-time>.yaml`. Store this file securely—it contains your full configuration.

Import restores configuration from a previously exported backup file. Click the file selector to browse for your backup file, then click **Import**. Status messages indicate which items were imported successfully.

Important: Import should only be done on a fresh SFTP Gateway instance. Imported items do not overwrite existing ones—if a user already exists on the server, that user is skipped during import, even if the backup file contains different settings (such as a different password).

Common uses for backup/recovery:

- **Disaster recovery**: Restore configuration after a server failure

- **Migration**: Move configuration to a new SFTP Gateway instance

- **Environment cloning**: Copy configuration from production to staging

Encryption Algorithms

The Encryption Algorithms section lets you control which SSH algorithms the SFTP server supports. You can enable or disable individual algorithms across four categories:

Category	Purpose
Key Exchange	Algorithms for securely establishing a shared session key
Public Keys	Algorithms for authenticating the server and users
SSH2 Ciphers	Algorithms for encrypting data in transit
HMACs	Algorithms for verifying message integrity

To configure encryption algorithms:

1. Navigate to **Settings** > **Encryption Algorithms**

2. Click **Edit** to enter editing mode
3. Enable or disable individual algorithms using the checkboxes
4. Changes are saved automatically

By default, only strong algorithms are enabled. Enable weaker algorithms only if you need to support legacy SFTP clients that don't support modern cryptography. Click **Reset to Default** to restore the default configuration.

For more information, see the Encryption Algorithms Reference in the knowledge base.

Application Properties

Application properties configure advanced settings that are not available in the Admin Web Interface. These settings control behaviors like password policies and brute force protection.

Configuration File Location

```
/opt/sftpgw/application.properties
```

Password Policy

Customize SFTP user password requirements:

```
password.policy.requireUpper=true
password.policy.requireDigit=true
password.policy.requireSpecial=true
password.policy.minLength=8
password.policy.maxLength=50
```

Set any of these to `false` or adjust the length requirements to match your organization's password policy.

Brute Force Protection

SFTP Gateway includes brute force protection that bans IP addresses after repeated failed login attempts:

- **SFTP**: 10 failed attempts in 5 minutes results in a 5-hour IP ban
- **Admin Web UI**: 10 failed attempts results in a 1-hour lockout per IP

When to disable: If SFTP Gateway sits behind a proxy or firewall that masks client IPs, all traffic appears to come from the proxy's IP address. In this case, failed login attempts from the internet would eventually ban the proxy's IP, blocking all incoming traffic. Disable brute force protection in these environments.

To disable IP banning, add:

```
sftp.ban.enabled=false
```

Session Timeouts

Control how long idle sessions remain open:

```
# Disconnect after 20 minutes of inactivity (default: 1200 seconds)
sftp.connection-idle-timeout-seconds=1200

# Time allowed for authentication (default: 120 seconds)
sftp.auth-idle-timeout-seconds=120
```

S3 Upload Settings

For large file uploads to S3:

```
# Maximum size for simple (non-multipart) uploads in bytes
# Default: ~117 MB. Increase for large files to avoid multipart overhead.
features.file-system.aws-s3.max-simple-upload-size-bytes=123289600
```

Verbose Logging

For troubleshooting difficult issues, enable verbose logging:

```
logging.level.com.sftpgateway.backend.sftp.logging.MaverickLogService=DEBUG
```

For even more detail, use `TRACE` instead of `DEBUG`.

Warning: Verbose logging fills up disk space quickly. Disable this setting once you are no longer actively troubleshooting.

For more details, see SFTP Gateway Logging.

Additional Properties

For the complete list of application properties, see the Application Properties Reference in the knowledge base.

Applying Configuration Changes

After modifying application properties:

1. Save the configuration file
2. Restart the service:

```
sudo su
service sftpgw-admin-api restart
```

API Reference

SFTP Gateway provides a REST API for programmatic management.

Why Use the API

The Admin Web Interface works well for managing a handful of users, but the API is better suited for:

- **Automation**: Provision new users automatically when employees join or external partners are onboarded
- **Bulk operations**: Migrate hundreds of users from an existing system without manual data entry
- **Scheduled backups**: Automate configuration exports on a schedule (e.g., nightly backups via cron)
- **Integration**: Connect SFTP Gateway to ticketing systems or custom workflows

How API Authentication Works

All API calls require a bearer token. The flow is:

1. **Log in** with your Admin Web Interface credentials
2. **Receive a bearer token** in the response headers
3. **Include the token** in the `Authorization` header for all subsequent API calls

The token is valid for the duration of your session. If a request returns `401 Unauthorized`, obtain a new token by logging in again.

Authentication

Log in with your Admin Web Interface credentials to obtain a bearer token:

```
POST /backend/login
```

Request:

```json
{
  "username": "admin",
  "password": "your-password"
}
```

The response headers include an `Authorization` header with your bearer token. Include this token in all subsequent requests:

```
Authorization: Bearer <token>
```

Example: List Users

```
GET /backend/api/users
```

Example: Create User

```
POST /backend/api/users
```

Common Workflow

A typical automation flow for provisioning a new SFTP user:

1. **Create a Cloud Connection** — Define the cloud storage bucket and credentials
2. **Create a Folder** — Create a folder in the virtual file system and wire it to the Cloud Connection
3. **Create the SFTP User** — Create the user and set their home directory to the folder
4. **Set authentication** — You can set the password during user creation or update. For SSH keys, create an SSH key object via the API and add it to the user's array of SSH keys. There's also a convenience endpoint for generating a new key pair.

API Documentation

The full API documentation is provided as a Swagger document. To view it in a human-readable format, load the document into swagger.io.

For examples in bash and Python, see the Admin API Reference in the knowledge base.

High Availability Deployment

A single SFTP Gateway instance is a single point of failure. If that instance goes down—whether due to hardware failure, network issues, or maintenance—SFTP users cannot connect and file transfers stop.

For some use cases, brief downtime is acceptable. But if your organization relies on SFTP for time-sensitive file transfers—such as receiving financial transactions, EDI documents, or data feeds that downstream systems depend on—you cannot afford to have the SFTP service unavailable. Files may be lost or delayed, and upstream systems may fail when they can't deliver their payloads.

High availability (HA) deployment solves this problem by running multiple SFTP Gateway instances behind a load balancer. If one instance fails, the load balancer routes traffic to the remaining healthy instances. Users experience no interruption, and file transfers continue without manual intervention.

When to Use HA

Consider HA deployment if:

- Your SFTP service must be available 24/7

- Upstream systems send files on a schedule and expect the service to be ready

- Downtime would cause missed SLAs, compliance issues, or operational disruptions

- You need to perform maintenance without taking the service offline

- You need to scale out to handle burst traffic

For development, testing, or low-criticality workloads, a single instance may be sufficient.

Architecture Requirements

Load Balancer Configuration

SFTP traffic requires a **Layer 4 (TCP) load balancer** because it runs over TCP 22. Use a Network Load Balancer (NLB) on AWS, a Standard Load Balancer with TCP rules on Azure, or a TCP/UDP Network Load Balancer on GCP.

Configure the load balancer to forward:

- **Port 22** → SFTP Gateway instances (SFTP traffic)

- **Port 443** → SFTP Gateway instances (Admin Web Interface)

Note: Some customers deploy an additional Application Load Balancer (ALB) for the Admin Web Interface to handle SSL termination at the load balancer. This requires additional nginx configuration on the VM since nginx expects to terminate SSL itself. For most deployments, this is unnecessary.

SSL/TLS Certificates

The Admin Web Interface uses a self-signed SSL certificate by default. Since only administrators access this interface (not end users), most customers accept the browser warning and use the self-signed certificate as-is.

If your security requirements prohibit self-signed certificates, you can install a valid certificate on each instance:

- Purchase a commercial SSL certificate from a provider like Comodo or GoDaddy

- Use Let’s Encrypt with DNS validation to obtain the SSL certificate

- Let the cloud provider manage SSL certificates (e.g., ACM on AWS) if using a load balancer

Shared State

Historically, deploying SFTP in HA has been difficult. Traditional SFTP implementations use OpenSSH, which stores user accounts in local files (`/etc/passwd`), writes uploaded files to the local filesystem, and keeps configuration in local files. This local state creates a synchronization problem—how do you keep user accounts, files, and configuration in sync across multiple instances?

SFTP Gateway solves this by eliminating local state entirely:

Component	Traditional SFTP (OpenSSH)	SFTP Gateway
User accounts	Local files (`/etc/passwd`)	PostgreSQL database
Configuration	Local files	PostgreSQL database
Uploaded files	Local filesystem	Written directly to cloud storage
Audit logs	Local log files	Streamed to cloud logging service

SFTP Gateway uses a custom Java application that implements the SFTP protocol, giving it the flexibility to do things that would be impossible with OpenSSH. Because all state is externalized, there's no synchronization delay—every instance sees the same users, configuration, and files immediately.

Database: In HA deployments, all instances connect to a shared PostgreSQL database (e.g., Amazon RDS, Azure Database for PostgreSQL, or Cloud SQL).

File uploads: Files are written directly to cloud storage, not staged on the local filesystem. Any instance can serve any user's files with no replication lag.

Logs: Audit logs are streamed to the cloud logging service (e.g., CloudWatch on AWS). This means you don't need to SSH into individual instances to check if a file transferred—all logs are centralized. This is especially important in HA because instances may be cycled out by the auto-scaling group, taking their local logs with them.

Platform-Specific Deployment

We provide HA templates for each cloud platform as reference implementations. These templates are designed for simplicity—use them as a starting point and iterate until they meet your specific requirements.

AWS

The AWS Marketplace offers HA CloudFormation templates:

- **New Network template**: Provisions VPC, subnets, and all network resources. Easier for new deployments.

- **Existing Network template**: Lets you select an existing VPC and subnets. Use this when deploying into an established network environment.

Azure

Azure HA templates are available in the knowledge base:

- HA ARM Template

- HA Terraform Template

GCP

GCP HA templates are available for both GDM and Terraform. Terraform is preferred as GDM is being phased out by GCP.

Custom Deployments

In some environments, you may only be allowed to deploy specific resources (e.g., the auto-scaling group but not the load balancer, database, or network). For custom template modifications, contact support about premier support options.

Health Checks

Configure load balancer health checks to use the dedicated health endpoint:

- **Protocol**: HTTPS
- **Port**: 443
- **Path**: `/backend/actuator/health`
- **Interval**: 30 seconds
- **Threshold**: 2-3 failures before marking unhealthy

This endpoint returns the health status of the SFTP Gateway application. Our HA templates have this configured by default.

Launch Configuration (for Custom Templates)

If you're writing your own Terraform or CloudFormation templates from scratch, the userdata must create a launch configuration file that tells the instance it's running in HA mode.

Create `/opt/sftpgw/launch_config.env` with:

```
ARCHITECTURE=HA
CLOUD_PROVIDER=aws  # or azure, gcp
DB_HOST=<database-hostname>
```

Example for Azure:

```
ARCHITECTURE=HA
CLOUD_PROVIDER=azure
DB_HOST=${azurerm_postgresql_flexible_server.sftpgw.fqdn}
```

The `ARCHITECTURE=HA` flag tells the instance to use the external PostgreSQL database instead of the local embedded database. Without this, each instance would use its own database and state would not be shared.

Static IP for Client Whitelisting

SFTP clients sometimes need to whitelist the IP address of your SFTP server in their firewalls. In an HA deployment, provide clients with the load balancer's static IP address.

- **Azure and GCP**: The load balancer is assigned a single static public IP address. Find this IP in the Azure Portal or GCP Console under the load balancer resource.
- **AWS**: The CloudFormation templates associate an Elastic IP with the NLB in one of the availability zones. You can find this IP in the EC2 Console under Elastic IPs, or in the NLB configuration.

Provide this IP to clients who need to whitelist your SFTP endpoint.

Server Administration

This section covers administrative tasks that require SSH access to the SFTP Gateway server itself.

SSH Access to the Server

SFTP Gateway has moved the SSH port (OpenSSH) from the default 22 to 2222. This was necessary so our custom Java application could listen on port 22 to serve SFTP.

Port	Purpose	Who Uses It
22	SFTP file transfers	SFTP users (clients)
2222	SSH admin access	Administrators

To access the server for administration tasks, connect on **port 2222**:

```
# AWS (Amazon Linux)
ssh -i /path/to/key.pem ec2-user@<server-ip> -p 2222

# Azure
ssh -i /path/to/key.pem azureuser@<server-ip> -p 2222

# GCP (Ubuntu)
ssh -i /path/to/key.pem ubuntu@<server-ip> -p 2222
```

The SSH key is the key pair you specified when launching the instance (not an SFTP user key).

Common mistake: Use port 2222 rather than 22 for admin SSH. Port 22 is reserved for SFTP client connections and will not give you a shell.

Service Management

SFTP Gateway runs as systemd services. Common commands:

```
# Check service status
sudo systemctl status sftpgw-admin-api.service

# Restart the service (apply configuration changes)
sudo systemctl restart sftpgw-admin-api.service

# View service logs
sudo journalctl -u sftpgw-admin-api.service -f
```

Log Files

SFTP Gateway writes logs to `/opt/sftpgw/log/`. Both audit logs and application logs include the date in the filename.

Log	Purpose
`sftp-audit-*.log`	File transfer activity, user logins
`application-*.log`	Admin API, cloud connection errors

Viewing Logs

```
cd /opt/sftpgw/log/
tail -f *
```

This tails all log files, catching any new activity on the most recent log file, regardless of the filename.

SFTP User Instructions

This section contains instructions you can share with your SFTP users. Consider copying this section into a separate document for distribution.

For SFTP Users: Getting Started

Welcome! This section explains how to connect to SFTP Gateway and transfer files.

What You Need

1. **Connection details** from your administrator:
    - SFTP server hostname or IP address
    - Your username
    - Either a password OR an SSH private key file

2. **An SFTP client** installed on your computer (FileZilla, WinSCP, or command line)

Generating SSH Key Pairs

If your administrator uses SSH key authentication, you may need to generate a key pair and send your **public key** to your administrator. Never share your private key.

On Mac/Linux

1. Open Terminal
2. Generate a key pair:

   ```
   ssh-keygen -t ed25519 -C "your-email@example.com"
   ```

3. When prompted for a file location, press Enter for the default (`~/.ssh/id_ed25519`)
4. Optionally enter a passphrase for additional security
5. Your keys are created:
   - **Private key**: `~/.ssh/id_ed25519` (keep this secure, never share)
   - **Public key**: `~/.ssh/id_ed25519.pub` (send this to your administrator)
6. View your public key to copy/send:

   ```
   cat ~/.ssh/id_ed25519.pub
   ```

On Windows (using PuTTYgen)

1. Download and install PuTTY
2. Open **PuTTYgen**
3. Click **Generate** and move your mouse randomly to generate entropy
4. Once generated:
   - Save the **private key** (.ppk file) securely
   - Copy the text from “Public key for pasting” and send it to your administrator
5. For OpenSSH format private key (for some clients):
   - Go to **Conversions** > **Export OpenSSH key**

On Windows (using ssh-keygen)

Windows 10/11 includes OpenSSH. Open PowerShell:

```
ssh-keygen -t ed25519 -C "your-email@example.com"
```

Keys are saved to `C:\Users\YourUsername\.ssh\`

Connecting with FileZilla

1. Open FileZilla

2. Go to **File** > **Site Manager**
3. Click **New Site**
4. Configure:
    - **Protocol**: SFTP - SSH File Transfer Protocol
    - **Host**: Server hostname from your administrator
    - **Port**: 22 (or as specified)
    - **Logon Type**: Key file (or Normal for password)
    - **User**: Your username
    - **Key file**: Browse to your private key (if using keys)
5. Click **Connect**

Connecting with WinSCP

1. Open WinSCP
2. In the Login dialog:
    - **File protocol**: SFTP
    - **Host name**: Server hostname
    - **User name**: Your username
3. For SSH key authentication:
    - Click **Advanced** > **Authentication**
    - Browse to your private key file
    - If your key is `.pem` format, WinSCP will prompt to convert to `.ppk`
4. Click **Login**

Connecting via Command Line

```
# With SSH key
sftp -i ~/.ssh/id_ed25519 username@server-hostname

# With password (you'll be prompted)
sftp username@server-hostname
```

Basic SFTP Commands

Command	Description
`ls`	List remote files
`cd <dir>`	Change remote directory
`lcd <dir>`	Change local directory
`pwd`	Print remote working directory
`lpwd`	Print local working directory
`put <file>`	Upload a file
`get <file>`	Download a file
`mkdir <dir>`	Create a directory

Command	Description
`rm <file>`	Delete a file
`exit`	Disconnect

Transferring Files

1. Connect to the SFTP server
2. Navigate to your upload folder (e.g., `cd uploads`)
3. Upload files: drag and drop in your client, or use `put filename`
4. Files are written directly to cloud storage—no sync delay

Common Issues

Unable to log in: Verify that the hostname or IP address is correct. Check that your username is entered exactly as provided (usernames are case-sensitive). For password authentication, confirm you're using the correct password. For SSH key authentication, ensure you're using the matching private key.

Connection timeout: Your IP address might not be allowed through the firewall. Contact your administrator to verify the security group permits inbound connections from your IP on port 22. To find your IP address, go to ipchicken.com, an easy-to-remember website.

Debugging connection problems: For detailed diagnostic output, connect from the command line with verbose logging: `sftp -vvv username@hostname`

Security Best Practices

For Administrators

1. **Restrict administrative access by IP**: See Restricting Administrative Access below
2. **Deploy resources in private subnets**: Expose only the Network Load Balancer in your DMZ while keeping SFTP Gateway instances protected in private subnets
3. **Use SSH key authentication**: Disable password authentication when possible
4. **Use instance profile credentials**: Instance profiles automatically rotate credentials, unlike static access keys
5. **Principle of least privilege**: Avoid broad permissions, such as S3 full access. The server would have access to all buckets, including existing ones with sensitive data. Restrict permissions to only the specific buckets needed.
6. **Monitor audit logs**: Audit logs reveal important activity patterns—scripted attacks from specific IPs, misbehaving client scripts causing performance issues, or unusual access patterns. See AWS Log Observability Setup for using tools like Grafana to visualize logs.

7. **Use strong encryption algorithms**: Only enable weaker algorithms if legacy clients require them
8. **Regular backups**: Export configuration regularly for disaster recovery
9. **Keep software updated**: Apply OS security patches promptly and stay current with the latest version of SFTP Gateway, which includes security fixes

Restricting Administrative Access

SFTP Gateway exposes three administrative ports that should be restricted to administrator IP addresses:

Port	Service	Recommendation
22	SFTP (user connections)	Open to the internet (by design)
80/443	Admin Web Interface	Restrict to administrator IPs
2222	SSH admin access	Restrict to administrator IPs

Ideally, port 22 would also be restricted to an IP allowlist, but this is often unfeasible when SFTP users connect from various locations. When port 22 must remain open to the internet, per-user IP allowlists provide an additional layer of security for users with predictable IPs. Note that SSH administrative access has been moved to port 2222, so port 22 only exposes the SFTP service.

How to restrict:

- **AWS**: Configure Security Group inbound rules to allow ports 80, 443, and 2222 only from your administrator’s IP addresses or office network range
- **Azure**: Configure Network Security Group rules to restrict these ports
- **GCP**: Configure firewall rules to restrict these ports

If using CloudFormation or ARM templates, the administrative IP range is typically a deployment parameter that configures these restrictions automatically.

Frequently Asked Questions

General

Q: What cloud storage providers are supported? A: Amazon S3, Azure Blob Storage, and Google Cloud Storage.

Q: Can users connect with any SFTP client? A: Yes, any standard SFTP client works (FileZilla, WinSCP, command line, etc.).

Q: Are files stored on the SFTP Gateway server? A: No, files are written directly to cloud storage. There is no intermediate staging area.

Q: When do folders appear in cloud storage? A: Folders are created in cloud storage when a user first logs in via SFTP, not when you create them in the UI.

Q: How do I check what version of SFTP Gateway I'm running? A: Scroll to the bottom of the web admin portal. The version number is displayed in the footer.

User Management

Q: Can I generate SSH keys for users? A: Yes, use the **Generate Key** button when adding SSH keys. The private key downloads automatically for you to send to the user.

Q: How do I reset a user's password? A: Edit the user in the Admin UI and set a new password in the SFTP Password field. This is found under Advanced Options.

Q: Can I restrict users to specific IP addresses? A: Yes, use the **IP Address Allowlist** in the user's Advanced Options. Enter IP addresses in CIDR format (e.g., `192.168.1.100/32`).

Security

Q: Is data encrypted in transit? A: Yes, SFTP encrypts all data as part of the protocol.

Q: Is data encrypted at rest? A: This depends on your Cloud Connection settings. Use SSE-S3 or KMS encryption for S3.

Q: How do I support legacy SFTP clients? A: Go to **Settings** > **Encryption Algorithms** and enable the legacy algorithms required by your client.

Backup and Recovery

Q: How do I back up my configuration? A: Go to **Settings** > **Backup & Recovery** > **Export** to download a YAML file with all users, folders, and settings.

Q: Can I migrate to a new server? A: Yes, export from the old server and import on the new server using the Backup & Recovery feature.

Getting Help

- **Knowledge base**: help.thorntech.com
- **Support**: support@thorntech.com

SFTP Gateway is developed by Thorn Technologies. For the latest documentation, visit help.thorntech.com.