Ngenea Hub

Ngenea Hub harnesses the power of Ngenea to provide global workflows, enabling your data to be where you need it, when you need it.

Installation

Notice

Hub versions 1.18+ have additional requirements

If an upgrade is being performed from a version < 1.18 please note changes in the Installation and Upgrade guides

A deployment of Ngenea Hub comprises two main components:

- **Ngenea Hub** - the central management point, from which all tasks are managed
- **Ngenea Worker** - worker agents, installed on individual Ngenea servers, that execute the Ngenea tasks

Ngenea Worker are installed on one or more nodes in a Site, which typically represents a single Ngenea cluster or location.

Ngenea Hub needs to be accessible from all Ngenea Workers on the following ports:

- `6379/tcp`
- `5672/tcp`
- `8000/tcp`

  `8000/tcp` is required for Hub versions >= 1.18

Installing Ngenea Hub

Ngenea Hub can be installed in a number of ways, use one of the methods described below.

CentOS / Redhat - Online Installation

Configure Docker Authentication

Note: This step is not required on PixStor systems

Configure Docker authentication for `eurepo.arcapix.com`.

```
docker login eurepo.arcapix.com
```

Installing Ngenea Hub

Transfer the `ngenea-hub` rpm to the target system.

Install the `ngenea-hub` package via yum.

```
yum install ngenea-hub-<version>.rpm
```

Optionally, create an initial Ngenea Hub configuration file at `/etc/sysconfig/ngeneahub`. This file contains the credentials which will be required for deploying workers, and can also be edited to use external queue systems prior to starting the Ngenea Hub service. This file will be created automatically if it does not exist when the service is started.

```
ngeneahubctl createconfig
```

Enable and start the Ngenea Hub service.

```
systemctl enable --now ngeneahub
```

Check the status of the service with:

```
ngeneahubctl status
```

CentOS / Redhat - Offline installation

The `ngeneahub` service will attempt to pull the required docker images from the Ngenea software repository servers. In situations where this is not possible (due to network restrictions, for instance), the containers can be installed via additional RPM: `ngenea-hub-images`, available at the same location as the main RPM.

Once the RPMs are transferred to the target system, they can be installed using rpm.

```
rpm -ivh ngenea-hub-<version>.rpm ngenea-hub-images-<version>.rpm
```

Cloud Deployment

Coming soon: image-based deployment of Ngenea Hub in the cloud.

Container Native Deployment

It is possible to deploy Ngenea Hub using standard container management tools and processes.

Please contact us to discuss.

Ngenea Worker

Pre-requisite: Ngenea Server software installed and configured

Install the `ngenea-worker` package via rpm.

```
yum install ngenea-worker
```

Add the appropriate worker configuration.

Note: The following section is new for version 1.18+

Run the join command to ensure TLS communications. Provide the username and password of a registered hub admin user to successfully authenticate. Refer to Hub Initial Configuration to add a hub admin user.

```
ngenea-worker join --user <USERNAME>
```

Example of ngenea-worker join with worker configuration option `api_secure=false`:

```
root@myserver:/root ngenea-worker join --user hubadmin
hubadmin's password:
Attempting to auth the hub using the provided credentials...
Authenticated as user onprem-worker.
Storing the client certificates for the worker...
Client certificate stored at /etc/pki/tls/certs/onprem.crt
Client private key stored at /etc/pki/tls/private/onprem.key
Storing the root CA for NgeneaHub...
Root CA stored at /etc/pki/tls/certs/ng-hub-ca.crt
```

Enable and start the `ngenea-worker` systemd unit.

```
systemctl enable --now ngenea-worker
```

Product Interoperability Matrix

Ngenea Hub requires specific versions of Ngenea.

- All Ngenea Hub managed sites must use the same version of Ngenea
- Mixed versions are unsupported

The following table defines the supported Ngenea versions for each version of Ngenea Hub.

Ngenea Hub version	Ngenea minimum version	Ngenea maximum version
1.28.0	1.26.1	1.26.1
1.26.0	1.25.0	1.25.0
1.26.0	1.24.1	1.24.1
1.25.0	1.24.1	1.24.1
1.24.0	1.21.0	1.22.0
1.23.0	1.21.0	1.22.0
1.22.0	1.21.0	1.22.0
1.21.0	1.21.0	1.21.0
1.20.0	1.21.0	1.21.0
1.19.0	1.21.0	1.21.0
1.17.3	1.20.1	1.20.1
1.17.0	1.19.0	1.19.0
1.13.0-1.16.0	1.16.0	1.19.0
1.10.0-1.12.0	1.15.0	1.16.0
1.9.0	1.14.0	1.14.0
1.8.0	1.13.0	1.14.0
1.7.0	1.12.0	1.12.0
1.6.0	1.12.0	1.12.0
1.5.0	1.12.0	1.12.0
<=0.6.0	1.9.0	1.11.0

Once installed, additional features such as Search may be set up as described in Feature Set-up

Ngenea Hub software must align with Ngenea software. Ensure to review the Product Interoperability Matrix and align software versions accordingly.

Upgrade

Note: Hub versions 1.18+ have additional requirements

If an upgrade is being performed from a version < 1.18 please note changes in the Installation and Upgrade guides

Ensure to review the Product Interoperability Matrix and align software versions accordingly.

Before you start

Before upgrading, you must wait for any pending or active jobs to complete, otherwise they may be lost or present an incorrect future state.

Scheduled workflows must be temporarily disabled prior to upgrading to prevent new jobs being submitted during the upgrade process.

Backup Workflows

Ngenea Hub ships with some default workflows. New releases may make changes to these workflows, so any customisations to them may be lost during upgrade. For safety, workflows should be backed-up before upgrading.

The easiest way to backup workflows is using ngclient

```
ngclient workflows list > workflows_backup.json
```

See NGCLIENT-WORKFLOWS for more information.

Stop Services

First, shutdown Ngenea Worker on all nodes

```
systemctl stop ngenea-worker
```

Note that any Ngenea Worker packages pre-1.12.0 will use the older syntax:

```
systemctl stop ngenea-worker@SITENAME
```

Then shutdown Ngenea Hub

```
systemctl stop ngeneahub
```

Warning: Not stopping the workers before upgrading may result in jobs being stuck as `PENDING`. If this happens, restarting the workers using `systemctl restart ngenea-worker` will allow jobs to start running again.

Upgrade Packages

Download the latest RPMs from the Download page.

Upgrade Ngenea Hub

```
yum upgrade ngenea-hub-<version>.rpm
```

As with Installation, for offline situations, the Ngenea Hub base and image rpms can be upgraded with

```
rpm -Uvh ngenea-hub-<version>.rpm ngenea-hub-images-<version>.rpm
```

Upgrade Ngenea Worker on all nodes

```
yum upgrade ngenea-worker-<version>.rpm
```

Upgrade Configurations

Warning: If upgrading from Ngenea Hub version 1.17 or older, you must convert worker configuration to 1.18+ format. Please refer to the appropriate worker configuration and worker join steps as described in Ngenea Worker. Installation of the worker is not required as this has been achieved in the prior Upgrade Packages step.

Restart Services

First, startup Ngenea Hub

```
systemctl start ngeneahub
```

Then start Ngenea Worker on all nodes

```
systemctl start ngenea-worker
```

If any scheduled workflows were disabled, they can now safely be re-enabled.

Validation

Check the status of the Ngenea Hub service with:

```
ngeneahubctl status
```

Check the status of Ngenea Worker service with

```
systemctl status ngenea-worker
```

Restoring Workflows

Check the workflows post-update. If there are any issue or inconsistencies with the upgraded workflows, they can be restored from the backup created pre-upgrade. This is also done using `ngclient`.

Any workflow which is missing can be re-imported using

```
ngclient workflows import <workflow_file>
```

Any workflow which has changed can be restored using

```
ngclient workflows update <id> <workflow_file>
```

Note, `ngclient` only allows importing or updating single workflows at a time. The `workflow_file` passed to the above commands must only contain a single workflow definition.

Configuration

Hub Initial Configuration

Once all services are up, create an admin user with:

```
ngeneahubctl adduser
```

Register an initial site (replacing `SITENAME`) (see Worker Installation for enabling a site's worker agents).

```
ngeneahubctl  addsite SITENAME
```

Log into the UI at http://server.address:8000.

Hub Configuration

Settings

The main configuration file for Ngenea Hub is at `/etc/sysconfig/ngeneahub`. This is an environment file which holds the information required for connecting to the various backend services.

Mandatory Settings

Setting	Description
DJANGO_SECRET	Secret string used secure signed data within django
POSTGRES_DB	Internal database name
POSTGRES_USER	Internal database username
POSTGRES_PASSWORD	Internal database password

Optional settings

Setting	Description
REDIS_HOST	

Setting	Description
	Address of the Redis queue results store. Defaults to the container service address.
WORKERS	The number of gunicorn workers to spawn for serving API requests. Default to 8.
CONSUMER_TIMEOUT	The timeout for rabbitmq consumer delivery acknowledgement in seconds. Default: 10800000 (3 hours)
HUB_PORT	User configurable hub port
WEB_BIND_IP	User configurable web bind IP
PUBLIC_URL	User configurable base url for the hub stack to be served from, must not end in a trailing slash.
HEARTBEAT	(bool) Key for Disabling/Enabling celery heartbeats, default: true (enabled)
GOSSIP	(bool) Key for Disabling/Enabling celery gossip, default: false (disabled)
MINGLE	(bool) Key for Disabling/Enabling celery mingle, default: false (disabled)
REDIS_HEALTH_CHECK_INTERVAL	(int) The Redis backend supports health checks. This value must be set as an integer whose value is the number of seconds between health checks. default: 60
REDIS_TCP_BACKLOG	(int) In high requests-per-second environments you need a high backlog in order to avoid slow client connections issues to redis. Default: 511
REDIS_SOCKET_TIMEOUT	(int) When there are network issues redis backend connection sockets can become stale, this timeout setting will reset the socket connection after this value in seconds after becoming idle and resume operation. Default: 60
CELERY_SOCKET_TIMEOUT	(int) When there are network issues redis broker sockets can become stale, this timeout setting will re-acquire the socket after becoming idle for this value in

Setting	Description
	seconds and resume operation. Default: 60
CELERY_CONNECTION_TIMEOUT	(int) When there are network issues redis broker connection via the acquired sockets can become stale, this timeout setting will reset the connection after becoming idle for this value in seconds and resume operation. Default: 60
EXPIRE_OLD_JOBS_INTERVAL	(cron) schedule for when old job expiration will be run. When the task runs, jobs older than the configured jobs_ttl will be expired. Default: 0 0 * * * (minutes can be random from 0-59)
REMOVE_OLD_SEARCH_RESULTS_INTERVAL	(cron) schedule for when search result removal will run. When the task runs, search results older than the configured search_result_ttl will be expired. Default: 0 0 * * * (minutes can be random from 0-59)
INVALIDATE_CANCELLED_JOB_TASKS_INTERVAL	(cron) schedule for when cancelled jobs are revoked. When the task runs, any tasks still active in a cancelled job will be automatically cancelled. Default: 0 * * * *
CLEANUP_OLD_EVENTS_INTERVAL	(cron) schedule for when old snapdiff events will be cleaned up. When the task runs, events for all but the 2 most recent completed snapdiff jobs per workflow will be deleted. Default: 0 * * * *
INACTIVE_TASKS_INTERVAL	(cron) schedule for when inactive tasks will be invalidated. When the task runs, any STARTED task which is not actually running in a worker will be marked as FAILED. Default: 0 * * * *

Server Configurations

Some settings are stored in the Ngenea Hub DB.

They can be viewed and changed via the REST API `/api/configurations/` endpoint.

See Configuration for more details.

Docker Compose configuration

The `docker-compose` file is stored in `/usr/share/ngeneahub/docker/docker-compose.yml`.

This can be extended by creating an override file at `/usr/share/ngeneahub/docker/docker-compose.override.yml`.

Worker Configuration

Note: This configuration is new for versions 1.18+. Best practice is to backup any pre-1.18 worker configuration prior to conversion to the 1.18+ format.

The Ngenea Worker configuration should be added to `/etc/ngenea/ngenea-worker.conf`. The configuration is in ini format. For example:

```
[settings]
site = site1
api_host = localhost
api_port = 8000
api_secure = true
redis_port = 6379
gpfs_nodes = ["localhost"]
```

Note: `api_secure=true` requires a valid SSL certificate and is not compatible with self-signed SSL certificates.

ngenea-worker defaults to `api_secure=true` if the `api_secure` setting is not specified.

Refer to: External SSL for provisioning SSL certificates

Alternately, specify `api_secure=false` to disable use of SSL.

The following is a list of available settings:

Option	Type	Default	Required	Description
site	string		Yes	The name of the queue to listen to
api_host	string		Yes	The base url for Ngenea Hub, this will be the url without the https protocol or port.
api_port	string		Yes	The port that the Ngenea Hub is being hosted on, by default

Option	Type	Default	Required	Description
				within Ngenea Hub it is 8000.
api_secure	string		No	If the API is behind a secure HTTPS connection, by default this is true. Refer to: External SSL for provisioning SSL certificates if `api_secure=true`.
api_secure_verify	string		No	If the API requests should verify certificates, by default this is true.
threads	int	10	No	The number of concurrent tasks that can be run.
heartbeat	bool	true	No	Key for Disabling/ Enabling celery heartbeats, by default Enabled.
gossip	bool	false	No	Key for Disabling/ Enabling celery gossip, by default Disabled.
mingle	bool	false	No	Key for Disabling/ Enabling celery mingle, by default Disabled.
redis_health_check_interval	int	60	No	The Redis backend supports health checks. This value must be set as an integer whose value is the number of seconds between health checks.
redis_socket_timeout	int	60	No	The Redis results backend supports a socket connection timeout, this value must be set as an integer whose value is the number of seconds.
celery_socket_timeout	int	60	No	The Redis celery broker supports a socket timeout, this value must be set as an integer whose value is the number of seconds.

Option	Type	Default	Required	Description
celery_connection_timeout	int	60	No	The Redis celery broker supports a socket connection timeout, this value must be set as an integer whose value is the number of seconds.
gpfs_nodes	list		No	A list of nodes to run the snapdiff policy scan on as a list of hostnames
enable_plugins	bool	false	No	Key for Disabling/ Enabling worker plugin behaviour (currently in alpha), by default Disabled.
loglevel	str	INFO	No	Key for setting the worker loglevel (more specific to worker main process). valid choices are INFO, DEBUG, WARNING, CRITICAL, ERROR (by default INFO).

Note: The initial access credentials can be found in the `/etc/sysconfig/ngeneahub` configuration file on the Ngenea Hub server. By default, the `BROKER_PASSWORD` is used for both the `broker_url` and the `result_backend` passwords, and the `BROKER_USER` is used for the `broker_url` username.

Passing Custom Config File to Ngenea Worker

The custom config file will be present in `/etc/ngenea/ngenea_config_arg.conf` file to run the systemd services.The format of the file is given below.

```
CONFIG=/etc/ngenea/ngenea-worker-custom.conf
```

Note: By default, this file will have the CONFIG commented to allow the default config to pass through if custom config file is not specified.

Uncomment the `CONFIG` in `/etc/ngenea/ngenea_config_arg.conf` file. Create a new worker config file (as ngenea-worker.conf) with different site name. Also site should be added to hub and worker using the commands listed below.

```
ngeneahubctl addsite <sitename>
ngenea-worker join --user <username> --site <sitename>
```

To Run ngenea-worker with multiple sites

To run services with two sites,

1. Create new service file same as `/usr/lib/systemd/system/ngenea-worker.service` with different filename (Eg: ngenea-worker-site1.service).
2. Create new custom config file same as `/etc/ngenea/ngenea_config_arg.conf` with different filename (Eg: ngenea_config_arg1.conf).
3. Create new worker config file same as `/etc/ngenea/ngenea-worker.conf` with different filename and give the filename in `ngenea_config_arg1.conf`.
4. In `ngenea-worker-site1.service`, change the value of `EnvironmentFile=...` to new custom config file created (Eg: EnvironmentFile=/etc/ngenea/ngenea_config_arg1.conf).
5. Now run `systemctl start ngenea-worker.service` and `systemctl start ngenea-worker-site1.service`.

Setting Ngenea Worker debug level in systemd service script

In `/usr/lib/systemd/system/ngenea-worker.service` file, set the value of `Environment=DYNAMO_DEBUG=..` to `true` as below. It will be `false` by default. This debug level is more specific to tasks logging.

```
Environment=DYNAMO_DEBUG=true
```

Feature Set-up

To use certain features in Ngenea Hub, additional set-up is required as described in this section.

Search

The search feature provides the ability to search for files across one or more sites.

This page describes the steps required to set up the search feature.

For information on how to use the search feature, see Search

Prerequisites

The search feature is only supported on workers running on a PixStor.

There are two backends which can provide search functionality to Ngenea Hub: **PixStor Search** and **PixStor Analytics**.

This document assumes that the desired backend has already been deployed, as described in the PixStor deployment guide. Be aware that neither backend will have been deployed by default.

Configuration

Search Backend

By default, Ngenea Hub will use the Analytics backend. If you want to use the PixStor Search backend instead, change the `search_backend` to `pixstor_search`, as described in Global Configurations

```
$ curl -s -X PATCH 'http://example.com/api/configurations/' -H
'Accept: application/json' -H "Authorization: Bearer
$JWT_ACCESS_TOKEN" -H 'Content-Type: application/json' -d
'{"search_backend": "pixstor_search"}'
```

All sites **must** use the same backend - all analytics or all PixStor Search.

Elasticsearch URL

If you are using the Analytics backend and the worker is running on **PixStor 6**, you will need to change the `elasticsearch_url` for the site to `http://localhost:19200`, as described in Site-specific Configurations

```
$ curl -s -X PATCH 'http://example.com/api/sites/1/' -H 'Accept:
application/json' -H "Authorization: Api-Key $APIKEY" -H 'Content-
Type: application/json' -d '{"elasticsearch_url": "localhost:19200"}'
```

If not set, the `elasticsearch_url` will default to `localhost:9200`, which is correct for workers running on PixStor 5.

Different sites may configure a different `elasticsearch_url`, for example if one is PixStor 5 and another is PixStor 6.

Search UI

Once configured, search can be used via the REST API as described in Search

To use search via the Ngenea Hub UI, you must first enable the `searchui` feature flag, as described in Feature Flags

For example, using ngclient

```
ngclient features enable searchui
```

Cloud Functions

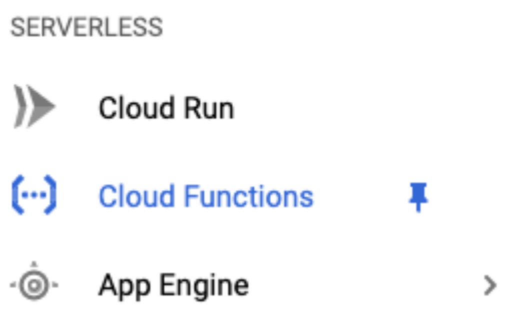
Cloud to Hub is a function intended to be running in the cloud. It runs in response to cloud storage events, triggering a job in Ngenea Hub to reflect the change onto another system (e.g. PixStor). This allows for keeping multiple systems in sync, using cloud storage as the source of truth.

The code is designed to work with any of the supported platforms (see above), and any event (create, delete, ...) with only changes to the `config.json` file, as described below.

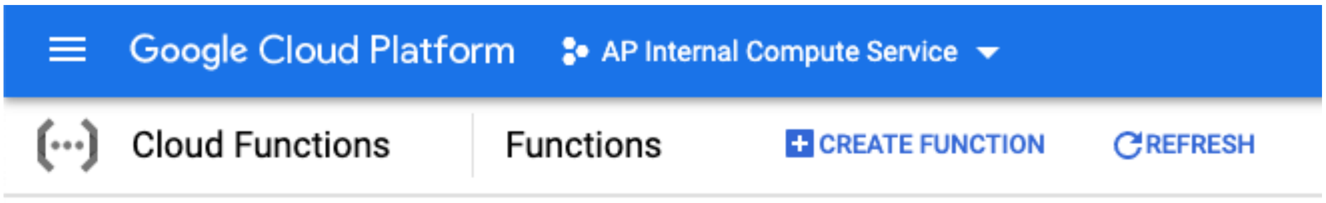
Currently supports:

GCP Cloud Function

In the GCP console menu under Serverless, select **Cloud Functions**



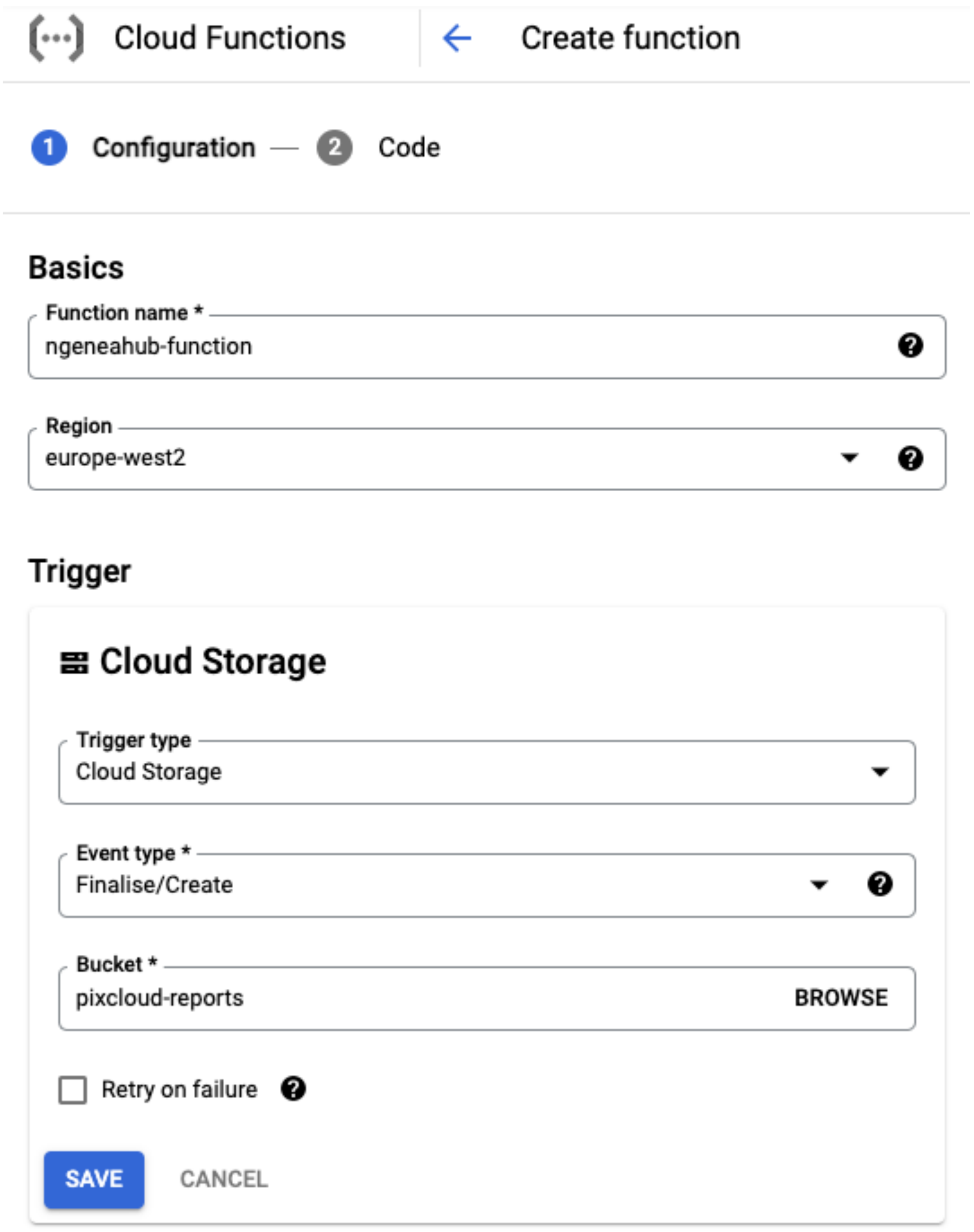
Choose **Create Function**



In the first Configuration page:

1. Give the function a name
2. Select a region to run from. Usually best to be in the same region as the bucket we'll be using
3. Under trigger select **Cloud Storage** type
4. And **Finalise/Create** for the event type
5. Select the bucket to monitor

Then press **Save**



Basics

Function name *
ngeneahub-function

Region
europe-west2

Trigger

Cloud Storage

Trigger type
Cloud Storage

Event type *
Finalise/Create

Bucket *
pixcloud-reports

Retry on failure

If you wish to create a new service account for this function, use the following `gcloud` command to create a new service account and assign it the role `storage.objectViewer`

```
PROJECT_ID='GCP-PROJECT-1'
SERVICE_ACCOUNT_ID='ngeneahub-function'
ROLE_NAME='roles/storage.objectViewer'

gcloud iam service-accounts create $SERVICE_ACCOUNT_ID \
    --description='A service account to give the {{ brand_name }}
function read access to GCS buckets' \
    --display-name=$SERVICE_ACCOUNT_ID

gcloud projects add-iam-policy-binding $PROJECT_ID \
    --member="serviceAccount:
```

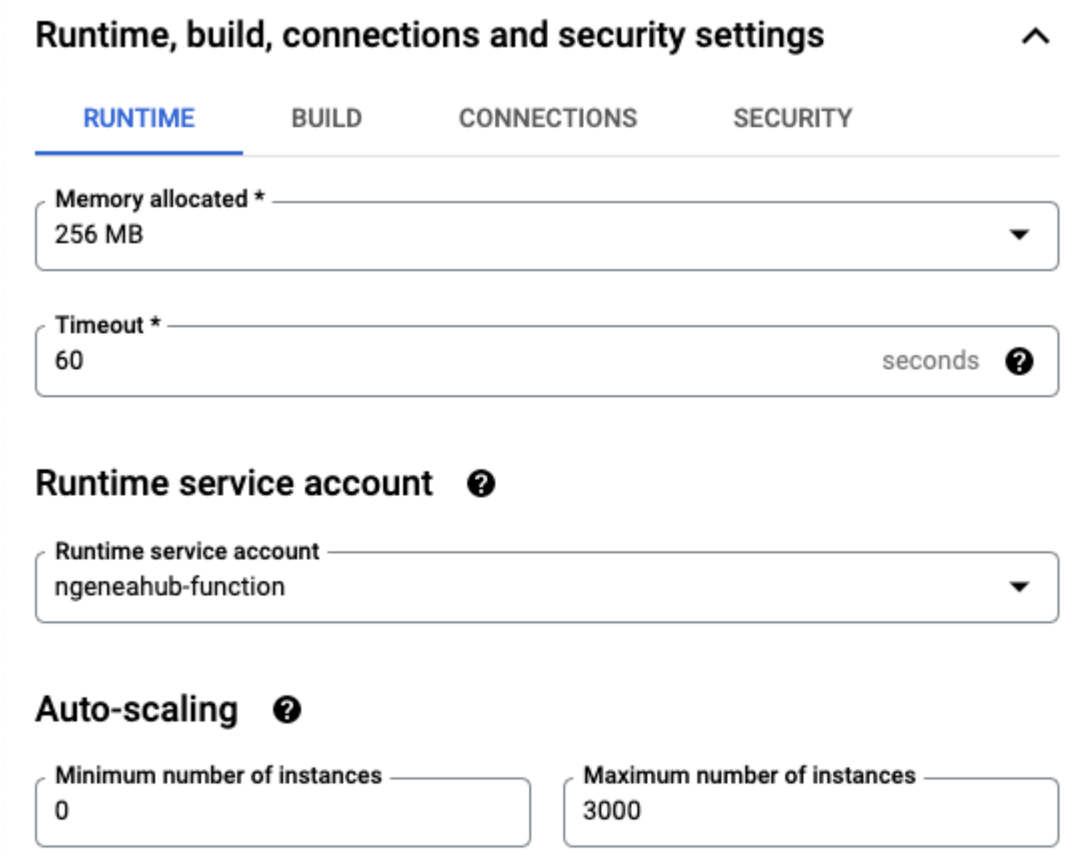
```
$SERVICE_ACCOUNT_ID@$PROJECT_ID.iam.gserviceaccount.com" \
    --role=$ROLE_NAME
```

Open up the **RUNTIME, BUILD AND CONNECTIONS SETTINGS** section

Under the **RUNTIME** tab at the bottom, select a **Runtime** service account that has the following permissions as a minimum (or the newly created service account from above):

- `storage.objectViewer`

Select **Next** to continue


Runtime, build, connections and security settings

Memory allocated *: 256 MB

Timeout *: 60 seconds

Runtime service account

Runtime service account: ngeneahub-function

Auto-scaling

Minimum number of instances: 0

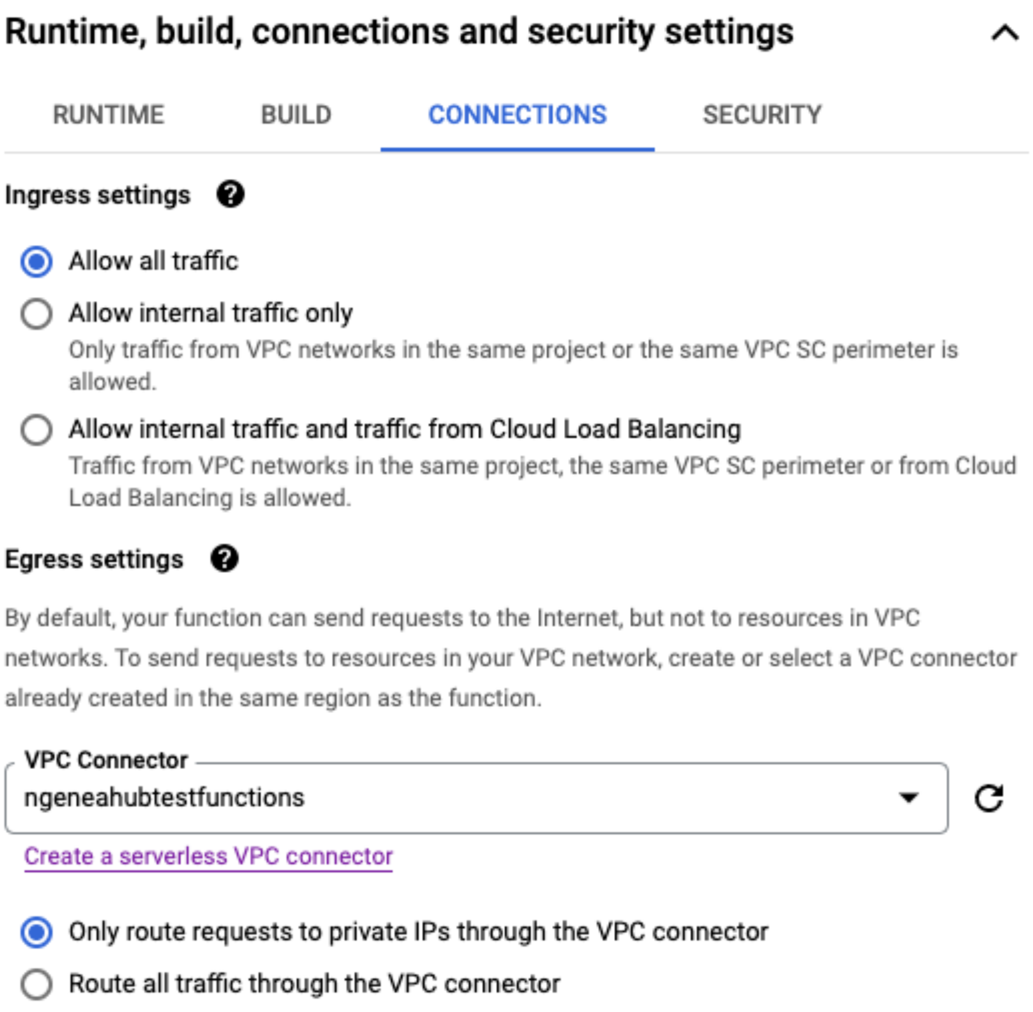
Maximum number of instances: 3000

If the Ngenea Hub doesn't have an external IP to connect to, you'll need a **VPC Connector** for the function to be able to access the Ngenea Hub private IP.

The creation of the **VPC Connector** is out of scope of these docs.

To select an existing **VPC Connector**, under the **RUNTIME, BUILD AND CONNECTIONS SETTINGS** section, select **Connections**.

From the VPC Connector drop down menu, select an existing connector and check the **Only route requests to private IPs through the VPC connector** radiobox.


Runtime, build, connections and security settings

Ingress settings

- [✓ Allow all traffic]
- Allow internal traffic only
  Only traffic from VPC networks in the same project or the same VPC SC perimeter is allowed.
- Allow internal traffic and traffic from Cloud Load Balancing
  Traffic from VPC networks in the same project, the same VPC SC perimeter or from Cloud Load Balancing is allowed.

Egress settings

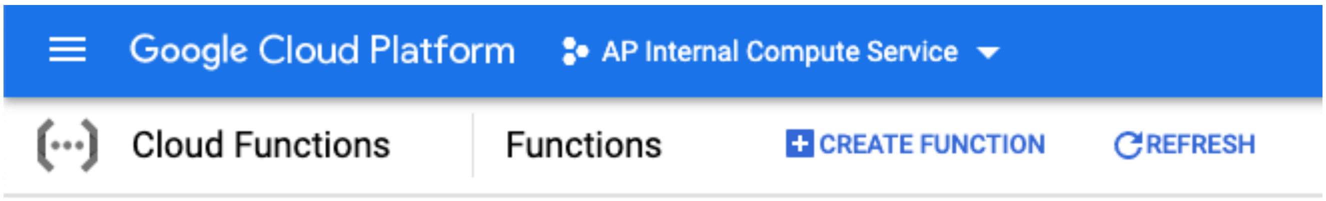
By default, your function can send requests to the Internet, but not to resources in VPC networks. To send requests to resources in your VPC network, create or select a VPC connector already created in the same region as the function.

VPC Connector: ngeneahubtestfunctions

- [✓ Only route requests to private IPs through the VPC connector]
- Route all traffic through the VPC connector

In the Code config section

1. Change the **Runtime** to **Python 3.9**
2. The **Entry Point** is **main**
3. Select **ZIP upload** in the **Source code**
4. Choose the GCP zip previously downloaded from the ../../download page
5. Select a **Stage bucket** for use while deploying. You can use the bucket we'll be monitoring
6. Select **Next** to build the **Cloud Function**

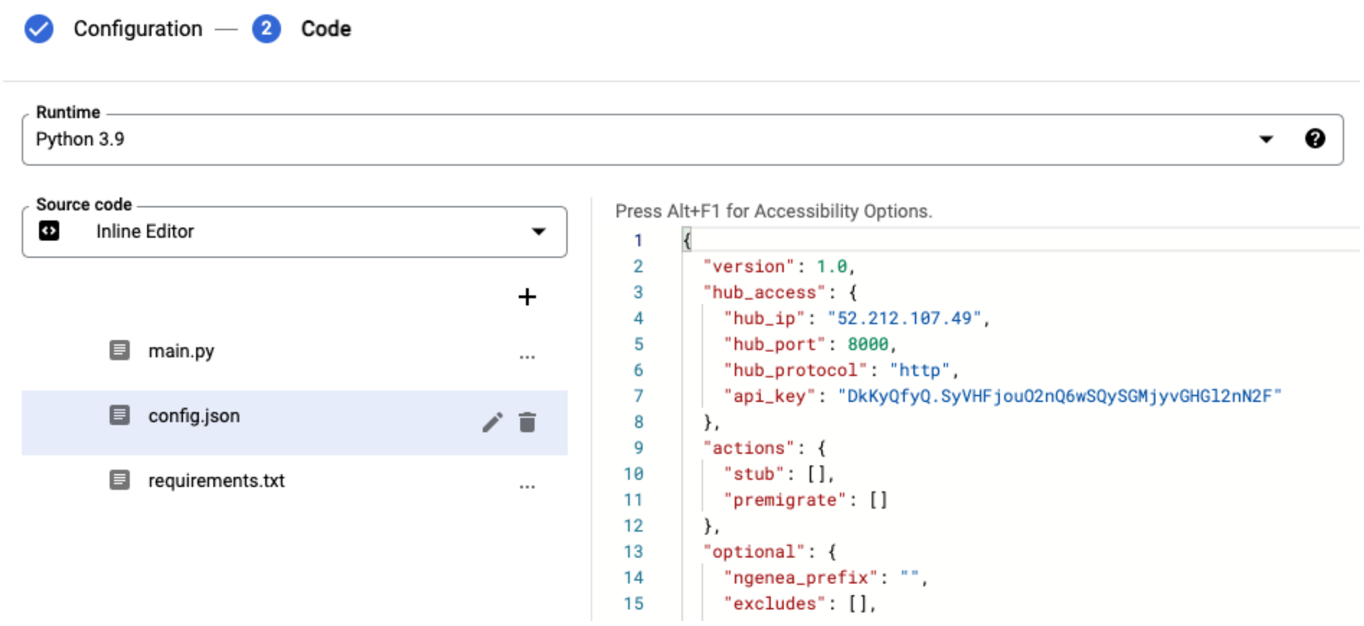


Once built you need to edit the default `config.json` file

Choose your new function and click **EDIT**

Select Next to get to the code edit section

Select the `config.json` file to edit



```
{
  "version": 1.0,
  "hub_access": {
    "hub_ip": "52.212.107.49",
    "hub_port": 8000,
    "hub_protocol": "http",
    "api_key": "DkKyQfyQ.SyVHFjouO2nQ6wSQySGMjyvGHGl2nN2F"
  },
  "actions": {
    "stub": [],
    "premigrate": []
  },
  "optional": {
    "ngenea_prefix": "",
    "excludes": [],
```

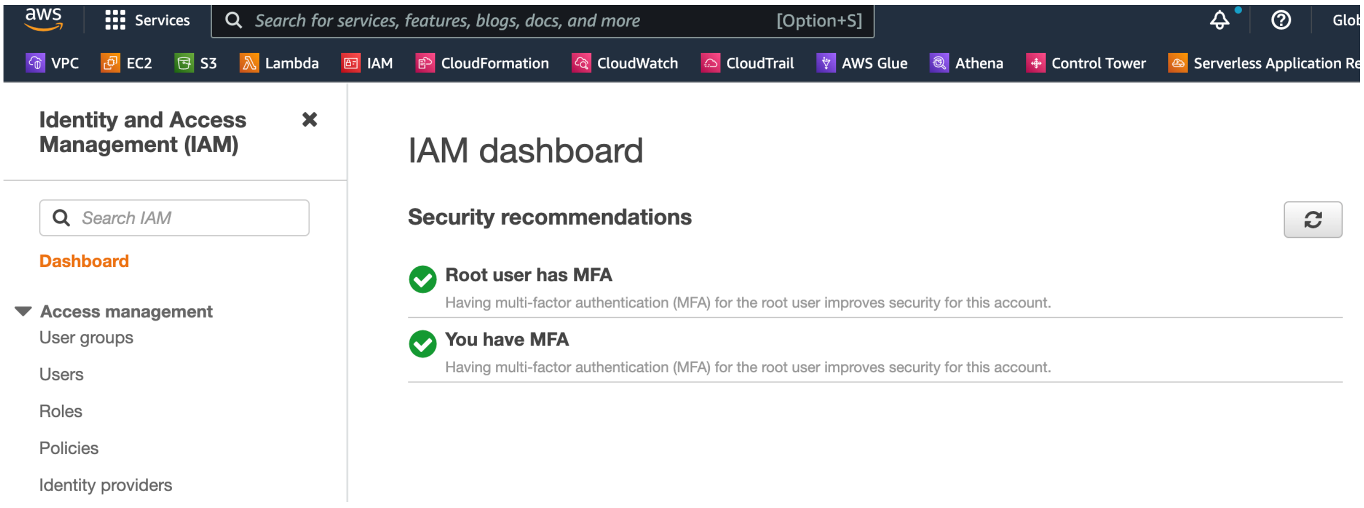
Edit the config file based on the docs from Cloud Functions

Select Deploy to save the changes. This can take 1-2 mins to update

AWS Lambda Function

Create the IAM policy and role

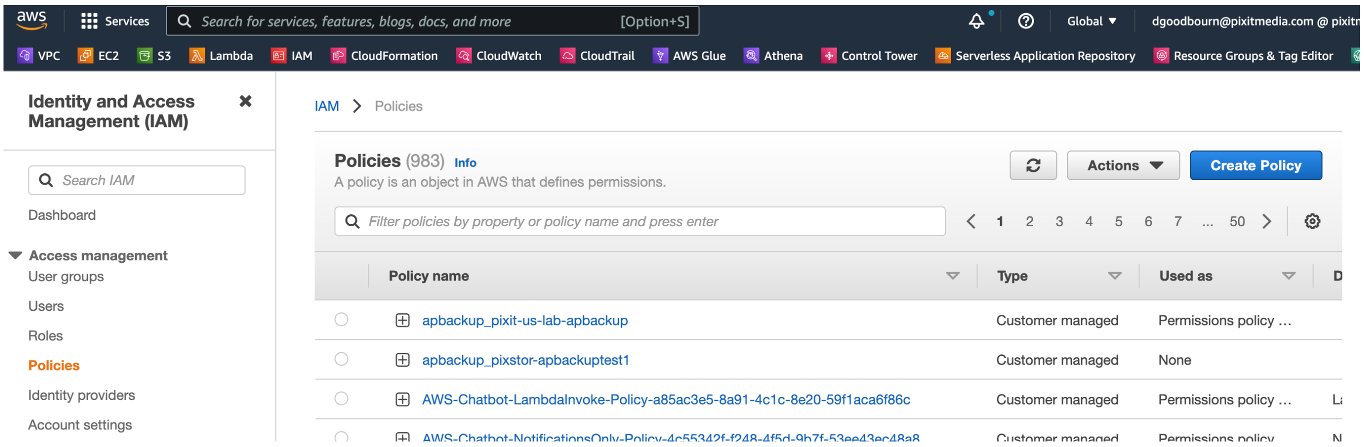
In the AWS console go to the IAM service



Root user has MFA
Having multi-factor authentication (MFA) for the root user improves security for this account.

You have MFA
Having multi-factor authentication (MFA) for the root user improves security for this account.

Select the **Policies** then **Create Policy**


aws
Services
Search for services, features, blogs, docs, and more [Option+S]
Global
dgoodbourn@pixitmedia.com @ pixitr
VPC
EC2
S3
Lambda
IAM
CloudFormation
CloudWatch
CloudTrail
AWS Glue
Athena
Control Tower
Serverless Application Repository
Resource Groups & Tag Editor
Identity and Access Management (IAM)
Search IAM
Dashboard
Access management
User groups
Users
Roles
Policies
Identity providers
Account settings
IAM > Policies
Info
Actions
Create Policy
Filter policies by property or policy name and press enter
1 2 3 4 5 6 7 ... 50

Policies (983)
A policy is an object in AWS that defines permissions.

Policy name	Type	Used as
apbackup_pixit-us-lab-apbackup	Customer managed	Permissions policy ...
apbackup_pixstor-apbackuptest1	Customer managed	None
AWS-Chatbot-LambdaInvoke-Policy-a85ac3e5-8a91-4c1c-8e20-59f1aca6f86c	Customer managed	Permissions policy ...
AWS-Chatbot-NotificationsOnly-Policy-4c55342f-f248-4f5d-9b7f-53ee43ec48a8	Customer managed	Permissions policy

Move to the **JSON** tab and replace the text with the following:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "logs:CreateLogGroup",
            "Resource": "arn:aws:logs:*:<<AWS_ACCOUNT_ID>>:*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:*:<<AWS_ACCOUNT_ID>>:log-group:/aws/
lambda/<<LAMBDA_NAME>>:*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "iam:GetUser",
            "Resource": "*"
        }
    ]
}
```

Replace **<<AWS_ACCOUNT_ID>>** with your account id. This must be the Id without the `-` in the name, i.e. `123456789012`, not `1234-5678-9012`

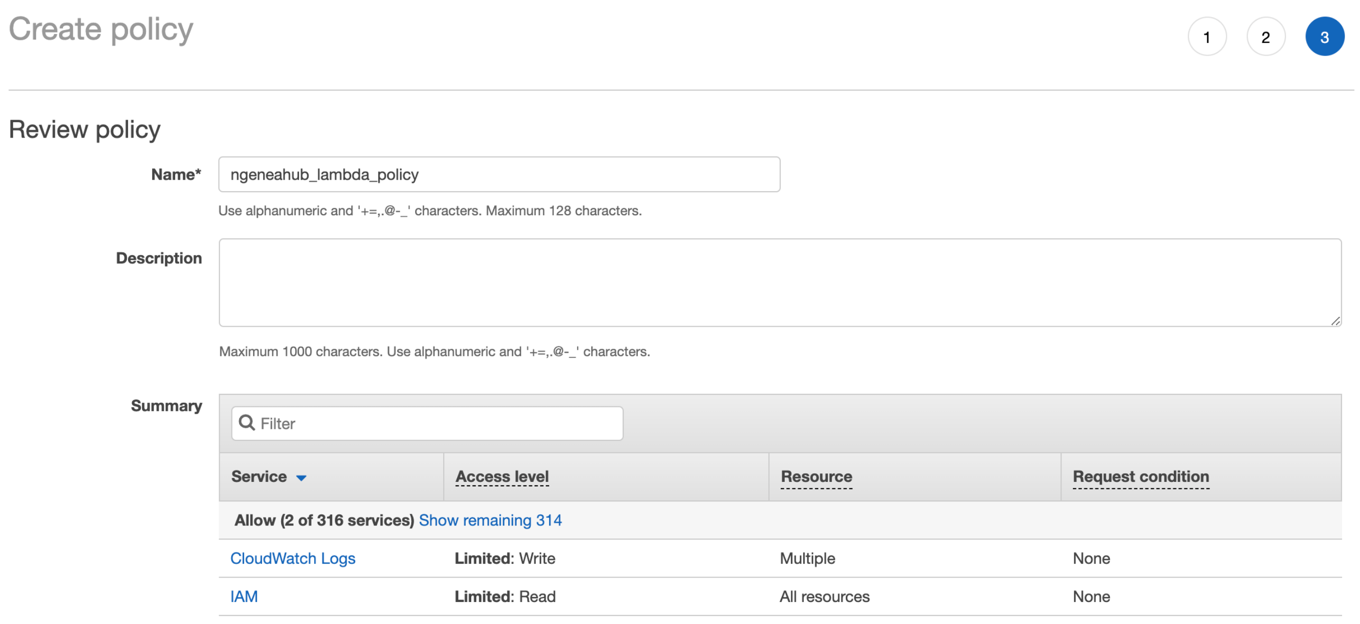
Replace **<<LAMBDA_NAME>>** with the name of your Lambda function

Select **Next**

If you require a tag, add it now, otherwise just select **Review**

Name the policy **ngeneahub_lambda_policy**

Then select **Create Policy**


Create policy

1 2 3

Review policy

Name* ngeneahub_lambda_policy

Use alphanumeric and '+=,.@-_' characters. Maximum 128 characters.

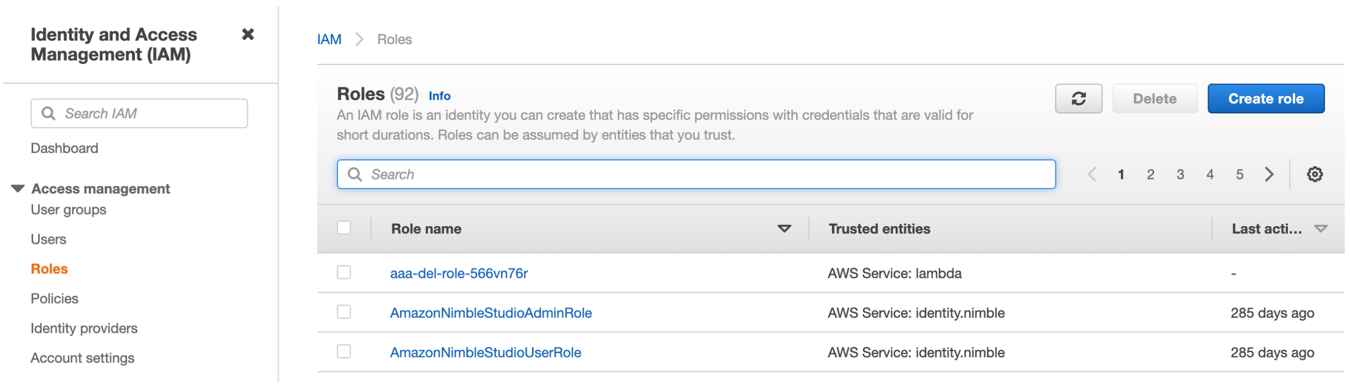
Description

Maximum 1000 characters. Use alphanumeric and '+=,.@-_' characters.

Summary

Service	Access level	Resource	Request condition
Allow (2 of 316 services) Show remaining 314			
CloudWatch Logs	**Limited**: Write	Multiple	None
IAM	**Limited**: Read	All resources	None

Next create a new **Role**


Identity and Access Management (IAM)

Search IAM

Dashboard

Access management
User groups
Users
Roles
Policies
Identity providers
Account settings

IAM > Roles

Roles (92) Info

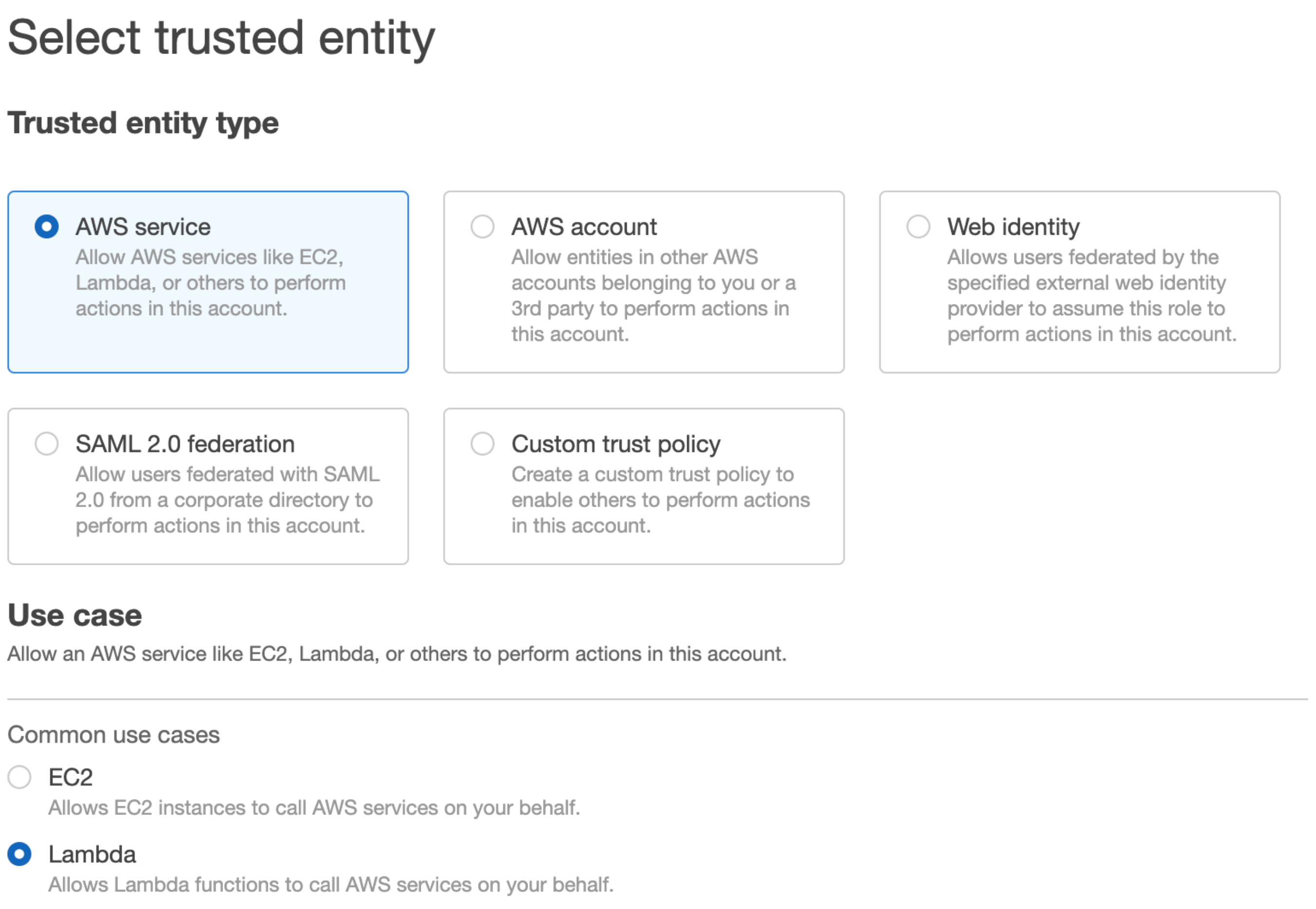
An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.

Delete | Create role

Search

	Role name	Trusted entities	Last acti...
☐	aaa-del-role-566vn76r	AWS Service: lambda	-
☐	AmazonNimbleStudioAdminRole	AWS Service: identity.nimble	285 days ago
☐	AmazonNimbleStudioUserRole	AWS Service: identity.nimble	285 days ago

Select the **AWS service** for the trusted entity and a **Lambda** for the use case


Select trusted entity

Trusted entity type

- [✓ AWS service] Allow AWS services like EC2, Lambda, or others to perform actions in this account.
- AWS account: Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.
- Web identity: Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.
- SAML 2.0 federation: Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.
- Custom trust policy: Create a custom trust policy to enable others to perform actions in this account.

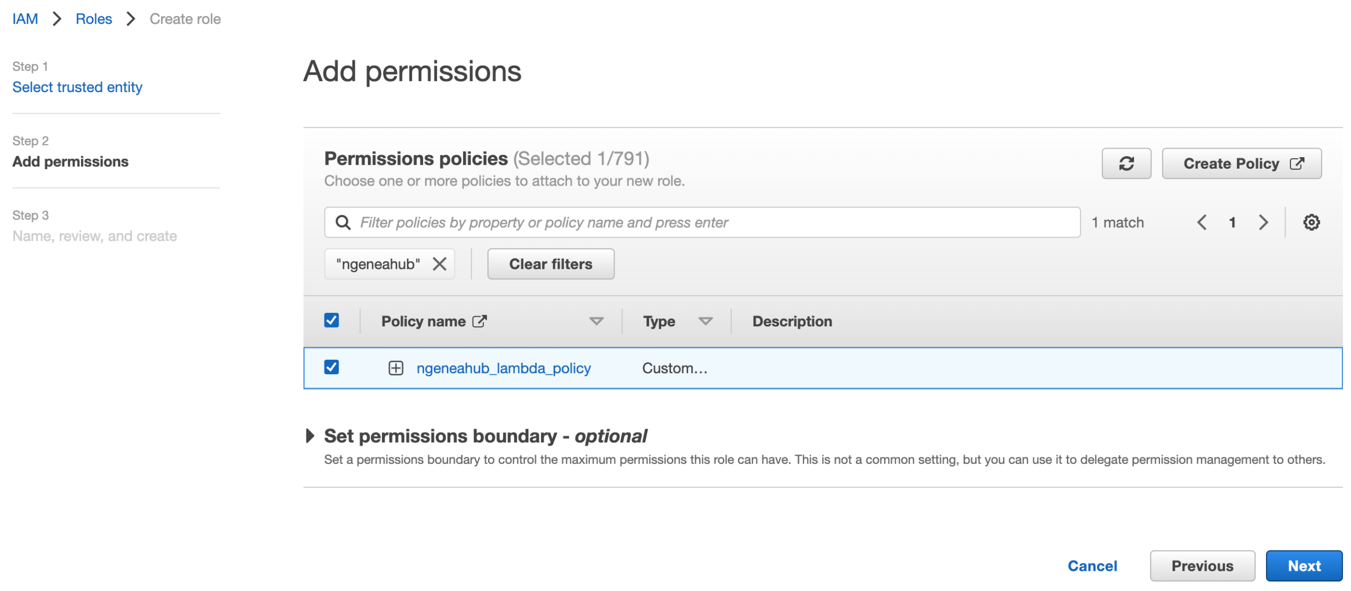
Use case

Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

Common use cases

- EC2: Allows EC2 instances to call AWS services on your behalf.
- [✓ Lambda] Allows Lambda functions to call AWS services on your behalf.

Search for the policy we just created above and select it, to attach the policy to this role. Then select **Next**


IAM > Roles > Create role

Step 1 Select trusted entity
Step 2 **Add permissions**
Step 3 Name, review, and create

Add permissions

Permissions policies (Selected 1/791)
Choose one or more policies to attach to your new role.

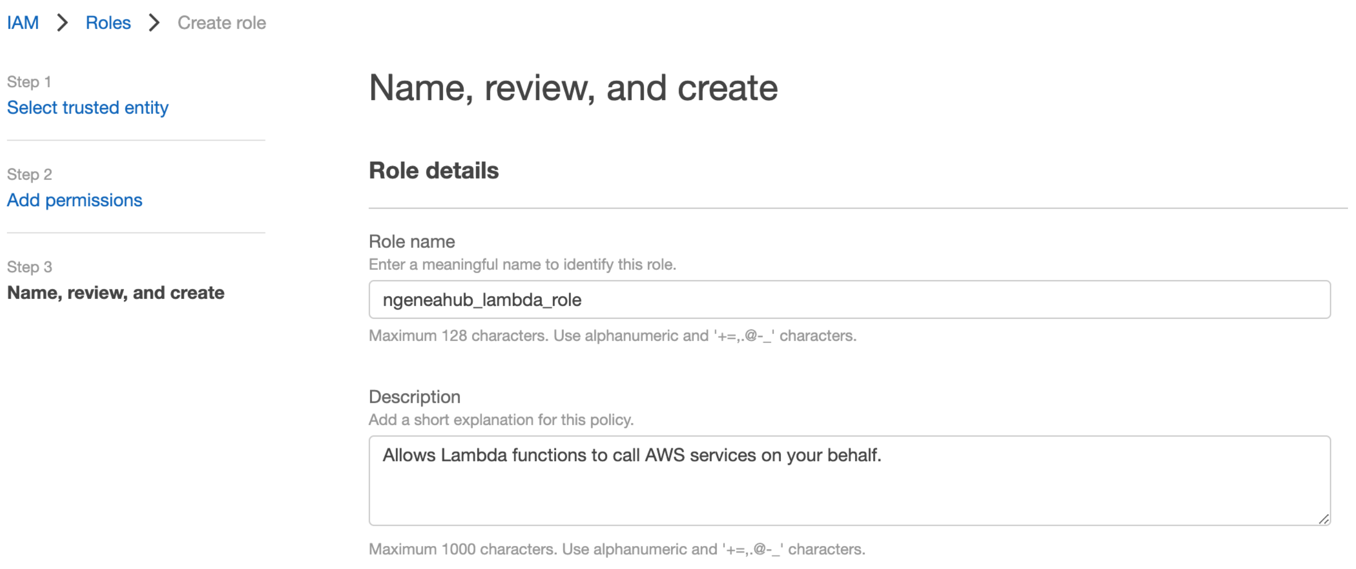
Filter policies by property or policy name and press enter — 1 match

"ngeneahub" — Clear filters

✓	Policy name	Type	Description
✓	ngeneahub_lambda_policy	Custom...	

Set permissions boundary - *optional*
Set a permissions boundary to control the maximum permissions this role can have. This is not a common setting, but you can use it to delegate permission management to others.

Name the role **ngeneahub_lambda_role** and select **Create Role**


IAM > Roles > Create role

Step 1
Select trusted entity

Step 2
Add permissions

Step 3
Name, review, and create

Name, review, and create

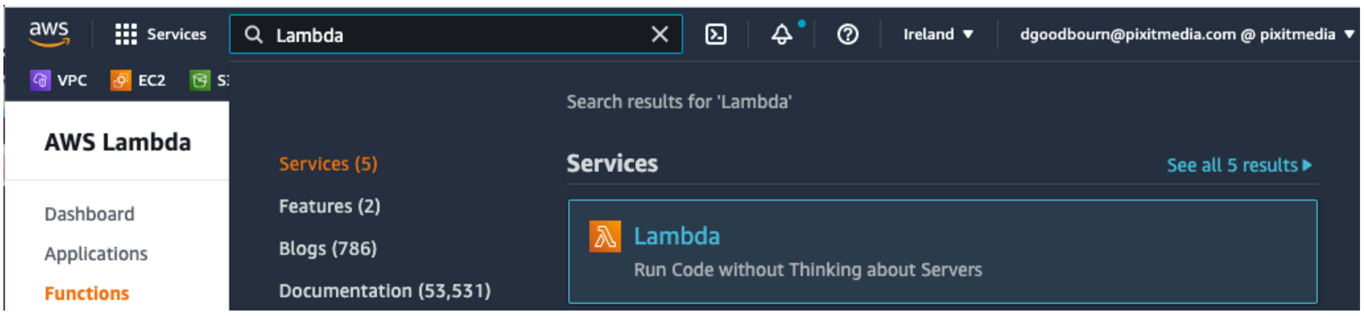
Role details

Role name
Enter a meaningful name to identify this role.
ngeneahub_lambda_role
Maximum 128 characters. Use alphanumeric and '+=,.@-_' characters.

Description
Add a short explanation for this policy.
Allows Lambda functions to call AWS services on your behalf.
Maximum 1000 characters. Use alphanumeric and '+=,.@-_' characters.

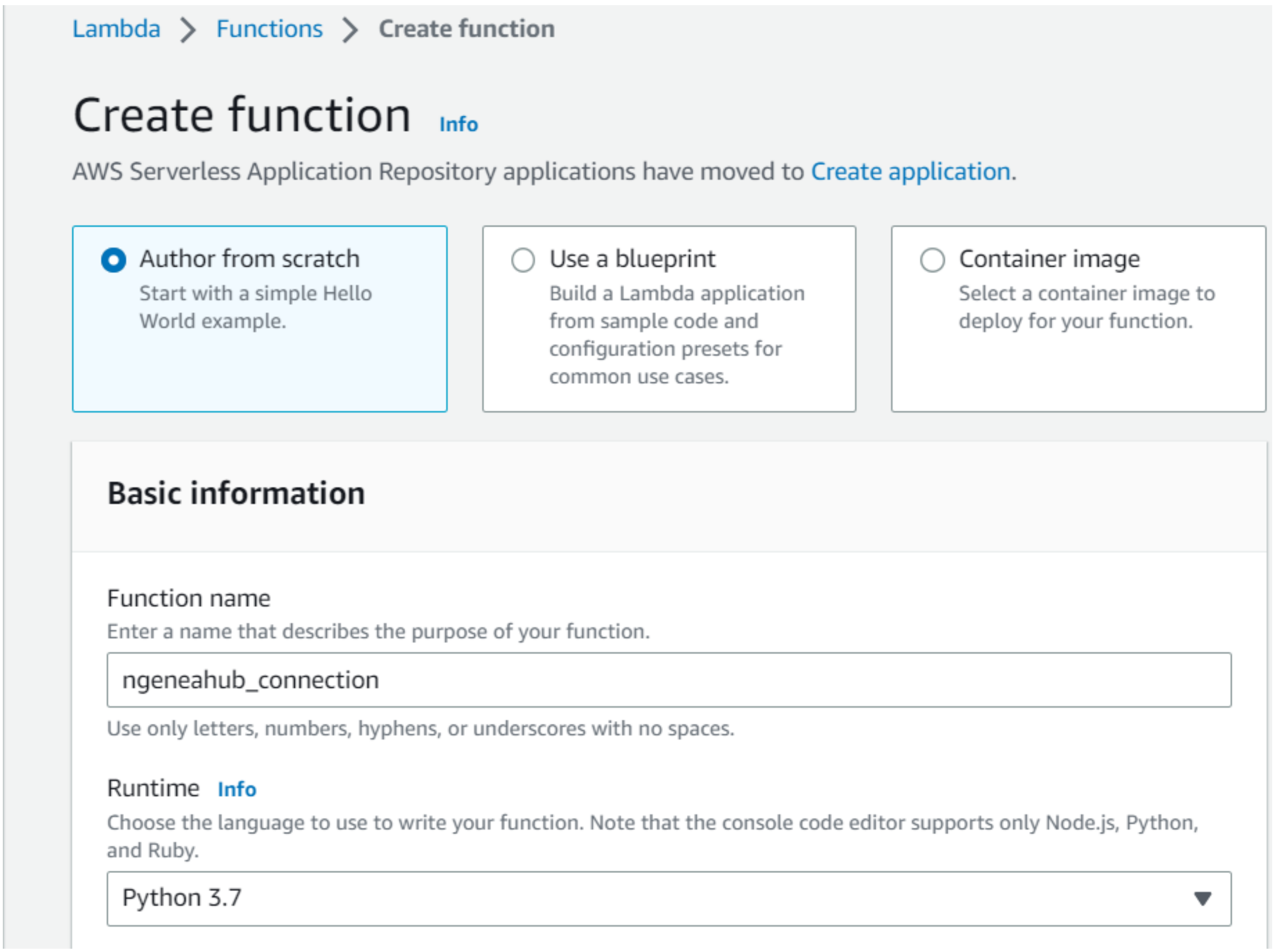
Create the Lambda function

In the AWS console go to the **Lambda** service



Click the **Create function** button

- Create a new function with the **Author from scratch** option selected
- Name the function
- And select **Python 3.7** from the **Runtime** list


Lambda > Functions > Create function

Create function Info

AWS Serverless Application Repository applications have moved to Create application.

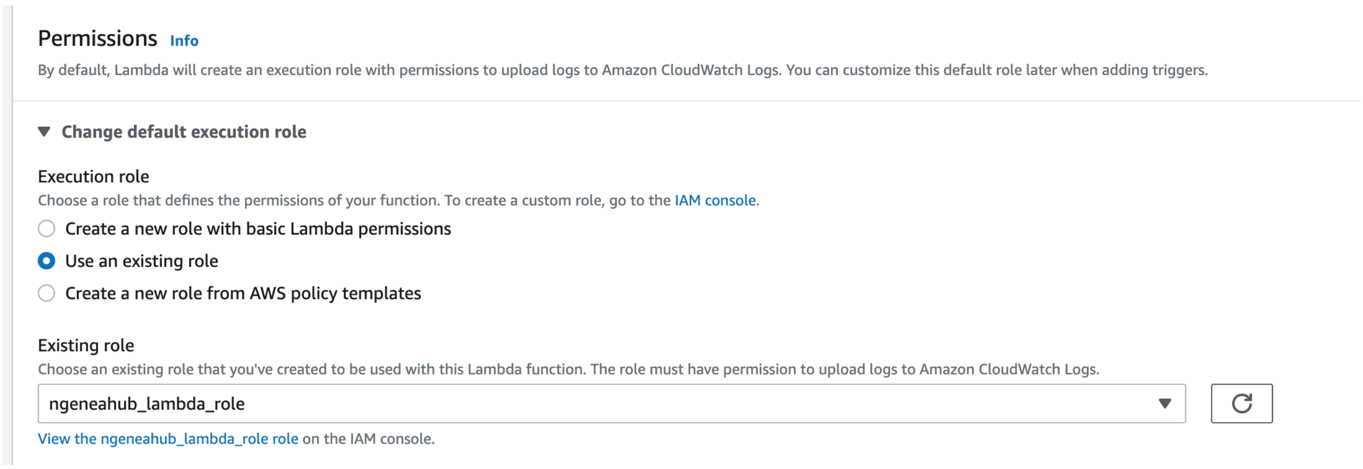
- [✓ Author from scratch] Start with a simple Hello World example.
- Use a blueprint: Build a Lambda application from sample code and configuration presets for common use cases.
- Container image: Select a container image to deploy for your function.

Basic information

Function name
Enter a name that describes the purpose of your function.
ngeneahub_connection
Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime Info
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Python 3.7

Under **Change default execution role**, select **Use an existing role** and find the role we created above


Permissions Info
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

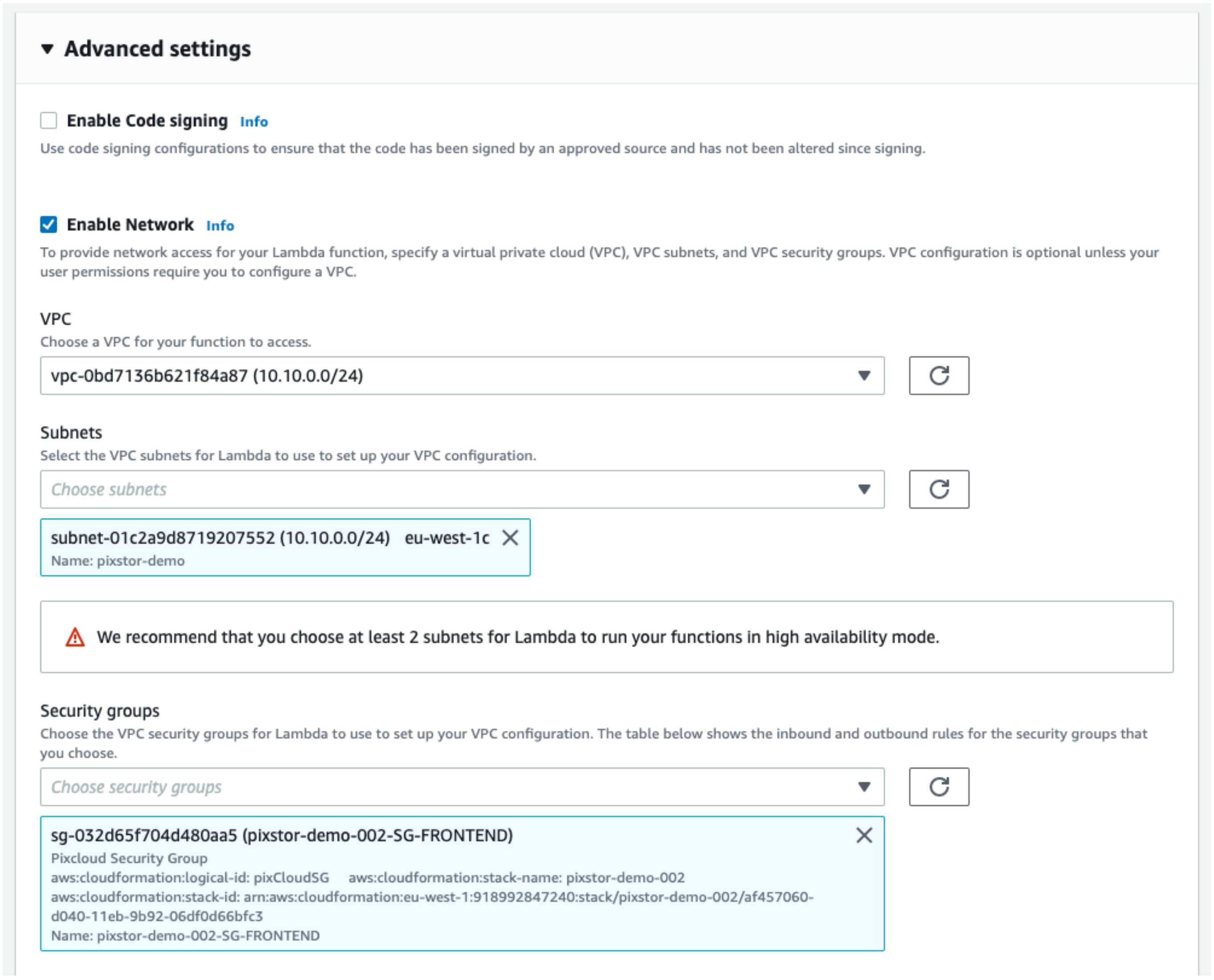
▼ Change default execution role

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.
- Create a new role with basic Lambda permissions
- [✓ Use an existing role]
- Create a new role from AWS policy templates

Existing role
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.
ngeneahub_lambda_role
View the ngeneahub_lambda_role role on the IAM console.

If your Ngenea Hub doesn’t have an external IP to connect to, you’ll need to select the **Enable Network** option under the **Advanced settings**

- Select the **VPC** the Ngenea Hub is located in
- Choose the **Subnet** it’s in
- And select a **Security group** that gives access to the Ngenea Hub over port 8000


▼ Advanced settings

Enable Code signing Info

Use code signing configurations to ensure that the code has been signed by an approved source and has not been altered since signing.

[✓] Enable Network Info

To provide network access for your Lambda function, specify a virtual private cloud (VPC), VPC subnets, and VPC security groups. VPC configuration is optional unless your user permissions require you to configure a VPC.

VPC

Choose a VPC for your function to access.

vpc-0bd7136b621f84a87 (10.10.0.0/24)

Subnets

Select the VPC subnets for Lambda to use to set up your VPC configuration.

Choose subnets

subnet-01c2a9d8719207552 (10.10.0.0/24) eu-west-1c

Name: pixstor-demo

We recommend that you choose at least 2 subnets for Lambda to run your functions in high availability mode.

Security groups

Choose the VPC security groups for Lambda to use to set up your VPC configuration. The table below shows the inbound and outbound rules for the security groups that you choose.

Choose security groups

sg-032d65f704d480aa5 (pixstor-demo-002-SG-FRONTEND)

Pixcloud Security Group

aws:cloudformation:logical-id: pixCloudSG aws:cloudformation:stack-name: pixstor-demo-002

aws:cloudformation:stack-id: arn:aws:cloudformation:eu-west-1:918992847240:stack/pixstor-demo-002/af457060-d040-11eb-9b92-06df0d66bfc3

Name: pixstor-demo-002-SG-FRONTEND

A policy will also have to be created to permit the function to traverse the VPC's networking, this can be done by adjusting the previously created IAM policy:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "logs:CreateLogGroup",
            "Resource": "arn:aws:logs:*:<<AWS_ACCOUNT_ID>>:*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:*:<<AWS_ACCOUNT_ID>>:log-group:/aws/
lambda/<<LAMBDA_NAME>>:*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "iam:GetUser",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetUser",
                "ec2:DescribeNetworkInterfaces",
                "ec2:CreateNetworkInterface",
                "ec2:DeleteNetworkInterface"
            ],
            "Resource": "*"
        }
    ]
}
```

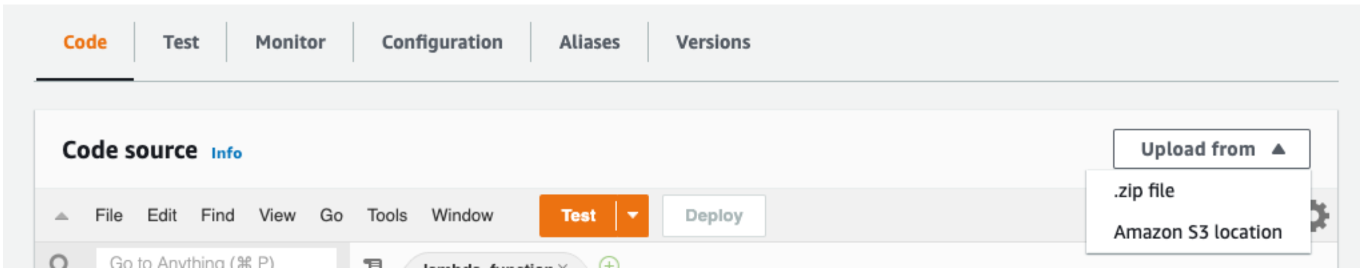
By adding this additional section, the function can traverse correctly.

With this configured, we can now create out function by selecting **Create function**

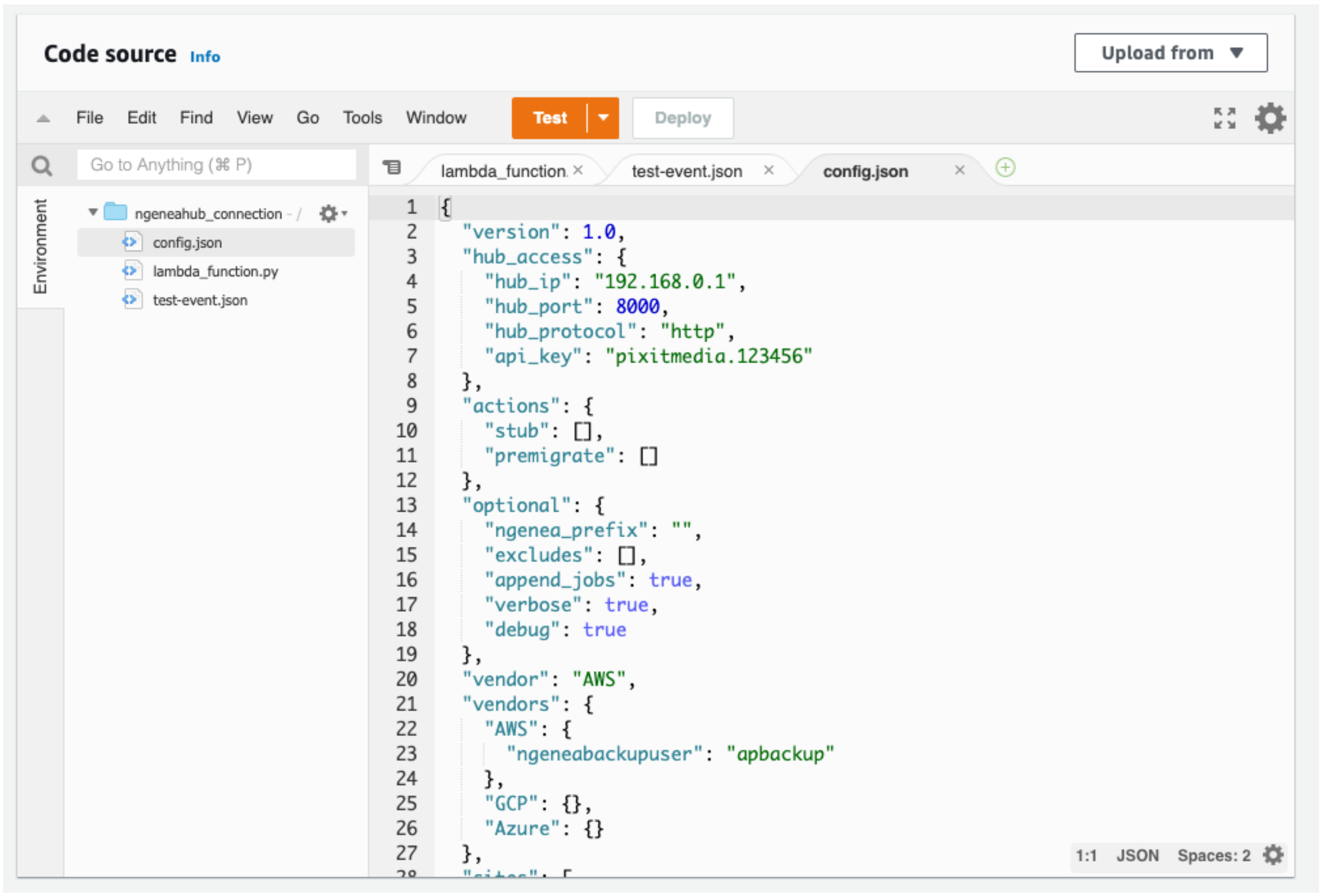
Once the function is created - This can take a few minutes if you've chosen to create the **Network** connection.

Under the **Code** tab, you'll need to upload the zip file containing all the code

Choose the AWS zip previously downloaded from the ../../download page



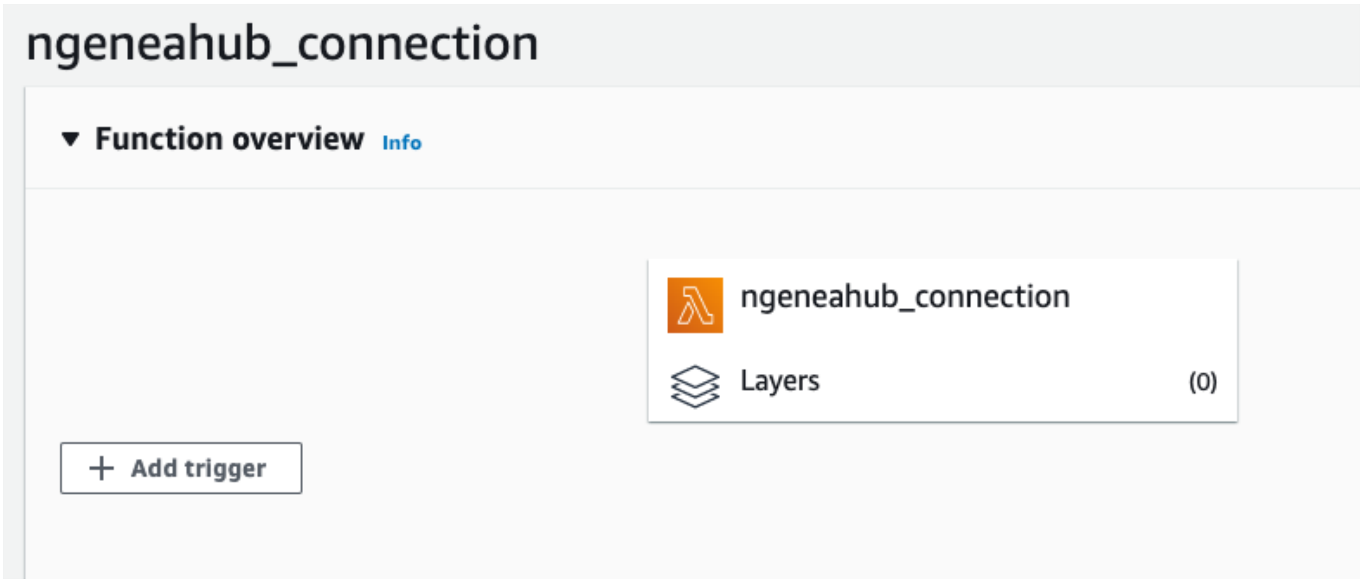
Once the code is uploaded, you need to edit the `config.json` file with all the relevant details based on the docs from Cloud Functions



```
{
  "version": 1.0,
  "hub_access": {
    "hub_ip": "192.168.0.1",
    "hub_port": 8000,
    "hub_protocol": "http",
    "api_key": "pixitmedia.123456"
  },
  "actions": {
    "stub": [],
    "premigrate": []
  },
  "optional": {
    "ngenea_prefix": "",
    "excludes": [],
    "append_jobs": true,
    "verbose": true,
    "debug": true
  },
  "vendor": "AWS",
  "vendors": {
    "AWS": {
      "ngeneabackupuser": "apbackup"
    },
    "GCP": {},
    "Azure": {}
  },
```

Once you've updated the `config.json` file, save it by selecting the **Deploy** button

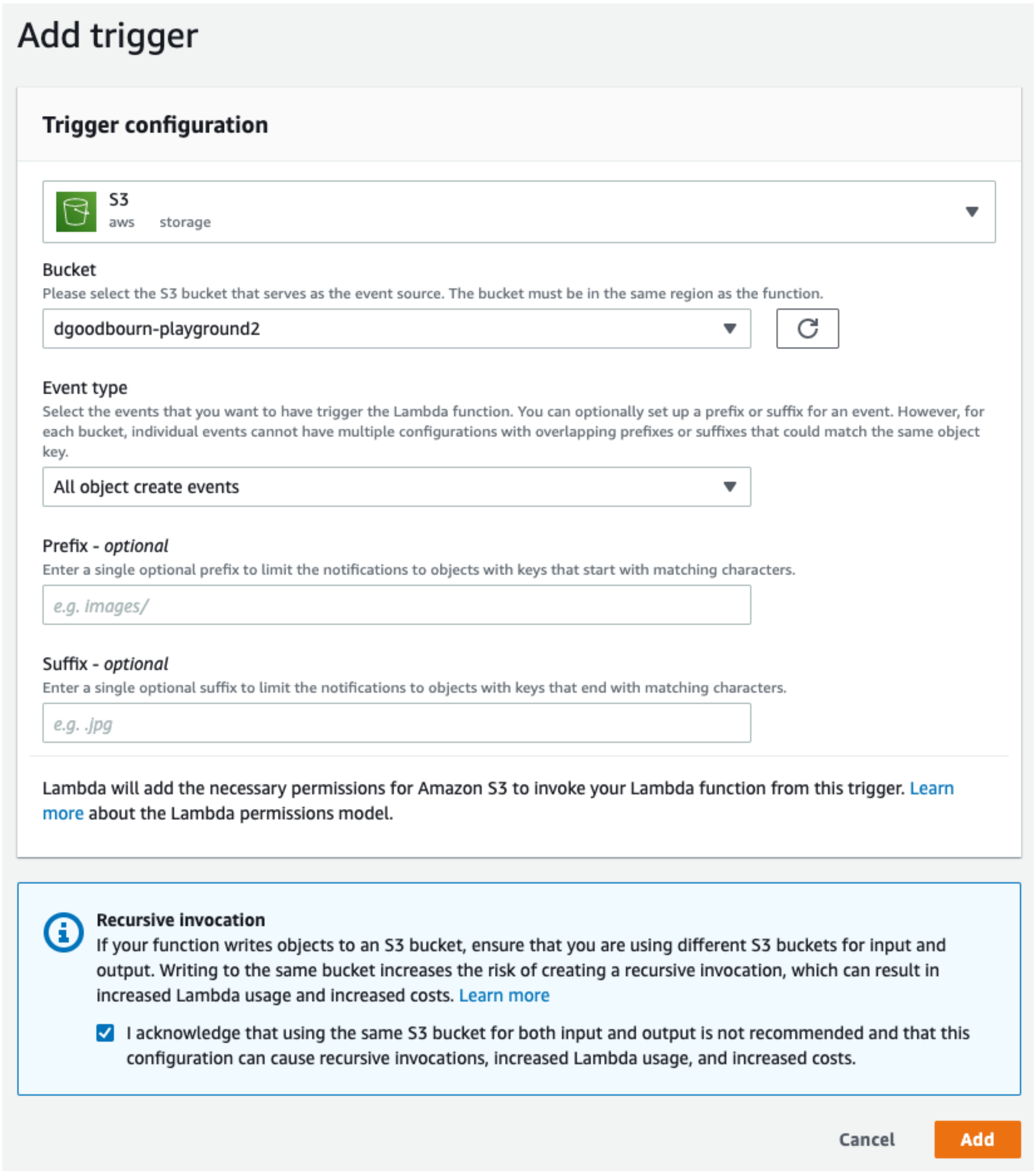
Now the Lambda function is created, we need to assign it to a bucket event. Select **Add trigger** under the **Function overview**



Choose:

- **S3** as the **Trigger**
- Select your bucket
- **All object create events** for the **Event type**
- Check the **Recursive Invocation** checkbox warning

Then just press **Add**


Add trigger

Trigger configuration

S3
aws storage

Bucket
Please select the S3 bucket that serves as the event source. The bucket must be in the same region as the function.
dgoodbourn-playground2

Event type
Select the events that you want to have trigger the Lambda function. You can optionally set up a prefix or suffix for an event. However, for each bucket, individual events cannot have multiple configurations with overlapping prefixes or suffixes that could match the same object key.
All object create events

Prefix - *optional*
Enter a single optional prefix to limit the notifications to objects with keys that start with matching characters.
e.g. images/

Suffix - *optional*
Enter a single optional suffix to limit the notifications to objects with keys that end with matching characters.
e.g. .jpg

Lambda will add the necessary permissions for Amazon S3 to invoke your Lambda function from this trigger. Learn more about the Lambda permissions model.

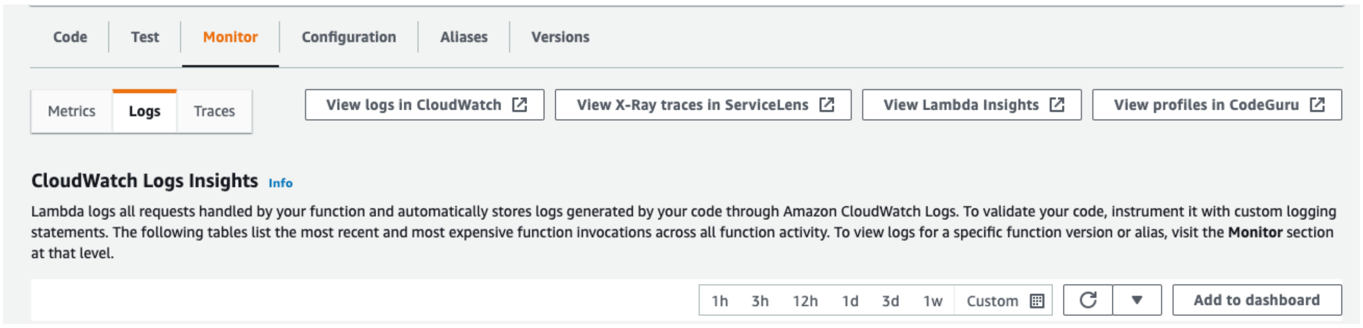
Recursive invocation
If your function writes objects to an S3 bucket, ensure that you are using different S3 buckets for input and output. Writing to the same bucket increases the risk of creating a recursive invocation, which can result in increased Lambda usage and increased costs. Learn more

✓ I acknowledge that using the same S3 bucket for both input and output is not recommended and that this configuration can cause recursive invocations, increased Lambda usage, and increased costs.

You now have a trigger assigned to the Lambda function

To monitor the Lambda function, under the **Monitor** tab you can see all different types of monitoring

Using the **View logs in CloudWatch** is the best place to see any out from the Lambda function



CloudWatch Logs Insights Info

Lambda logs all requests handled by your function and automatically stores logs generated by your code through Amazon CloudWatch Logs. To validate your code, instrument it with custom logging statements. The following tables list the most recent and most expensive function invocations across all function activity. To view logs for a specific function version or alias, visit the **Monitor** section at that level.

Coming soon:

- Azure

Configuration

The configuration file is in JSON format.

version

Indicates the config format version. Currently, only `1.0` is supported.

hub_access

Defines setting for interacting with Ngenea Hub

- **hub_ip**: IP addresss of the Ngenea Hub REST API to use
- **hub_port**: port for the Ngenea Hub REST API, typically `8000`
- **hub_protocol**: either `http` or `https`
- **api_key**: API 'client key' to authenticate Ngenea Hub with
- **workflow**: name of the workflow to submit for the event file, e.g. `reverse_stub`
- **workflow_flags**: mapping of settings to pass to the workflow, e.g. `{"hydrate": true}`

For new files, one would typically use **workflow** `reverse_stub` or `recall`, which ship with Ngenea Hub by default. There is no default 'delete' workflow in Ngenea Hub, so one must be created - e.g.

```
{
  "name": "delete_file",
  "label": "delete_file",
  "icon_classes": [
    "fa fa-cloud fa-stack-2x text-success",
    "fa fa-angle-up fa-stack-2x text-light"
  ],
  "discovery": null,
  "enabled": true,
  "visible": true,
  "fields": [],
  "filter_rules": [
    {
      "type": "all",
      "state": "all",
      "action": [
        {
          "name": "dynamo.tasks.delete_paths_from_gpfs",
          "recursive": false
        }
      ],
      "description": "Delete the file at a given path"
    }
  ]
}
```

sites

List of sites to reflect the events to.

- **site**: name of the site, as registered in Ngenea Hub
- **default**: (optional) default mode for 'recall' type workflows. One of `stub`, `premigrate`
- **skip_from_ngenea**: (optional) if True then any file that was created via ngenea will be skipped for this site

default is used if the event path doesn't match any **action** (see below), and if `hydrate` isn't explicitly set in **workflow_flags** (see above)

skip_from_ngenea is useful when e.g. Ngenea Hub is being used to sync files between sites. Files which are being transmitted via the cloud bucket don't need to be automatically recalled onto either site. On the other hand, files uploaded directly to the cloud still need to be recalled.

For GCP, we may not be able to determine the source of a delete event. In that case, deletes will always be reflected to all sites.

actions

Mapping of actions -- `stub` or `premigrate` -- to path prefixes.

This is used to determine whether a file should be hydrated by a 'recall' type workflow. If a path matches multiple actions, the longest match wins. For example, using

```
{
    "stub": ["data"],
    "premigrate": ["data/cats"]
}
```

the path `data/cats/cat-01.jpg` would be premigrated, while `data/cats-02.jpg` would be stubbed.

If not specified, the site-specific **default** (see above) will be used, unless missing or unless `hydrate` is explicitly set in **workflow_flags**. If none of these settings are configured, the default behaviour is to stub.

optional

Optional settings

- **ngenea_prefix**: prefix which maps a cloud path to a local path, e.g. with prefix `/mmfs1` the cloud path `data/cats-01.jpg` is mapped to `/mfs1/data/cats-01.jpg`. Default: `''`
- **excludes**: list of strings used to exclude paths. The strings are treated as sub-strings which can match anywhere in the (cloud) path string. Default: `[]`
- **append_jobs**: if `true`, tasks will be grouped under the same job id (per hour). If `false`, each task will get its own job. Default: `false`
- **verbose**: set logging output to `info` level. Default: `false`
- **debug**: set logging output to `debug` level. Takes precedence over **verbose**. Default: `false`

vendor

The vendor that the function is being run on.

Currently supported values: `AWS`, `GCP`

vendors

Vendor specific settings. Currently only used by `AWS`

AWS

- **ngeneabackupuser**: name of the user ngenea uses for AWS. Used to identify whether a file came from ngenea for **skip_from_ngenea**

Complete Example

```
{
    "version": 1.0,
    "hub_access": {
        "hub_ip": "192.168.0.1",
        "hub_port": 8000,
        "hub_protocol": "http",
        "api_key": "pixitmedia.123456",
        "workflow": "reverse_stub",
        "workflow_flags": {
            "hydrate": false,
            "overwrite": true
        }
    },
    "sites": [
        {
            "site": "uk",
            "default": "stub"
        }
    ],
    "actions": {
        "stub": [],
        "premigrate": []
    },
    "optional": {
        "ngenea_prefix": "",
        "excludes": [],
        "append_jobs": true,
        "verbose": true,
        "debug": true
    },
    "vendor": "AWS",
    "vendors": {
        "AWS": {
            "ngeneabackupuser": ""
        },
        "GCP": {},
        "Azure": {}
    }
}
```

Ngenea Worker Plugins

Warning: This feature is currently in alpha

When constructing workflows to run on Ngenea Hub, there may not be a task that covers the behaviour required for a given workflow. To alleviate this, there is now support for custom plugin tasks to be added to any instance of Ngenea Worker that can cover that missing behaviour.

These are defined as custom tasks for the celery https://docs.celeryq.dev/en/stable/ task distribution framework.

Creating your plugin

Warning: This feature is currently in alpha.

A plugin for the hub is a python module that contains one or multiple celery tasks that can be discovered by the Ngenea Worker, these tasks can then be used in workflows in Ngenea Hub.

Example plugin creation

Using the plugin scripts located at `ngenea-worker plugins` a template plugin can be created in the plugin directory at `/var/lib/ngenea-worker/plugins`. All plugins will persist in this location and will need to be in this directory to be correctly installed.

For the following example, we will be creating a plugin named `echo_args` that will return the expected key word arguments provided to the task itself.

To begin, create a plugin template for the example `echo_args` using:

```
ngenea-worker plugins create echo_args
```

This will create the template plugin project at `/var/lib/ngenea-worker/plugins/echo_args`. To implement custom logic, navigate to `/var/lib/ngenea-worker/plugins/echo_args` in which contains the python module file `echo_args/echo_args.py`. Within this file there is the basic example task `example_task`:

```
@shared_task(bind=True, name="dynamo.custom.example_task")
def example_task(self, *args, paths: List = None, jobid: int = None,
**kwargs):
    print(paths)
    print(jobid)
    return {"status": "success"}
```

The logic within this task can be replaced along with the name of the function to our echo task:

Note: The name provided to `shared_task` will be the task name that is used to call the task within a workflow.

```
@shared_task(bind=True, name="dynamo.custom.echo_args")
def echo_args(self, *args, paths: List = None, jobid: int = None,
**kwargs):
    return {"paths": paths, "jobid": jobid, **kwargs}
```

With the new task in place, the entry point for the plugin will have to be adjusted in `setup.py`.

Note: Ensure that all custom task entry points are defined under `worker_plugin`

Initial `setup.py` entry points:

```
...
    "worker_plugin": [
        "echo_args=echo_args.echo_args:example_task",
    ]
...
```

In this case, the changes are as follows:

```
...
    "worker_plugin": [
        "echo_args=echo_args.echo_args:echo_args",
    ]
...
```

External Dependencies

If the logic in the plugin requires an external dependency, it can be installed into the worker using the `setup.py` within the template plugin generated via `ngenea-worker plugins create`. To add additional dependencies, they can be added to the `install_requires` section:

Note: Ensure that the celery package and version is un-edited within the plugin requirements, otherwise it may have unexpected effects on the functionality of Ngenea Worker.

```
install_requires=[
    "celery>=5.0.3",
    "example-module==0.1.0"
]
```

Installing custom plugins

To install the newly created plugin refer to the plugin installation page.

Managing plugins

Warning: This feature is currently in alpha.

After plugins have been created, they can be managed using the following scripts.

Installing your plugin

All the plugins that have been created within `/var/lib/ngenea-/worker/plugins` can be installed through running:

```
ngenea-worker plugins install
```

This will install all of the plugin packages within the plugins directory. These packages will not update within the Ngenea Worker unless it has had a version number increase in the `setup.py`.

Single plugins can also be installed by providing the name of the package explicitly:

```
ngenea-worker plugins install PACKAGE_NAME
```

After installing the desired plugins, the Ngenea Worker service will need to be restarted:

```
systemctl restart ngenea-worker
```

Uninstalling your plugin

Uninstalling any of the plugins can be done through the uninstall script:

Warning: This can remove any modules within the environment that Ngenea Worker runs within, use caution when uninstalling any modules that aren't directly within `/var/lib/ngenea-worker/plugins` as it could cause Ngenea Worker to not function correctly.

```
ngenea-worker plugins uninstall <PACKAGE_NAME>
```

To uninstall all of the modules in the plugin directory it can be done through:

```
pushd /var/lib/ngenea-worker/plugins/
ngenea-worker plugins uninstall * -y
popd
```

Custom Tasks

Warning: This feature is currently in alpha

In addition to the predefined tasks, Ngenea Worker plugins allow custom tasks to be defined.

Once created, custom tasks can be included in Custom Workflows.

Custom tasks must accept the same standard arguments, and return a payload in the same format, as predefined tasks do.

The Structure of a Task

Arguments

Tasks must accept the following keyword arguments:

name	description	example	default
`jobid`	ID of the job the task is associated with.	`100`	`None`
`paths`	List of paths (may be files, folders or symlinks) to be processed. Each path is given as a dict with a "path" key, and can also have other keys	`[{"path": "/mmfs1/data/my-fileset/file1", "size": 264321}, {"path": "/mmfs1/data/my-fileset/file2", "size": 15}]`	`None`

name	description	example	default
	such as "size" and "message".		

To pass specific keyword arguments to a task, include these in the workflow definition as hardcoded values or Runtime fields.

In addition, tasks should accept arbitrary other keyword arguments via a `**kwargs` parameter, but do not need to process them.

Here is an example function signature, from the definition of the predefined task `dynamo.tasks.migrate`:

```
def migrate(self, *args, jobid=None, paths=None,
skip_modified_during_migrate=False, **kwargs)
```

Return Payload

Ngenea Hub expects task results to be returned in a specific format.

There is a class `FileActionStatus` that the worker code uses as a convenience for constructing the return payload. This may be used as an alternative to constructing the return payload from scratch.

Using the FileActionStatus class

The class is imported like this:

```
from arcapix.dynamo.server.status import FileActionStatus
```

A status object is instantiated like this:

```
status = FileActionStatus(taskname, input_paths, jobid, queue_name)
```

where `input_paths` is the list of path objects as passed to the task in the `paths` parameter, and `queue_name` is `"<site>-custom"` for worker plugins.

For each path operated on by the task, call the `.add(key, path)` method on the status object, where `key` is the state of the file:

```
status.add("processed", "/mmfs1/data/my-fileset/file1")
```

The status object keeps track of files as they are added. At the completion of the task, return the results like this:

```
return status.dump()
```

Constructing a return payload from scratch

Tasks return a dict containing at least these 3 keys:

```
{
    "jobid": <job_id>,
    "paths": <list of path objects that were processed or skipped>,
    "status": {
        "task": <task_name>,
        "input_paths": <list of path objects input to the task>,
        "input_total": <number of paths input to the task>,
        "summary": <dict giving number of files keyed by state>,
        "details": <dict giving list of file objects keyed by state>,
        "started": <timezone-aware datetime for when the task
started>
    }
}
```

The `paths` key lists the paths to be passed to the next task in the workflow.

Here is an example payload:

```
{
    "jobid": 100,
    "paths": [{"path": "/mmfs1/data/my-fileset/file1"}],
    "status": {
        "task": "my-plugin",
        "input_paths": [{"path": "/mmfs1/data/my-fileset/file1"},
{"path": "/mmfs1/data/my-fileset/file2"}],
        "input_total": 2,
        "summary": {
            "failures": 1,
            "processed": 1,
            "skipped": 0
        },
        "details": {
            "failures": [{"path": "/mmfs1/data/my-fileset/file2",
"message": ["Reason xyz for the failure"]}],
            "processed": [{"path": "/mmfs1/data/my-fileset/file1"}],
            "skipped": []
        },
        "started": "2023-04-25T14:56:31.253043",
    }
}
```

Supported file states

These are the supported file states for tasks that perform operations on files:

- "processed": file was successfully processed
- "skipped": file was not processed because they were already in the desired state
- "aborted": file was not processed and should not be processed by subsequent tasks in the workflow
- "failures": file could not be processed because of an error
- "inprogress": file is still being processed (this state is rarely used)

Generally, the files that should be passed to the next task are those that were processed or skipped.

Support for Streaming Paths from API

This applies to delete and move handlers in a snapdiff workflow.

To fetch the streaming paths, the function `expand_paths` from the module `arcapix.dynamo.server.utils.inputs` has to be used like below.

```
   from arcapix.dynamo.server.utils.inputs import expand_paths
```

Warning: `expand_paths` provides a generator function fetching pages of paths on demand. Developers must ensure that memory starvation of Hub is negated by following the principles of generator functions thereby avoiding storing large number of values in memory concurrently.

Example Plugins

Warning: This feature is currently in alpha

Here are a couple examples of functionality that can be added to the hub, these can be dropped into any template project.

Email notification task

The intent of this plugin is to email a list of staff members at the end of a workflow to ensure that they are informed of its completion, it has extra key word arguments that allow the editing of the subject:

```
import sys

from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.utils import formatdate
from smtplib import SMTP
from typing import List

from celery import shared_task


def send_email(email_to: List[str], email_from: str, subject: str,
message: str, server: str = "localhost"):
    """
    Sends an email through an already configured SMTP server

    :param email_to: List of email recipients
    :param email_from: Address that email derives from
    :param subject: Email subject
    :param message: Message to send
    :param server: Target SMTP server address
    :return: None
    """
    smtp_message = MIMEMultipart()
    smtp_message['From'] = email_from
    smtp_message['To'] = ', '.join(email_to)
    smtp_message['Date'] = formatdate(localtime=True)
    smtp_message['Subject'] = subject

    smtp_message.attach(MIMEText(message))

    try:
        smtp = SMTP(server)
        smtp.sendmail(email_from, email_to, smtp_message.as_string())
        smtp.close()
    except OSError as error:
        # All SMTP errors are derived from OSError and catching all
SMTP errors from the base exception is not allowed
        print('Error sending notification email: %s', error)


@shared_task(bind=True, name="dynamo.custom.email_staff")
def email_staff(
    self,
    *args,
    paths: List = None,
    jobid: int = None,
    staff_members: List = None,
    message: str = None,
    subject: str = None,
    server: str = None,
    from_address: str = None,
    **kwargs
):
    if not message:
        message = f"Job {jobid} has completed successfully
processing {len(paths)} paths"

    if not from_address:
        from_address = "ngenea-worker@pixitmedia.com"

    if not subject:
        subject = "Ngenea Worker Job completed successfully"

    if not server:
        server = "localhost"

    send_email(email_to=staff_members, email_from=from_address,
subject=subject, message=message, server=server)

    return {"status": "success"}


if __name__ == "__main__":
    sys.exit(0)  # pragma: no cover
```

The `setup.py` will need editing to make the entry point the name of the function `email_staff` in the module that had been created for this example plugin.

To make use of this new task in the hub, here is an example workflow that will email staff members after all the data has been migrated with a custom subject and receiving email address:

```
{
    "name": "migrate_notif",
    "label": "Migrate with notif",
    "icon_classes": [
        "fa fa-cloud fa-stack-2x text-primary",
        "fa fa-refresh fa-stack-1x text-light"
    ],
    "discovery": null,
    "enabled": true,
    "visible": true,
    "fields": [],
    "filter_rules": [
        {
            "type": "all",
            "state": "all",
            "action": [
                {
                    "name": "dynamo.tasks.migrate"
                },
                {
                    "name": "dynamo.custom.email_staff",
                    "staff_members": [
                        "johnsmith@organisation.com",
                        "admin@organisation.com"
                    ],
                    "subject": "Project number 7 job complete",
                    "from_address": "notifications@organisation.com"
                }
            ]
        }
    ]
}
```

Running additional script

After running a set of tasks, you may need to execute a script on the host machine of the Ngenea Worker instance. The following plugin allows the execution of an arbitrary script located at `/opt/cloud_script.sh`:

Note: Any script run through this plugin will be run with root access

```python
import sys

from subprocess import run as run_script
from subprocess import TimeoutExpired, SubprocessError, PIPE, STDOUT

from typing import List

from celery import shared_task

DEFAULT_SCRIPT = "/opt/cloud_script.sh"


@shared_task(bind=True, name="dynamo.custom.run_cloud_script")
def run_cloud_script(
    self,
    *args,
    paths: List = None,
    jobid: int = None,
    script_location: str = None,
    timeout: int = 600,
    additional_args: List = None,
    use_paths: bool = False,
    **kwargs
):
    status = {"success": False}

    try:
        args = [script_location if script_location else
DEFAULT_SCRIPT]
        if additional_args:
            args = args + additional_args

        if use_paths:
            # Appends the paths to the arguments
            args = args + [path["path"] for path in paths]

        result = run_script(
            args,
            stdout=PIPE,
            stderr=STDOUT,
            check=True,
            timeout=timeout,
        )

        status["log"] = str(result.stdout)

        if result.returncode == 0:
            status["success"] = True

    except TimeoutExpired:
        status["log"] = "Called script timed out"
    except SubprocessError as sp_err:
        status["log"] = str(sp_err)
    return {"status": "success"}


if __name__ == "__main__":
    sys.exit(0)  # pragma: no cover
```

The `setup.py` will need editing to make the entry point the name of the function `run_cloud_script` in the module that had been created for this example plugin.

To make use of this new task in the hub, here is an example workflow that will run the default cloud script after all of the data has been migrated:

```
{
    "name": "migrate_cloud_script",
    "label": "Migrate with notif",
    "icon_classes": [
        "fa fa-cloud fa-stack-2x text-primary",
        "fa fa-refresh fa-stack-1x text-light"
    ],
    "discovery": null,
    "enabled": true,
    "visible": true,
    "fields": [],
    "filter_rules": [
        {
            "type": "all",
            "state": "all",
            "action": [
                {
                    "name": "dynamo.tasks.migrate"
                },
                {
                    "name": "dynamo.custom.run_cloud_script"
                }
            ]
        }
    ]
}
```

Disaster Recovery / Cold Failover

It's possible to configure Ngenea Hub to be ablo to cold-failover to another node if it's running on a PixStor.

Setup

Configure datastore

Configure Ngenea Hub to store it's persistent data on the GPFS filesystem so it can be read by multiple nodes. This is done by settings the following setting in `/etc/sysconfig/ngeneahub`

```
DATA_DIR=/mmfs1/.arcapix/ngeneahub/data
```

Configure Networking

It's strongly recommended to configure a floating IP that can be used for the Ngenea Worker to connect to. This will allow cold failover without having to reconfigure workers.

This can be done by setting the following settings in `/etc/sysconfig/ngeneahub`:

- `SERVICE_CIDR`. Set this to the IP and netmask of the IP you want to be managed by ngeneahub. e.g. `192.168.2.3/24` for the IP `192.168.2.3` on a network with a netmask of `255.255.255.0`
- `SERVICE_INTERFACE`. Set this to the name of the interface the IP adress should be added to. e.g. `man0`

Configure the workers to use this IP by editing `/etc/ngenea/ngenea-worker.conf` on each worker node and modifying `broker_url` and `result_backend`

Install Ngenea Hub

Install Ngenea Hub on multiple nodes as usual. Make sure `/etc/sysconfig/ngeneahub` are in sync across these nodes. Enable and start the service on one node only. Leave the service disabled and stopped on the other nodes.

Performing failover

In the case of a node failure, after confirming the services are no longer running on the other node, the following seteps can be peformed to bring the serive up on another node:

important You must be certain the service is not running anywhere else before continuing, otherwise data loss can occur.

- Remove the lock file from `${DATA_DIR}/.lock`.
- Start the Ngenea Hub service

Migration from local datastore

AFter setting `DATA_DIR` in `/etc/sysconfig/ngeneahub` and restarting the service, data will automatically be migrated. This is a one-way operation.

External SSL

This document provides information on

- NGINX Installation on vanilla centos 7
- Configuration for SSL Termination and Reverse proxies to ngeneahub

Installing NGINX

Adding the EPEL Software Repository

```
sudo yum install epel-release
```

Installing NGINX

```
sudo yum install nginx
```

Starting Nginx service

```
sudo systemctl start nginx
```

Check Nginx service Status

```
sudo systemctl status nginx
```

Nginx status output should look like this

```
Output
● nginx.service - The nginx HTTP and reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; disabled;
vendor preset: disabled)
   Active: active (running) since Mon 2022-01-24 20:14:24 UTC; 5s ago
  Process: 1898 ExecStart=/usr/sbin/nginx (code=exited, status=0/
SUCCESS)
  Process: 1896 ExecStartPre=/usr/sbin/nginx -t (code=exited,
status=0/SUCCESS)
  Process: 1895 ExecStartPre=/usr/bin/rm -f /run/nginx.pid
(code=exited, status=0/SUCCESS)
 Main PID: 1900 (nginx)
   CGroup: /system.slice/nginx.service
           ├─1900 nginx: master process /usr/sbin/nginx
           └─1901 nginx: worker process

Jan 24 20:14:24 centos-updates systemd[1]: Starting The nginx HTTP
and reverse proxy server...
Jan 24 20:14:24 centos-updates nginx[1896]: nginx: the configuration
file /etc/nginx/nginx.conf syntax is ok
Jan 24 20:14:24 centos-updates nginx[1896]: nginx: configuration
file /etc/nginx/nginx.conf test is successful
Jan 24 20:14:24 centos-updates systemd[1]: Started The nginx HTTP
and reverse proxy server.
```

The service should be `active`

To stop the Nginx service

```
sudo systemctl stop nginx
```

To disable the Nginx service

```
sudo systemctl disable nginx
```

To Enable the Nginx service

```
sudo systemctl enable nginx
```

Configuration for SSL Termination and Reverse Proxy using OpenSSL

Create Self-Signed Certificates for Nginx

Create the Certificate Configuration file named `localhost.conf`

```

[req]
default_bits       = 2048
default_keyfile    = localhost.key
distinguished_name = req_distinguished_name
req_extensions     = req_ext
x509_extensions    = v3_ca

[req_distinguished_name]
countryName                 = Country Name (2 letter code)
countryName_default         = US
stateOrProvinceName         = State or Province Name (full name)
stateOrProvinceName_default = New York
localityName                = Locality Name (eg, city)
localityName_default        = Rochester
organizationName            = Organization Name (eg, company)
organizationName_default    = localhost
organizationalUnitName      = organizationalunit
organizationalUnitName_default = Development
commonName                  = Common Name (e.g. server FQDN or YOUR
name)
commonName_default          = localhost
commonName_max              = 64

[req_ext]
subjectAltName = @alt_names

[v3_ca]
subjectAltName = @alt_names

[alt_names]
```

```
DNS.1    = example.com
DNS.2    = 127.0.0.1
```

Create the Certificate using OpenSSL using below command

```
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout
localhost.key -out localhost.crt -config localhost.conf
```

Copy the Certificate Key Pair to the Certificates folder `/etc/ssl/certs`

```
sudo cp localhost.crt /etc/ssl/certs/localhost.crt
sudo cp localhost.key /etc/ssl/private/localhost.key
```

Creating configuration file for Nginx

Create a configuration file in `/etc/nginx/conf.d/proxy.conf` and Update the Nginx Configuration File to Load the Certificate Key Pair

```
    server {

        listen 80;
        listen 443 ssl http2;
        listen [::]:443 ssl http2; #for IPv6

        server_name example.com;

        #specify the certificate files to use
        ssl_certificate /etc/ssl/certs/localhost.crt;
        ssl_certificate_key /etc/ssl/private/localhost.key;

        ssl_protocols TLSv1.2 TLSv1.1 TLSv1;

        root /usr/share/nginx/html;

        #Serving index.html file when requesting /
        index index.html;

        #Reverse proxy for requests
        location / {
                proxy_set_header        Host $host:8000;
                proxy_set_header        X-Real-IP $remote_addr;
                proxy_set_header        X-Forwarded-For
$proxy_add_x_forwarded_for;
                proxy_set_header        X-Forwarded-Proto $scheme;

                proxy_pass http://localhost:8000/;#if running
outside docker you can use 127.0.0.1 or localhost instead of
host.docker.internal, or with docker with network_mode: "host"
                proxy_redirect off;
                }
        }
}
```

Reload the Nginx service after you have made some configuration changes

```
sudo systemctl reload nginx
```

Configuration for SSL Termination and Reverse Proxy using Certbot and LetsEncrypt (Another method)

Add trusted SSL Certificates from Letsencrypt

We need to redirect all unencrypted HTTP connections to HTTPS. This is done with certbot and letsencrypt certificates. The certbot will obtain free certificates and also handle the renewal process automatically. To do that we will install certbot and also a plugin for our NGINX server.

```
sudo yum install certbot python3-certbot-nginx
```

Once we have installed those packages, we can obtain our certificates.

```
sudo certbot --nginx -d example.com
```

It will ask you if you want to redirect all traffic from HTTP to HTTPS. Select yes (2). This automatically makes some changes to our NGINX default configuration.

```
server {

  server_name example.com;

  location / {
    proxy_pass http://127.0.0.1:8000;
  }

    listen [::]:443 ssl ipv6only=on; # managed by Certbot
    listen 443 ssl; # managed by Certbot
    ssl_certificate /etc/letsencrypt/live/example.com/fullchain.pem;
# managed by Certbot
    ssl_certificate_key /etc/letsencrypt/live/example.com/
privkey.pem; # managed by Certbot
    include /etc/letsencrypt/options-ssl-nginx.conf; # managed by
Certbot
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by
Certbot

}
server {
    if ($host = example.com) {
        return 301 https://$host$request_uri;
    } # managed by Certbot

  listen 80 default_server;
  listen [::]:80 default_server;
  server_name example.com;
    return 404; # managed by Certbot
}
```

HTTPS Configuration for NgeneaHub

To configure NgeneaHub for HTTPS, a configuration file named 'nghub.conf' should be created under etc/nginx/conf.d/pixstor/ folder. It should contain the configuration below:

```
location /ngeneahub/ {
     proxy_pass http://localhost:8000;
     proxy_http_version 1.1;
     proxy_set_header Host $host;
     proxy_set_header X-Forwarded-Proto $scheme;
     proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;

}
location /ngeneahub/ws {
     proxy_pass http://localhost:8000;
     proxy_http_version 1.1;
     proxy_set_header Host $host;
     proxy_set_header Upgrade $http_upgrade;
     proxy_set_header Connection "Upgrade";
     proxy_set_header X-Forwarded-Proto $scheme;
     proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
}
```

Reload the Nginx service after you have made some configuration changes

```
sudo systemctl reload nginx
```

To auto-renew the certificates, Run

```
certbot renew
```

Usage

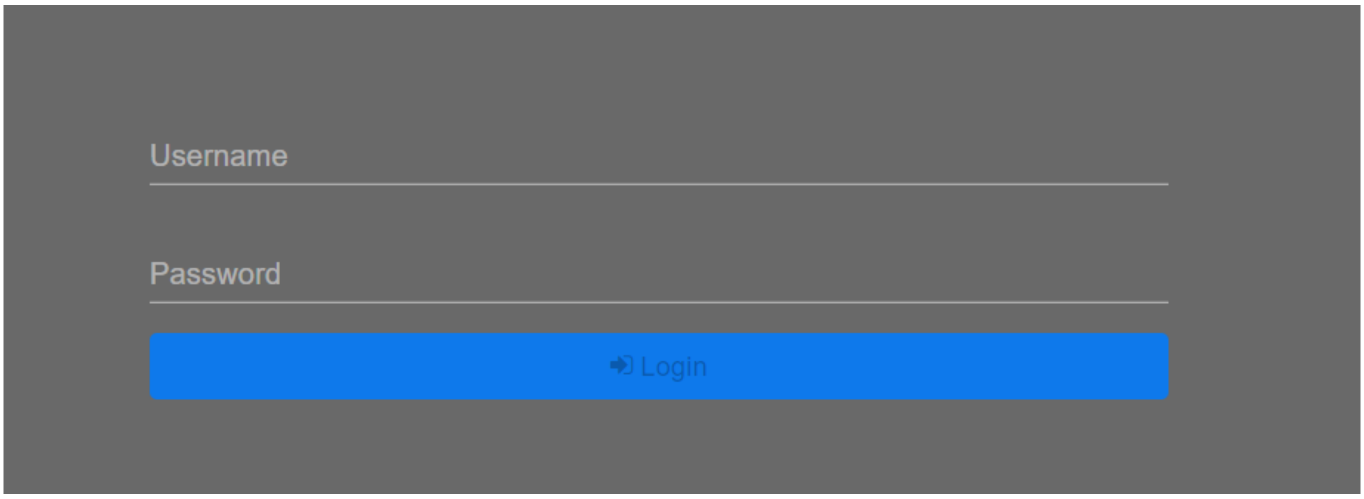
Web Interface

To access Ngenea Hub, go to http://example.com:8000/.

Authentication

Login

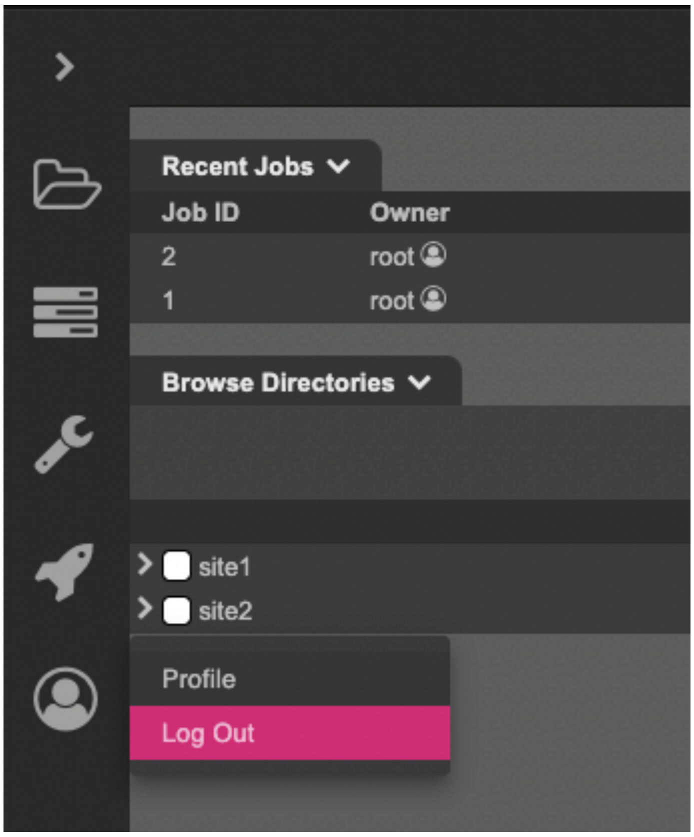
Upon navigating to the Ngenea Hub UI a login screen is presented.



Enter a valid username and password before pressing the `Login` button to authenticate.

Logout

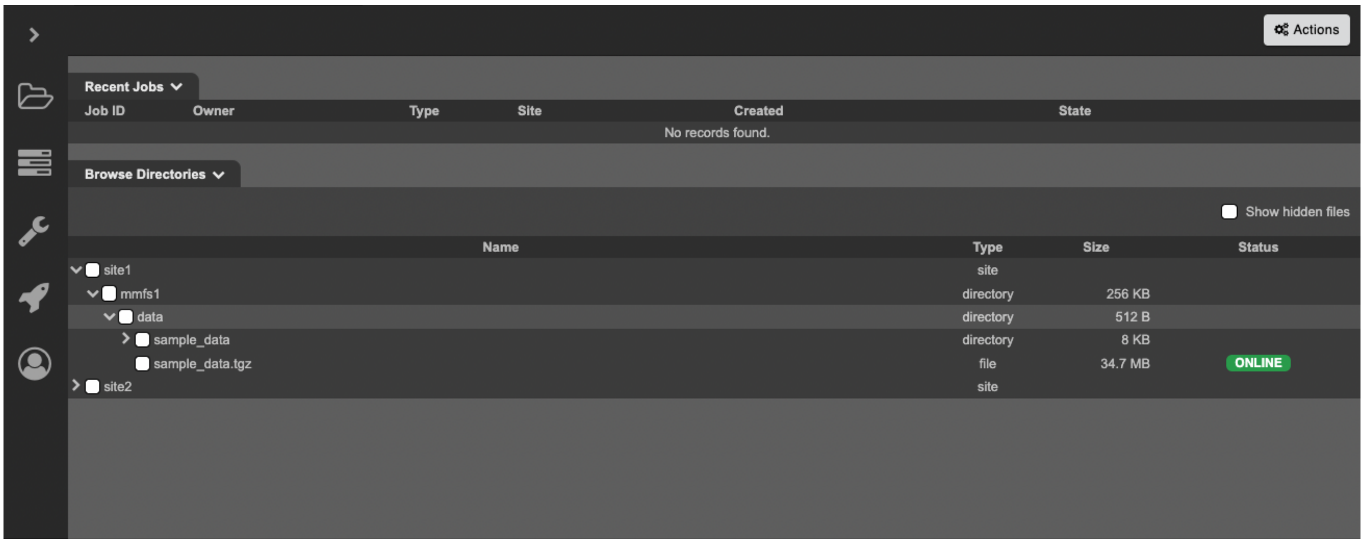
To end your session, select Logout from the `Man Icon` at the bottom left hand corner of the UI.



Job ID	Owner
2	root
1	root

Browser

Upon logging in, select the directory icon on the left hand side of the UI. This takes to the Browser page.



Job ID	Owner	Type	Site	Created	State
No records found.					

Name	Type	Size	Status
site1	site		
mmfs1	directory	256 KB	
data	directory	512 B	
sample_data	directory	8 KB	
sample_data.tgz	file	34.7 MB	ONLINE
site2	site		

The Browser page allows users to see a list of recent jobs as well as browse all available directories across all Sites for the purpose of either migrating, pre-migrating, recalling, or sending files.

Recent Jobs

The Recent Jobs section shows a list of the 5 most recent jobs that were initiated via the UI.

The view presents several columns:

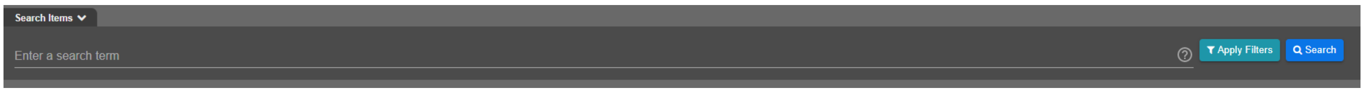
Option	Description
Job ID	Shows the identification number associated with the job
Owner	Name of the user who created the job
Type	Shows the job's type, i.e. Migrate, Recall, Premigrate, or Send
Site	Name of the Site where files were migrated from
Created	Shows the job's creation time
State	Shows the job's state, i.e. success or failure

Search

Note: Before you can use the search feature, additional set-up is necessary, as described in the search feature page.

Search section contains a search bar to discover the contents of the configured Sites.

The view presents a search bar, a button for managing the search filters and another button for starting and stopping the search. The view also presents `(?)` button to get more instructions about starting a search.

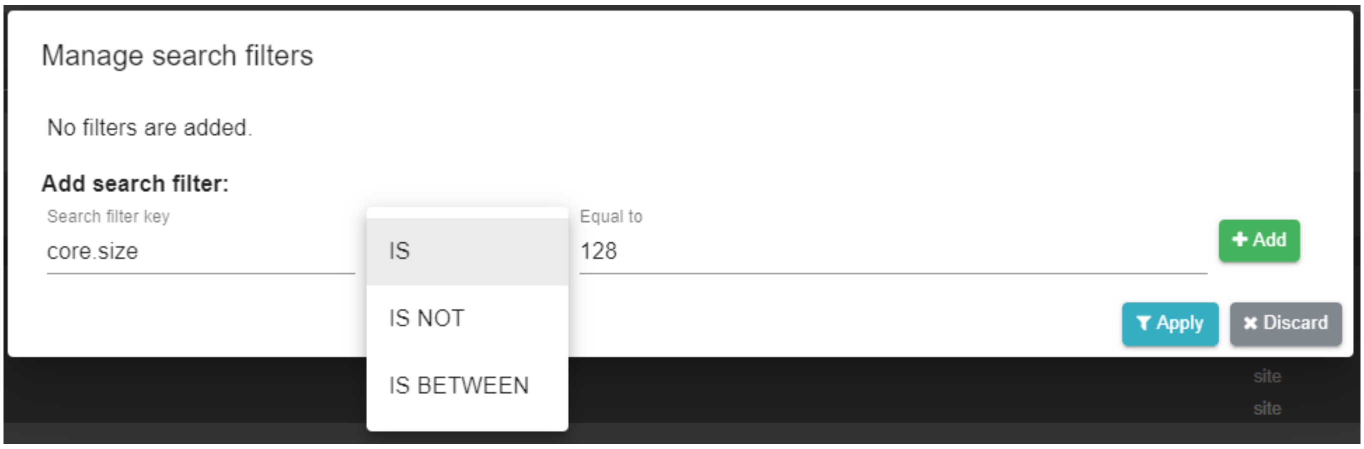


To search for a term, type the search term in the search bar and click `Search` button (or hit `Enter` key) to start a search. The default search filter is "core.filename".

Managing the search filters from the dialog

For more complex search operations, add more filters by clicking `Apply Filters` button and use the dialog opened after that. Type the search filter key in the input box with the label `Search filter key` and then pick an operator from the select box with the label `Operator`.

For the operator `IS`, the dialog is seen as in the image below:


Manage search filters

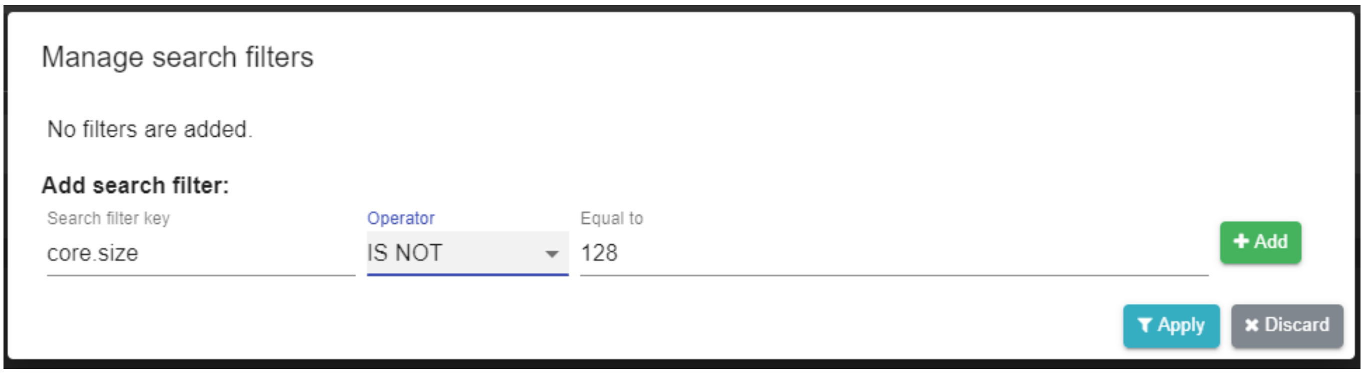
No filters are added.

Add search filter:

Search filter key: core.size

Equal to: 128

For the operator `IS NOT`, the dialog is seen as in the image below:


Manage search filters

No filters are added.

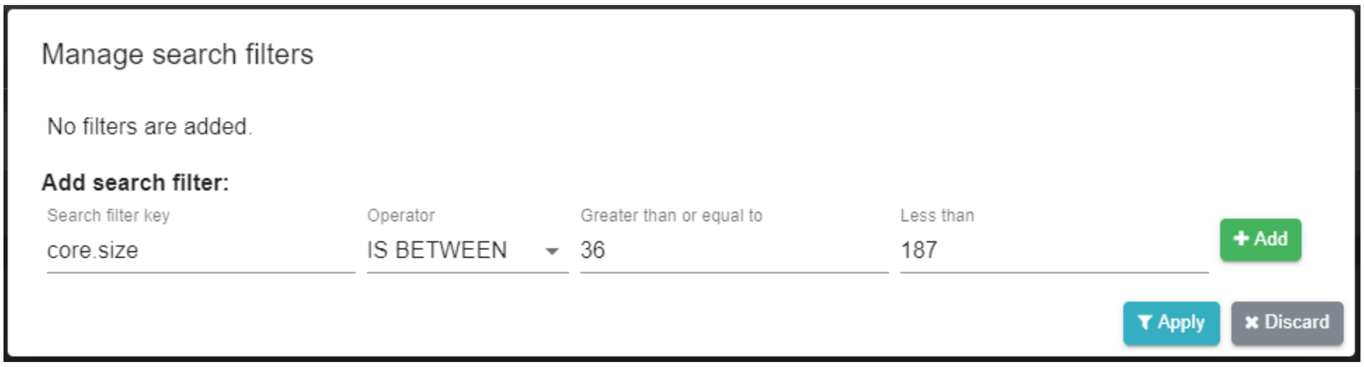
Add search filter:

Search filter key: core.size

Operator: IS NOT

Equal to: 128

For the operator `IS BETWEEN`, the dialog is seen as in the image below:


Manage search filters

No filters are added.

Add search filter:

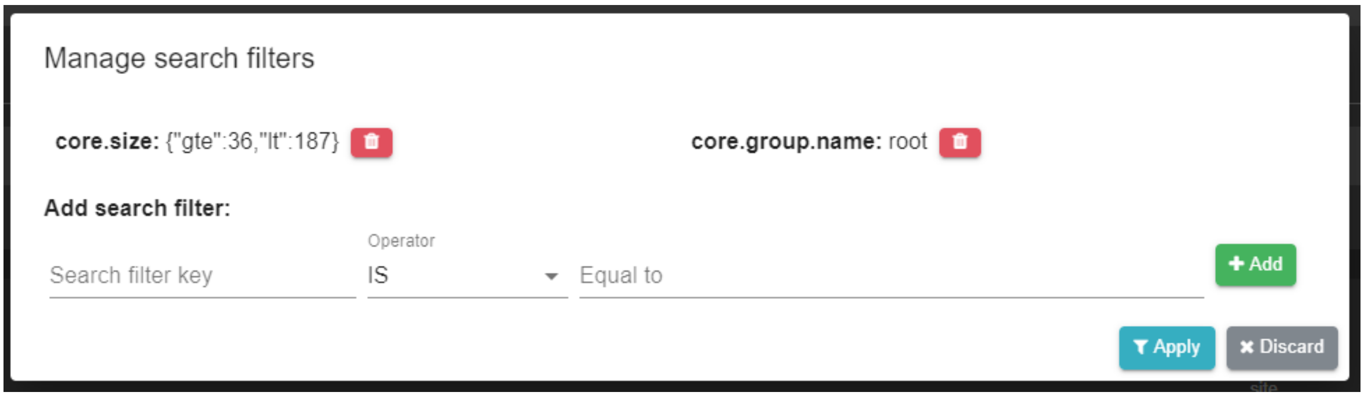
Search filter key: core.size

Operator: IS BETWEEN

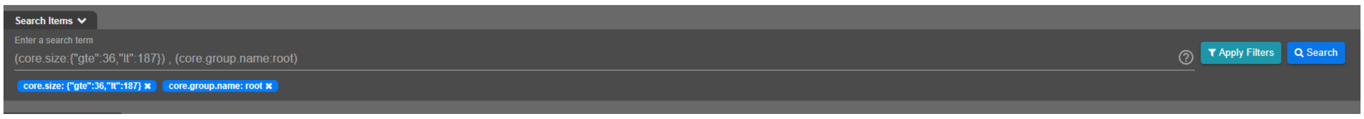
Greater than or equal to: 36

Less than: 187

Click `Add` button to add the search filter. The dialog is seen as in the image below, when "core.size" is selected to be greater than or equal to 36 and less than 187, and "core.group.name" is selected to be "root".

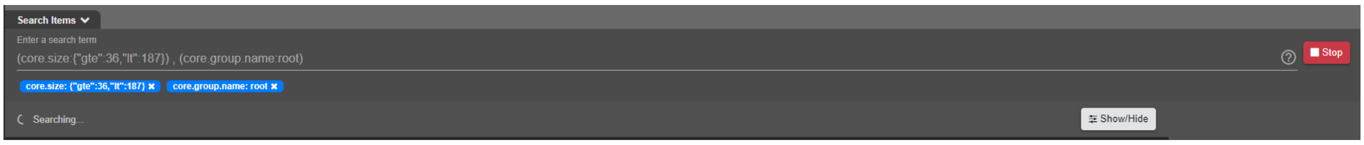


Once the selection is finished, click `Apply` button to save & close the dialog (this action does not submit the search). See that filters are shown as a query in the search bar and they are shown with badges under the search bar. The search section is seen as in the image below, after the search filters are selected:

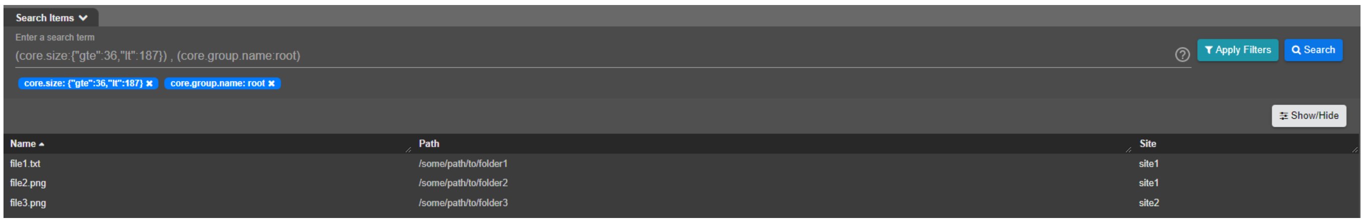

Search Items
Enter a search term
(core.size:{"gte":36,"lt":187}) , (core.group.name:root)
Apply Filters
Search
core.size: {"gte":36,"lt":187}
core.group.name: root

Submitting the search and viewing search results

Click `Search` button, or hit `Enter` key to start the search. The search section will contain search results from each configured site, matching with the search filters. To stop the search, click `Stop` button. The search section is seen as in the image below after the search is in progress.


Search Items
Enter a search term
(core.size:{"gte":36,"lt":187}) , (core.group.name:root)
Stop
core.size: {"gte":36,"lt":187}
core.group.name: root
Searching...
Show/Hide

The search section is seen as in the image below after some results are found.


Search Items
Enter a search term
(core.size:{"gte":36,"lt":187}) , (core.group.name:root)
Apply Filters
Search
core.size: {"gte":36,"lt":187}
core.group.name: root
Show/Hide

Name	Path	Site
file1.txt	/some/path/to/folder1	site1
file2.png	/some/path/to/folder2	site1
file3.png	/some/path/to/folder3	site2

Advanced usage: Managing the search filters from the search bar

You can also make complex search operations by typing the search term in the format below:

```
(<key1>:<value1>) , (<key2>:<value2>) , (<key3>:<value3>) ...
```

For multiple search filters, you need to separate them by paranthesis. For using various comparison operators, here are the filter formats:

Operation	**Format**
Equal to	`(<key1>: <value1>)`

Operation	Format
Less than	`(<key2>: {"lt": <value2> })`
Greater than or equal to	`(<key3>: {"gte": <value3> })`
Combined	`(<key4>: {"gte": <value4>, "lt": <value5> })`

Browse Directories

The Browse Directories section contains a list of configured Sites and correspondent directories under the Sites' filesystems.

The view presents several columns:

Option	Description
Name	Shows the name of the Site, directories, and files
Type	Shows whether the listed item is a Site, directory, or file
Size	Shows the directories and files size
Status	Shows whether files are online or offline

User can select one or multiple directories, or one or multiple files to migrate, premigrate, recall them or send them to a different Site.

Migrate

To Migrate a directory or file, expand the Site containing said directory and file. Select the directory or file you wish to Migrate by ticking their relevant boxes.

Click the "Actions" button at the top right hand side of the page and select "Migrate".

A new Job is created and it is shown at the top of the "Recent Jobs" list. Job's State will display a progress bar until completion.

Once job is complete, the State will either show as Success or Failed.

Premigrate

To Premigrate a directory or file, expand the Site containing said directory and file. Select the directory or file you wish to Premigrate by ticking their relevant boxes.

Click the "Actions" button at the top right hand side of the page and select "Premigrate".

A new Job is created and it is shown at the top of the "Recent Jobs" list. Job's State will display a progress bar until completion.

Once job is complete, the State will either show as Success or Failed.

Recall

To Recall a directory or file, expand the Site containing said directory and file. Select the directory or file you wish to Recall by ticking their relevant boxes.

Click the "Actions" button at the top right hand side of the page and select "Recall".

A new Job is created and it is shown at the top of the "Recent Jobs" list. Job's State will display a progress bar until completion.

Once job is complete, the State will either show as Success or Failed.

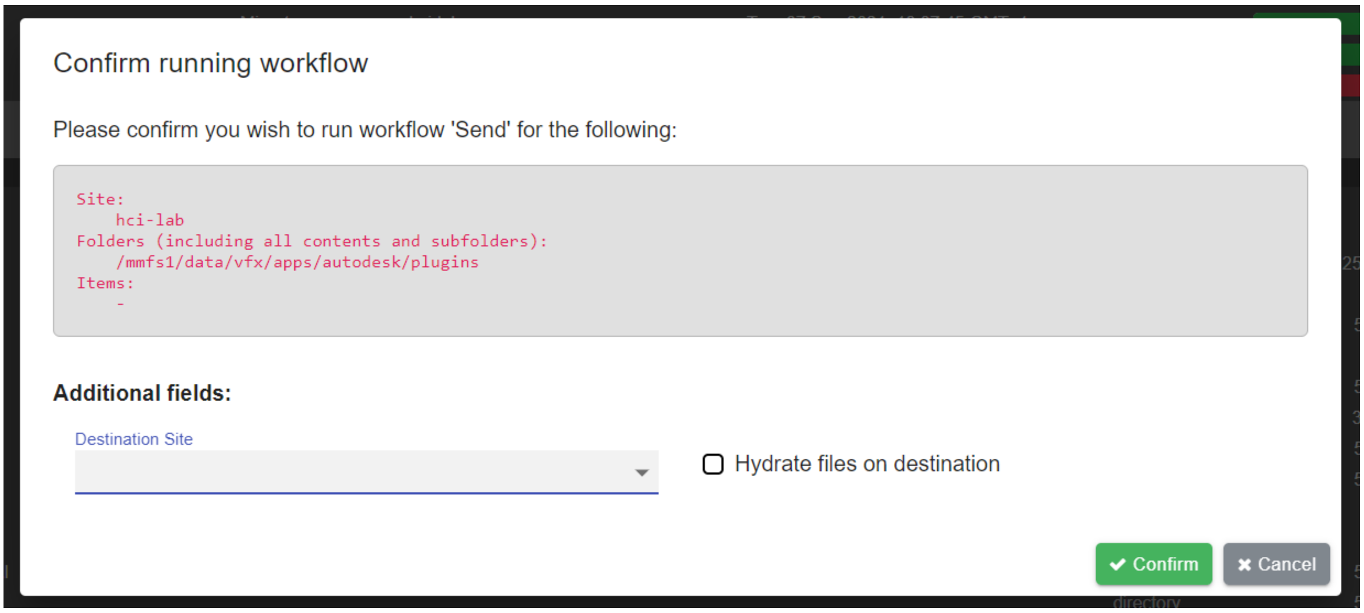
Send

Premigrate behaviour change: Prior to Ngenea Hub 1.8.0, the send workflow would migrate data on the source site. This has been changed to pre-migrate.

To Send a directory or file from one Site to another, expand the Site containing said directory and file. Select the directory or file you wish to Send by ticking their relevant boxes.

Click the "Actions" button at the top right hand side of the page and select "Send".

Select the Site you wish to send the directory and/or files to. Tick the "Hydrate files on destination" if required, and click "Confirm".



Confirm running workflow

Please confirm you wish to run workflow 'Send' for the following:

```
Site:
    hci-lab
Folders (including all contents and subfolders):
    /mmfs1/data/vfx/apps/autodesk/plugins
Items:
    -
```

Additional fields:

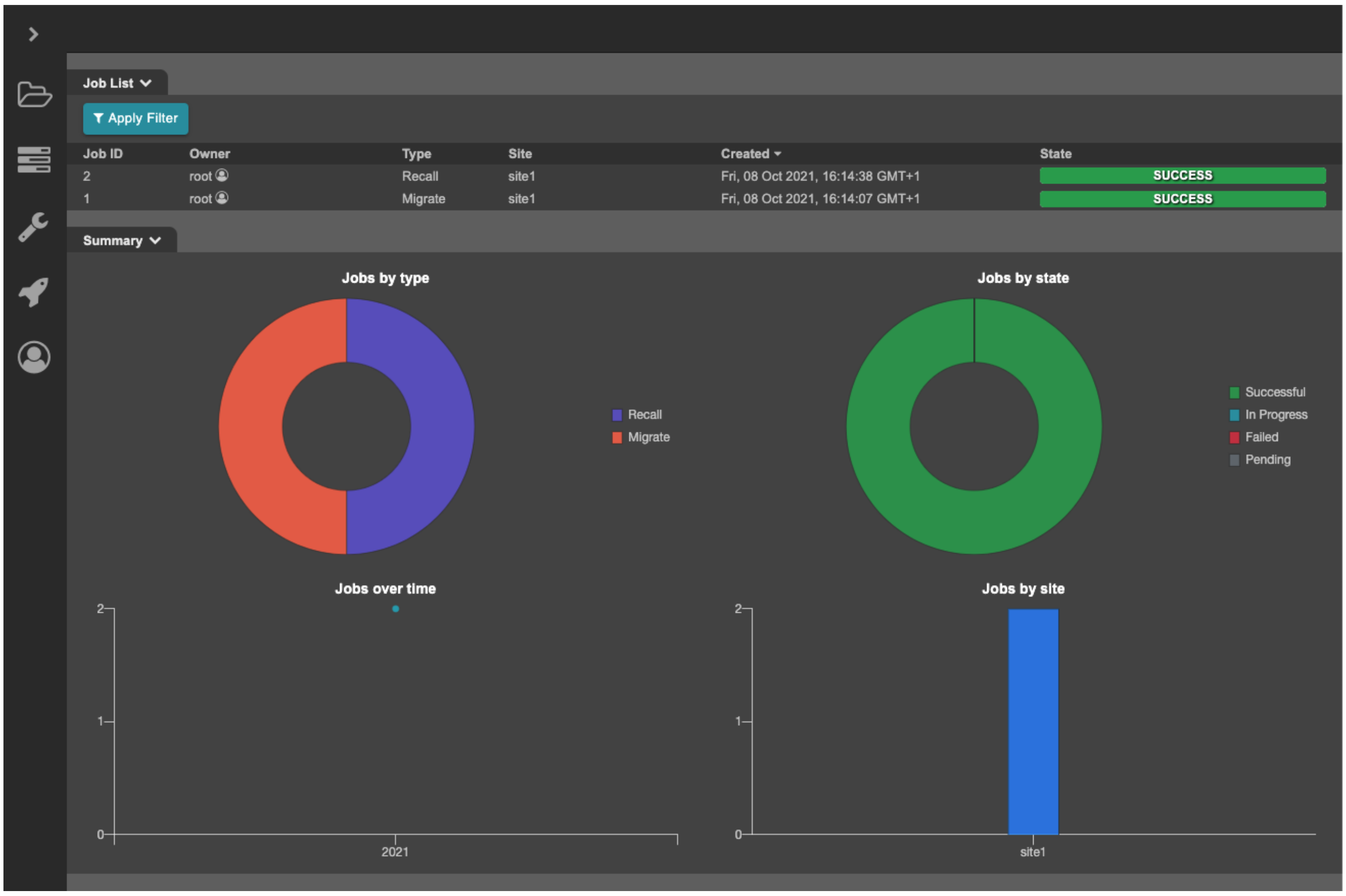
A new Job is created and it is shown at the top of the "Recent Jobs" list. Job's State will display a progress bar until completion.

Once job is complete, the State will either show as Success or Failed.

Expanding the receiving Site's Directories now shows the path that was replicated from the sending Site.

Jobs

The Jobs page shows a list of all the jobs that were initiated via the UI.



Job ID	Owner	Type	Site	Created	State
2	root	Recall	site1	Fri, 08 Oct 2021, 16:14:38 GMT+1	SUCCESS
1	root	Migrate	site1	Fri, 08 Oct 2021, 16:14:07 GMT+1	SUCCESS

The view presents several columns:

Option	**Description**
Job ID	Shows the identification number associated with the job
Owner	Name of the user who created the job
Type	Shows the job's type, i.e. Migrate, Recall, Premigrate, or Send
Site	Name of the Site where files were migrated from
Created	Shows the job's creation time
State	Shows the job's state, i.e. success or failure

Each column can be sorted in ascending and descending order.

Pagination

To select whether to view 20, 50, or 100 Jobs at the time, choose the relevant option in the "Items per Page" dropdown.

Clicking on the right and left arrow next to "Items per Page" will take you to the next/previous pages.

Apply Filter

Jobs list can be filtered by time period, job type, and job state.

To filter the list, select the "Apply Filter" button on the top left hand side of the UI.

Select one or a combination of filters, and click "Apply".

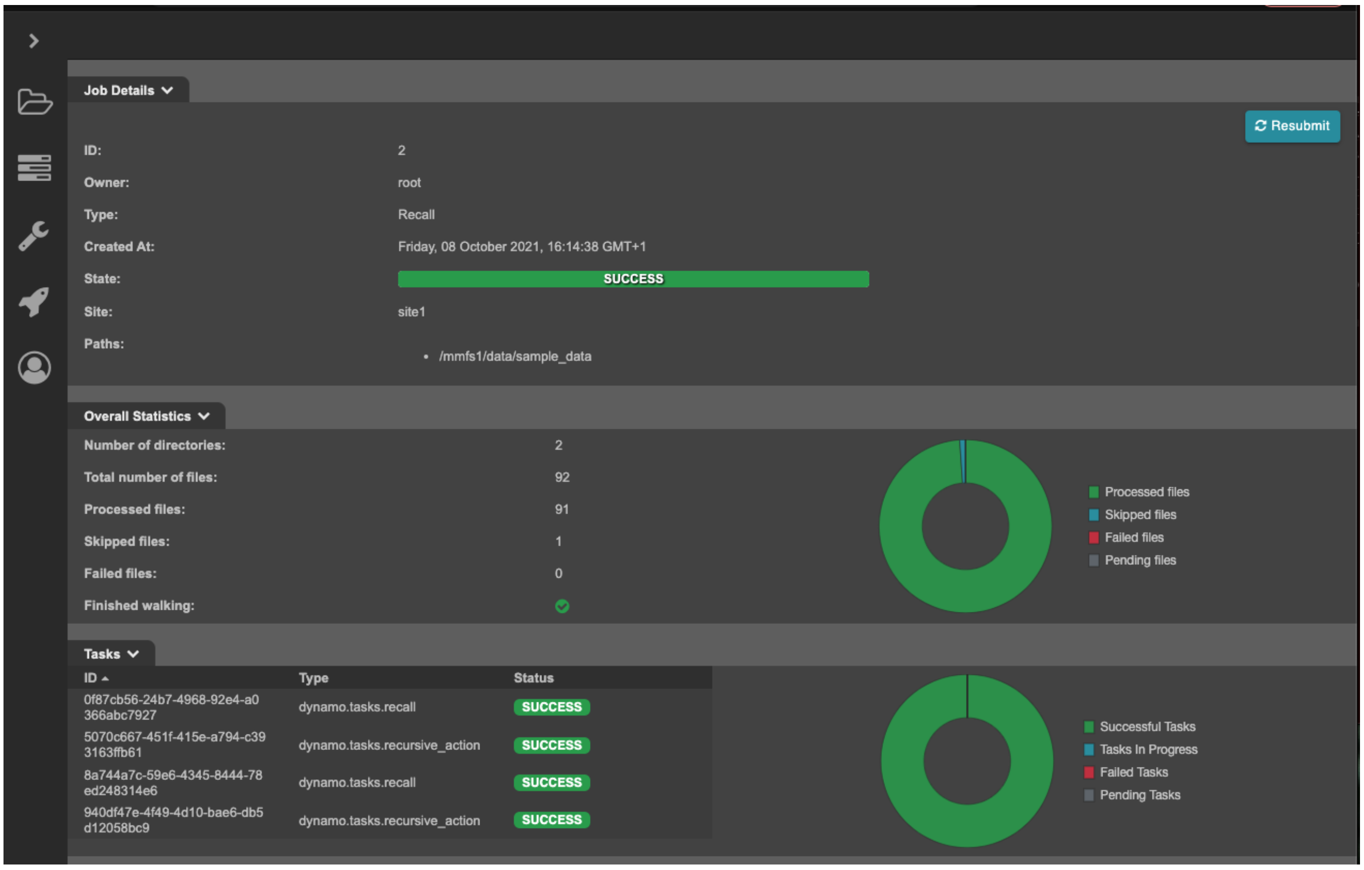
Job are now filtered as per your selection.

To remove a filter, simply select the "x" next to the applied filter.

Job ID

On the Jobs page, click on a Job ID to see additional information regarding the Job.

The Job Details, Overall Statistics, and Tasks tab are displayed.



Job Details

ID:	2
Owner:	root
Type:	Recall
Created At:	Friday, 08 October 2021, 16:14:38 GMT+1
State:	SUCCESS
Site:	site1
Paths:	• /mmfs1/data/sample_data

Overall Statistics

Number of directories:	2
Total number of files:	92
Processed files:	91
Skipped files:	1
Failed files:	0
Finished walking:	✓

Tasks

ID	Type	Status
0f87cb56-24b7-4968-92e4-a0366abc7927	dynamo.tasks.recall	SUCCESS
5070c667-451f-415e-a794-c393163ffb61	dynamo.tasks.recursive_action	SUCCESS
8a744a7c-59e6-4345-8444-78ed248314e6	dynamo.tasks.recall	SUCCESS
940df47e-4f49-4d10-bae6-db5d12058bc9	dynamo.tasks.recursive_action	SUCCESS

Each tab shows specific Job details, some of which are clickable:

- Overall Statistics --> Total number of files, processed files, skipped files, and failed files.
- Tasks --> ID

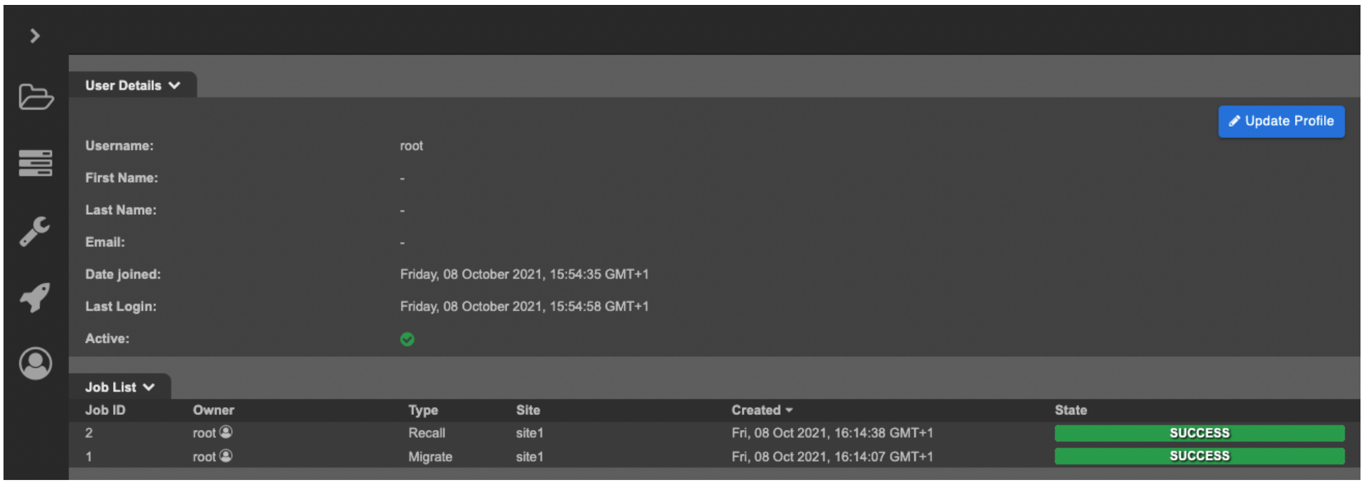
Selecting any of the clickable items opens a dialogue showing the relevant output.

Jobs can also be resubmitted by clicking the "Resubmit" button at the top right hand side of the page.

Owner

On the Jobs page, click on any Owner to see additional information regarding the user who initiated the Job.

Selecting an Owner takes to a page that shows details about the user, as well as a list of Jobs initiated by said user.


User Details

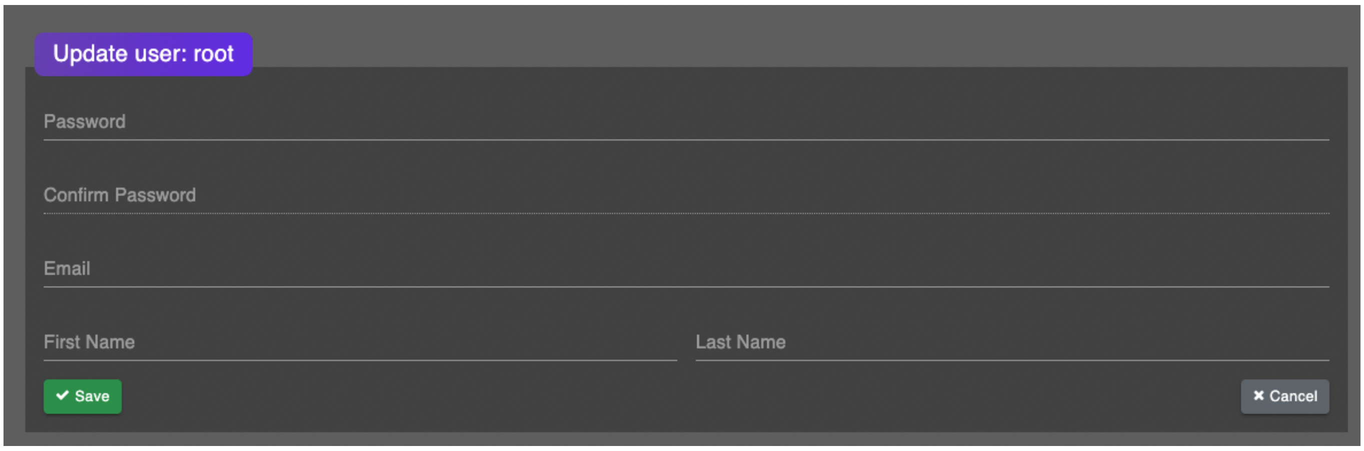
Username:	root
First Name:	-
Last Name:	-
Email:	-
Date joined:	Friday, 08 October 2021, 15:54:35 GMT+1
Last Login:	Friday, 08 October 2021, 15:54:58 GMT+1
Active:	✓

Job List

Job ID	Owner	Type	Site	Created	State
2	root	Recall	site1	Fri, 08 Oct 2021, 16:14:38 GMT+1	SUCCESS
1	root	Migrate	site1	Fri, 08 Oct 2021, 16:14:07 GMT+1	SUCCESS

If you selected your own user, you will see an "Update Profile" on the top right hand side of the page.

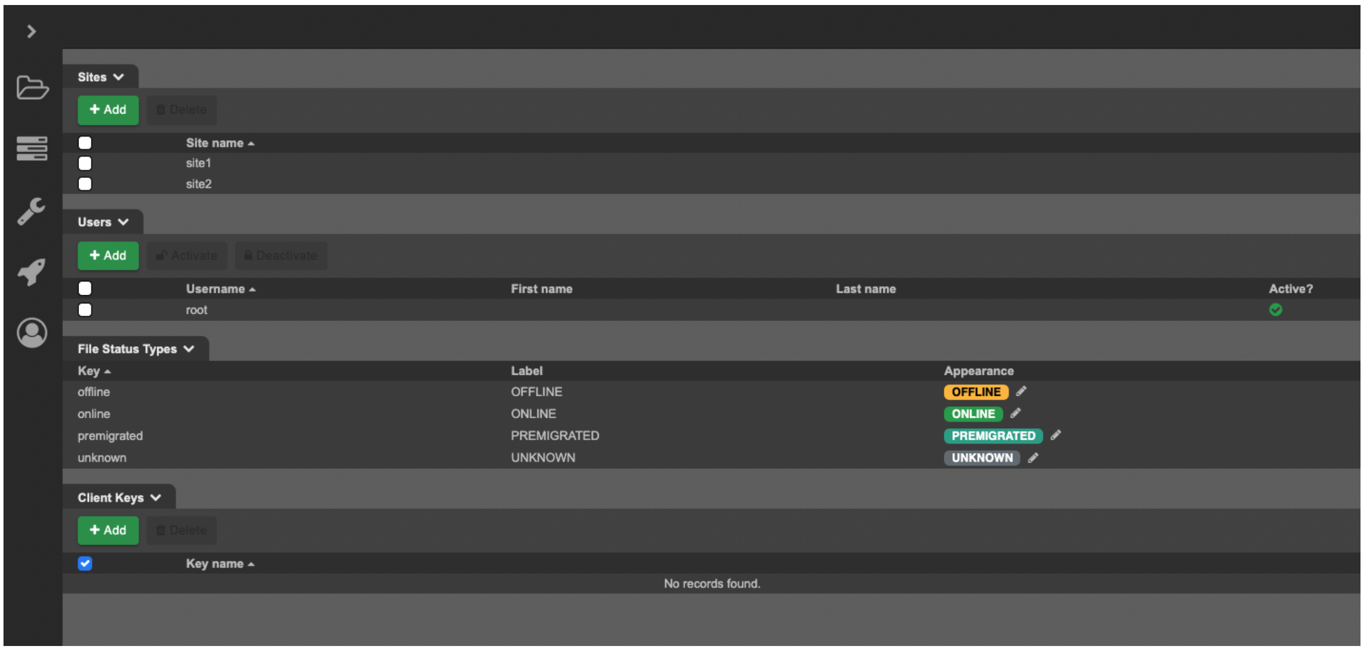
This takes you to the "Update User" page where you can change your own password, email, first name and last name.



The Job List's layout is the same as the one shown in Jobs page and has the same functionalities.

Administration

The Administration page allows you to add and delete Sites, add, activate and deactivate Users, as well as changing the name and colours of the labels found in the Browse Directories section.


Sites

+ Add | Delete

	Site name
☐	site1
☐	site2

Users

+ Add | Activate | Deactivate

	Username	First name	Last name	Active?
☐	root			✓

File Status Types

Key	Label	Appearance
offline	OFFLINE	OFFLINE
online	ONLINE	ONLINE
premigrated	PREMIGRATED	PREMIGRATED
unknown	UNKNOWN	UNKNOWN

Client Keys

+ Add | Delete

☑	Key name

No records found.

Sites

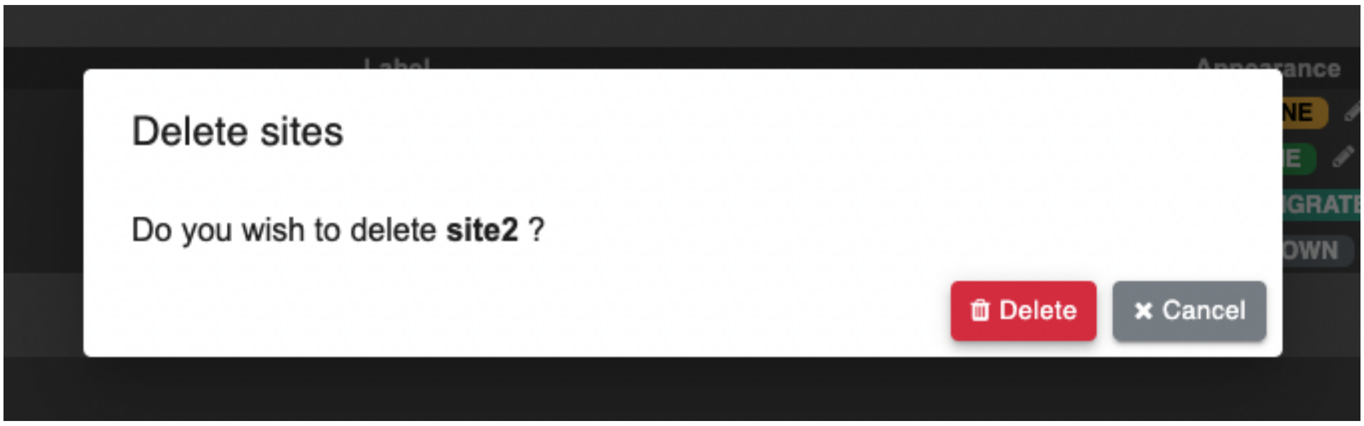
The Sites tab contains the list of Sites that have been configured. This list can be sorted by Site name in ascending and descending order.

Add Site

To add a Site, select the "Add" button, enter a Site name, and confirm by clicking "Add"

Delete Site

To delete a Site, select one or multiple Sites by ticking their correspondent boxes. Click "Delete" and then confirm deletion once the "Delete Site" dialogue is presented.


Delete sites

Do you wish to delete **site2** ?

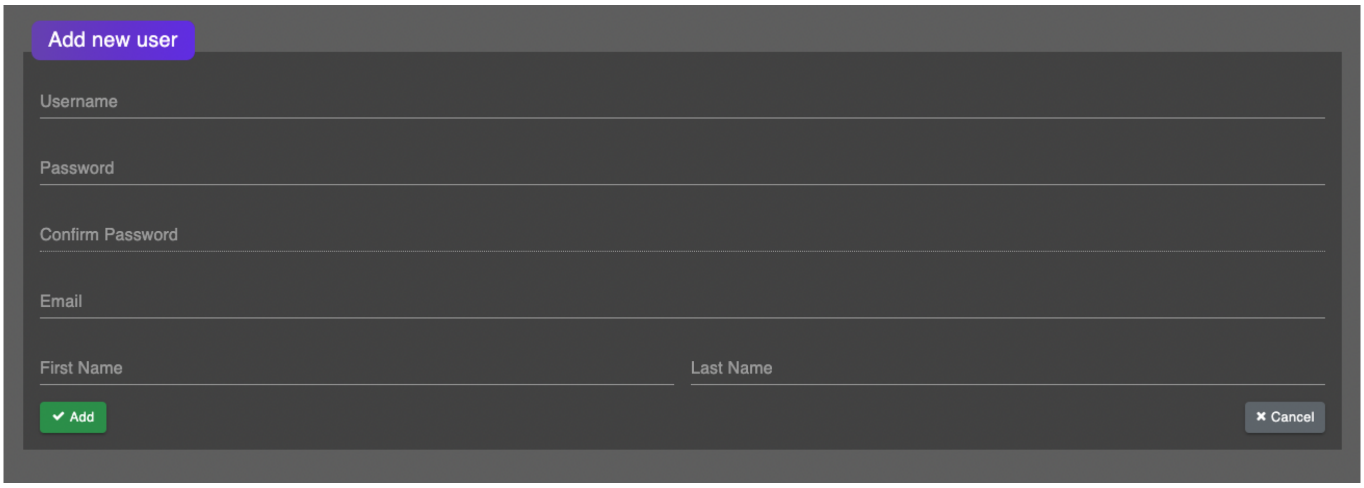
Delete | ✖ Cancel

Users

The Users tab contains the list of Users who are allowed to use the UI. This list can be sorted by Username in ascending and descending order.

Add User

To add a new User, select the "Add" button, then enter a Username, Password, Email, First Name, and Last Name.



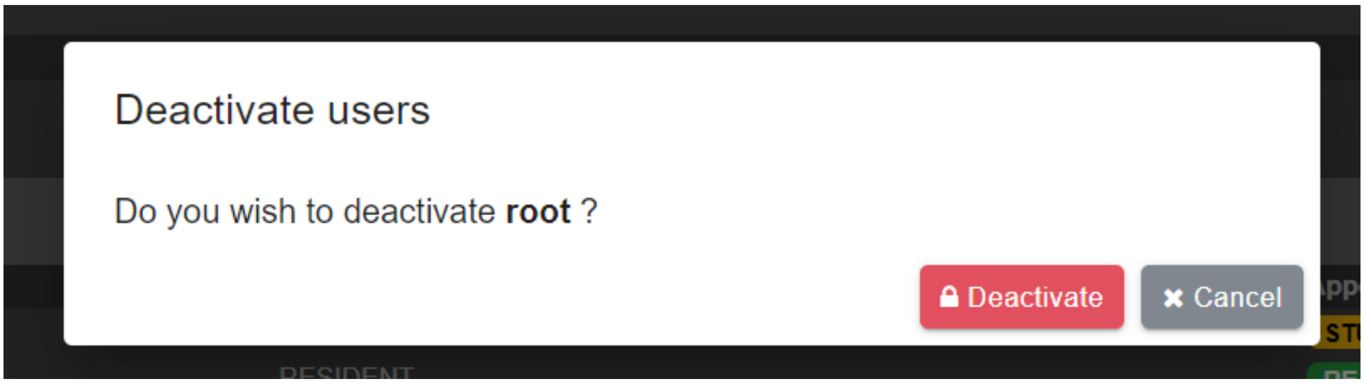
Confirm by clicking "Add".

Activate User

To activate an existing User, select the "Activate" button, and then confirm activation once the "Activate user" dialog is presented.

Deactivate User

To deactivate a User, select one or multiple Users by ticking their correspondent boxes. Click "Deactivate" and then confirm deactivation once the "Deactivate user" dialogue is presented.



Deactivate users

Do you wish to deactivate **root** ?

User will no longer have access to the UI but still exists and can be reactivated at any time.

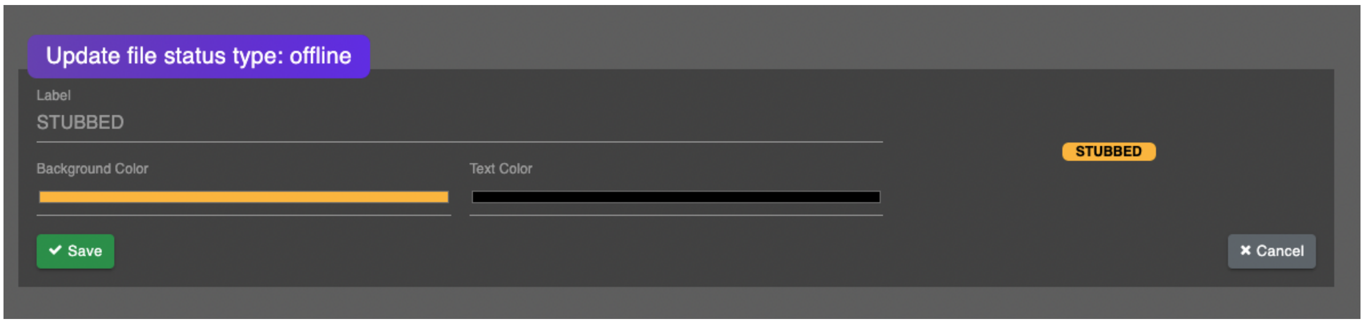
File Status Types

The File Status Types tab lists the labels that are used to indicate a file's status.

The labels are customisable as both colour and label name can be changed.

To change the appearance of a label, select the pencil symbol next to the label that needs updating.

"Update File Status Type" page is displayed:



Change label name, background colour, or text colour as per your preference.

Click "Save" to confirm the update.

API

Reference

The API reference can be found at your Ngenea Hub install at `/api/docs/`. This section does not attempt to duplicate the reference, but instead provide some usage examples.

Authentication

Authentication to the API can be performed in 2 ways

- JWT Authentication
- Client Keys

The first one is for interacting with the API interactively and is therefore most likely not suitable for building automated workflows. On the other hand, client keys are valid until they are revoked and are more suitable for automation.

JWT Authentication

To use the API directly, authentication tokens should be generated to prevent sending the username and password repeatedly. You need to generate these tokens by sending your username-password pair to the login endpoint: `/auth/token/`

```
curl -s -X POST 'http://localhost:8000/api/auth/token/'  -H 'Accept:
application/json'  -H 'Content-Type: application/json' -d
'{"username": "dfoster",  "password": "******"}' | jq -r
{
  "access": <access_token>,
  "refresh": <refresh_token>,
}
```

There are 2 types of authentication tokens:

- Access token
- Refresh token

Access tokens are used for doing API requests. You need to include the token in the Authorization header to use any other endpoint:

```
curl -s -X GET 'http://localhost:8000/api/jobs/'  -H 'Accept:
application/json'  -H 'Content-Type: application/json'  -H
"Authorization: Bearer $JWT_ACCESS_TOKEN" | jq
{
  "count": 0,
  "next": null,
  "previous": null,
  "results": [],
  "stats": {
    "type": {
      "migrate": 0,
      "premigrate": 0,
      "recall": 0
    },
    "state": {
      "SUCCESS": 0,
      "FAILURE": 0,
      "STARTED": 0,
      "PENDING": 0,
      "ERROR": 0
    },
    "created": {},
    "site": {}
  }
}
```

On the other hand, refresh tokens are used for refreshing the access token. For security purposes, access tokens expire in 1 hour and refresh tokens expire in 1 day. When an expired token is used, one of `HTTP 401 Unauthorized` and `HTTP 403 Forbidden` errors is received. In that case, you need to refresh the access token with `/api/token/refresh/` endpoint:

```
curl -s -X POST 'http://localhost:8000/api/auth/token/refresh/'  -H
'Accept: application/json'  -H 'Content-Type: application/json' -d
'{"refresh": "<refresh_token>"}' | jq -r
{
  "access": <new_access_token>,
}
```

Refresh tokens can also be expired too. In that case, you need to send your credentials (username and password) again to obtain new token pair.

Client Keys

Creating Client Keys

Note: The UI does not currently support creating client keys and therefore have to be done via the API directly

Before we can authenticate using the client key, we need to temporarily authenticate using JWT to be able to create a client key.

To get a valid JWT access token using curl and `jq`:

```
export JWT_TOKEN=$(curl -s -X POST 'http://localhost:8000/api/auth/
token/'  -H 'Accept: application/json'  -H 'Content-Type:
application/json' -d '{"username": "dfoster",  "password":
"******"}' | jq -r .access)
echo $JWT_TOKEN
<token>
```

This can now be used to create a client key:

```
curl -s -X POST 'http://localhost:8000/api/auth/clientkeys/'  -H
'Accept: application/json'  -H 'Content-Type: application/json'  -H
"Authorization: Bearer $JWT_TOKEN" -d '{"name": "my_automation_key"}'
| jq '.'
{
  "url": "http://localhost:8000/api/auth/clientkeys/1/",
  "id": 1,
  "name": "my_automation_key",
  "api_key": "YOUR_API_KEY"
}
```

```
{warning} This is the only time the client key will be visible, make
sure it is recorded.
```

Using Client Keys

The key created in the previous section can now be used by setting the header `Authorization: Api-Key YOUR_API_KEY` against an API endpoint. For example:

```
export API_KEY=YOUR_API_KEY

curl -s -X GET 'http://localhost:8000/api/jobs/'  -H 'Accept:
application/json'  -H 'Content-Type: application/json'  -H
"Authorization: Api-Key $API_KEY" | jq
{
  "count": 0,
  "next": null,
  "previous": null,
  "results": [],
  "stats": {
    "type": {
      "migrate": 0,
      "premigrate": 0,
      "recall": 0
    },
    "state": {
      "SUCCESS": 0,
      "FAILURE": 0,
      "STARTED": 0,
      "PENDING": 0,
      "ERROR": 0
    },
    "created": {},
    "site": {}
  }
}
```

Submitting Workflow

To submit a workflow, the following parameters are required:

name	description
`workflow`	The name of the workflow to submit
`paths`	A list of paths to execute the workflow on
`site`	The name of the site where the workflow should be started from. Steps within a workflow may run on different sites.

In addition, the following optional parameters may be provided:

name	description
`discovery`	Name of the file discovery technique to use. Currently the only supported discovery is `recursive`. If no discovery is specified, `recursive` will be used as the default. If explicitly set to `null`, no discovery will be performed and the provided paths will be used 'as is'.
`fields`	Additional parameter for the workflow, typically used by custom workflows.

Migrate

Using a Client Key stored in a environment variable `TOKEN`, the following is an example of migrating a file using curl.

```
curl -s -X POST 'http://example.com/api/file/workflow/'  -H 'Accept:
application/json'  -H 'Content-Type: application/json'  -H
"Authorization: Api-Key $TOKEN"  -d '{"paths": ["/mmfs1/data/
sample_data.tgz"], "site": "dfoster1", "workflow": "migrate",
"discovery": null}'
```

Note: Since we're only migrating a single file, we don't need recursive discovery, so discovery has been set to `null` to disable it.

Trailing Slashes

In general, the endpoint used with `POST` requests must have a trailing slash.

For example, a `POST` request made to `http://example.com/api/file/workflow` would error, but if made to `http://example.com/api/file/workflow/` it would work.

Monitoring and Management

systemd service

Ngenea Hub is controlled via the `ngeneahub` systemd service

ngeneahubctl cli tool

The `ngeneahubctl` tool can be used to manually stop/start the services, outside of systemd, for debugging

Docker containers

Ngenea Hub uses a collection of docker containers, which can be managed by standard Docker monitoring/management tools and processes:

Container Name	Description
`ngeneahub_app_1`	Web application
`ngeneahub_jobrefresh_1`	Maintenance task controller
`ngeneahub_db_1`	Application database (Postgres)
`ngeneahub_redis_1`	Task results backend and celery broker

Health endpoints

To view the state of all sites and related nodes within known to Ngenea Hub, a GET request can be performed to `/api/health` to view all of the sites, nodes and the hub service itself. The states are currently based on how many nodes are online for each site using the following states:

State	Description
`ok`	All nodes are functional

`warning`	Some nodes are offline within a site
`critical`	One or more sites are completely offline

An example output of the health endpoint can be seen below:

~~~
```json {

    "overall_health": "ok", "hub_status": {

        "health": "ok"

    }, "site_status": [

        {
           "site": "site1", "health": "ok", "nodes": [

               {
                  "name":  "pixstor-east-ng-test",  "health":  "ok",
                  "online": true

               }

           ]

        }, {

            "site": "site2", "health": "ok", "nodes": [

                {
                   "name": "pixstor-west-ng-test", "health":
                   "ok", "online": true

                }

            ]

        }

    ]
~~~

```
}
```

A request can also be performed to specific sites using `/api/sites/ID/health/` to view the site specific health status:

~~~
```json {

    "site": "site1", "health": "ok", "nodes": [

        {
           "name": "pixstor-east-ng-test", "health": "ok", "online": true

        }
~~~

```
    ]
}
```

Custom Workflows

Defining workflows

It's possible to define custom workflows which use pre-defined rules as building blocks to create your workflow.

> **Note:** Custom workflows are not currently exposed via the UI. Use the API `/api/workflows/` endpoint to create custom workflows

A workflow definition requires the following parameters:

Name	Description
`name`	The unique name for this workflow. For easy of submission again the API, this should not contain spaces.
`label`	The human readable name for this workflow, can contain spaces.
`icon_classes`	List of icon classes to represent the workflow in the UI. Font Awesome is useful here.
`filter_rules`	A list of rules to apply to provided files that match defined states. Described in more detail below.
`fields`	A list of runtime fields. Described in more detail below.

Additionally, you can optionally provide:

Name	Description
`discovery`	Which discovery task the workflow should be used by default, this can be either recursive or snapdiff.
`discovery_options`	A json containing any additional options to pass to the workflow default discovery. Described in more detail below.

Filter Rules

Filter rules are defined in JSON. They are a list of individual rules in a mapping format that will be performed on each matching file result when a discovery task is complete. If called through the API with no discovery task provided, rules will be applied to any states provided in the workflow input.

Steps are defined in JSON. Steps is a list of individual steps that will be performed serially. Each rule must contain the following:

Name	**Description**	**Required**
`state`	The state of a result provided by the discovery task with any given path, an example of that could be "processed" or "modified" more details about this are in the discovery section.	Yes
`type`	The type of result the rule will apply to, the only valid types are: `file|directory|symlink|all`	Yes
`action`	A list of tasks to perform on files that match the state and type	Yes
`include`	A list of globs to apply to provided files to limit actions to just them.	No
`exclude`	A list of globs to apply to provided files. Described in more detail below.	No
`ignore_site_includes`	Whether to ignore any global includes defined on the site the workflow will run on	No
`ignore_site_excludes`	Whether to ignore any global excludes defined on the site the workflow will run on	No

These rules control which actions will be performed on certain files based on their given state that they have been given following specific discovery tasks such as `snapdiff` or provided in the initial input of a workflow. These states can allow direct control of workflows performed on files provided, allowing multiple workflow paths within the same job by utilizing multiple rules controlling specific states with additional control with include and exclude path rules.

Alongside rules bound to a state, there are two special states that rules can be used, these being `default` and `all`. Rule sets cannot have both `default` and `all` rules within them, but it is possible to have multiple of one type with different sets of exclude and include rules to allow for more granular control.

Rules defined with `default` as their `state` and `type` will perform their action on paths that have not been captured by all other rules within a given rule set. This means that if there are specific file states that need to be actioned differently, paths that do not match any other rules actioned against without ignoring those non-matching paths.

The other special rule type is rules with the `state` and `type` of `all`. This rule will perform its action on all paths regardless of their provided type and state. This is an additional operation so if another rule has an explicit rule provided it will perform multiple actions on the same path, for each matching rule in rule set. Simple workflows are typically composed of a single rule with the state and type of `all` as this will simply process all paths provided to it.

Within each rule, there must be a list of actions to perform on the resulting file provided within the `action` key. These actions will be performed serially. Each action must be a mapping that contain the following in each entry:

Name	**Description**	**Required**
`name`	The name of the task to run, e.g. `dynamo.tasks.migrate`	Yes
`site`	The name of the site to run against, if this is not provided it will use the site provided within the workflow call.	No

If steps have optional arguments, these can be passed as additional key:value pairs in these step definition mapping to pass those optional arguments.

As an example we can define a generic rule that captures every type of file and state and sends it to a second site, this would be useful for a bulk move using the `recursive` discovery task to cover all types of files in directories provided to the task:

Example 1 - Send to london

```
{
    "state": "all",
    "type": "all",
    "action": [
        {
            "name": "dynamo.tasks.migrate"
        },
        {
            "name": "dynamo.tasks.reverse_stub",
            "site": "london"
        }
    ]
}
```

Runtime fields

A workflow needs to be able to accept parameters as it submitted. Taking example #1 above, "london" doesn't want to be hardcoded as the destination site, as that would mean a new workflow would need to be defined for each possible destination.

Instead, fields can be defined, that in turn will need to be provided at workflow submission time. Fields are defined as a mapping with the following keys:

Name	Description	Required
`name`	The name of the field.	Yes
`label`	The friendly name for this field, used for presenting in the UI	Yes
`type`	The type of the field, valid options are: • `string` - a free text field • `int` - a free text field that will be validated a integer • `bool` - a checkbox • `choices` - A dropdown box representing a list of choices, populated from choices list of objects. • `enum[enum_type]` - A dropdown box representing a choice of option, populated from enum_type. `enum_type` can be one of the following ◦ `site` - A list of all the sites Ngenea Hub has defined • `list` - a list of values of any scalar type	Yes
`default`	The default value for runtime fields	optional

The following is an example of a custom field definition for providing a site to an action step:

Example 2 - Custom field definition

```
[
    {
        "name": "target_site",
        "label": "Site to migrate to",
        "type": "enum[site]"
    }
]
```

Custom field defintion with default value

```
[
    {
        "name": "target_site",
        "label": "site to migrate to",
        "type": "enum[site]",
        "default": "london"
    }
]
```

If default value is specified in runtime fields, it will take the default value for fields while running workflow if the user input is not given otherwise it will always use the user input.

Back in the definition of an action step, any value that is prefixed with a `*` will be used as a field name and the value replaced instead of a literal string.

The following example, modifies example #1 to use the custom field as defined in example #3:

Example 3 - Updated rule now using custom fields

```
{
    "state": "all",
    "type": "all",
    "action": [
        {
            "name": "dynamo.tasks.migrate"
        },
        {
            "name": "dynamo.tasks.reverse_stub",
            "site": "*target_site"
        }
    ]
}
```

So, a complete request to create a workflow that will process all file and state types with a dynamic "site" field will look like:

Example 4 - Full workflow request

```
{
    "name": "send_file",
    "label": "Send files from one site to another",
    "icon_classes": ["fa fa-cloud fa-stack-2x text-primary", "fa fa-
refresh fa-stack-1x text-light"],
    "filter_rules": [
        {
            "state": "all",
            "type": "all",
            "action": [
                {
                    "name": "dynamo.tasks.migrate"
                },
                {
                    "name": "dynamo.tasks.reverse_stub",
                    "site": "*target_site"
                }
            ]
        }
    ],
    "fields": [
        {
            "name": "target_site",
            "label": "Site to migrate to",
            "type": "enum[site]"
        }
    ]
}
```

The following is an example of a custom field definition for providing a choices to an action step:

Example 5 - Custom field definition

```
[
    {
        "name": "sync_policy",
        "label": "sync_policy",
        "type": "choices",
        "choices": [
            {
               "label": "Newest",
               "value": "newest"
            },
            {
               "label": "Sourcesite",
               "value": "sourcesite"
            }
        ]
    }
]
```

choices support both string and integer type values.

Back in the definition of an action step, any value that is prefixed with a `*` will be used as a field name and the value replaced instead of a literal string.

The following example, uses the custom field in action:

Example 6 - Updated rule now using custom fields

```
{
    "state": "all",
    "type": "all",
    "action": [
        {
            "name": "dynamo.tasks.migrate"
        },
        {
            "name": "dynamo.tasks.reverse_stub",
            "sync_policy": "*sync_policy"
        }
    ]
}
```

So, a complete request to create a workflow that will process all file and state types with a static choices field will look like:

Example 7 - Full workflow request

```
{
    "name": "send_file",
    "label": "Send files from one site to another",
    "icon": "<span class='fa-stack'><i class='fa fa-cloud fa-
stack-2x text-primary'></i><i class='fa fa-angle-right fa-stack-2x
text-light'></i></span>"
    "filter_rules": [
        {
            "state": "all",
            "type": "all",
            "action": [
                {
                    "name": "dynamo.tasks.migrate"
                },
                {
                    "name": "dynamo.tasks.reverse_stub",
                    "sync_policy": "*sync_policy"
                }
            ]
        }
    ],
    "fields": [
        {
            "name": "sync_policy",
            "label": "sync_policy",
            "type": "choices",
            "choices": [
               {
                  "label": "Newest",
                  "value": "newest"
               },
               {
                  "label": "Sourcesite",
                  "value": "sourcesite"
               }
            ]
        }
    ]
}
```

Running Workflows

Once a workflow has been defined, it can be performed through the file browser by selecting files and directories and clicking the actions button. It is then possible to select the workflow you wish to call, this workflow call will not use a discovery task unless a directory is selected, in that case it will make use of the `recursive` discovery step.

This can also be performed via a POST request to `/api/file/workflow`. When called through the API, you have the option to provide a discovery step, these steps can expand the initial paths provided to them to either recursively perform actions or perform something like a file difference scan.

Name	**Description**	**Type**	**Required**
`paths`	A list of paths to perform the workflows against, these can be just strings of file absolute file paths or can be JSON with the keys of "path" and "state", detailed example in example 7	JSON List	Yes
`site`	The site to perform the workflow against	String	Yes
`fields`	The runtime fields for a workflow	String	Yes
`discovery`	The discovery phase to use for this workflow run, this will override any defaults	String	No
`job`	The ID of a job that this workflow should be run within	Integer	No

Following the example workflow defined above, you can call the workflow to recursively send all files within any paths provided using the following POST to `/api/file/workflow`:

Example 8 - Calling example workflow

```
{
    "paths": [
        "/mmfs1/data/project_one",
        "/mmfs1/data/project_two"
    ],
    "site": "london",
    "workflow": "send_file",
    "discovery": "recursive",
    "fields": {
        "target_site": "dublin",
    }
}
```

This will now migrate all files within `/mmfs1/data/project_one` and `/mmfs1/data/project_two` and then recall them at the site defined as `dublin`.

If there is a more complex workflow that have been defined that includes rules for specific states, the input paths can include this state information. This behaviour can be only be used when no discovery state is provided, an example of a custom rule set using could be:

Example 9 - Calling workflow with state data

```
{
    "name": "migrate_state",
    "label": "Stateful file migration",
    "filter_rules": [
        {
            "type": "all",
            "state": "modified",
            "action": {
                "name": "dynamo.tasks.migrate"
            }
        {
            "type": "all",
            "state": "moved",
            "action": {
                "name": "dynamo.tasks.delete_paths_from_gpfs"
            }
        }
    ],
    "discovery": null,
    "fields": []
}
```

Here is a simple rule set that will migrate all paths provided with the state `modified` and will delete all paths provided with the state `moved`. With this example workflow provided you can perform a POST to `/api/file/workflow` with the following JSON:

Example 10 - Calling workflow with state data

```
{
    "paths": [
        {
            "path": "/mmfs1/data/project_one",
            "state": "modified"
        {
            "path": "/mmfs1/data/project_two",
            "state": "moved"
        }
    ],
    "site": "london",
    "workflow": "migrate_state",
    "discovery": null,
    "fields": {}
}
```

Using multiple state based rules with different include and exclude path filters, you could achieve more complex behaviour in workflow calls for more finite control.

Discovery Steps

Discovery steps can make complex large bulk operations much more manageable to call, allowing you to provide a single path that expands to cover all the contents of a path, or to see time based differences for a given path.

Note: If a workflow is submitted without a discovery task explicitly provided, it will default to using the discovery task defined as the default during the workflow's creation, visible via the workflow's "discovery" attribute. To avoid this, it is possible to expliclty pass `null` as the discovery task via the API to skip any discovery phase and additional processing on the paths provided and instead process the actions specified using the rules, without any additional checks.

Name	Description	Supported states
`recursive`	Performs a recursive expansion of the initial provided paths. This allows paths to be expanded to cover all sub file and directories, it will then perform the defined action for all the generic rules in a workflow against all resulting files.	`all`
`snapdiff`	Performs a time based file scan on an independant fileset between the last time a scan was performed. It will retrieve all file differences between those moments in time and the state of that file.	`created|updated|moved|deleted|all`

For more complex discovery steps such as `snapdiff`, there are defined states that files, directories and links can be in once it has completed its scan. This allows more explicit control of file and state control within a single call to a workflow. If for example you want all results with the type of `file` that have the state `created` to be sent to another site without any temporary files, a rule to cover that could be:

Example 11 - Custom rule for filtering snapdiff discovery results

```
{
    "state": "created",
    "type": "file",
    "exclude": ["*.temp"],
    "action": [
        {
            "name": "dynamo.tasks.migrate"
        },
        {
            "name": "dynamo.tasks.reverse_stub",
            "site": "*target_site"
        }
    ]
}
```

Discovery Options

Additional options can be passed to the discovery task by setting `discovery_options` on the workflow. The supported options are those described in Discovery Steps.

Example 12 - Passing options to the discovery

```
{
    "discovery": "recursive",
    "discovery_options": {
        "skip_missing": false
    }
}
```

Discovery options will be used with the workflow default discovery only. If a different discovery is set at runtime, the discovery options will be ignored. Discovery options cannot currently be overwritten at runtime.

Includes and Excludes

Includes and excludes can be used to select paths that individual filter rules should apply to, or generally limit which paths should be handled during a workflow run.

Include/exclude patterns behave like unix shell pattern matching ('globbing'). The 'wildcard' asterisk character `*` will match any characters within a string. Patterns must match whole paths; partial matches are not supported, except through the use of wildcards.

Includes and excludes are combined as "a path matching any includes and not any excludes". For example `{"include": ["/mmfs1/data/*"], "exclude": ["*.tmp"]}` would match only files in `/mmfs1/data`, but not files in that directory with the `.tmp` extension. If no includes are defined, then all files are considered included (unless explicitly excluded).

There are three places where path include and exclude patterns can be defined:

- on a site
- within a filter rule
- at runtime, when a workflow is submitted

Site patterns can be used to apply includes and excludes globally, to all workflows and workflow steps. If defined, these will be appended to any patterns defined within a filter rule or at runtime.

For example, if a rule defines `{"exclude": ["*.tmp"]}` and the site defines `{"exclude": ["*.cache"]}`, then the combined excludes for that rule would be `{"exclude": ["*.tmp", "*.cache"]}`

If not desired, this behaviour can be overridden by specifying `ignore_site_includes` / `ignore_site_excludes` either on a per-rule basis, or for all rules by passing those parameters when submitting a workflow run.

For workflows involving multiple sites, such as send and sync, only the primary (source) site patterns will be considered.

If includes and excludes are passed when submitting a workflow, they will be applied to all filter rules within the workflow, replacing any patterns already defined within the workflow rules. Any site patterns will be appended to the runtime patterns, unless 'ignore' is specified.

Jobs

Housekeeping

Job details are kept for a time after job completion. During this time, they can be viewed in the UI.

But older jobs are periodically culled from the database, to keep it to a manageable size. The housekeeping process runs once a day, and removes results for any job that completed more than 90 days ago (by default). A different 'time-to-live' (TTL) can be set using the `jobs_ttl` configuration - see Configuration for more information.

Site Sync

Ngenea Hub provides the facility for syncing data from one site to another.

Site Sync applies changes in one direction. See also Bidirectional Site Sync for applying changes in both directions.

Synchronisation is achieved by utilising a scheduled workflow which periodically discovers file and directory changes within an Independent Fileset on a source site and applying those changes to a target site. Changes are applied by sending newly created or recently modified files and directories, including deleting or moving files or directories in place on the target site as necessary to match the source site.

The following walkthrough details how to set up a sync between two sites ('Site Sync') using the Ngenea Hub UI.

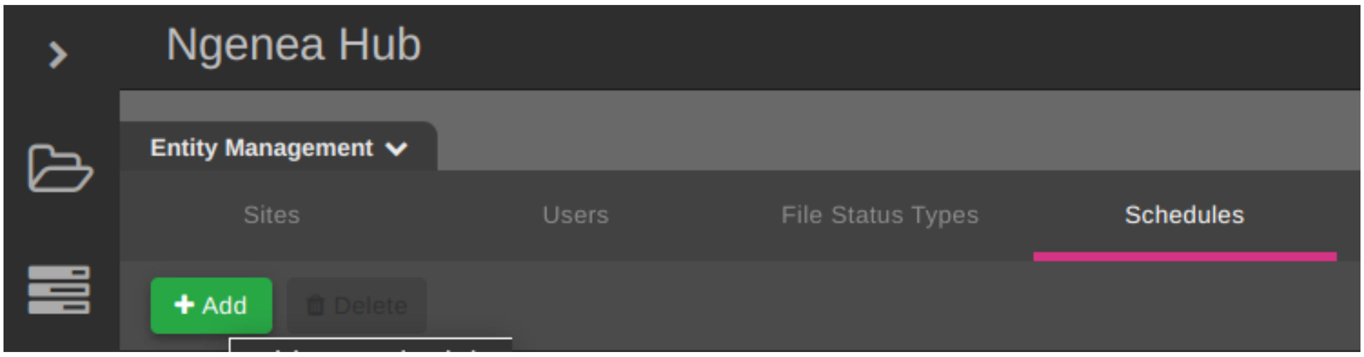
The REST API can also be utilised to achieve the same setup.

Schedule

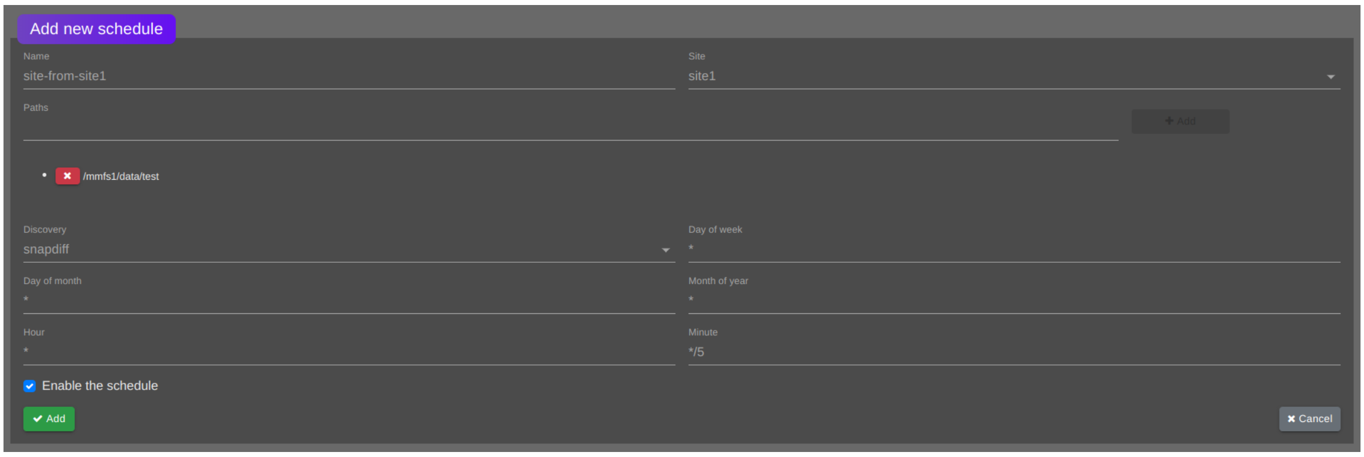
A schedule defines when and how often a sync occurs.

In the Ngenea Hub UI, navigate to the `Administration` page, and select the `Schedules` tab.

To create a new schedule, click the `Add` button.



Configure the schedule by entering appropriate details into the `Add new schedule` dialog:


Add new schedule

Name: site-from-site1
Site: site1
Paths
+ Add
- /mmfs1/data/test

Discovery: snapdiff
Day of week: *
Day of month: *
Month of year: *
Hour: *
Minute: */5

☑ Enable the schedule

Add
Cancel

1. Enter a descriptive `Name` for the schedule, E.G. "sync-to-site2"
2. Select the `source` site from the dropdown menu
3. Set the `Discovery` to `snapdiff`. Snapdiff discovery uses snapshots to track changes over time within an Independent Fileset.

4. Set the `Path` to the top level directory (path) of the Independent Fileset to be synchronised
5. Set the time criteria
6. Click the `Add` button to add the new schedule

Time criteria

A schedule utilises cron syntax. By default, all the fields are populated with `*` - this means the schedule will run every minute. This is likely too frequent as the calculation of file changes could take longer than the time between the schedule running every minute. Sizing for an ideal setup will balance the need to sync files quickly versus how long the sync snapdiff stage takes to run. This can be most effectively achieved through observation.

Examples:

- To enact the schedule every 5 minutes, specify the `Minute` field as follows: `*/5`.
- To enact the schedule once an hour, set the `Minute` field to a number of minutes past the hour. Setting `Minute` to `5` will run the sync once an hour at 5 minutes past the hour.
- To enact the schedule at set times throughout the hour, change the `Minute` field separating entries by commas (E.G.) `7,23,46`

Workflow

Schedules enact a workflow. A workflow performs one or more series of tasks to files and/or directories.

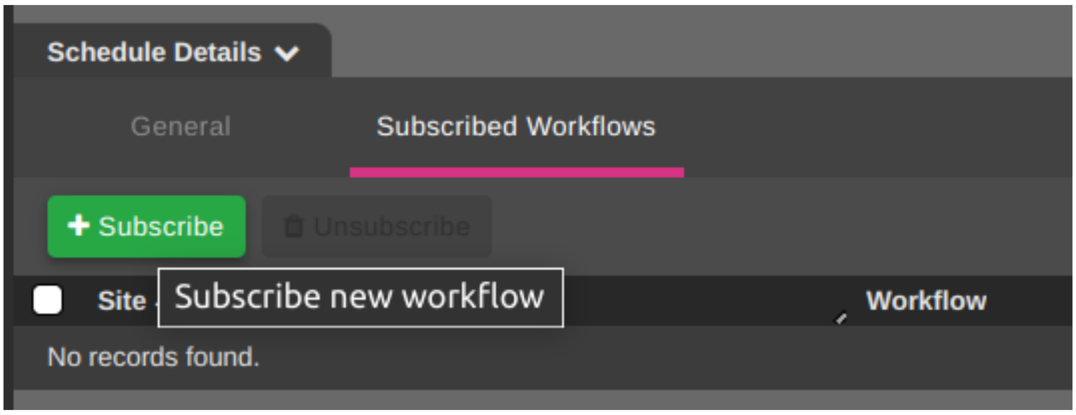
In the Ngenea Hub Hub UI, navigate to the `Administration` page, and select the `Schedules` tab.

Select the prior created schedule in the `Schedules` tab, which then displays the `Schedule Details` page.

Subscribing a workflow

To associate a workflow with a schedule undertake the following actions:

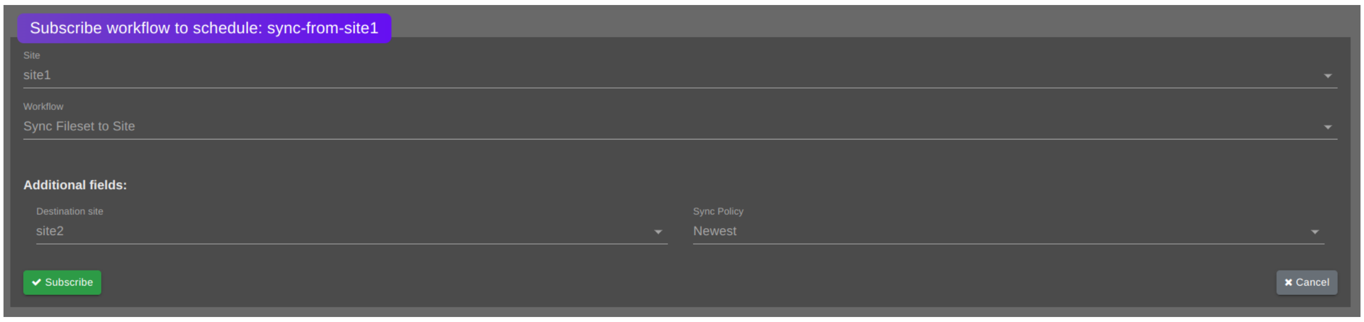
1. Click `Subscribe` and in the create dialogue set `Site` to the source site from which to synchronise.



Ngenea Hub provides a built-in workflow for synchronising changes from an Independent Fileset on a source site to a target site.

1. Select `Sync Fileset to Site` from the `Workflow` dropdown menu.

Configure the workflow subscription settings by entering appropriate details into the `Subscribe workflow to schedule` dialog:


Subscribe workflow to schedule: sync-from-site1

Site: site1

Workflow: Sync Fileset to Site

Additional fields:

Destination site: site2

Sync Policy: Newest

1. Select the target site in the `Destination site`
2. Select an appropriate `Sync Policy`
3. Click `Subscribe` to associate the workflow to the schedule

Sync Policy

The `Sync Policy` setting controls how conflicts are resolved. A conflict may occur if (E.G.) different versions of the same file exist on both source and target sites. By default, the newest version of the file will be retained.

Outcome

Now associated (subscribed), files and/or directories will synchronise at the next scheduled execution time. When executing the scheduled workflow, two jobs appear within the schedule details page. One for the snapdiff discovery, and one for the subscribed sync workflow.

The first enactment of the sync will synchronise all files and directories in the Independent Fileset from the source site. Further enactments will synchronise files and/or directories created, modified, deleted or moved since the last enactment and the time of the most recent enactment.

Additional Options

Associating more than one workflow to the same schedule can be utilised to synchronise from the same source site to another target site. Additional workflows cause creation of an additional job, and run simultaneously with other syncs enacted from the same schedule.

Troubleshooting

If you encounter issues with sync, refer to the sync troubleshooting guide page for guidance.

Bidirectional Site Sync

Ngenea Hub provides the facility for syncing data between sites.

Bidirectional Site Sync is similar to Site Sync, but applies changes in both directions. If a Bidirectional Site Sync runs between sites A and B, changes on site A will be applied on site B, followed by changes on site B being applied on site A.

Synchronisation is achieved by utilising a scheduled workflow which periodically discovers file and directory changes within an Independent Fileset on each of the sites and applies those changes to the other site. Changes are applied by sending newly created or recently modified files and directories, including deleting or moving files or directories in place on the target site as necessary to match the other site.

A prerequisite for a working Bidirectional Site Sync is that the Independent Fileset has been created on both sites.

The principles of setting up Bidirectional Site Sync are similar (but with some differences explained below) to setting up a Site Sync, so please refer to the walkthrough on the Site Sync page for detailed instructions. It can be done using the Ngenea Hub UI or the REST API.

Schedule

Discovery

Create a new schedule. Unlike with Site Sync, for Bidirectional Site Sync set the schedule's `Discovery` to `bidirectional_snapdiff`.

This discovery type operates identically to the `snapdiff` discovery type, with snapshots used to track changes over time within an Independent Fileset.

A separation is enforced between `bidirectional_snapdiff` schedules and schedules with other discovery types. Whereas the `Discovery` of other schedules may be changed, a change from or to the `bidirectional_snapdiff` discovery type is not allowed. This is to ensure consistency of the specific workflow rules that apply to Bidirectional Site Sync.

Time criteria

Unlike with Site Sync, a Bidirectional Site Sync will not be triggered if the previous sync is still running. This means that some syncs will be skipped if scheduled closer together than the time it takes to complete a sync.

Workflow

Subscribing a workflow

Select the `Bi-directional Site Sync` workflow.

This is the only workflow that may be subscribed to a `bidirectional_snapdiff` schedule, and it cannot be subscribed to other types of schedule.

A `bidirectional_snapdiff` schedule may have at most one subscribed workflow.

Outcome

When executing the scheduled workflow, only one job is created and appears within the schedule details page. This job includes the tasks for both the snapdiff discoveries (one on each site), as well as the tasks that enact the sync in each direction.

Troubleshooting

If you encounter issues with sync, refer to the sync troubleshooting guide page for guidance.

Search

Note: Before you can use the search feature, additional set-up is necessary, as described in the search feature page.

The search endpoint provides the ability to search for files across multiple sites, and aggregate the results.

Search is performed in two steps - submitting a query, and retrieving the results.

Submitting a query

Search performs a query by submitting asynchronous tasks to each requested site. The sites then perform the actual search and return results as available.

A search is initiated by `POST`ing a query to the search endpoint

```
curl -s -X POST 'http://example.com/api/search/'  -H 'Accept:
application/json'  -H 'Content-Type: application/json'  -H
"Authorization: Api-Key $TOKEN"  -d '{"path": "/mmfs1/data",
"sites": ["site1"], "recursive": true, "filters": {"hsm.status":
"migrated"}}'
```

The search request payload is made of

name	description
`path`	Directory to query
`sites`	List of one or more sites to search. Default: all sites
`recursive`	Whether to search the `path` recursively. Default: false, only immediate children of `path` will be returned.

name	description
`filters`	A collection of filters against arbitrary metadata, see below. Default: None
`metadata_fields`	A list of metadata fields to include in search results. This can include specific field names (e.g. `hsm.status`), or namespace wildcards (e.g. `core.*`) to select all fields in a given namespace. Default: all available fields.
`merge`	If the same file exists on multiple site, this will cause them to be merged in the results (see below). Default: false

Upon successful submission, the request will return status `201 (Created)`, and a response body which includes the url for retrieving search results (see below)

```
{"id":1,"url":"http://example.com/api/search/1/"}
```

Filters

Filters are a collection of filters to apply to arbitrary file metadata.

The specific metadata available to be filtered on depends on the search backend being used. The fields in the following examples may not be available for all backends.

At a minimum, one can expect to be able to filter on `core.filename`, the file basename. For example to filter only jpeg files, `{"core.filename": "*.jpg"}`

Possible filter types are

type	description	example
exact match	match a value exactly	`{"core.filename": "cats-01.jpg"}`, `{"core.size": 0}`
match list	match any of the values in the list (value1 OR value2 OR ...)	`{"core.group.name": ["editor", "admin"]}`
wildcard	any string value containing an asterisk (`*`) is treated as a wildcard	`{"core.filename", "*.jpg"}`
range	numerical or date range, using any combination of less-than (`lt`), less-than-equal (`lte`), greater-than (`gt`), greater-than-equal (`gte`)	`{"core.modificationtime": { "gte": "2021-01-01", "lt": "2021-02-01"}}`, `{"core.size": {"gt": 1000000000}}`
negation		

type	description	example
	exclude anything matching a given filter	`{"not": {"core.filename": ".DS_Store"}}`

Filters are combined as AND, e.g. `{"core.extension": ".jpg", "hsm.status": "migrated"}` matches `.jpg` files which are HSM migrated.

Retrieving results

When search results are read, they can be retrieved using the url returned when the query was submitted.

```
$ curl 'http://example.com/api/search/1/' -H "Authorization: Api-Key
$TOKEN"
{
  "count": 1,
  "next": null,
  "previous": null,
  "items": [
    {
      "href": "http://example.com/api/file/?
path=%2Fmmfs1%2Fdata%2Fhello.txt&site=site1",
      "site": "site1",
      "path": "/mmfs1/data",
      "name": "hello.txt",
      "metadata": {
          "core.accesstime": "2021-10-12T16:27:28",
          "core.changetime" : "2021-10-12T16:28:45",
          "core.directory" : "/mmfs1/data",
          "core.extension" : ".txt",
          "core.filename" : "hello.txt",
          "core.group.id" : 0,
          "core.group.name" : "root",
          "core.hash.sha512": "db3974a97...94d2434a593",
          "core.modificationtime" : "2021-10-12T16:28:45",
          "core.pathname": "/mmfs1/data/hello.txt",
          "core.size" : 12,
          "core.user.id" : 0,
          "core.user.name" : "root",
          "gpfs.filesetname" : "root",
          "gpfs.filesystem" : "mmfs1",
          "gpfs.kballocated" : 0,
          "gpfs.poolname" : "sas1",
          "hsm.status" : "migrated"
          "ngenea.pathname" : "data/hello.txt",
          "ngenea.size" : 12,
          "ngenea.target" : "awss3",
          "ngenea.uuid": "acf1a307-5b6a-43b0-8fb2-d2b366e88008",
      }
    }
  ],
  "metadata_fields": ["core.accesstime", ...],
  "complete": true,
  "errors": {"site2": "Search backend is offline"}
}
```

Results from different sites may not arrive at the same time. The `complete` field indicates whether all sites what returned their results. This includes when a site returns with an error.

Results from different sites are 'concatenated', meaning if the same file exists on multiple sites, there will be separate result items for the file for each site.

The `metadata` field on each item contains arbitrary file metadata. The specific metadata will vary depending on the search backend being used. In the case of the PixStor Search backend, the available fields will vary depending on file type, and which plugins were used when the files were ingested.

If `metadata_fields` was specified when the query was submitted, the `metadata_fields` entry in the response will match, with any wildcards expanded to list the avilable fields which match those wildcards. Otherwise, the `metadata_fields` entry will list all the available metadata fields which could be returned from the search backend. Individual files may not have all the listed fields.

All search backends format results to be namespaced, similar to PixStor Search, for consistency.

If an error occurs while performing the search on any of the sites, the `errors` entries will provide a mapping of site names and error messages.

Parameters

Search results are paginated. The following parameters can be used to control what results are returned

name	description
`page`	Numbered page of results to fetch. Default: 1
`page_size`	Maximum number of results to return per page. Default: 20
`sort`	One or more fields to sort results on, separated by commas, e.g. `?sort=name,site`. Field names can be prefixed with `-` to reverse order. For fields in `metadata`, the field name is specified as is, e.g. `?sort=-core.accesstime`. Default: arbitrary order.

Merged results

When a search is submitted with `"merge": true`, the search results will be 'merged'.

This means that entries for matching files from different sites will be combine. An entry is considered to be matching if it has the same full path.

```
$ curl 'http://example.com/api/search/2/' -H "Authorization: Api-Key
$TOKEN"
{
  "count": 1,
  "next": null,
  "previous": null,
  "items": [
    {
      "path": "/mmfs1/data",
      "name": "hello.txt",
      "metadata": {
          "core.accesstime": "2021-10-12T16:27:28",
          "core.changetime" : "2021-10-12T16:28:45",
          "core.directory" : "/mmfs1/data",
          "core.extension" : ".txt",
          "core.filename" : "hello.txt",
          "core.group.id" : 0,
          "core.group.name" : "root",
          "core.hash.sha512": "db3974a97...94d2434a593",
          "core.modificationtime" : "2021-10-12T16:28:45",
          "core.pathname": "/mmfs1/data/hello.txt",
          "core.size" : 12,
          "core.user.id" : 0,
          "core.user.name" : "root",
          "gpfs.filesetname" : "root",
          "gpfs.filesystem" : "mmfs1",
          "gpfs.kballocated" : 0,
          "gpfs.poolname" : "sas1",
          "hsm.status" : "migrated"
          "ngenea.pathname" : "data/hello.txt",
          "ngenea.size" : 12,
          "ngenea.target" : "awss3",
          "ngenea.uuid": "acf1a307-5b6a-43b0-8fb2-d2b366e88008",
      },
      "status": {
         "site1": true,
         "site2": false
      }
    }
  ],
  "metadata_fields": ["core.accesstime", ...],
  "complete": true
}
```

Merged results no longer have the `site` and `href` fields. In their place is a `status` field, which maps sites to whether the file is 'resident' on that site.

A file is considered resident if the file is not migrated, or is premigrated ('hydrated'). A file is considered not resident if the file is migrated (stubbed), or not present at all.

Max Results

There is a hard limit on the number of results returned, per site. By default, each site will return, at most, 200 results.

Fetching a lot of results makes queries slower and, since results are stored in the DB, storing more results uses more space. One the other hand, the limiting may lead to some matches not being returned.

The maximum number of results per site is controlled by the `search_max_results` configuration - see Configuration for more info.

Result limiting is applied when the search query is submitted, not when results are retrieved. If you change `search_max_results`, you will need to resubmit your query to fetch any additional matches.

Note, some backends have a hard limit of 10,000 results.

Housekeeping

The results from a query are stored, so they can be retrieved multiple times without performing a new query.

However, over time, the files on each site will change, and the stored results may no longer accurately reflect the active file system.

Therefore, old results are periodically culled. The housekeeping process runs once a day, and removes results for any search which was submitted more than a week ago (by default). A different 'time-to-live' (TTL) can be set using the `search_result_ttl` configuration - see Configuration for more information.

Results can also be manually removed by performing a `DELETE` request against the given search result endpoint

```
curl -X DELETE 'http://example.com/api/search/1/' -H "Authorization: Api-Key $TOKEN"
```

Controlling Bandwidth Usage

Ngenea Hub can control the amount of traffic speed for each node within each site that processes files through Ngenea. This will limit both outgoing and incoming traffic with defined cloud services.

Enabling the bandwidth feature in the UI

Regardless of enabling this feature, it is possible to change the bandwidth via REST and will enable changing of bandwidth when the `Site` instance has its `bandwidth` attribute changed.

This can be enabled by enabling the feature flag, details on how to do this can be found on the Feature Flags page.

This will enable the UI within the Site details page and represents the bandwidth limit in Mb/s.

Checking Node status

Each site within Ngenea Hub is the collection of all nodes running an instance of Ngenea Worker using the name of a given site. These are nodes within a cluster that are collectively listening to the same queue for a given site. Each time any worker comes online on a new node or a known node, this is tracked within Ngenea Hub using `Node` objects. These are automatically created when first starting up a worker, after creation their online status can be monitored.

You can view the nodes for each site within `/api/nodes/` this will be a complete list of all nodes known to Ngenea Hub.

For bandwidth control, there will need to be existing nodes known to Ngenea Hub, otherwise the bandwidth rules cannot be applied. If your worker coming online was not tracked due to timing issues, you can manually scan for existing nodes using one of the Actions.

Registering Datastores

In order to control the bandwidth for all nodes under a site, the site will need to have the Ngenea policy targets defined as `Datastore` instances within Ngenea Hub.

The following example is for defining an AWS S3 target for Ngenea within Ngenea Hub:

```
{
    "name": "site1_amazon",
    "type": "S3",
    "bucket": "bucket01",
    "secretaccesskey": "secret-key",
    "accesskey": "access-key"
}
```

With this established datastore, all that is left to do is link the created datastore to a site so that when a change in bandwidth is applied, the site knows what service to limit the traffic to.

Cloud IPAddresses

Each datastore that points to a cloud target will have access to a list of all known IP addresses that are associated with that specific service. This will be used to limit all traffic between those IP ranges and the nodes currently running a worker instance when a bandwidth limit is applied.

These IP ranges are updated internally once a day in a scheduled task.

Using manual IP addresses

Manual IP addresses can be used to override the list of cloud related IPs, this can be useful to control bandwidth to custom endpoint targets such as services like minio making use of POST `api/ipaddresses/`. Using the following example to add an address to the list of IPAddresses:

> **Note:** This will disable the use of all cloud addresses for a datastore and will instead only use the custom IP addresses.

```
{
    "ipaddr": "208.65.153.237",
    "datastore": 1
}
```

This will ensure that all traffic between those nodes and the defined IP addresses will be limited to the max bandwidth limit

Linking a Datastore to a Site

With our example site `site1` creating a `SiteLink` between `site1` and the datastore `site1_amazon` allows the bandwidth to be applied to all S3 targets on all nodes that are running the Ngenea Worker service.

For this example all local data for this Ngenea is located in `/mmfs1/data/aws_data` with data being placing data in the bucket `bucket01` under `aws_data/`.

The following `SiteLink` can be used to represent this:

```
{
    "site": "site1",
    "datastore": "site1_amazon",
    "site_path": "/mmfs1/data/aws_data/",
    "datastore_path": "aws_data/"
}
```

This will ensure that each `Node` under `site1` will limit its traffic to all related addresses.

Applying the bandwidth

> **Note:** This will effect all traffic on each node running a worker to the cloud services defined with the sites linked datastores.

With the SiteLink in place, changing the bandwidth attribute to the Mb/s desire on the site instance via the API route PATCH `/api/site/{id}`:

```
{
  "bandwidth": 1000
}
```

Using this or the UI, this will cause the hub to signal the worker to limit traffic using the IP ranges defined for the datastores.

This total bandwidth will be divided between all nodes for any given site, so if a site has a bandwidth of 1Gb/s then both nodes will be limited to 500mb/s.

Configuration

Global Configurations

Some configurations are stored in the Ngenea Hub configuration file, as described in Hub Configuration. These are generally static, or sensitive settings. Changes to these settings require a service restart.

In addition, there are configurations which can be changed on-the-fly, typically to change Ngenea Hub behaviour. These settings can be viewed and changed via the REST API, as described below.

Available Settings

Name	Description	Default
jobs_ttl	How long job details should be stored after job completion, in days.	90
search_backend	Backend to use when performing searches. Currently supported backends: `analytics`, `pixstor_search`.	`analytics`
search_result_ttl	How long search results should be stored, in days.	7
search_max_results	Maximum number of search results to fetch from the search backend, per site. Fetching more results will make queries slower and will require more storage space. Fetching fewer results may lead to some files being missing. Note, some backends have a hard limit of 10,000 results.	200
snapshot_create_delete_retry_timeout	Time in seconds after which workers will give up retrying snapshot create and delete operations. These occur in the `dynamo.tasks.snapdiff` and `dynamo.tasks.rotate_snapshot` tasks. The default 1000s will give 4 or	1000

Name	Description	Default
	more retries. Set to 0 to disable retries.	
stat_timeout	How long to wait for results from the `/api/file/` endpoint, in seconds.	10
stat_refresh_period	How long a files details will be retained as a cache within the file browser, in seconds.	10
task_invalidation_timeout	How long in minutes for a task in the STARTED state will wait before being invalidated.	360
snapdiff_stream_timeout	Idle timeout when waiting for delete and move tasks to complete during snapdiff workflows, in minutes.	10

Note: If the file details are not returned for large fileset, increase the default value of `stat_refresh_period` in the configuration tab of hub admin page http://:/admin or do a patch on configuration API.

REST API

Configurations can be listed and set via the Ngenea Hub REST API.

Note: The configurations endpoint does not support client key authentication. You must use JWT Authentication.

To list the current configuration settings,

```
$ curl -s 'http://example.com/api/configurations/' -H 'Accept:
application/json' -H 'Authorization: Bearer $JWT_ACCESS_TOKEN'
{
    "search_backend": "analytics",
    "search_max_results": 200,
    "search_result_ttl": 7,
    ...
}
```

To change one or more configuration settings, make a `PATCH` request the same endpoint

```
$ curl -s -X PATCH 'http://example.com/api/configurations/' -H
'Accept: application/json' -H "Authorization: Bearer
$JWT_ACCESS_TOKEN" -H 'Content-Type: application/json' -d
'{"search_max_results": 500}'
{
    "search_max_results": 500,
    ...
}
```

Settings Migration

In versions 1.9.0 and earlier, some of the above settings were configured via the Ngenea Hub config file (Hub Configuration).

Upon updating to version 1.10.0 or above, any values currently set in that config file will be captured. Thereafter, any changes to those settings within the config file will be ignored.

Site-specific Configurations

Some configuration options can be set on a per-site level, and may differ between sites.

These can be viewed and changed via the REST API, as described below. They can also be viewed and changed in the Ngenea Hub UI, from the 'Sites' tab on the Administration page.

Available Settings

Name	**Description**	**Default**
bandwidth	Limit the bandwidth for the site (In MB/s). In the UI, it is hidden behind the `bandwidth_controls` feature flag (see Feature Flags)	not set (unlimited)
elasticsearch_url	URL used to interact with Elasticsearch when `search_backend` is set to `analytics` (see **Global Configurations** above). The URL is evaluated on the node(s) on which the site worker is running.	`localhost:9200`
file_batch_gb	Limit the total size of file data in a batch, in gigabytes. See **File Batching** below	1
file_batch_size	Limit the total number of files in a batch. See **File Batching** below	40
lock_threshold	The `snapdiff` discovery uses locking to prevent multiple snapdiff running against the same fileset at once. To prevent stale locks, lock are considered 'expired' after the `lock_threshold`, given in seconds.	86400 (one day)
include	A list of include glob patterns which will apply to all workflows run against this site	not set
exclude	A list of exclude glob patterns which will apply to all workflows run against this site	not set

File Batching

The file list generated by discovery tasks may be broken into smaller batches before passing them to workflow steps.

This makes the overall job execution more granular. Individual tasks will be smaller and faster. This also makes it easier to cancel a job, given that only PENDING tasks can be cancelled.

On the other hand, if the batching is too small, the large number of tasks generated may saturate the job queue, blocking out tasks from other jobs.

File batching is based on both `file_batch_gb` and `file_batch_size`. Whichever limit results in a smaller batch is the one which is used. For example, given 100 files of 500MB each, a `file_batch_gb` of 1 and `file_batch_size` of 10 will result in 50 batches of 2 files each (1GB total per batch), because 1GB (2 files) is smaller than 10 files (5GB).

REST API

Configurations can be listed and set via the Ngenea Hub REST API.

The sites endpoint supports client key authentication

To list the current site configuration settings,

```
$ curl -s 'http://example.com/api/sites/1/' -H 'Accept: application/json' -H 'Authorization: Api-Key $APIKEY'
{
    "name": "site1",
    "elasticsearch_url": "localhost:19200",
    "file_batch_size": 100,
    ...
}
```

Note - configurations are only included when fetching a specific site, not when listing all sites.

To change one or more configuration settings, make a `PATCH` request the same endpoint

```
$ curl -s -X PATCH 'http://example.com/api/sites/1/' -H 'Accept: application/json' -H "Authorization: Api-Key $APIKEY" -H 'Content-Type: application/json' -d '{"file_batch_gb": 5}'
{
    "file_batch_gb": 5,
    ...
}
```

Feature Flags

Feature flags control whether selected pre-release features are enabled.

Certain features may be included in a release which aren't yet fully implemented, or fully tested. By default, these features are disabled and 'hidden', so should not affect normal functionality.

However, these features may be enabled on a 'preview' basis, on the understanding that they may be incomplete or unstable.

Warning: Do not enable preview features unless you are willing to accept the potential risks.

Once a feature is finalised and stable, it will be released officially, and the corresponding feature flag will be removed.

Available Features

The following features are currently available.

name	description	stability	default
`searchui`	Enable search features in the Ngenea Hub UI	inprogress	False
`bandwidth_controls`	Enable bandwidth controls in the Ngenea Hub UI	inprogress	False

REST API

Features can be listed, enabled, or disabled via the Ngenea Hub REST API.

To list the available features, and whether they're currently enabled

```
$ curl -s 'http://example.com/api/features/'  -H 'Accept:
application/json'  -H "Authorization: Api-Key $TOKEN"
{
    "count": 1,
    "next": null,
    "results": [
        {
            "name": "searchui",
            "description": "Enable search features in the Ngenea Hub
UI",
            "enabled": false
        },
        {
            "name": "bandwidth_controls",
            "description": "Enable bandwidth controls in the Ngenea
Hub UI",
            "enabled": false,
        }
    ]
}
```

Individual features are keyed by their name, e.g. `http://example.com/api/features/searchui/`

To enable a feature, make a `PATCH` request against the desired feature

```
$ curl -s -X PATCH 'http://example.com/api/features/searchui/'  -H
'Accept: application/json'  -H "Authorization: Api-Key $TOKEN" -H
'Content-Type: application/json' -d '{"enabled": true}'
[
    {
        "name": "searchui",
        "description": "Enable search features in the Ngenea Hub UI",
        "enabled": true
    },
    ...
]
```

And similarly, to disable a feature

```
curl -s -X PATCH 'http://example.com/api/features/searchui/'  -H
'Accept: application/json'  -H "Authorization: Api-Key $TOKEN" -H
'Content-Type: application/json' -d '{"enabled": false}'
```

Note: It may be necessary to restart the Ngenea Hub service for a feature change to take effect.

ngclient

Alternatively, feature flags can be interacted with using ngclient.

To list available features and whether they're currently enabled

```
$ ngclient features list
 [X]  searchui               Enable search features in the UI
 [ ]  bandwidth_controls     Enable bandwidth controls in the UI
```

To enable a feature

```
ngclient features enable bandwidth_controls
```

And to disable a feature

```
ngclient features disable bandwidth_controls
```

See ngclient features for more information.

Actions

Actions allow administrative users to perform certain operations, as described below.

Available Actions

discover_nodes

Ngenea Hub monitors for when new nodes come online.

The `discover_nodes` action can be used to manually scan for nodes. This may be necessary if a node was previously manually removed from Ngenea Hub

This action takes no arguments.

REST API

Actions are submitted via the `/api/actions/` endpoint, and typically execute asynchronously. The state of the action can be viewed via the same endpoint.

Note: The actions endpoint does not support client key authentication. You must use JWT Authentication.

To submit an action, make a `POST` request against the `/api/actions/` endpoint

```
$ curl -s -X POST 'http://example.com/api/actions/' -H 'Accept:
application/json' -H "Authorization: Bearer $JWT_ACCESS_TOKEN" -H
'Content-Type: application/json' -d '{"action": "discover_nodes"}'
{
    "id": 12345,
    ...
}
```

This returns a unique id, which can be used to check the status and results of the action

```
$ curl -s 'http://example.com/api/actions/12345/' -H 'Accept:
application/json' -H "Authorization: Bearer $JWT_ACCESS_TOKEN"
{
    "action": "discover_nodes",
    "user": "myuser",
    "state": "SUCCESS"
    "results": {...}
}
```

Limitations

Site Sync

Site Sync must only be used in one direction

The intent of `site_sync` is for one source site to synchronise all required changes to any amount of destination sites and not consider the state of the destination site(s). Site Sync must only be utilised when data is required to be synchronised without concern for any active changes on destination site(s).

Site Sync only considers changes within an applicable time window for synchronisation

Synchronisation is not ‘event driven’. Changes are collated within a window of time (defined by the associated schedule), and sent as a group.

The order in which certain events occurred cannot be determined. For example; delete determination; where no data point exists to determine the ‘change time’ [ctime] of a file or directory due to deletion prior to the sync time window.

During bidirectional synchronisation, when conflicting create/modify and delete events occur for files, the create/modify event takes precedence over the delete event to prevent data loss. In such scenarios, a newer version of the file which was prior deleted will be present after synchronisation. Directory behaviour is not affected.

Site Sync supports Independent Filesets

Site Sync methodology is incompatible with any requirement to synchronise an entire file system, nor is use of Dependent Filesets, or arbitrary directory trees supported.

Destination site Independent Filesets must exist prior to synchronisation

Site Sync does not create Independent Filesets on destination site(s) prior to synchronisation. Destination Independent Fileset creation is an administrative function and must be undertaken prior to configuration and operation of a Site Sync methodology to the destination site.

Directory deletion is not supported

Site sync does not support directory deletion. Deletion of a large file tree structure on a source site will delete files within the directory structure, resulting in an empty directory tree on the destination site.

Bidirectional Site Sync

Bidirectional Site Sync implements eventual consistency

Site Sync adheres to the principle of eventual consistency whereby one or more subsequent Site Sync jobs or tasks are required to be enacted for source and destination sites to be in sync . Prior to all required Site Sync jobs enacting the synchronisation status is viewed as partially synchronised. Each subsequent job increases the totality of synchronisation.

Data which has failed to be synchronised in prior synchronisations is placed into subsequent synchronisation runs, leading to eventual consistency.

Bidirectional Site Sync is only supported via schedules

Ref: Schedules

A bidirectional Site Sync is created via defining a schedule using the `bidirectional_snapdiff` discovery and subscribing the `bidirectional_sync` workflow to the schedule.

A path may only be managed by one schedule per site, and at most one workflow may be subscribed to a `bidirectional_snapdiff` schedule.

Hub does not support multiple bidirectional synchronisation with the same source site for the same Independent Fileset (E.G. between site1 and site2, and between site1 and site3).

The `bidirectional_snapdiff` discovery causes all iterative Independent Fileset changes to be tracked. Identified changes are compared to across iterative runs. When both sets of changes have been evaluated, only appropriately valid changes are synchronised to the destination site.

Bidirectional synchronisation is sequential

When setting up a bidirectional site sync ‘site1’ is the site which is configured when creating the schedule, and ‘site2’ is the site configured as the `destinationsite` when subscribing the workflow to the schedule.

A bidirectional site sync will first synchronise changes from site1 to site2, and then from site2 to site1.

A failure while synchronising site1 to site2 will not block the reverse direction sync from enacting.

The sequential nature of synchronisation ensures conflict situations where the synchronisation from site2 to site1 wins by virtue of the last write [most recent write at any site] of a file taking precedence.

Swapping files / Last write wins

Synchronisation of a set of file moves whereby the actions include synchronisation determination of the paths of two files being swapped is complex and can result in conflicts. Where identical file paths exist at the destination site, file moves will fail rather than overwriting the existing files.

Manual intervention is required before a synchronisation will again succeed. N.B.: the replaying of the swap of the files on the destination site is not sufficient to resolve the conflict.

Renaming or deleting files and directories causes re-sending of data

Where a file is moved on site1 and the same file is deleted on site2 during an active synchronisation, the moved file from site1 will be resent to site2. This behaviour will be observed even if the delete event occurred later chronologically.

Renaming or files and directories on both bidirectional Site Sync sites causes duplication of data

This behaviour is observed when a file or directory is moved on both sites to different locations during an active synchronisation. E.G.:

- `/mmfs1/data/path1` is moved to `/mmfs1/data/path2` on `site1`
- `/mmfs1/data/path1` is moved to `/mmfs1/data/path3` on `site2`

This scenario results in duplicate data at both sites in both `/mmfs1/data/path2` and `/mmfs1/data/path3`.

Creation or deletion of empty directories on site 1 does not synchronise to site 2

Site Sync does not perform deletions of empty directories on a destination site and does not create empty directories on a destination site.

Troubleshooting

This section outlines steps for troubleshooting issues with Ngenea Hub

Service Status

To check the status of Ngenea Hub and its individual services

```
ngeneahubctl status
```

To check the status of Ngenea Worker

```
systemctl status ngenea-worker
```

Service Logs

The full logs for Ngenea Hub can be viewed with

```
journalctl -u ngeneahub
```

To view Ngenea Worker logs

```
journalctl -u ngenea-worker
```

Specific Features

Site Sync

Site sync - both one-way and bidirectional - may fail due to conflicts which cannot be automatically resolved. This page outlines options for intervening to resolve such conflicts.

Snapshot Rotation

By default, snapdiff snapshots will always rotate, even if an error occurs during sync.

Traditionally, snapshots are rolled-back on error. However, this can lead to changes being replayed inappropriately, leading to further errors and conflicts. Because of this, the default behaviour was changed to always rotate.

The downside to this approach is that some changes may be missed by sync. For example, if an network issues prevent a file from being synced, that file will not be re-synced unless or until it changes again. Note, however, there are retries within a sync run to mitigate such temporary issues.

If a sync job does fail, it will be necessary to determine the source of the error and manually resolve any issues. The job details will report on which paths failed overall. Looking at the individual task details will give more information on where and why a specific path failed.

To disable this behaviour, so that snapshots will rollback on error, set the runtime field `snapdiff_rotate_on_error` to False

Manual Resolution

In some cases it is possible to resolve conflicts by manually applying changes.

For example, if a file is moved on site A and deleted on site B, sync will fail because there is no file to move (or delete, depending on the sync direction) on the target site. In this case, maunally deleting the file on site A, or re-sending the file (in its new location) onto site B will resolve the conflict. Thereafter, sync will be able to run without issue.

Re-sync all

Another option is to re-sync everything from scratch. This is the safest option, as it ensures that no file changes are lost.

Note that sync will skip any files which are already in the correct state, so a re-sync won't take as long as syncing to a brand new target.

Snapdiff-based sync uses filesystem snapshots to track file changes over time. The snapdiff discovery uses a 'last snap' file to record the last snapshot which was successfully synced.

This last snap file is located at `/mmfs1/.rotate/ngenea-worker.lastsnap.name.<fileset_name>.id.<fileset_id>`

By removing the last snap file, the next sync run will behave as if it has not been run before, and so will sync everything. Once sync has run successfully, you can safely delete the snapshot which was previously recorded in the last snap file (before that file was deleted).

Force rotate

The riskiest option is to force a snapshot rotation. This effectively says that you don't care about the current failure and just want the sync to move on.

As discussed above, this is the default behaviour. Note that this may result in some file changes not being synced. For a safer option, see **Re-sync all** above.

The following steps can be used as a one-off when the default rotate-on-error function has been disabled.

To force a rotate, you should first temporarily disable sync. This can be done by setting the sync schedule to disabled in the Ngenea Hub UI.

Next, create a new snapshot of the fileset. The name is expected to be of the form `ngenea-worker.snapdiff.<timestamp>`.

Update the 'last snap' file (described above), replacing the currently recorded snapshot name with the name of the new snapshot you just created.

This last snap file is located at `/mmfs1/.rotate/ngenea-worker.lastsnap.name.<fileset_name>.id.<fileset_id>`

Finally, re-enable sync. Sync will now pick up new file changes starting from the point at which the new snapshot was created.

Once sync has run successfully, you can safely delete the snapshot which was previously recorded in the last snap file (before that file was updated).

Locking Errors

Lock files are used to ensure that a sync for a given fileset will not run if one is already running.

In this case, any new sync job will fail, and under the snapdiff task details, you will see an error like

```
SnapdiffLockError: Could not perform snapdiff as one is currently
running for provided fileset
```

Under rare circumstances, a lock may not be correctly cleaned up, preventing syncs from running, even though there are none currently active.

In that case, the lock file can be removed manually. First, ensure there there aren't any syncs running. For extra safety, temporarily disable any scheduled syncs.

The lock file is located at `/mmfs1/.rotate/snapdiff.name.<fileset_name>.id.<fileset_id>.lock`

Any snapdiff lock file will automatically expire after 24 hours by default. The lifetime can be changed using the `lock_threshold` site setting

Reference

Default Workflows

This section documents all the workflows which come installed in Ngenea Hub by default. See Custom Workflows for guidance on how to create your own workflows.

`migrate`

Migrate one or more files from a site.

discovery: `recursive`

steps:

- dynamo.tasks.migrate

fields:

- `lock_level` (choice): Filesystem locking level for ngenea operations. One of: `partial`, `implicit`. Default=`partial`

`premigrate`

Premigrate one or more files from a site.

discovery: `recursive`

steps:

- dynamo.tasks.migrate
  - `premigrate`: True

fields:

- `lock_level` (choice): Filesystem locking level for ngenea operations. One of: `partial`, `implicit`. Default=`inplicit`

`recall`

Recall one or more files on to a site.

discovery: `recursive`

steps:

- dynamo.tasks.recall

fields:

- `lock_level` (choice): Filesystem locking level for ngenea operations. One of: `partial`, `implicit`. Default=`inplicit`

`send`

Send files from one site to another via cloud storage.

discovery: `recursive`

steps:

- dynamo.tasks.remove_location_xattrs_for_moved
- dynamo.tasks.migrate
  - `premigrate`: True
- dynamo.tasks.reverse_stub

fields:

- `destinationsite` (string): site to send files to
- `hydrate` (bool): hydrate files on the destination site

`site_sync`

Sync a fileset from one site to another via cloud storage.

The `snapdiff` discovery looks for changes within the fileset on the source site since the last time the workflow was invoked. These changes are then synced to the destination site.

The workflow should be invoked with a single path which is the link point of the independent fileset to be synced.

discovery: `snapdiff`

fields:

- `destinationsite` (string): site to sync changes to
- `sync_preference` (string): determines how conflicts should be resolved on the remote site. One of: newest, local, ignore

created

Files with state `created` are sent to the destination site, subject to `sync_preference`

steps:

- dynamo.tasks.remove_location_xattrs_for_moved
- dynamo.tasks.migrate
  - `premigrate`: True
  - `overwrite`: True
  - `abort_missing`: True
- dynamo.tasks.reverse_stub
  - `overwrite`: True

updated

Files with state `updated` are sent to the destination site, subject to `sync_preference`

steps:

- dynamo.tasks.check_sync_state
- dynamo.tasks.remove_location_xattrs_for_moved
- dynamo.tasks.migrate
  - `premigrate`: True
  - `overwrite`: True
  - `abort_missing`: True
- dynamo.tasks.reverse_stub
  - `overwrite`: True

moved

Files with state `moved` are moved 'in-place' on the destination site

steps:

- dynamo.tasks.move_paths_on_gpfs

deleted

Files with state `deleted` are removed on the destination site

steps:

- dynamo.tasks.delete_paths_from_gpfs

Hidden Workflows

This section documents all the workflows which come installed in Ngenea Hub by default that are not provided in the UI. Some of these are intended for advanced users making use of the API, and some are used for GPFS policy automation.

transparent_recall

This is the workflow that will be performed on dehydrated files on the system when accessed, if the policy is configured to make use of Ngenea Hub within its Transparent Recall policy.

Typically, all of these Transparent Recalls are placed into a single job within Ngenea Hub.

steps:

- dynamo.tasks.recall
  - `lock_level`: "partial"

reverse_stub

This workflow is designed to create stubs files or fully hydrated files using remote content on a given site without the need for an existing file on the site for a given path. Due to this requiring the data to be within the cloud and not present on the filesystem, it requires the paths to target cloud targets through the API.

steps:

- dynamo.tasks.reverse_stub

fields:

- `hydrate` (bool): If the file should be fully hydrated during the recall operation
- `overwrite` (bool): If the operation should overwrite a local file that already exists

delete_file

This workflow deletes one or more file/folder paths from the filesystem.

steps:

- dynamo.tasks.delete_paths_from_gpfs

fields:

- `recursive` (bool): If non-empty directories should be deleted. if recursive is `false` additional jobs will be needed to delete empty directories. Defaults to `false`

Discovery Steps

This section documents all the currently supported discovery steps in Ngenea Hub. See Custom Workflows for guidance on how to use these in your own workflows.

`dynamo.tasks.recursive_action` - `recursive`

Navigates down any folder tree provided to it and actions any rule defined with `all` as its type and state on all found files.

Argument	Type	Default	Description
`skip_missing`	`bool`	`False`	Allows the processing of any files directly provided to the discovery step regardless of it being on the filesystem.

`dynamo.tasks.snapdiff` - `snapdiff`

For use only with a GPFS Independent Fileset, it will create a GPFS snapshot and it will then process the list of differences between the time of the initial run and subsequent runs. On the first run of this, it will ingest all the files within that Fileset.

Argument	Type	Default	Description
`skip_old_ctimes`	`bool`	`False`	Skips any files within the snapdiff difference list if the ctime of the file is older than the oldest snapshot.
`condense_moves`	`bool`	`True`	If a directory is moved, this will condense all the move operations into a single operation for move based tasks, otherwise it will action against every effected file.

Workflow Steps

This section documents all the currently supported steps in Ngenea Hub. See Custom Workflows for guidance on how to use these steps in your own workflows.

`dynamo.tasks.migrate`

Migrates a list of files to a pre-defined remote target using Ngenea.

Argument	Type	Default	Description
`premigrate`	`bool`	`False`	retain the content of every migrated file and do not set the OFFLINE flag for the file.migrating.
`stub_size`	`int`	`0`	retain a segment of every migrated file starting from its beginning and having a specified approximate length in bytes.
`overwrite`	`bool`	`False`	overwrite remote objects if they already exist--do not create remote object instances with various UUID suffixes
`fail_on_mismatch`	`bool`	`False`	fail a file migration if a remote object exists but has different hash or metadata. In that case, the task errors
`lock_level`	`string`	`implicit`	Defined the locking mode that ngenea will use when performing the migrate
`endpoint`	`string`		specify the endpoint to migrate
`abort_missing`	`bool`	`False`	allow the migrate task to make any missing files end up in the "aborted" state instead of "failed"

`dynamo.tasks.recall`

Recalls a list of files to a pre-defined remote target using Ngenea.

Argument	Type	Default	Description
`skip_hash`	`bool`	`False`	If the recall should skip checking the hash of the file
`endpoint`	`string`		specify which endpoint(site) to recall from
`lock_level`	`string`	`partial`	Defines the locking level ngenea will use during the recall
`default_uid`	`string`		When a file is recalled, it uses this UID if one is not set on the remote object
`default_gid`	`string`		When a file is recalled, it uses this GID if one is not set on the remote object
`update_atime`	`bool`	`false`	When a files is recalled, update its access time (atime) to 'now'
`update_mtime`	`bool`	`false`	When a files is recalled, update its modification time (mtime) to 'now'
`delete_remote`	`bool`	`false`	If set, when a file is recalled, it deletes the file in the remote location

`dynamo.tasks.reverse_stub`

Recalls a list of files to a pre-defined remote target using Ngenea.

Argument	Type	Default	Description
`hydrate`	`bool`	`False`	If the file should be premigrated instead of a regular stub
`stub_size`	`int`	`0`	The max file size before files will be stubbed for this task
`skip_hash`	`bool`	`False`	If the recall should skip checking the hash of the file
`overwrite`	`bool`	`False`	Overwrite local files if they already exist, except files with only metadata changes.
`endpoint`	`string`		specify which endpoint(site) to recall from.
`retry_stale`	`string`	None	Controls if the worker should attempt to retry file failures due to stale file handles. This string can be either stub for only removing reverse stubbed files or all.
`lock_level`	`string`	`implicit`	Defines the locking level ngenea will use during the recall
`default_uid`	`string`		When a file is recalled, it uses this UID if one is not set on the remote object
`default_gid`	`string`		When a file is recalled, it uses this GID if one is not set on the remote object
`update_atime`	`bool`	`false`	When a files is recalled, update its access time (atime) to 'now'
`update_mtime`	`bool`	`false`	When a files is recalled, update its modification time (mtime) to 'now'
`conflict_preference`	`string`	`None`	Dictates what state the local file should be to pass the check. Options are "newest" which passes if the local file is the latest version of the file on either site, "local" which accepts the local file version regardless of the check and "ignore" which always uses the other sites file version.

`dynamo.tasks.ngenea_sync_metadata`

Sync the local ngenea metadata on a file with the remote target using Ngenea.

Argument	Type	Default	Description
`skip_hash`	`bool`	`False`	If the sync should skip checking the hash of the file
`endpoint`	`string`		Specify which endpoint(site) to recall from
`default_uid`	`string`		When a file is synced, it uses this UID if one is not set on the remote object
`default_gid`	`string`		

Argument	Type	Default	Description
			When a file is synced, it uses this GID if one is not set on the remote object

`dynamo.tasks.delete_paths_from_gpfs`

Removes a list of files from a GPFS filesystem.

Argument	Type	Default	Description
`recursive`	`bool`	`False`	If any directory path is provided and this is set, it will remove the entire file tree, otherwise it will only remove empty directories. It is important to note that the recursive behaviour of removing the entire directory tree will not apply to filesets or if the target directory contains files or directories that are restricted from being deleted such as `snapshots`, in such cases the task will silently ignore those files or directories and report the task as successful.

`dynamo.tasks.check_sync_state`

Checks a provided site against the calling sites to ensure that the local file is in a specified state compared to another site. Using this task will also perform `dynamo.tasks.stat_paths` on the provided site before execution.

Argument	Type	Default	Description
`sync_preference`	`string`	`ignore`	Dictates what state the local file should be to pass the check. Options are "newest" which passes if the local file is the latest version of the file on either site, "local" which accepts the local file version regardless of the check and "ignore" which always uses the other sites file version.
`site`	`string`		The target site to compare
`abort_outdated`	`bool`	`False`	If this bool is set files that do not need to be executed will be marked as aborted as opposed to skipped
`hash_includes_acl`	`bool`	`False`	If this bool is set the metadata comparison for directories will also compare Access Control Lists

`dynamo.tasks.move_paths_on_gpfs`

Moves files on the filesystem using provided paths with a `source` key.

This task moves files in two steps (via an intermediate temporary location), to avoid move conflicts (for example, this task correctly handles cases where files are 'swapped': `fileA` moved to `fileB`, and `fileB` moved to `fileA`).

Argument	Type	Default	Description
`delete_remote_xattrs`	`bool`	`False`	If set, after a file has been moved all remote location xattrs will be removed
`source_missing_signature`	`json`	`null`	A signature to send any paths to where the source is missing. If this isn't set, a missing source is treated as a failure.
`target_max_age`	`int`	`null`	If the target file exists, it is overwritten by default. If this optional parameter is set, the target has to have a ctime <= this timestamp for the move to proceed. Otherwise, the source file is removed. Given in seconds since epoch. Primarily used for sync workflows.

`dynamo.tasks.one_step_move_paths_on_gpfs`

Moves files on the filesystem using provided paths with a `source` key.

This task is not used in the default Ngenea Hub workflows. It is less robust than `dynamo.tasks.move_paths_on_gpfs`, because it moves files directly from source to their new location (in one step). So for example, trying to 'swap' files (`fileA` moved to `fileB`, and `fileB` moved to `fileA`) with this task will effectively result in one of these files being deleted on the target site.

However, this task is for cases where doing moves in two steps is not an acceptable option.

Argument	Type	Default	Description
`delete_remote_xattrs`	`bool`	`False`	If set, after a file has been moved all remote location xattrs will be removed
`source_missing_signature`	`json`	`null`	A signature to send any paths to where the source is missing. If this isn't set, a missing source is treated as a failure.
`target_max_age`	`int`	`null`	If the target file exists, it is overwritten by default. If this optional parameter is set, the target has to have a ctime <= this timestamp for the move to proceed. Otherwise, the source file is

Argument	Type	Default	Description
			removed. Given in seconds since epoch. Primarily used for sync workflows.

`dynamo.tasks.remove_location_xattrs_for_moved`

This task removes all remote location xattrs on all provided paths.

This step takes no additional arguments.

`dynamo.tasks.move_in_cloud`

Moves a file on the filesystem's related cloud storage platform using provided paths with a `source` key.

This step takes no additional arguments.

`dynamo.tasks.remove_from_cloud`

Deletes a file on the filesystem's related cloud storage platform using provided paths.

This step takes no additional arguments.

`dynamo.tasks.ensure_cloud_file_exists`

Ensures all files provided to the task exist on the filesystem's related cloud storage platform. If some do not, it will attempt to retry this check an additional two more times before failing.

This step takes no additional arguments.

Job States

When jobs are created on the call of a workflow, they can end up in specific state that

State	Description
`Pending`	The job is being populated with tasks through its discovery task and will begin when paths have been collected
`Started`	Some tasks within the job have started to be processed
`Success`	All tasks in a job have completed successfully
`Failure`	There was an error or failure when attempting a task within a job, meaning the job could not complete
`Skipped`	Based on the output of the provided discovery task, no tasks needed to be created so there is no work to
`Cancelled`	

State	Description
	The job has been manually closed via request and all remaining task have been cancelled

Task States

When jobs create tasks, after performing their action on the provided files they can end up in specific state as seen below

State	Description
`Pending`	This task is has been created but has not yet been picked up by a site
`Started`	This task is now running on site
`Success`	This task has completed successfully
`Failure`	There was a unexpected error when attempting a task within a job, meaning the job could not complete and could not provide structured output
`Error`	There was a captured error when attempting a task within a job, meaning the job will not have processed all paths but has structured output of what has been completed. Any paths which were successfully processed will be handled by subsequent tasks in a task chain.
`Skipped`	This task has will have no work to perform so it has been automatically skipped by another task
`Cancelled`	Either a previous task has failed or the job has been manually cancelled, causing this task to no longer run

Note: Job and task states are updated asynchronously. There may be, for example, a short delay between a job/task completing and its state being reported as such.

File States

As a file is processed by a workflow, each task reports whether the state of the file following the task. The containing job reports the state of the file after all workflow steps have completed.

A file can have any of the following states

Processed

The file was successfully processed by the workflow step without issues

Skipped

The files was not processed because it was already in the expected state, e.g. a delete was skipped because the file wasn't found, a migration was skipped because the file was already offline.

Because it is in the expected state, skipped files can be processed by subsequent tasks in the workflow.

Failed

An error occurred meaning the file couldn't be processed. Because of this error, the file will not be processed by later tasks in the workflow.

The overall job will be marked as failed if it contains any failed files. Workflows which use the `snapdiff` discovery will be 'rolled back' in this case.

Failed files typically indicate an issue that needs to be manually resolved, such as network issues.

Aborted

An error occurred meaning the file couldn't be processed. Because the file was aborted, it will not be processed by later tasks in the workflow.

The overall job is **not** marked as failed if it contains aborted files. Workflows which use the `snapdiff` discovery will 'rotate' in this case.

The intended use case for this state is in scheduled (recurring) jobs, where the any aborted files are expected to be successfully processed in a later run.

For example, if a file changes while it is being sent, the send will be aborted to prevent data corruption. But because the file was modified, it will be identified as modified in the next snapdiff scan and processed again; effectively retried.

API

`GET /actions/`

Query Parameters:

- **page** (integer) -- A page number within the paginated result set. When not given, first page is retrieved by default.
- **page_size** (integer) -- Number of results to return per page. Page size parameter can be a number between `20` and `100`. For disabling pagination and retrieving all results, `0` should be given. When page size parameter is empty or `<20`, `20` results are returned by default. When page size parameter `>100`, `100` results are returned by default.

Status Codes:

- 200 OK --

Response JSON Object:

- **count** (integer) -- (required)
- **next** (string) --
- **previous** (string) --
- **results[].action** (string) -- (required)
- **results[].id** (integer) -- (read only)
- **results[].state** (string) --

`POST /actions/`

Request JSON Object:

- **action** (string) -- (required)

Status Codes:

- 201 Created --

Response JSON Object:

- **action** (string) -- (required)

`GET /actions/{id}/`

Parameters:

- **id** (string) --

Status Codes:

- 200 OK --

Response JSON Object:

- **action** (string) -- (required)
- **id** (integer) -- (read only)
- **results** (string) -- (read only)
- **state** (string) --
- **user** (integer) --

`GET /auth/clientkeys/`

API endpoint for managing client keys

Query Parameters:

- **page** (integer) -- A page number within the paginated result set. When not given, first page is retrieved by default.
- **page_size** (integer) -- Number of results to return per page. Page size parameter can be a number between `20` and `100`. For disabling pagination and retrieving all results, `0` should be given. When page size parameter is empty or `<20`, `20` results are returned by default. When page size parameter `>100`, `100` results are returned by default.

Status Codes:

- 200 OK --

Response JSON Object:

- **count** (integer) -- (required)
- **next** (string) --
- **previous** (string) --
- **results[].id** (integer) -- (read only)
- **results[].name** (string) -- Name of the client key (required)
- **results[].url** (string) -- (read only)

`POST /auth/clientkeys/`

API endpoint for managing client keys

Request JSON Object:

- **api_key** (string) -- (read only)
- **id** (integer) -- (read only)
- **name** (string) -- Name of the client key (required)
- **url** (string) -- (read only)

Status Codes:
- 201 Created --

Response JSON Object:
- **api_key** (string) -- (read only)
- **id** (integer) -- (read only)
- **name** (string) -- Name of the client key (required)
- **url** (string) -- (read only)

`GET /auth/clientkeys/{id}/`

API endpoint for managing client keys

Parameters:
- **id** (string) --

Status Codes:
- 200 OK --

Response JSON Object:
- **id** (integer) -- (read only)
- **name** (string) -- Name of the client key (required)
- **url** (string) -- (read only)

`PATCH /auth/clientkeys/{id}/`

API endpoint for managing client keys

Parameters:
- **id** (string) --

Request JSON Object:
- **id** (integer) -- (read only)
- **name** (string) -- Name of the client key (required)
- **url** (string) -- (read only)

Status Codes:
- 200 OK --

Response JSON Object:
- **id** (integer) -- (read only)
- **name** (string) -- Name of the client key (required)
- **url** (string) -- (read only)

`DELETE /auth/clientkeys/{id}/`

API endpoint for managing client keys

Parameters:
- **id** (string) --

Status Codes:
- 204 No Content --

`POST /auth/token/`

Request JSON Object:
- **password** (string) -- (required)
- **username** (string) -- (required)

Status Codes:
- 201 Created --

Response JSON Object:

- **password** (string) -- (required)
- **username** (string) -- (required)

`GET /auth/token/publickey/`

API endpoint for retrieving public key that is used for token verification.

Status Codes:

- 200 OK --

`POST /auth/token/refresh/`

Takes a refresh type JSON web token and returns an access type JSON web token if the refresh token is valid.

Request JSON Object:

- **access** (string) -- (read only)
- **refresh** (string) -- (required)

Status Codes:

- 201 Created --

Response JSON Object:

- **access** (string) -- (read only)
- **refresh** (string) -- (required)

`POST /auth/token/verify/`

Verifies that the token is not expired AND the token owner exists in the database AND the token owner is an active user.

Request JSON Object:

- **token** (string) -- (required)
- **type** (string) -- Token type e.g: access or refresh (required)

Status Codes:

- 201 Created --

Response JSON Object:

- **token** (string) -- (required)
- **type** (string) -- Token type e.g: access or refresh (required)

`GET /configurations/`

API endpoint for viewing and setting configurations.

Status Codes:

- 200 OK --

Response JSON Object:

- **jobs_ttl** (integer) -- Time to store job details after the job completes, in days
- **search_backend** (string) -- Search backend
- **search_max_results** (integer) -- Maximum search results
- **search_result_ttl** (integer) -- Maximum time to store search results in days
- **snapdiff_stream_timeout** (integer) -- Maximum amount of minutes to wait for task results
- **snapshot_create_delete_retry_timeout** (integer) -- Maximum time for workers to retry snapshot create and delete operations, in seconds

- **stat_refresh_period** (integer) -- Maximum time to wait for results from stat in seconds
- **stat_timeout** (integer) -- Maximum time to wait for results from stat in seconds
- **task_invalidation_timeout** (integer) -- Maximum amount of minutes before a task in the STARTED state is considered invalid

`PATCH /configurations/`

API endpoint for viewing and setting configurations.

Request JSON Object:

- **jobs_ttl** (integer) -- Time to store job details after the job completes, in days
- **search_backend** (string) -- Search backend
- **search_max_results** (integer) -- Maximum search results
- **search_result_ttl** (integer) -- Maximum time to store search results in days
- **snapdiff_stream_timeout** (integer) -- Maximum amount of minutes to wait for task results
- **snapshot_create_delete_retry_timeout** (integer) -- Maximum time for workers to retry snapshot create and delete operations, in seconds
- **stat_refresh_period** (integer) -- Maximum time to wait for results from stat in seconds
- **stat_timeout** (integer) -- Maximum time to wait for results from stat in seconds
- **task_invalidation_timeout** (integer) -- Maximum amount of minutes before a task in the STARTED state is considered invalid

Status Codes:

- 200 OK --

Response JSON Object:

- **jobs_ttl** (integer) -- Time to store job details after the job completes, in days
- **search_backend** (string) -- Search backend
- **search_max_results** (integer) -- Maximum search results
- **search_result_ttl** (integer) -- Maximum time to store search results in days
- **snapdiff_stream_timeout** (integer) -- Maximum amount of minutes to wait for task results
- **snapshot_create_delete_retry_timeout** (integer) -- Maximum time for workers to retry snapshot create and delete operations, in seconds
- **stat_refresh_period** (integer) -- Maximum time to wait for results from stat in seconds
- **stat_timeout** (integer) -- Maximum time to wait for results from stat in seconds
- **task_invalidation_timeout** (integer) -- Maximum amount of minutes before a task in the STARTED state is considered invalid

`GET /datastores/`

API endpoint for managing DataStores.

Query Parameters:

- **page** (integer) -- A page number within the paginated result set. When not given, first page is retrieved by default.

- **page_size** (integer) -- Number of results to return per page. Page size parameter can be a number between `20` and `100`. For disabling pagination and retrieving all results, `0` should be given. When page size parameter is empty or `<20`, `20` results are returned by default. When page size parameter `>100`, `100` results are returned by default.

Status Codes:

- 200 OK --

Response JSON Object:

- **count** (integer) -- (required)
- **next** (string) --
- **previous** (string) --
- **results[].accesskey** (string) --
- **results[].accesskeyid** (string) --
- **results[].bucket** (string) --
- **results[].container** (string) --
- **results[].credentialsfile** (string) --
- **results[].endpoint** (string) --
- **results[].id** (integer) -- (read only)
- **results[].name** (string) -- DataStore Name (required)
- **results[].region** (string) --
- **results[].secretaccesskey** (string) --
- **results[].storageaccount** (string) --
- **results[].type** (string) -- Site Type (required)
- **results[].url** (string) -- (read only)

`POST /datastores/`

API endpoint for managing DataStores.

Request JSON Object:

- **accesskey** (string) --
- **accesskeyid** (string) --
- **bucket** (string) --
- **container** (string) --
- **credentialsfile** (string) --
- **endpoint** (string) --
- **id** (integer) -- (read only)
- **name** (string) -- DataStore Name (required)
- **region** (string) --
- **secretaccesskey** (string) --
- **storageaccount** (string) --
- **type** (string) -- Site Type (required)
- **url** (string) -- (read only)

Status Codes:

- 201 Created --

Response JSON Object:

- **accesskey** (string) --
- **accesskeyid** (string) --
- **bucket** (string) --
- **container** (string) --
- **credentialsfile** (string) --

- **endpoint** (string) --
- **id** (integer) -- (read only)
- **name** (string) -- DataStore Name (required)
- **region** (string) --
- **secretaccesskey** (string) --
- **storageaccount** (string) --
- **type** (string) -- Site Type (required)
- **url** (string) -- (read only)

`GET /datastores/{id}/`

API endpoint for managing DataStores.

Parameters:

- **id** (string) --

Status Codes:

- 200 OK --

Response JSON Object:

- **accesskey** (string) --
- **accesskeyid** (string) --
- **bucket** (string) --
- **container** (string) --
- **credentialsfile** (string) --
- **endpoint** (string) --
- **id** (integer) -- (read only)
- **name** (string) -- DataStore Name (required)
- **region** (string) --
- **secretaccesskey** (string) --
- **storageaccount** (string) --
- **type** (string) -- Site Type (required)
- **url** (string) -- (read only)

`PATCH /datastores/{id}/`

API endpoint for managing DataStores.

Parameters:

- **id** (string) --

Request JSON Object:

- **accesskey** (string) --
- **accesskeyid** (string) --
- **bucket** (string) --
- **container** (string) --
- **credentialsfile** (string) --
- **endpoint** (string) --
- **id** (integer) -- (read only)
- **name** (string) -- DataStore Name (required)
- **region** (string) --
- **secretaccesskey** (string) --
- **storageaccount** (string) --
- **type** (string) -- Site Type (required)
- **url** (string) -- (read only)

Status Codes:

- 200 OK --

Response JSON Object:

- **accesskey** (string) --
- **accesskeyid** (string) --
- **bucket** (string) --
- **container** (string) --
- **credentialsfile** (string) --
- **endpoint** (string) --
- **id** (integer) -- (read only)
- **name** (string) -- DataStore Name (required)
- **region** (string) --
- **secretaccesskey** (string) --
- **storageaccount** (string) --
- **type** (string) -- Site Type (required)
- **url** (string) -- (read only)

`DELETE /datastores/{id}/`

API endpoint for managing DataStores.

Parameters:

- **id** (string) --

Status Codes:

- 204 No Content --

`GET /features/`

API endpoint for managing feature flags.

Query Parameters:

- **page** (integer) -- A page number within the paginated result set. When not given, first page is retrieved by default.
- **page_size** (integer) -- Number of results to return per page. Page size parameter can be a number between `20` and `100`. For disabling pagination and retrieving all results, `0` should be given. When page size parameter is empty or `<20`, `20` results are returned by default. When page size parameter `>100`, `100` results are returned by default.

Status Codes:

- 200 OK --

Response JSON Object:

- **count** (integer) -- (required)
- **next** (string) --
- **previous** (string) --
- **results[].description** (string) -- Description of what the feature does (read only)
- **results[].enabled** (boolean) -- Whether the feature has been enabled
- **results[].name** (string) -- Name of the feature (read only)

`GET /features/{name}/`

API endpoint for managing feature flags.

Parameters:
- **name** (string) --

Status Codes:
- 200 OK --

Response JSON Object:
- **description** (string) -- Description of what the feature does (read only)
- **enabled** (boolean) -- Whether the feature has been enabled
- **name** (string) -- Name of the feature (read only)

`PATCH /features/{name}/`

API endpoint for managing feature flags.

Parameters:
- **name** (string) --

Request JSON Object:
- **description** (string) -- Description of what the feature does (read only)
- **enabled** (boolean) -- Whether the feature has been enabled
- **name** (string) -- Name of the feature (read only)

Status Codes:
- 200 OK --

Response JSON Object:
- **description** (string) -- Description of what the feature does (read only)
- **enabled** (boolean) -- Whether the feature has been enabled
- **name** (string) -- Name of the feature (read only)

`GET /file/`

Retrieves list of files under given path for given site.

Query Parameters:
- **path** (string) -- Target directory path
- **site** (string) -- Site name
- **details** (boolean) -- Show details of children objects
- **cache_ttl** (integer) -- How long the cache will last for the target path

Status Codes:
- 200 OK --

`GET /file/test/`

API endpoint for managing files.

Status Codes:
- 200 OK --

`POST /file/workflow/`

Performs a workflow on a list of files

Request JSON Object:
- **discovery** (string) -- Discovery name
- **exclude[]** (string) --
- **fields** (object) --
- **ignore_site_excludes** (boolean) --
- **ignore_site_includes** (boolean) --

- **include[]** (string) --
- **job** (integer) -- Job ID
- **paths[]** (object) --
- **site** (string) -- Site name (required)
- **workflow** (string) -- Workflow name (required)

Status Codes:

- 201 Created --

Response JSON Object:

- **discovery** (string) -- Discovery name
- **exclude[]** (string) --
- **fields** (object) --
- **ignore_site_excludes** (boolean) --
- **ignore_site_includes** (boolean) --
- **include[]** (string) --
- **job** (integer) -- Job ID
- **paths[]** (object) --
- **site** (string) -- Site name (required)
- **workflow** (string) -- Workflow name (required)

GET /filesets/

Retrieve list of filesets on a given site.

Query Parameters:

- **site** (string) -- Site name

Status Codes:

- 200 OK --

GET /filestatustypes/

API endpoint for managing file status types.

Query Parameters:

- **page** (integer) -- A page number within the paginated result set. When not given, first page is retrieved by default.
- **page_size** (integer) -- Number of results to return per page. Page size parameter can be a number between `20` and `100`. For disabling pagination and retrieving all results, `0` should be given. When page size parameter is empty or `<20`, `20` results are returned by default. When page size parameter `>100`, `100` results are returned by default.

Status Codes:

- 200 OK --

Response JSON Object:

- **count** (integer) -- (required)
- **next** (string) --
- **previous** (string) --
- **results[].background_color** (string) -- (required)
- **results[].key** (string) -- (required)
- **results[].label** (string) -- (required)
- **results[].text_color** (string) -- (required)
- **results[].url** (string) -- (read only)

GET /filestatustypes/{key}/

API endpoint for managing file status types.

Parameters:
- **key** (string) --

Status Codes:
- 200 OK --

Response JSON Object:
- **background_color** (string) -- (required)
- **key** (string) -- (required)
- **label** (string) -- (required)
- **text_color** (string) -- (required)
- **url** (string) -- (read only)

`PATCH /filestatustypes/{key}/`

API endpoint for managing file status types.

Parameters:
- **key** (string) --

Request JSON Object:
- **background_color** (string) -- (required)
- **label** (string) -- (required)
- **text_color** (string) -- (required)

Status Codes:
- 200 OK --

Response JSON Object:
- **background_color** (string) -- (required)
- **label** (string) -- (required)
- **text_color** (string) -- (required)

`GET /groups/`

API endpoint for managing groups.

Query Parameters:
- **page** (integer) -- A page number within the paginated result set. When not given, first page is retrieved by default.
- **page_size** (integer) -- Number of results to return per page. Page size parameter can be a number between `20` and `100`. For disabling pagination and retrieving all results, `0` should be given. When page size parameter is empty or `<20`, `20` results are returned by default. When page size parameter `>100`, `100` results are returned by default.

Status Codes:
- 200 OK --

Response JSON Object:
- **count** (integer) -- (required)
- **next** (string) --
- **previous** (string) --
- **results[].name** (string) -- (required)
- **results[].permissions[]** (string) --
- **results[].users** (string) -- (required)

`POST /groups/`

API endpoint for managing groups.

Request JSON Object:
- **name** (string) -- (required)
- **permissions[]** (string) --
- **users** (string) -- (required)

Status Codes:
- 201 Created --

Response JSON Object:
- **name** (string) -- (required)
- **permissions[]** (string) --
- **users** (string) -- (required)

`GET /groups/{id}/`

API endpoint for managing groups.

Parameters:
- **id** (string) --

Status Codes:
- 200 OK --

Response JSON Object:
- **name** (string) -- (required)
- **permissions[]** (string) --
- **users** (string) -- (required)

`PATCH /groups/{id}/`

API endpoint for managing groups.

Parameters:
- **id** (string) --

Request JSON Object:
- **name** (string) -- (required)
- **permissions[]** (string) --
- **users** (string) -- (required)

Status Codes:
- 200 OK --

Response JSON Object:
- **name** (string) -- (required)
- **permissions[]** (string) --
- **users** (string) -- (required)

`DELETE /groups/{id}/`

API endpoint for managing groups.

Parameters:
- **id** (string) --

Status Codes:
- 204 No Content --

`GET /health/`

Status Codes:

- 200 OK --

`GET /ipaddresses/`

API endpoint for managing IP addresses.

Query Parameters:

- **page** (integer) -- A page number within the paginated result set. When not given, first page is retrieved by default.
- **page_size** (integer) -- Number of results to return per page. Page size parameter can be a number between `20` and `100`. For disabling pagination and retrieving all results, `0` should be given. When page size parameter is empty or `<20`, `20` results are returned by default. When page size parameter `>100`, `100` results are returned by default.
- **datastore_id** (integer) -- Data store ID that the IP is assigned to

Status Codes:

- 200 OK --

Response JSON Object:

- **count** (integer) -- (required)
- **next** (string) --
- **previous** (string) --
- **results[].datastore.id** (integer) -- (read only)
- **results[].datastore.name** (string) -- DataStore Name (required)
- **results[].datastore.url** (string) -- (read only)
- **results[].id** (integer) -- (read only)
- **results[].ipaddr** (string) -- IP Address (IPv4) (required)
- **results[].url** (string) -- (read only)

`POST /ipaddresses/`

API endpoint for managing IP addresses.

Request JSON Object:

- **datastore** (integer) -- (required)
- **id** (integer) -- (read only)
- **ipaddr** (string) -- (required)
- **url** (string) -- (read only)

Status Codes:

- 201 Created --

Response JSON Object:

- **datastore** (integer) -- (required)
- **id** (integer) -- (read only)
- **ipaddr** (string) -- (required)
- **url** (string) -- (read only)

`GET /ipaddresses/{id}/`

API endpoint for managing IP addresses.

Parameters:

- **id** (string) --

Status Codes:

- 200 OK --

Response JSON Object:

- **datastore.id** (integer) -- (read only)
- **datastore.name** (string) -- DataStore Name (required)
- **datastore.url** (string) -- (read only)
- **id** (integer) -- (read only)
- **ipaddr** (string) -- IP Address (IPv4) (required)
- **url** (string) -- (read only)

`PATCH /ipaddresses/{id}/`

API endpoint for managing IP addresses.

Parameters:

- **id** (string) --

Request JSON Object:

- **datastore** (integer) -- (required)
- **id** (integer) -- (read only)
- **ipaddr** (string) -- (required)
- **url** (string) -- (read only)

Status Codes:

- 200 OK --

Response JSON Object:

- **datastore** (integer) -- (required)
- **id** (integer) -- (read only)
- **ipaddr** (string) -- (required)
- **url** (string) -- (read only)

`DELETE /ipaddresses/{id}/`

API endpoint for managing IP addresses.

Parameters:

- **id** (string) --

Status Codes:

- 204 No Content --

`GET /jobs/`

API endpoint for managing jobs.

Query Parameters:

- **page** (integer) -- A page number within the paginated result set. When not given, first page is retrieved by default.
- **page_size** (integer) -- Number of results to return per page. Page size parameter can be a number between `20` and `100`. For disabling pagination and retrieving all results, `0` should be given. When page size parameter is empty or `<20`, `20` results are returned by default. When page size parameter `>100`, `100` results are returned by default.
- **created** (string) -- Time period string for filtering jobs by time. Leave `null` for displaying jobs in all times.
- **created_time_from** (string) -- Start time for filtering jobs by creation time in UTC. Discarded when `created` parameter is given.
- **created_time_to** (string) -- End time for filtering jobs by creation time in UTC. Discarded when `created` parameter is given.

- **completed_time_from** (string) -- Start time for filtering jobs by completion time in UTC.
- **completed_time_to** (string) -- End time for filtering jobs by completion time in UTC.
- **jobtype** (string) -- Job type
- **state** (array) -- Job states
- **owner_ids** (array) -- Job owner user IDs. Send `-1` for the `Unknown` owner.
- **clientkey_ids** (array) -- Job clientkey IDs
- **schedule_ids** (array) -- Job schedule IDs
- **site_id** (integer) -- Job site ID
- **input_paths_prefix** (string) -- Path prefix for the job input paths
- **hide_noop** (boolean) -- Hide successful jobs with no processed files

Status Codes:

- 200 OK --

Response JSON Object:

- **count** (integer) -- (required)
- **next** (string) --
- **previous** (string) --
- **results[].clientkey** (string) -- (read only)
- **results[].completed** (string) -- Time of the job completion
- **results[].created** (string) -- Time of the job creation
- **results[].dir_walk_complete** (string) -- (read only)
- **results[].fields** (object) --
- **results[].id** (integer) -- (read only)
- **results[].jobtype** (string) -- (required)
- **results[].numabortedfiles** (integer) --
- **results[].numcancelledfiles** (integer) --
- **results[].numfailedfiles** (integer) --
- **results[].numfiles** (integer) --
- **results[].numprocessedfiles** (integer) --
- **results[].numskippedfiles** (integer) --
- **results[].owner** (string) -- (read only)
- **results[].runtime** (number) --
- **results[].schedule** (string) -- (read only)
- **results[].site** (string) -- Site Name (required)
- **results[].started** (string) -- Time the job started executing
- **results[].state** (string) --
- **results[].url** (string) -- (read only)

`POST /jobs/`

API endpoint for managing jobs.

Request JSON Object:

- **discovery** (string) -- Path discovery method
- **jobtype** (string) -- (required)
- **paths[]** (string) --
- **site** (string) -- Site Name (required)
- **state** (string) --

Status Codes:

- 201 Created --

Response JSON Object:

- **discovery** (string) -- Path discovery method
- **jobtype** (string) -- (required)
- **paths[]** (string) --
- **site** (string) -- Site Name (required)
- **state** (string) --

`GET /jobs/recent/`

Retrieves last N jobs as recent jobs. N = 5 by default (defined in dynamohub/settings/base.py).

Query Parameters:

- **page** (integer) -- A page number within the paginated result set. When not given, first page is retrieved by default.
- **page_size** (integer) -- Number of results to return per page. Page size parameter can be a number between `20` and `100`. For disabling pagination and retrieving all results, `0` should be given. When page size parameter is empty or `<20`, `20` results are returned by default. When page size parameter `>100`, `100` results are returned by default.

Status Codes:

- 200 OK --

Response JSON Object:

- **count** (integer) -- (required)
- **next** (string) --
- **previous** (string) --
- **results[].clientkey** (string) -- (read only)
- **results[].completed** (string) -- Time of the job completion
- **results[].created** (string) -- Time of the job creation
- **results[].dir_walk_complete** (string) -- (read only)
- **results[].discovery** (string) -- Path discovery method
- **results[].fields** (object) --
- **results[].id** (integer) -- (read only)
- **results[].jobtype** (string) -- (required)
- **results[].numabortedfiles** (integer) --
- **results[].numcancelledfiles** (integer) --
- **results[].numfailedfiles** (integer) --
- **results[].numfiles** (integer) --
- **results[].numprocessedfiles** (integer) --
- **results[].numskippedfiles** (integer) --
- **results[].owner** (string) -- (read only)
- **results[].paths** (string) -- (read only)
- **results[].runtime** (number) --
- **results[].schedule** (string) -- (read only)
- **results[].site** (string) -- Site Name (required)
- **results[].started** (string) -- Time the job started executing
- **results[].state** (string) --
- **results[].url** (string) -- (read only)

`GET /jobs/stats/`

API endpoint for managing jobs.

Query Parameters:

- **page** (integer) -- A page number within the paginated result set. When not given, first page is retrieved by default.
- **page_size** (integer) -- Number of results to return per page. Page size parameter can be a number between `20` and `100`. For disabling pagination and retrieving all results, `0` should be given. When page size parameter is empty or `<20`, `20` results are returned by default. When page size parameter `>100`, `100` results are returned by default.
- **created** (string) -- Time period string for filtering jobs by time. Leave `null` for displaying jobs in all times.
- **created_time_from** (string) -- Start time for filtering jobs by creation time in UTC. Discarded when `created` parameter is given.
- **created_time_to** (string) -- End time for filtering jobs by creation time in UTC. Discarded when `created` parameter is given.
- **completed_time_from** (string) -- Start time for filtering jobs by completion time in UTC.
- **completed_time_to** (string) -- End time for filtering jobs by completion time in UTC.
- **jobtype** (string) -- Job type
- **state** (array) -- Job states
- **owner_ids** (array) -- Job owner user IDs. Send `-1` for the `Unknown` owner.
- **clientkey_ids** (array) -- Job clientkey IDs
- **schedule_ids** (array) -- Job schedule IDs
- **site_id** (integer) -- Job site ID
- **input_paths_prefix** (string) -- Path prefix for the job input paths
- **hide_noop** (boolean) -- Hide successful jobs with no processed files

Status Codes:

- 200 OK --

Response JSON Object:

- **count** (integer) -- (required)
- **next** (string) --
- **previous** (string) --
- **results[].clientkey** (string) -- (read only)
- **results[].completed** (string) -- Time of the job completion
- **results[].created** (string) -- Time of the job creation
- **results[].dir_walk_complete** (string) -- (read only)
- **results[].discovery** (string) -- Path discovery method
- **results[].fields** (object) --
- **results[].id** (integer) -- (read only)
- **results[].jobtype** (string) -- (required)
- **results[].numabortedfiles** (integer) --
- **results[].numcancelledfiles** (integer) --
- **results[].numfailedfiles** (integer) --
- **results[].numfiles** (integer) --
- **results[].numprocessedfiles** (integer) --
- **results[].numskippedfiles** (integer) --
- **results[].owner** (string) -- (read only)
- **results[].paths** (string) -- (read only)
- **results[].runtime** (number) --
- **results[].schedule** (string) -- (read only)

- **results[].site** (string) -- Site Name (required)
- **results[].started** (string) -- Time the job started executing
- **results[].state** (string) --
- **results[].url** (string) -- (read only)

`GET /jobs/{id}/`

API endpoint for managing jobs.

Parameters:
- **id** (string) --

Status Codes:
- 200 OK --

Response JSON Object:
- **clientkey** (string) -- (read only)
- **completed** (string) -- Time of the job completion
- **created** (string) -- Time of the job creation
- **dir_walk_complete** (string) -- (read only)
- **discovery** (string) -- Path discovery method
- **fields** (object) --
- **id** (integer) -- (read only)
- **jobtype** (string) -- (required)
- **numabortedfiles** (integer) --
- **numcancelledfiles** (integer) --
- **numfailedfiles** (integer) --
- **numfiles** (integer) --
- **numprocessedfiles** (integer) --
- **numskippedfiles** (integer) --
- **owner** (string) -- (read only)
- **paths** (string) -- (read only)
- **runtime** (number) --
- **schedule** (string) -- (read only)
- **site** (string) -- Site Name (required)
- **started** (string) -- Time the job started executing
- **state** (string) --
- **url** (string) -- (read only)

`PATCH /jobs/{id}/`

API endpoint for managing jobs.

Parameters:
- **id** (string) --

Request JSON Object:
- **discovery** (string) -- Path discovery method
- **jobtype** (string) -- (required)
- **paths[]** (string) --
- **site** (string) -- Site Name (required)
- **state** (string) --

Status Codes:
- 200 OK --

Response JSON Object:

- **discovery** (string) -- Path discovery method
- **jobtype** (string) -- (required)
- **paths[]** (string) --
- **site** (string) -- Site Name (required)
- **state** (string) --

`DELETE /jobs/{id}/`

API endpoint for managing jobs.

Parameters:

- **id** (string) --

Status Codes:

- 204 No Content --

`POST /jobs/{id}/cancel/`

Cancels the pending and started tasks currently on the MQ for the given ID's job.

Parameters:

- **id** (string) --

Status Codes:

- 201 Created --

Response JSON Object:

- **clientkey** (string) -- (read only)
- **completed** (string) -- Time of the job completion
- **created** (string) -- Time of the job creation
- **dir_walk_complete** (string) -- (read only)
- **discovery** (string) -- Path discovery method
- **fields** (object) --
- **id** (integer) -- (read only)
- **jobtype** (string) -- (required)
- **numabortedfiles** (integer) --
- **numcancelledfiles** (integer) --
- **numfailedfiles** (integer) --
- **numfiles** (integer) --
- **numprocessedfiles** (integer) --
- **numskippedfiles** (integer) --
- **owner** (string) -- (read only)
- **paths** (string) -- (read only)
- **runtime** (number) --
- **schedule** (string) -- (read only)
- **site** (string) -- Site Name (required)
- **started** (string) -- Time the job started executing
- **state** (string) --
- **url** (string) -- (read only)

`GET /jobs/{id}/files/`

Retrieves the files related with a job, with their execution status.

Parameters:

- **id** (string) --

Query Parameters:

- **category** (string) --
- **page** (integer) -- A page number within the paginated result set. When not given, first page is retrieved by default.
- **page_size** (integer) -- Number of results to return per page. Page size parameter can be a number between `20` and `100`. For disabling pagination and retrieving all results, `0` should be given. When page size parameter is empty or `<20`, `20` results are returned by default. When page size parameter `>100`, `100` results are returned by default.

Status Codes:

- 200 OK --

Response JSON Object:

- **clientkey** (string) -- (read only)
- **completed** (string) -- Time of the job completion
- **created** (string) -- Time of the job creation
- **dir_walk_complete** (string) -- (read only)
- **discovery** (string) -- Path discovery method
- **fields** (object) --
- **id** (integer) -- (read only)
- **jobtype** (string) -- (required)
- **numabortedfiles** (integer) --
- **numcancelledfiles** (integer) --
- **numfailedfiles** (integer) --
- **numfiles** (integer) --
- **numprocessedfiles** (integer) --
- **numskippedfiles** (integer) --
- **owner** (string) -- (read only)
- **paths** (string) -- (read only)
- **runtime** (number) --
- **schedule** (string) -- (read only)
- **site** (string) -- Site Name (required)
- **started** (string) -- Time the job started executing
- **state** (string) --
- **url** (string) -- (read only)

`POST /jobs/{id}/resubmit/`

Resubmits the job with given id. If the job is not finished yet, this action will not have an effect.

Parameters:

- **id** (string) --

Status Codes:

- 201 Created --

`GET /nodes/`

Query Parameters:

- **page** (integer) -- A page number within the paginated result set. When not given, first page is retrieved by default.

- **page_size** (integer) -- Number of results to return per page. Page size parameter can be a number between `20` and `100`. For disabling pagination and retrieving all results, `0` should be given. When page size parameter is empty or `<20`, `20` results are returned by default. When page size parameter `>100`, `100` results are returned by default.

Status Codes:

- 200 OK --

Response JSON Object:

- **count** (integer) -- (required)
- **next** (string) --
- **previous** (string) --
- **results[].id** (integer) -- (read only)
- **results[].last_heartbeat** (string) -- Time the node sent its last heartbeat event
- **results[].name** (string) -- Hostname for a given worker node (required)
- **results[].online** (string) -- (read only)
- **results[].site** (string) -- (required)
- **results[].url** (string) -- (read only)

`GET /nodes/{id}/`

Parameters:

- **id** (string) --

Status Codes:

- 200 OK --

Response JSON Object:

- **id** (integer) -- (read only)
- **last_heartbeat** (string) -- Time the node sent its last heartbeat event
- **name** (string) -- Hostname for a given worker node (required)
- **online** (string) -- (read only)
- **site** (string) -- (required)
- **url** (string) -- (read only)

`PATCH /nodes/{id}/`

Parameters:

- **id** (string) --

Request JSON Object:

- **id** (integer) -- (read only)
- **last_heartbeat** (string) -- Time the node sent its last heartbeat event
- **name** (string) -- Hostname for a given worker node (required)
- **online** (string) -- (read only)
- **site** (string) -- (required)
- **url** (string) -- (read only)

Status Codes:

- 200 OK --

Response JSON Object:

- **id** (integer) -- (read only)
- **last_heartbeat** (string) -- Time the node sent its last heartbeat event
- **name** (string) -- Hostname for a given worker node (required)
- **online** (string) -- (read only)
- **site** (string) -- (required)

- **url** (string) -- (read only)

DELETE /nodes/{id}/

Parameters:

- **id** (string) --

Status Codes:

- 204 No Content --

GET /pki/ca/pub/

Status Codes:

- 200 OK --

GET /pki/redis/private/

Status Codes:

- 200 OK --

GET /pki/redis/pub/

Status Codes:

- 200 OK --

POST /pki/site/

Status Codes:

- 201 Created --

GET /schedules/

Query Parameters:

- **page** (integer) -- A page number within the paginated result set. When not given, first page is retrieved by default.
- **page_size** (integer) -- Number of results to return per page. Page size parameter can be a number between `20` and `100`. For disabling pagination and retrieving all results, `0` should be given. When page size parameter is empty or `<20`, `20` results are returned by default. When page size parameter `>100`, `100` results are returned by default.

Status Codes:

- 200 OK --

Response JSON Object:

- **count** (integer) -- (required)
- **next** (string) --
- **previous** (string) --
- **results[].day_of_month** (string) -- The day setting for the cron schedule
- **results[].day_of_week** (string) -- The week setting for the cron schedule
- **results[].discovery** (string) --
- **results[].discovery_options** (object) --
- **results[].enabled** (boolean) -- If the schedule should be enabled
- **results[].hour** (string) -- The hour setting for the cron schedule
- **results[].id** (integer) -- (read only)
- **results[].managed_paths** (object) -- Path of managed filesystem elements
- **results[].minute** (string) -- The minute setting for the cron schedule
- **results[].month_of_year** (string) -- The month setting for the cron schedule
- **results[].name** (string) -- Schedule Name (required)

- **results[].site** (string) -- (required)
- **results[].url** (string) -- (read only)

`POST /schedules/`

Request JSON Object:

- **day_of_month** (string) -- The day setting for the cron schedule
- **day_of_week** (string) -- The week setting for the cron schedule
- **discovery** (string) --
- **discovery_options** (object) --
- **enabled** (boolean) -- If the schedule should be enabled
- **hour** (string) -- The hour setting for the cron schedule
- **id** (integer) -- (read only)
- **managed_paths** (object) -- Path of managed filesystem elements
- **minute** (string) -- The minute setting for the cron schedule
- **month_of_year** (string) -- The month setting for the cron schedule
- **name** (string) -- Schedule Name (required)
- **site** (string) -- (required)
- **url** (string) -- (read only)

Status Codes:

- 201 Created --

Response JSON Object:

- **day_of_month** (string) -- The day setting for the cron schedule
- **day_of_week** (string) -- The week setting for the cron schedule
- **discovery** (string) --
- **discovery_options** (object) --
- **enabled** (boolean) -- If the schedule should be enabled
- **hour** (string) -- The hour setting for the cron schedule
- **id** (integer) -- (read only)
- **managed_paths** (object) -- Path of managed filesystem elements
- **minute** (string) -- The minute setting for the cron schedule
- **month_of_year** (string) -- The month setting for the cron schedule
- **name** (string) -- Schedule Name (required)
- **site** (string) -- (required)
- **url** (string) -- (read only)

`GET /schedules/{id}/`

Parameters:

- **id** (string) --

Status Codes:

- 200 OK --

Response JSON Object:

- **day_of_month** (string) -- The day setting for the cron schedule
- **day_of_week** (string) -- The week setting for the cron schedule
- **discovery** (string) --
- **discovery_options** (object) --
- **enabled** (boolean) -- If the schedule should be enabled
- **hour** (string) -- The hour setting for the cron schedule
- **id** (integer) -- (read only)
- **managed_paths** (object) -- Path of managed filesystem elements
- **minute** (string) -- The minute setting for the cron schedule

- **month_of_year** (string) -- The month setting for the cron schedule
- **name** (string) -- Schedule Name (required)
- **site** (string) -- (required)
- **url** (string) -- (read only)

`PATCH /schedules/{id}/`

Parameters:

- **id** (string) --

Request JSON Object:

- **day_of_month** (string) -- The day setting for the cron schedule
- **day_of_week** (string) -- The week setting for the cron schedule
- **discovery** (string) --
- **discovery_options** (object) --
- **enabled** (boolean) -- If the schedule should be enabled
- **hour** (string) -- The hour setting for the cron schedule
- **id** (integer) -- (read only)
- **managed_paths** (object) -- Path of managed filesystem elements
- **minute** (string) -- The minute setting for the cron schedule
- **month_of_year** (string) -- The month setting for the cron schedule
- **name** (string) -- Schedule Name (required)
- **site** (string) -- (required)

Status Codes:

- 200 OK --

Response JSON Object:

- **day_of_month** (string) -- The day setting for the cron schedule
- **day_of_week** (string) -- The week setting for the cron schedule
- **discovery** (string) --
- **discovery_options** (object) --
- **enabled** (boolean) -- If the schedule should be enabled
- **hour** (string) -- The hour setting for the cron schedule
- **id** (integer) -- (read only)
- **managed_paths** (object) -- Path of managed filesystem elements
- **minute** (string) -- The minute setting for the cron schedule
- **month_of_year** (string) -- The month setting for the cron schedule
- **name** (string) -- Schedule Name (required)
- **site** (string) -- (required)

`DELETE /schedules/{id}/`

Parameters:

- **id** (string) --

Status Codes:

- 204 No Content --

`GET /schedules/{parent_lookup_schedule}/workflows/`

Parameters:

- **parent_lookup_schedule** (string) --

Query Parameters:

- **page** (integer) -- A page number within the paginated result set. When not given, first page is retrieved by default.

- **page_size** (integer) -- Number of results to return per page. Page size parameter can be a number between `20` and `100`. For disabling pagination and retrieving all results, `0` should be given. When page size parameter is empty or `<20`, `20` results are returned by default. When page size parameter `>100`, `100` results are returned by default.

Status Codes:

- 200 OK --

Response JSON Object:

- **count** (integer) -- (required)
- **next** (string) --
- **previous** (string) --
- **results[].fields** (object) -- Mapping of path to operation for task usage
- **results[].id** (integer) -- (read only)
- **results[].site** (string) -- (required)
- **results[].url** (string) -- (read only)
- **results[].workflow** (string) -- (required)

`POST /schedules/{parent_lookup_schedule}/workflows/`

Parameters:

- **parent_lookup_schedule** (string) --

Request JSON Object:

- **fields** (object) -- Mapping of path to operation for task usage
- **id** (integer) -- (read only)
- **site** (string) -- (required)
- **workflow** (string) -- (required)

Status Codes:

- 201 Created --

Response JSON Object:

- **fields** (object) -- Mapping of path to operation for task usage
- **id** (integer) -- (read only)
- **site** (string) -- (required)
- **workflow** (string) -- (required)

`GET /schedules/{parent_lookup_schedule}/workflows/{id}/`

Parameters:

- **id** (string) --
- **parent_lookup_schedule** (string) --

Status Codes:

- 200 OK --

Response JSON Object:

- **fields** (object) -- Mapping of path to operation for task usage
- **id** (integer) -- (read only)
- **site** (string) -- (required)
- **url** (string) -- (read only)
- **workflow** (string) -- (required)

`PATCH /schedules/{parent_lookup_schedule}/workflows/{id}/`

Parameters:

- **id** (string) --
- **parent_lookup_schedule** (string) --

Request JSON Object:

- **fields** (object) -- Mapping of path to operation for task usage
- **id** (integer) -- (read only)
- **site** (string) -- (required)
- **url** (string) -- (read only)
- **workflow** (string) -- (required)

Status Codes:

- 200 OK --

Response JSON Object:

- **fields** (object) -- Mapping of path to operation for task usage
- **id** (integer) -- (read only)
- **site** (string) -- (required)
- **url** (string) -- (read only)
- **workflow** (string) -- (required)

`DELETE /schedules/{parent_lookup_schedule}/workflows/{id}/`

Parameters:

- **id** (string) --
- **parent_lookup_schedule** (string) --

Status Codes:

- 204 No Content --

`GET /search/`

API endpoint for file search

Query Parameters:

- **page** (integer) -- A page number within the paginated result set. When not given, first page is retrieved by default.
- **page_size** (integer) -- Number of results to return per page. Page size parameter can be a number between `20` and `100`. For disabling pagination and retrieving all results, `0` should be given. When page size parameter is empty or `<20`, `20` results are returned by default. When page size parameter `>100`, `100` results are returned by default.

Status Codes:

- 200 OK --

Response JSON Object:

- **count** (integer) -- (required)
- **next** (string) --
- **previous** (string) --
- **results[].id** (integer) -- (read only)
- **results[].url** (string) -- (read only)

`POST /search/`

API endpoint for file search

Request JSON Object:

- **filters** (object) -- Metadata filters to apply to search
- **merge** (boolean) -- Whether matching files should be merged
- **metadata_fields** (object) -- Available metadata fields from this search
- **path** (string) -- Directory to search (required)
- **recursive** (boolean) -- Search the target path recursively

- **sites[]** (string) --

Status Codes:

- 201 Created --

Response JSON Object:

- **filters** (object) -- Metadata filters to apply to search
- **merge** (boolean) -- Whether matching files should be merged
- **metadata_fields** (object) -- Available metadata fields from this search
- **path** (string) -- Directory to search (required)
- **recursive** (boolean) -- Search the target path recursively
- **sites[]** (string) --

`GET /search/metadata_fields/`

API endpoint for file search

Query Parameters:

- **page** (integer) -- A page number within the paginated result set. When not given, first page is retrieved by default.
- **page_size** (integer) -- Number of results to return per page. Page size parameter can be a number between `20` and `100`. For disabling pagination and retrieving all results, `0` should be given. When page size parameter is empty or `<20`, `20` results are returned by default. When page size parameter `>100`, `100` results are returned by default.

Status Codes:

- 200 OK --

Response JSON Object:

- **count** (integer) -- (required)
- **next** (string) --
- **previous** (string) --
- **results[].id** (integer) -- (read only)
- **results[].url** (string) -- (read only)

`GET /search/{id}/`

Get paginated results for a given search id.

Parameters:

- **id** (string) --

Query Parameters:

- **page** (integer) -- Number of the page of results to return
- **page_size** (integer) -- Number of results to return per page
- **sort** (string) -- One or more fields to sort results by

Status Codes:

- 200 OK --

Response JSON Object:

- **href** (string) -- (read only)
- **metadata** (object) -- File metadata
- **name** (string) -- Directory or file name (required)
- **path** (string) -- Directory or file path (required)
- **site** (string) -- Site Name

`DELETE /search/{id}/`

API endpoint for file search

Parameters:

- **id** (string) --

Status Codes:

- 204 No Content --

GET /sitelinks/

API endpoint for managing sitelinks.

Query Parameters:

- **page** (integer) -- A page number within the paginated result set. When not given, first page is retrieved by default.
- **page_size** (integer) -- Number of results to return per page. Page size parameter can be a number between `20` and `100`. For disabling pagination and retrieving all results, `0` should be given. When page size parameter is empty or `<20`, `20` results are returned by default. When page size parameter `>100`, `100` results are returned by default.
- **site_id** (integer) -- Site ID
- **datastore_id** (integer) -- Data store ID

Status Codes:

- 200 OK --

Response JSON Object:

- **count** (integer) -- (required)
- **next** (string) --
- **previous** (string) --
- **results[].datastore** (string) -- (required)
- **results[].datastore_path** (string) -- (required)
- **results[].id** (integer) -- (read only)
- **results[].site** (string) -- (required)
- **results[].site_path** (string) -- (required)
- **results[].url** (string) -- (read only)

POST /sitelinks/

API endpoint for managing sitelinks.

Request JSON Object:

- **datastore** (string) -- (required)
- **datastore_path** (string) -- (required)
- **id** (integer) -- (read only)
- **site** (string) -- (required)
- **site_path** (string) -- (required)
- **url** (string) -- (read only)

Status Codes:

- 201 Created --

Response JSON Object:

- **datastore** (string) -- (required)
- **datastore_path** (string) -- (required)
- **id** (integer) -- (read only)
- **site** (string) -- (required)
- **site_path** (string) -- (required)
- **url** (string) -- (read only)

`GET /sitelinks/{id}/`

API endpoint for managing sitelinks.

Parameters:
- **id** (string) --

Status Codes:
- 200 OK --

Response JSON Object:
- **datastore** (string) -- (required)
- **datastore_path** (string) -- (required)
- **id** (integer) -- (read only)
- **site** (string) -- (required)
- **site_path** (string) -- (required)
- **url** (string) -- (read only)

`PATCH /sitelinks/{id}/`

API endpoint for managing sitelinks.

Parameters:
- **id** (string) --

Request JSON Object:
- **datastore** (string) -- (required)
- **datastore_path** (string) -- (required)
- **id** (integer) -- (read only)
- **site** (string) -- (required)
- **site_path** (string) -- (required)
- **url** (string) -- (read only)

Status Codes:
- 200 OK --

Response JSON Object:
- **datastore** (string) -- (required)
- **datastore_path** (string) -- (required)
- **id** (integer) -- (read only)
- **site** (string) -- (required)
- **site_path** (string) -- (required)
- **url** (string) -- (read only)

`DELETE /sitelinks/{id}/`

API endpoint for managing sitelinks.

Parameters:
- **id** (string) --

Status Codes:
- 204 No Content --

`GET /sites/`

API endpoint for managing sites.

Query Parameters:
- **page** (integer) -- A page number within the paginated result set. When not given, first page is retrieved by default.

- **page_size** (integer) -- Number of results to return per page. Page size parameter can be a number between `20` and `100`. For disabling pagination and retrieving all results, `0` should be given. When page size parameter is empty or `<20`, `20` results are returned by default. When page size parameter `>100`, `100` results are returned by default.

Status Codes:

- 200 OK --

Response JSON Object:

- **count** (integer) -- (required)
- **next** (string) --
- **previous** (string) --
- **results[].id** (integer) -- (read only)
- **results[].name** (string) -- Site Name (required)
- **results[].url** (string) -- (read only)

POST /sites/

API endpoint for managing sites.

Request JSON Object:

- **bandwidth** (integer) -- speed for site (in Mb/s)
- **elasticsearch_url** (string) -- URL of the Elasticsearch server to use for the Analytics search backend on this site
- **exclude** (object) -- Global workflow excludes for this site
- **file_batch_gb** (integer) -- File batch GB
- **file_batch_size** (integer) -- File batch size
- **gpfs_iscan_buckets** (integer) -- Number of buckets to use for the gpfs snapdiff policy
- **gpfs_iscan_threads** (integer) -- Number of threads to use for the gpfs snapdiff policy
- **id** (integer) -- (read only)
- **include** (object) -- Global workflow includes for this site
- **lock_threshold** (integer) -- Threshold for soft locking snapshot rotations
- **name** (string) -- Site Name (required)
- **pixstor_search_url** (string) -- The base URL for querying the PixStor API
- **url** (string) -- (read only)

Status Codes:

- 201 Created --

Response JSON Object:

- **bandwidth** (integer) -- speed for site (in Mb/s)
- **elasticsearch_url** (string) -- URL of the Elasticsearch server to use for the Analytics search backend on this site
- **exclude** (object) -- Global workflow excludes for this site
- **file_batch_gb** (integer) -- File batch GB
- **file_batch_size** (integer) -- File batch size
- **gpfs_iscan_buckets** (integer) -- Number of buckets to use for the gpfs snapdiff policy
- **gpfs_iscan_threads** (integer) -- Number of threads to use for the gpfs snapdiff policy
- **id** (integer) -- (read only)
- **include** (object) -- Global workflow includes for this site

- **lock_threshold** (integer) -- Threshold for soft locking snapshot rotations
- **name** (string) -- Site Name (required)
- **pixstor_search_url** (string) -- The base URL for querying the PixStor API
- **url** (string) -- (read only)

`GET /sites/{id}/`

API endpoint for managing sites.

Parameters:

- **id** (string) --

Status Codes:

- 200 OK --

Response JSON Object:

- **bandwidth** (integer) -- speed for site (in Mb/s)
- **elasticsearch_url** (string) -- URL of the Elasticsearch server to use for the Analytics search backend on this site
- **exclude** (object) -- Global workflow excludes for this site
- **file_batch_gb** (integer) -- File batch GB
- **file_batch_size** (integer) -- File batch size
- **gpfs_iscan_buckets** (integer) -- Number of buckets to use for the gpfs snapdiff policy
- **gpfs_iscan_threads** (integer) -- Number of threads to use for the gpfs snapdiff policy
- **id** (integer) -- (read only)
- **include** (object) -- Global workflow includes for this site
- **lock_threshold** (integer) -- Threshold for soft locking snapshot rotations
- **name** (string) -- Site Name (required)
- **pixstor_search_url** (string) -- The base URL for querying the PixStor API
- **url** (string) -- (read only)

`PATCH /sites/{id}/`

API endpoint for managing sites.

Parameters:

- **id** (string) --

Request JSON Object:

- **bandwidth** (integer) -- speed for site (in Mb/s)
- **elasticsearch_url** (string) -- URL of the Elasticsearch server to use for the Analytics search backend on this site
- **exclude** (object) -- Global workflow excludes for this site
- **file_batch_gb** (integer) -- File batch GB
- **file_batch_size** (integer) -- File batch size
- **gpfs_iscan_buckets** (integer) -- Number of buckets to use for the gpfs snapdiff policy
- **gpfs_iscan_threads** (integer) -- Number of threads to use for the gpfs snapdiff policy
- **id** (integer) -- (read only)
- **include** (object) -- Global workflow includes for this site
- **lock_threshold** (integer) -- Threshold for soft locking snapshot rotations
- **name** (string) -- Site Name (required)

- **pixstor_search_url** (string) -- The base URL for querying the PixStor API
- **url** (string) -- (read only)

Status Codes:

- 200 OK --

Response JSON Object:

- **bandwidth** (integer) -- speed for site (in Mb/s)
- **elasticsearch_url** (string) -- URL of the Elasticsearch server to use for the Analytics search backend on this site
- **exclude** (object) -- Global workflow excludes for this site
- **file_batch_gb** (integer) -- File batch GB
- **file_batch_size** (integer) -- File batch size
- **gpfs_iscan_buckets** (integer) -- Number of buckets to use for the gpfs snapdiff policy
- **gpfs_iscan_threads** (integer) -- Number of threads to use for the gpfs snapdiff policy
- **id** (integer) -- (read only)
- **include** (object) -- Global workflow includes for this site
- **lock_threshold** (integer) -- Threshold for soft locking snapshot rotations
- **name** (string) -- Site Name (required)
- **pixstor_search_url** (string) -- The base URL for querying the PixStor API
- **url** (string) -- (read only)

`DELETE /sites/{id}/`

API endpoint for managing sites.

Parameters:

- **id** (string) --

Status Codes:

- 204 No Content --

`GET /sites/{id}/health/`

API endpoint for managing sites.

Parameters:

- **id** (string) --

Status Codes:

- 200 OK --

Response JSON Object:

- **bandwidth** (integer) -- speed for site (in Mb/s)
- **elasticsearch_url** (string) -- URL of the Elasticsearch server to use for the Analytics search backend on this site
- **exclude** (object) -- Global workflow excludes for this site
- **file_batch_gb** (integer) -- File batch GB
- **file_batch_size** (integer) -- File batch size
- **gpfs_iscan_buckets** (integer) -- Number of buckets to use for the gpfs snapdiff policy
- **gpfs_iscan_threads** (integer) -- Number of threads to use for the gpfs snapdiff policy
- **id** (integer) -- (read only)
- **include** (object) -- Global workflow includes for this site

- **lock_threshold** (integer) -- Threshold for soft locking snapshot rotations
- **name** (string) -- Site Name (required)
- **pixstor_search_url** (string) -- The base URL for querying the PixStor API
- **url** (string) -- (read only)

`GET /tasks/`

API endpoint for viewing tasks.

Query Parameters:

- **page** (integer) -- A page number within the paginated result set. When not given, first page is retrieved by default.
- **page_size** (integer) -- Number of results to return per page. Page size parameter can be a number between `20` and `100`. For disabling pagination and retrieving all results, `0` should be given. When page size parameter is empty or `<20`, `20` results are returned by default. When page size parameter `>100`, `100` results are returned by default.
- **tasktype** (string) -- Task type
- **state** (array) -- Task states
- **job_id** (integer) -- Job ID

Status Codes:

- 200 OK --

Response JSON Object:

- **count** (integer) -- (required)
- **next** (string) --
- **previous** (string) --
- **results[].job** (integer) -- (required)
- **results[].site** (string) -- (required)
- **results[].started** (string) -- Time that the task started running
- **results[].state** (string) -- (required)
- **results[].taskid** (string) -- Job task ID (required)
- **results[].tasktype** (string) -- (required)
- **results[].url** (string) -- (read only)

`GET /tasks/{taskid}/`

API endpoint for viewing tasks.

Parameters:

- **taskid** (string) --

Status Codes:

- 200 OK --

Response JSON Object:

- **completed** (string) -- Time of the task completion
- **job** (integer) -- (required)
- **moved_data** (integer) --
- **numabortedfiles** (integer) --
- **numcancelledfiles** (integer) --
- **numfailedfiles** (integer) --
- **numfiles** (integer) --
- **numprocessedfiles** (integer) --
- **numskippedfiles** (integer) --

- **paths** (string) -- (read only)
- **results** (string) -- (read only)
- **runtime** (string) -- (read only)
- **site** (string) -- (required)
- **started** (string) -- Time that the task started running
- **state** (string) -- (required)
- **taskid** (string) -- Job task ID (required)
- **tasktype** (string) -- (required)
- **url** (string) -- (read only)

`GET /tasks/{taskid}/files/`

API endpoint for viewing tasks.

Parameters:

- **taskid** (string) --

Query Parameters:

- **state** (string) --
- **type** (string) --
- **site** (string) --
- **page** (integer) -- A page number within the paginated result set. When not given, first page is retrieved by default.
- **page_size** (integer) -- Number of results to return per page. Page size parameter can be a number between `20` and `100`. For disabling pagination and retrieving all results, `0` should be given. When page size parameter is empty or `<20`, `20` results are returned by default. When page size parameter `>100`, `100` results are returned by default.

Status Codes:

- 200 OK --

Response JSON Object:

- **completed** (string) -- Time of the task completion
- **job** (integer) -- (required)
- **moved_data** (integer) --
- **numabortedfiles** (integer) --
- **numcancelledfiles** (integer) --
- **numfailedfiles** (integer) --
- **numfiles** (integer) --
- **numprocessedfiles** (integer) --
- **numskippedfiles** (integer) --
- **paths** (string) -- (read only)
- **results** (string) -- (read only)
- **runtime** (string) -- (read only)
- **site** (string) -- (required)
- **started** (string) -- Time that the task started running
- **state** (string) -- (required)
- **taskid** (string) -- Job task ID (required)
- **tasktype** (string) -- (required)
- **url** (string) -- (read only)

`GET /users/`

API endpoint for managing users.

Query Parameters:

- **page** (integer) -- A page number within the paginated result set. When not given, first page is retrieved by default.
- **page_size** (integer) -- Number of results to return per page. Page size parameter can be a number between `20` and `100`. For disabling pagination and retrieving all results, `0` should be given. When page size parameter is empty or `<20`, `20` results are returned by default. When page size parameter `>100`, `100` results are returned by default.

Status Codes:

- 200 OK --

Response JSON Object:

- **count** (integer) -- (required)
- **next** (string) --
- **previous** (string) --
- **results[].date_joined** (string) --
- **results[].email** (string) --
- **results[].first_name** (string) --
- **results[].groups[].id** (integer) -- (read only)
- **results[].groups[].name** (string) -- (required)
- **results[].groups[].url** (string) -- (read only)
- **results[].id** (integer) -- (read only)
- **results[].is_active** (boolean) -- Designates whether this user should be treated as active. Unselect this instead of deleting accounts.
- **results[].last_login** (string) --
- **results[].last_name** (string) --
- **results[].url** (string) -- (read only)
- **results[].username** (string) -- Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only. (required)

`POST /users/`

API endpoint for managing users.

Request JSON Object:

- **email** (string) --
- **first_name** (string) --
- **groups[]** (string) --
- **last_name** (string) --
- **password** (string) -- (required)
- **username** (string) -- Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only. (required)

Status Codes:

- 201 Created --

Response JSON Object:

- **email** (string) --
- **first_name** (string) --
- **groups[]** (string) --
- **last_name** (string) --
- **password** (string) -- (required)
- **username** (string) -- Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only. (required)

`GET /users/{username}/`

API endpoint for managing users.

Parameters:

- **username** (string) --

Status Codes:

- 200 OK --

Response JSON Object:

- **date_joined** (string) --
- **email** (string) --
- **first_name** (string) --
- **groups[].id** (integer) -- (read only)
- **groups[].name** (string) -- (required)
- **groups[].url** (string) -- (read only)
- **id** (integer) -- (read only)
- **is_active** (boolean) -- Designates whether this user should be treated as active. Unselect this instead of deleting accounts.
- **last_login** (string) --
- **last_name** (string) --
- **url** (string) -- (read only)
- **username** (string) -- Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only. (required)

`PATCH /users/{username}/`

API endpoint for managing users.

Parameters:

- **username** (string) --

Request JSON Object:

- **email** (string) --
- **first_name** (string) --
- **groups[]** (string) --
- **last_name** (string) --
- **password** (string) --

Status Codes:

- 200 OK --

Response JSON Object:

- **email** (string) --
- **first_name** (string) --
- **groups[]** (string) --
- **last_name** (string) --
- **password** (string) --

`DELETE /users/{username}/`

API endpoint for managing users.

Parameters:

- **username** (string) --

Status Codes:

- 204 No Content --

POST /users/{username}/activate/

Activates user account with given username.

Parameters:

- **username** (string) --

Status Codes:

- 201 Created --

POST /users/{username}/deactivate/

Deactivates user account with given username.

Parameters:

- **username** (string) --

Status Codes:

- 201 Created --

GET /workflows/

API endpoint for viewing workflows.

Query Parameters:

- **page** (integer) -- A page number within the paginated result set. When not given, first page is retrieved by default.
- **page_size** (integer) -- Number of results to return per page. Page size parameter can be a number between `20` and `100`. For disabling pagination and retrieving all results, `0` should be given. When page size parameter is empty or `<20`, `20` results are returned by default. When page size parameter `>100`, `100` results are returned by default.

Status Codes:

- 200 OK --

Response JSON Object:

- **count** (integer) -- (required)
- **next** (string) --
- **previous** (string) --
- **results[].allow_missing_paths** (boolean) -- To allow paths that does not exist
- **results[].discovery** (string) --
- **results[].discovery_options** (object) --
- **results[].enabled** (boolean) -- Is the workflow available for use?
- **results[].fields** (object) --
- **results[].filter_rules** (object) --
- **results[].icon_classes** (object) --
- **results[].id** (integer) -- (read only)
- **results[].label** (string) -- Friendly name of the workflow (required)
- **results[].name** (string) -- Name of Workflow (required)
- **results[].schedule_only** (boolean) -- Workflow only callable inside of a schedule
- **results[].visible** (boolean) -- Is the workflow visible on the UI?

POST /workflows/

API endpoint for viewing workflows.

Request JSON Object:

- **allow_missing_paths** (boolean) -- To allow paths that does not exist
- **discovery** (string) --
- **discovery_options** (object) --
- **enabled** (boolean) -- Is the workflow available for use?
- **fields** (object) --
- **filter_rules** (object) --
- **icon_classes** (object) --
- **id** (integer) -- (read only)
- **label** (string) -- Friendly name of the workflow (required)
- **name** (string) -- Name of Workflow (required)
- **schedule_only** (boolean) -- Workflow only callable inside of a schedule
- **visible** (boolean) -- Is the workflow visible on the UI?

Status Codes:

- 201 Created --

Response JSON Object:

- **allow_missing_paths** (boolean) -- To allow paths that does not exist
- **discovery** (string) --
- **discovery_options** (object) --
- **enabled** (boolean) -- Is the workflow available for use?
- **fields** (object) --
- **filter_rules** (object) --
- **icon_classes** (object) --
- **id** (integer) -- (read only)
- **label** (string) -- Friendly name of the workflow (required)
- **name** (string) -- Name of Workflow (required)
- **schedule_only** (boolean) -- Workflow only callable inside of a schedule
- **visible** (boolean) -- Is the workflow visible on the UI?

`GET /workflows/{id}/`

API endpoint for viewing workflows.

Parameters:

- **id** (string) --

Status Codes:

- 200 OK --

Response JSON Object:

- **allow_missing_paths** (boolean) -- To allow paths that does not exist
- **discovery** (string) --
- **discovery_options** (object) --
- **enabled** (boolean) -- Is the workflow available for use?
- **fields** (object) --
- **filter_rules** (object) --
- **icon_classes** (object) --
- **id** (integer) -- (read only)
- **label** (string) -- Friendly name of the workflow (required)
- **name** (string) -- Name of Workflow (required)
- **schedule_only** (boolean) -- Workflow only callable inside of a schedule
- **visible** (boolean) -- Is the workflow visible on the UI?

`PATCH /workflows/{id}/`

API endpoint for viewing workflows.

Parameters:

- **id** (string) --

Request JSON Object:

- **allow_missing_paths** (boolean) -- To allow paths that does not exist
- **discovery** (string) --
- **discovery_options** (object) --
- **enabled** (boolean) -- Is the workflow available for use?
- **fields** (object) --
- **filter_rules** (object) --
- **icon_classes** (object) --
- **id** (integer) -- (read only)
- **label** (string) -- Friendly name of the workflow (required)
- **name** (string) -- Name of Workflow (required)
- **schedule_only** (boolean) -- Workflow only callable inside of a schedule
- **visible** (boolean) -- Is the workflow visible on the UI?

Status Codes:

- 200 OK --

Response JSON Object:

- **allow_missing_paths** (boolean) -- To allow paths that does not exist
- **discovery** (string) --
- **discovery_options** (object) --
- **enabled** (boolean) -- Is the workflow available for use?
- **fields** (object) --
- **filter_rules** (object) --
- **icon_classes** (object) --
- **id** (integer) -- (read only)
- **label** (string) -- Friendly name of the workflow (required)
- **name** (string) -- Name of Workflow (required)
- **schedule_only** (boolean) -- Workflow only callable inside of a schedule
- **visible** (boolean) -- Is the workflow visible on the UI?

`DELETE /workflows/{id}/`

API endpoint for viewing workflows.

Parameters:

- **id** (string) --

Status Codes:

- 204 No Content --

Tools

The following sections document add-on tools for Ngenea Hub

ngclient

NGCLIENT

SYNOPSIS

ngclient authenticate (-u USERNAME [-p PASSWORD] | -T TOKEN) [-k NAME]

ngclient migrate path... [-s site] [-r] [-p] [options...]

ngclient recall path... [-s site] [-r] [options...]

ngclient send path... [-s source] -t target [options...]

ngclient workflows COMMAND

ngclient features COMMAND

DESCRIPTION

ngclient is a CLI wrapper for the default Ngenea Hub workflows - migrate, recall, and send. It also provides a mechanism for generating authentication tokens.

ngclient settings can be read from a config file, rather than being passed on the command line. See **ngenea-client.conf(5)** for more information on the configuration format. CLI flags take precedence over config file settings.

The authenticate command can be used to generate a client key from a username or access token. The generated client key will be printed to stdout. That client key can then be used with the workflow sub-commands, either via the --client-key flag, or by saving it to the **ngenea-client.conf(5)** configuration file.

The workflows command group contains sub-commands for interacting with workflows, such as listing workflows and importing new one. See **ngclient-workflows(1)** for more details.

The features command group contains sub-commands for listing, enabling, or disabling feature flags. See **ngclient-features(1)** for more details.

OPTION SUMMARY

```
     path...                One paths to call the workflow on

-T, --access-token TOKEN    Access token to authenticate with
-u, --username USERNAME     Username to authenticate with
-p, --password PASSWORD     Password for the authentication username
-k, --key-name NAME         Unique name for the client key

    --base-url              Base URL of the {{ brand_name }} API

-c, --config CONFIG         Alternative configuration file path
    --client-key KEY        Client API key to authenticate with

-s, --site SITE             Site to perform the workflow on
```

```
-t, --target TARGET         Site to send files to

-d, --no-wait               Exit after job is submitted, don't wait
for it to complete
    --timeout SECONDS       wait for completion timeout. If not set,
wait indefinitely

-r, --recursive             Perform task recursively.
-p, --premigrate            Premigrate files from site
-H, --hydrate               Hydrate files on the send target site

-h, --help                  Print help message and exit
```

OPTIONS

- **-T**, **--access-token**

  An access token to generate a client key with.

- **-u**, **--username**

  Username to generate a client key with.

- **-p**, **--password**

  Password to use in combination with --username

  If --username is specified and --password isn't, you will be prompted to enter a password interactively. This may be preferrable so that the password doesn't appear in shell history.

- **-k**, **--key-name**

  A unique name to assign when generating a client key.

  This will be displayed in the Ngenea Hub UI

  If not specified, a random uuid will be generated for the key name.

- **--base-url**

  Base URL of the Ngenea Hub API, which operations will be performed against.

  This can be used to perform Ngenea Hub operations on a remote server.

  If not specified, the default is http://localhost:8000/api

- **-c**, **--config**

  The path to an alternative configuration file.

  If not specified, the default configuration paths will be used. The default paths are in the user's HOME directory $HOME/.config/ngenea/ngenea-client.conf, and the global configuration at /etc/ngenea/ngenea-client.conf

See **ngenea-client.conf(5)** for more information on the configuration format.

Command line options take precedence over any corresponding config file settings.

- **-D**, **--no-verify**

  Disables TLS Verification

  By default, **ngclient** will verify TLS certificate at the remote end.

  With this flag, requests made by **ngclient** will accept any TLS certificate presented by the server, and will ignore hostname mismatches and/or expired certificates.

- **--client-key**

  Ngenea Hub authentication client key.

  This can be generated via the Ngenea Hub REST API, or using **ngclient(1)** authenticate

- **-s**, **--site**

  Site to use for workflows.

  For migrate and recall, this is the site where the workflows execute. For send, this is the source site, from which files are sent.

  Site does not have to match the node where **ngclient** is being called. This can be used to migrate/recall/send files from a remote site.

  Note - shell-globbing will be evaluated on the local node. For a remote site, the files that the glob would match may differ.

- **-t**, **--target**

  Target site for the send workflow

- **-d**, **--no-wait**

  Don't wait for workflows to complete.

  By default, **ngclient** will wait for the workflow to complete, subject to --timeout.

  With this flag, **ngclient** will exit immediately. The workflow will continue to execute independently. In that case, the workflow can be monitored in the Ngenea Hub UI

- **--timeout**

  How long to wait for workflows to complete, in seconds.

  If not specified, **ngclient** will wait indefinitely.

If the workflow doesn't complete within the timeout, the client will exit with an error. The workflow itself may continue to execute.

- **-r**, **--recursive**

  Migrate or recall files and directories recursively.

- **-p**, **--premigrate**

  Premigrate files

  Premigrated files are migrated, but the data is kept resident.

- **-H**, **--hydrate**

  Controls whether the send workflow 'hydrates' files on the target.

  If false, files are only reverse stubbed on the target.

- **-h**, **--help**

  Prints the help message.

EXAMPLES

GENERATE A CLIENT KEY

```
$ ngclient authenticate --username pixadmin -k ngclient-pixadmin
pixadmin's password:

jDBh2cRk6.LswQfylT2BtGiqtYUWhMB1iipJmQNgr
```

NOTE: The `authenticate` command doesn't read config values from the config file, so the --no-verify and --base-url arguments should be passed as cmd arguments if default values aren't required

RECALL A FILE

```
ngclient recall /mmfs1/data/hello.txt -s site1 --client-key
jDBh2cRk6.LswQfylT2BtGiqtYUWhMB1iipJmQNgr
```

For brevity, the site and client-key can be saved to the config file

PREMIGRATE A DIRECTORY RECURSIVELY

Assuming the site and client key has been saved to the config file

```
ngclient migrate /mmfs1/data/sample_data/cats -p -r
```

SEND A FILE TO A REMOTE SITE

```
ngclient send /mmfs1/data/hello.txt -s site1 -t site2 --hydrate
```

AVAILABILITY

Distributed as part of the ngenea-hub-client rpm, or the ngclient wheel (Python) for non-Red Hat based systems.

The ngclient wheel can be installed and run on any operating system.

Note - transparent_recall(1) is packaged along with ngclient, but transparent_recall will only work on Unix-based operating systems.

SEE ALSO

ngenea-client.conf(5), ngclient-workflows(1), ngclient-features(1), transparent_recall(1), ngmigrate(1), ngrecall(1)

LICENSE

---

ngclient-workflows

NGCLIENT-WORKFLOWS

SYNOPSIS

ngclient workflows list [workflow-id] [options...]

ngclient workflows import workflow-file [options...]

ngclient workflows update workflow-id workflow-file [options...]

ngclient workflows delete workflow-id [options...]

DESCRIPTION

The list command is used to list one or more existing workflows from Ngenea Hub. By default, workflows are output in json format, one per line. The --yaml flag can be used to output as yaml.

The import command can be used to import a new, custom workflow from a json or yaml formatted file.

Currently it is not possible to invoke these custom workflows from **ngclient**, once created. They can be invoked from the Ngenea Hub UI or via the REST API.

The update command can be used to update an existing workflow from a json or yaml formatted file. The file can contain only the fields you want to change to perform a partial update, or a whole workflow definition for a full replacement.

NOTE - it's not possible to make partial changes to the fields or filter_rules blocks. They can only be replaced as a whole.

The delete command can be used to delete an existing workflow, by id.

Base URL and API key settings can be read from a config file, rather than being passed on the command line. See **ngenea-client.conf(5)** for more information on the configuration format. CLI flags take precedence over config file settings.

Interacting with workflows requires Ngenea Hub authentication. The **ngclient(1)** authenticate command can be used to generate a client key from a username or access token.

OPTION SUMMARY

```
      workflow-id           Unique workflow identifier
      workflow-file         File containing a custom workflow
definition

    --yaml                  List workflows in yaml format

    --base-url              Base URL of the {{ brand_name }} API

-c, --config CONFIG         Alternative configuration file path
    --client-key KEY        Client API key to authenticate with

-h, --help                  Print help message and exit
```

OPTIONS

- **workflow-file**

  Path to a json or yaml formatted file, containing a workflow definition.

  If '-' is used, the workflow definition will be read from stdin.

  The workflow format is described in the main documentation, section '4.4. Custom Workflows'

- **--yaml**

  List workflows in yaml format.

By default, workflows are output in json format, one per line (jsonl).

The --yaml flag will output the workflows in structured yaml format. If multiple workflows are being listed, each one will be separated by a blank line.

- **--base-url**

  Base URL of the Ngenea Hub API, which operations will be performed against.

  This can be used to perform Ngenea Hub operations on a remote server.

  If not specified, the default is http://localhost:8000/api

- **-c**, **--config**

  The path to an alternative configuration file.

  If not specified, the default configuration paths will be used. The default paths are in the user's HOME directory $HOME/.config/ngenea/ngenea-client.conf, and the global configuration at /etc/ngenea/ngenea-client.conf

  See **ngenea-client.conf(5)** for more information on the configuration format.

  Command line options take precedence over any corresponding config file settings.

- **--client-key**

  Ngenea Hub authentication client key.

  This can be generated via the Ngenea Hub REST API, or using **ngclient(1)** authenticate

- **-h**, **--help**

  Prints the help message.

EXAMPLES

GENERATE A CLIENT KEY

```
$ ngclient authenticate --username pixadmin -k ngclient-pixadmin
pixadmin's password:

jDBh2cRk6.LswQfylT2BtGiqtYUWhMB1iipJmQNgr
```

The following examples assume that the client key has been saved in the default config file.

GET AN EXISTING WORKFLOW

```
$ ngclient workflows list 1
{"id": 1, "name": "migrate", "label": "Migrate", "icon_classes":
```

```
["fa fa-cloud fa-stack-2x text-success", "fa fa-angle-up fa-stack-2x
text-light"], "discovery": "recursive", "enabled": true, "visible":
true, "fields": [], "filter_rules": [{"type": "all", "state": "all",
"action": [{"name": "dynamo.tasks.migrate"}], "description":
"Migrates a file off from a given path"}]}
```

LIST ALL EXISTING WORKFLOWS IN YAML FORMAT

```
$ ngclient workflows list --yaml
id: 1
name: migrate
label: Migrate
discovery: recursive
...
enabled: true
visible: true

id: 2
name: premigrate
label: Premigrate
discovery: recursive
...
enabled: true
visible: true

...
```

(the above example output has been truncated)

IMPORT A CUSTOM WORKFLOW

Using the following workflow definition in json format

```
$ cat overwrite_workflow.json
{"name": "recall_overwrite", "label": "Overwrite On Recall",
"icon_classes": ["fa fa-cloud fa-stack-2x text-primary", "fa fa-
caret-down fa-stack-2x text-light"], "filter_rules": [{"type":
"all", "state": "all", "action": [{"name":
"dynamo.tasks.reverse_stub", "site": "*destinationsite",
"overwrite": true}]}], "fields": [{"name": "destinationsite",
"type": "enum[site]", "label": "Destination Site", "value": "site"}]}
```

Import the workflow as follows

```
ngclient workflows import overwrite_workflow.json
```

RENAME A WORKFLOW

With the change in yaml format, using '-' to read from stdin

```
echo "name: overwrite_on_recall" | ngclient workflows update 6 -
```

DELETE A WORKFLOW

```
ngclient workflows delete 6
```

AVAILABILITY

Distributed as part of the ngenea-hub-client rpm, or the ngclient wheel (Python) for non-Red Hat based systems.

The ngclient wheel can be installed and run on any operating system.

SEE ALSO

ngclient(1), ngenea-client.conf(5)

LICENSE

---

ngclient-features

NGCLIENT-FEATURES

SYNOPSIS

ngclient features list [options...]

ngclient features enable name [options...]

ngclient features disable name [options...]

DESCRIPTION

The list command is used to list available feature flags for .

The enable and disable commands can be used to enable a named feature in Ngenea Hub.

Base URL and API key settings can be read from a config file, rather than being passed on the command line. See **ngenea-client.conf(5)** for more information on the configuration format. CLI flags take precedence over config file settings.

Interacting with features requires Ngenea Hub authentication. The **ngclient(1)** authenticate command can be used to generate a client key from a username or access token.

OPTION SUMMARY

```
      name                     Name of the feature to enable or disable

    --json                     List features in json format

    --base-url                 Base URL of the {{ brand_name }} API

-c, --config CONFIG            Alternative configuration file path
    --client-key KEY           Client API key to authenticate with

-h, --help                     Print help message and exit
```

OPTIONS

- **--json**

  List features in json format.

  By default, the list command will report features in a table-based format. The --json flag will report features in json format instead, one per line.

- **--base-url**

  Base URL of the Ngenea Hub API, which operations will be performed against.

  This can be used to perform Ngenea Hub operations on a remote server.

  If not specified, the default is http://localhost:8000/api

- **-c**, **--config**

  The path to an alternative configuration file.

  If not specified, the default configuration paths will be used. The default paths are in the user's HOME directory $HOME/.config/ngenea/ngenea-client.conf, and the global configuration at /etc/ngenea/ngenea-client.conf

  See **ngenea-client.conf(5)** for more information on the configuration format.

  Command line options take precedence over any corresponding config file settings.

- **--client-key**

  Ngenea Hub authentication client key.

  This can be generated via the Ngenea Hub REST API, or using **ngclient(1)** authenticate

- **-h**, **--help**

  Prints the help message.

EXAMPLES

The following examples assume that the client key has been saved in the default config file.

LIST AVAILABLE FEATURES

```
$ ngclient features list
 [X]  searchui              Enable search features in the UI
 [ ]  bandwidth_controls    Enable bandwidth controls in the UI
 [ ]  rbac                  Enable role-based access controls
```

(The above are just examples and may not reflect actual feature flags)

ENABLE A FEATURE

```
ngclient features enable rbac
```

DISABLE A FEATURE

```
ngclient features disable searchui
```

AVAILABILITY

Distributed as part of the ngenea-hub-client rpm, or the ngclient wheel (Python) for non-Red Hat based systems.

The ngclient wheel can be installed and run on any operating system.

SEE ALSO

ngclient(1), ngenea-client.conf(5)

LICENSE

`ngenea-client.conf`

NGENEACLIENTCONF

SYNOPSIS

The Ngenea Hub client configuration files is used to configure **ngclient(1)** and **transparent_recall(1)**

The default config file locations are in the user's HOME directory $HOME/.config/ngenea/ngenea-client.conf, with a global config at /etc/ngenea/ngenea-client.conf.

If both configuration files exist, the user config will take precedence, with the global config used for any values not specified in the user config.

For example

```
# global config
[settings]
base_url = http://10.172.0.23:8000/api
site = default

# user config
[settings]
site = mysite
```

would result in base_url = http://10.172.0.23:8000/api, since it's not specified in the user config, and site = mysite since the value from the user config takes precedence.

NOTE - unless explicitly specified with the --config flag, both ngclient(1) and transparent_recall(1) will use this same default config files.

If a config file is explicitly specified with --config, the default configs will not be considered at all.

Command line options take precedence over any corresponding config file settings.

FILE FORMAT

ngenea-client.conf(5) uses an ini-style format.

It is made up of key = value lines under the [settings] section header.

```
[settings]
client_key = mykey
```

Boolean type values can be either true, false, yes, or no (case-insensitive)

Additional sections, or unrecognised keys are ignored.

PARAMETERS

- **base_url**

Base URL of the Ngenea Hub API, which operations will be performed against.

This can be used to perform Ngenea Hub operations on a remote server.

If not specified, the default is http://localhost:8000/api

- **client_key**

  Ngenea Hub authentication client key.

  This can be generated via the Ngenea Hub REST API, or using **ngclient(1)** authenticate

- **site**

  The default site to use for workflows.

  For migrate and recall, this is the site where the workflows execute. For send, this is the source site, from which files are sent.

- **wait**

  Whether to wait for workflows to complete.

  If true (default), tools will wait for the workflow to complete, subject to timeout.

  If false, tools will exit immediately. The workflow will continue to execute independently. In that case, the workflow can be monitored in the Ngenea Hub UI

- **timeout**

  How long to wait for workflows to complete, in seconds.

  If not set, tools will wait indefinitely.

  If the workflow doesn't complete within the timeout, the client will exit with an error. The workflow itself may continue to execute.

- **hydrate**

  For **ngclient(1)** send, controls whether sent files are hydrated on the target.

  If false, files are only reverse stubbed on the target.

- **api_secure_verify**

  Whether to enable/disable TLS Verification when communicating with Ngenea Hub REST API.

  This is particularly useful when Ngenea Hub REST API is behind a self-signed certificate.

  If false, TLS verification will be disabled.

If not specified, the default is true.

EXAMPLE

```
[settings]
base_url = http://mypixserver:8000/api
client_key = ...
site = mysite
wait = true
timeout = 180
hydrate = true
api_secure_verify = true
```

SEE ALSO

ngclient(1), transparent_recall(1)

LICENSE

---

transparent_recall

TRANSPARENT_RECALL

SYNOPSIS

transparent_recall file [--config CONF]

DESCRIPTION

transparent_recall is a tool for recalling individual files via Ngenea Hub

Performing recalls via Ngenea Hub allows for monitoring progress via the Ngenea Hub UI. Individual recall tasks performed on demand, but for reporting are grouped together into one job per hour.

transparent_recall can be called directly to recall files, but typically would be installed as a filesystem policy rule. See **TRANSPARENT RECALL POLICY** for more info.

OPTION SUMMARY

```
      file          One or more directory to export events from

    --config CONF   Alternative configuration file location
-h, --help          Print help message and exit
```

OPTIONS

- **--config**

  The path to an alternative configuration file.

  If not specified, the default configuration paths will be used. The default paths are in the user's HOME directory $HOME/.config/ngenea/ngenea-client.conf, and the global configuration at /etc/ngenea/ngenea-client.conf

  See the **CONFIGURATION** section below and ngenea-client.conf(5) for more information.

  Command line options take precedence over any corresponding config file settings.

- **-h**, **--help**

  Prints the help message.

CONFIGURATION

transparent_recall requires authentication to be able to perform recalls via Ngenea Hub. To authenticate, a valid client_key must be placed in the configuration file.

A client key can be generated via the Ngenea Hub REST API, or using the **ngclient(1)** authenticate command.

Minimally, the configuration must include this client_key, as well as the site where recalls are performed.

```
[settings]
client_key = ...
site = thissite
```

The site must match the the node where the recall was triggered.

transparent_recall will respect any **ngclient(1)** recall configuration options, except for recursive. This includes timeout; by default it will wait indefinitely for the recall to complete.

See **ngenea-client.conf(5)** for more information on the configuration format and additional options.

TRANSPARENT RECALL POLICY

Transparent recall, by definition, is intended to be triggered automatically when an offline file is opened for reading or writing.

To enable transparent recall functionality using **transparent_recall**, the following rules can be added to the filesystem placement policy.

```
/* BEGIN: NGENEA TRANSPARENT RECALL POLICY */

define(xattr_stbsz,[INTEGER(XATTR('dmapi.APXstbsz'))])

RULE FileOpen EVENT 'OPEN'
     ACTION(SetDataEvent(0, OP_READ, CASE WHEN xattr_stbsz IS NULL
THEN 0 ELSE xattr_stbsz END) AND
            SetDataEvent(1, OP_WRITE+OP_TRUNC, 0))
     WHERE LENGTH(XATTR('dmapi.APXguuid'))>0 AND
CountSubstr(MISC_ATTRIBUTES,'V')>0

RULE FileOpen_else EVENT 'OPEN' DIRECTORIES_PLUS

RULE FileData EVENT 'DATA'
     ACTION(system_lw('/usr/bin/python3 /usr/bin/transparent_recall
' || getDetail('path_name') )=0)
     WHERE LENGTH(XATTR('dmapi.APXguuid'))>0 AND
CountSubstr(MISC_ATTRIBUTES,'V')>0

RULE FileData_else EVENT 'DATA' DIRECTORIES_PLUS

/* END: NGENEA TRANSPARENT RECALL POLICY */
```

If transparent recall (EVENT) rules are already installed for ngenea (native), these rules should replace those equivalent rules.

Don't replace any rules besides the ngenea EVENT rules, e.g. don't replace any SET POOL rules.

See **mmchpolicy(1)** for how to change the filesystem placement policy

WARNING

Due to limitations of GPFS, the above policy will not work on any paths which contain whitespace. Transparent recall will simply error.

An alternative policy substitutes the system_lw line for

```
ACTION(system('/usr/bin/python3 /usr/bin/transparent_recall ''' ||
getDetail('path_name') || '''')=0)
```

This will handle paths containing whitespace, but not those containing single quotes. Again, transparent recall will error.

There is currently no approach which works in all cases. You must chose the version which is most compatible with the sorts of paths you have on your system.

TROUBLESHOOTING

When attempting to read an offline file, if the read process reports "Operation not permitted", and it's not due to permissions, the most likely cause is that the recall failed.

Logs for the **transparent_recall** command invocation can be found at /var/adm/ras/mmfs.log.latest

Logs for the transparent recall job can be viewed via Ngenea Hub.

WARNING - if reading a file triggers a recall, the read request will block until recall exits; it can't be interrupted (Ctrl+C) or killed (kill -9). If the recall job is 'stuck' and no timeout is set, the only way to make the read process exit is to kill the recall job via Ngenea Hub.

AVAILABILITY

Distributed as part of the ngenea-hub-client rpm, or the ngclient wheel (Python) for non-Red Hat based systems.

Note - **transparent_recall** makes use of flock(2), so can only be used on Unix-base operating systems.

SEE ALSO

ngclient(1), ngenea-client.conf(5), ngrecall(1), mmchpolicy(1)

LICENSE

Contact

Ngenea Hub is provided and supported by:

Pixit Media: https://pixitmedia.com/contact-us/

Arcastream: https://www.arcastream.com/contact/

```
Release notes - Ngenea Hub - 1.28.1: 2023-11-23
===============================================

Improvement
-----------

DYNAMOENG-1692: Increase Postgres working memory to reduce disk IO

Bug
---

DYNAMOENG-1691: Database transaction error during stuck task
invalidation


Release notes - Ngenea Hub - 1.28.0: 2023-11-17
===============================================

Improvement
-----------
DYNAMOENG-1610 - Workflow names and labels can now be much larger
DYNAMOENG-1650 - Added a flag to move_paths_on_gpfs to allow the
task to abort missing files
DYNAMOENG-1666 - Skipped jobs are now automatically cleared with
other inactive jobs

Bug
---
DYNAMOENG-1582 - Empty directory ACL syncing is now working
correctly in bidirectional sync
DYNAMOENG-1617 - Unexpected task results from transparent recalls
now correctly report in the hub
DYNAMOENG-1644 - ngclient now correctly updates the
discovery_options in workflow definition
DYNAMOENG-1679 - An error message when accessing files outside gpfs
on the browser in the worker has been removed to reduce noise
DYNAMOENG-1681 - Inactive tasks are now correctly invalidated after
the defined threshold


Release notes - Ngenea Hub - 1.27.1: 2023-08-24
===============================================

Bug
---
DYNAMOENG-1561 - Reverse stub error when syncing metadata for stub
files


Release notes - Ngenea Hub - 1.27.0: 2023-08-16
===============================================
Improvement
-----------
DYNAMOENG-1511 - An additional option for bypassing SSL validation
has been added to ngclient
DYNAMOENG-1491 - Snapdiff workflows now include the discovery_option
condense_moves
DYNAMOENG-1481 - Plugin tasks can now add to the total stats for a
job
DYNAMOENG-1500 - Running plugin installs now always installs the
latest version of the plugin code
DYNAMOENG-1521 - Job filters now time out after 30 seconds instead
of 10
DYNAMOENG-1535 - path_prefix in the job filters is now
input_path_prefix and is significantly faster
DYNAMOENG-1445 - Plugins can now be deleted on uninstall
DYNAMOENG-1495 - Task log messages when a job fails due to a system
link error have been improved
DYNAMOENG-1514 - Snapdiff task manager now waits 10 minutes by
default for a task to start instead of 1 minute and can be configured

Bug
---
DYNAMOENG-749 - File endpoint now correctly returns 404 if a path
does not exist
DYNAMOENG-1051 - Recursive delete workflows now only take one run to
remove all data
DYNAMOENG-1473 - Snapdiff task manager task now correctly waits for
the correct amount of events
DYNAMOENG-1479 - skip_old_ctimes in a snapdiff workflow now apply to
moved files
DYNAMOENG-1489 - PATCH operations on a workflow now correctly allow
you to change discovery_options
DYNAMOENG-1522 - The systemd service for ngenea-worker now correctly
restarts after all the threads exit completely
DYNAMOENG-1536 - The API no longer returns 500 when an empty set is
returned
DYNAMOENG-1531 - Offline files now have all of their ngenea metadata
updated during reverse_stub


Release notes - Ngenea Hub - 1.26.2: 2023-07-05
===============================================

Bug
---
DYNAMOENG-1509 - Fixed containers not being able to initiate new
threads preventing websocket connections


Release notes - Ngenea Hub - 1.26.1: 2023-06-30
===============================================

Bug
```

```
---
DYNAMOENG-1499 - Fixed issue with bidirectional_sync where it would
not run the second site on a task error


Release notes - Ngenea Hub - 1.26.0: 2023-06-22
===============================================

Feature
-------
DYNAMOENG-897 - The worker can now be passed a path to a different
configuration file via the CLI
DYNAMOENG-1440 - Users can now control all of the automated cleanup
tasks that run in the background of the hub
DYNAMOENG-1451 - A new field has been added to workflow definition
named "discovery_options" to control discovery behaviour
DYNAMOENG-1468 - Alpha plugin support can now be enabled within the
worker

Improvement
-----------
DYNAMOENG-857 - Snapdiff snapshot tracking files now includes the
fileset ID
DYNAMOENG-1352 - recursive_action now adds all found directories to
its total file count
DYNAMOENG-1474 - Ngenea based worker tasks now support "moved"
snapdiff rules
DYNAMOENG-1483 - Added a flag to abort a file when the file is
missing for the ctime check

Bug
---
DYNAMOENG-1224 - There is no longer a "could not read properties of
undefined" error after clicking search results
DYNAMOENG-1242 - Jobs now do not require to be cancelled twice in
snapdiff workflows
DYNAMOENG-1457 - Unexpected files are no longer created during a
bidirectional sync
DYNAMOENG-1469 - Files are correctly filtered out from a
bidirectional sync when there are old jobs without a started time
DYNAMOENG-1459 - Files created and renamed during a sync are not
longer absent on the target site
DYNAMOENG-1462 - Files with an old ctime are synced by default with
a toggle to enable filtering them
DYNAMOENG-1480 - Migrate tasks will no longer fail when a message
was already within the path details


Release notes - Ngenea Hub - 1.25.0: 2023-06-07
===============================================

Feature
-------
DYNAMOENG-203 - The Ngenea logging level can now be increased for
```

```
Ngenea based tasks
DYNAMOENG-467 - Job details can now be retrieved by clicking the pie
graph
DYNAMOENG-1395 - Recalls can now be called on files that do not
exist on the filesystem
DYNAMOENG-1452 - The hub ngenea tasks now support --sync-metadata
DYNAMOENG-1456 - All ngenea tasks now support --fail-on-mismatch
DYNAMOENG-1454 - Cloud SDK based tasks can now specify one ngenea
target to operate on

Improvement
-----------
DYNAMOENG-1055 - Skipped tasks now clearly state the paths skipped
when there is no work to be performed
DYNAMOENG-1376 - Check sync state now checks the xattrs, acls and
other metadata when comparing changes on directories
DYNAMOENG-1488 - Workflow field lists can now be edited within the
UI at runtime
DYNAMOENG-1042 - There are no longer un-needed warnings during
snapdiff operations

Bug
---
DYNAMOENG-1201 - Duplicate results are no longer created under
certain sync workflows
DYNAMOENG-1351 - Task resulting in the error "reversely stubbed
object does not exist" no longer occur with xattrs containing "max"
DYNAMOENG-1453 - Cloud SDK based worker tasks now work with all
discovery types
DYNAMOENG-1467 - Cloud SDK based tasks now correctly map matching
paths to all ngenea targets


Release notes - Ngenea Hub - 1.24.0: 2023-05-09
===============================================

Feature
-------
DYNAMOHUB-1341 - The Ngenea worker can now make use of multiple GPFS
nodes for the snapdiff policy and configurable threads
DYNAMOHUB-1372 - check_sync_state now has a flag to allow files to
result in the aborted state instead of skipped

Improvement
-----------
DYNAMOHUB-944 - Reverse stub now respects the sync_preference during
bi-directional syncs
DYNAMOHUB-1391 - The lambda function documentation has been updated
and improved
DYNAMOHUB-1400 - Task specific API keys are now more robust
DYNAMOHUB-1413 - ngenahubctl wipe now correctly removes volumes
DYNAMOHUB-1416 - The hydrate argument in tasks within
bi_directionalsync is now a run time field
```

```
Bug
---
DYNAMOHUB-1061 - The Ngenea worker no longer goes into a failed
state when its systemd service is stopped
DYNAMOHUB-1119 - remove_location_xattrs_for_moved now errors
correctly when paths do not match the regex provided for any Ngenea
policy are provided
DYNAMOHUB-1167 - Revoked is no longer a state that a task can enter
when cancelled
DYNAMOHUB-1176 - Hub containers no longer start on boot after the
service is disabled
DYNAMOHUB-1397 - The celery monitor container now correctly exits


Release notes - Ngenea Hub - 1.23.0: 2023-04-12
===============================================

Improvement
-----------
DYNAMOHUB-1347 - Files now have their change time checked before a
delete to ensure the file is not deleted after being edited

Bug
---
DYNAMOHUB-934 - Transparent recall jobs can now be correctly filtered
DYNAMOHUB-961 - Tasks can no longer have a negative run time
DYNAMOHUB-1056 - Certain characters are no longer parsed incorrectly
that resulted in incorrect paths in jobs
DYNAMOHUB-1023 - The snapdiff task now correctly parses backslash
escaped paths
DYNAMOHUB-1124 - Surrogate characters in paths are no longer causing
exceptions and are instead no longer supported
DYNAMOHUB-1168 - Character special files no longer cause the
websocket logic to error
DYNAMOHUB-1360 - UI schedule field now correctly supports offset mode
DYNAMOHUB-1363 - Internal hub processing directories are now ignored
by the recursive action discovery
DYNAMOHUB-1369 - Broken symlinks can now be deleted and moved in a
bidirectional sync
DYNAMOHUB-1384 - The UI no longer crashes when selecting a path for
a recursive schedule
DYNAMOHUB-1385 - Empty directories are no long sent back when both
sites in a bidirectional sync alter the same directory


Release notes - Ngenea Hub - 1.22.0: 2023-03-28
===============================================

Feature
-------
DYNAMOHUB-1342 - New single step move task:
dynamo.tasks.one_step_move_paths_on_gpfs

Improvement
```

```
-----------
DYNAMOHUB-1361 - Optimised the migration process for pre-1.21
releases to stop migrations taking multiple hours while drastically
lowering memory usage
DYNAMOHUB-1359 - Authentication errors on the task files endpoint
now have more verbose logging

Bug
---
DYNAMOHUB-1202 - Bidirectional sync tasks that run on a different
target site now report their site correctly
DYNAMOHUB-1309 - Non bidirectional snapdiff scheduled tasks no
longer attempt to run while one is on-going
DYNAMOHUB-1349 - Excluded or included files not longer cause
"Received 0 of the expected 1 results" in move or delete tasks
DYNAMOHUB-1371 - Fixed issue with file move and delete task keys in
non bidirectional sync runs

Release notes - Ngenea Hub - 1.21.0: 2023-03-09
===============================================

Feature
-------
DYNAMOHUB-527 - Symlinks are now supported by the recursive discovery

Improvement
-----------
DYNAMOHUB-1343 - Relative symlink moves are now supported

Bug
---
DYNAMOHUB-1336 - Directory symlinks are now correctly deleted
DYNAMOHUB-1186 - The deletion of a subscribed workflow no longer
causes it to run anyway
DYNAMOHUB-1356 - Deletes are no longer run on both sites during
certain conditions in a bidirectional_sync
DYNAMOHUB-1215 - Sites can now correctly be renamed


Release notes - Ngenea Hub - 1.20.0: 2023-02-24
===============================================

Feature
-------
DYNAMOHUB-746 - The processed files list can now be downloaded in
full in JSON

Bug
---
DYNAMOHUB-1320 - Tasks that log "critical: watchdog timeout expired"
now log the error the correct amount of times
DYNAMOHUB-1286 - Tasks that encounter files that fail with
"unexpected end of file" now correctly abort the file operation
DYNAMOHUB-1327 - Tasks stuck in pending due to their parent task not
```

```
having results now resolve themselves after 30 minutes


Release notes - Ngenea Hub - 1.19.0: 2023-02-23
===============================================

Features
--------
DYNAMOHUB-1267 - There are now more connection timeout settings for
robust redis broker and results connections

Improvements
------------
DYNAMOHUB-1314 - Each move and delete operation is now fully logged
within the worker
DYNAMOHUB-1316 - Celery workers have had an improvement to how they
handle latent connections
DYNAMOHUB-1160 - Synced file and directory deletes now do not replay
deletes in given circumstances
DYNAMOHUB-1312 - Snapshot operations are now retried if they fail
during all snapdiff tasks

Bug
---
DYNAMOHUB-1319 - Fixed "FileNotFoundError" from passing to further
tasks
DYNAMOHUB-1315 - Files that produce "no such file or directory" now
correctly abort during a migrate
DYNAMOHUB-1300 - Fixed GCS reverse_stub issue due to "Too many open
files"
DYNAMOHUB-1328 - Fixed issue caused by REDIS_HEALTH_CHECK_INTERVAL
in the hub

Release notes - Ngenea Hub - 1.18.0: 2023-02-15
===============================================

Features
--------

- DYNAMOHUB-617: Multiple jobs can now be restarted on the jobs page
- DYNAMOHUB-848: All redis communication now uses TLS
- DYNAMOHUB-932: Clicking outside of a model now closes that modal
- DYNAMOHUB-964: There is now an automatic cleanup of old jobs
- DYNAMOHUB-1161: Multiple state filters can now be chosen on the
job details page
- DYNAMOHUB-1173: dynamo.tasks.recall now supports the delete_remote
argument
- DYNAMOHUB-1197: Users can now use the API to retrieve the files
that have been processed without the full payload
- DYNAMOHUB-1205: The hub now uses redis for its broker instead of
rabbitmq
- DYNAMOHUB-1213: Sync now supports directories for all its
operations
- DYNAMOHUB-1241: Grafana now reports on the redis metrics
```

```
Improvements
------------

- DYNAMOHUB-643: Job statistics now take cancelled tasks in their
totals
- DYNAMOHUB-1153: The docker shared memory now has a different
default for the database
- DYNAMOHUB-1191: Moved file/folder conflict resolution has been
made more robust
- DYNAMOHUB-1195: Syncs will now perform actions delete, move and
then other operations in that order
- DYNAMOHUB-1245: Celery workers can now toggle the heartbeat,
mingle and gossip behaviours
- DYNAMOHUB-1255: The gossip, mingle and heartbeat are now enabled
by default instead of disabled
- DYNAMOHUB-1258: redis_backend_health_check_interval can now be
configured for the hub's redis service
- DYNAMOHUB-1260: Ngeneahub now makes use of ngenea 1.20
- DYNAMOHUB-1291: REDIS_TCP_BACKLOG can now be configured for the
hub's redis service
- DYNAMOHUB-1305: Optimise the storing of task results for faster
sync
- DYNAMOHUB-1306: Removed redundant 'Number of directories' from job
details page
- DYNAMOHUB-1315: Mark "no such file or directory" as 'aborted' in
migrate and reverse stub tasks

Bug
---

- DYNAMOHUB-707: Snapdiff workflows without any additional work no
longer show as having a pending file
- DYNAMOHUB-1099: The jobs page now loads signifcantly faster when
there are a large volume of jobs
- DYNAMOHUB-1142: Migration tasks no longer fail due to "no such
file or directory"
- DYNAMOHUB-1148: Syncs not correctly filter out previously
completed directory moves
- DYNAMOHUB-1151: Directory moves are now performed in order
- DYNAMOHUB-1175: The jobs details page no longer attempts to load
the failed files list unprompted
- DYNAMOHUB-1216: Syncs can now be cancelled and it will roll back
the snapshot correctly
- DYNAMOHUB-1220: Create and complete time filters now work as
expected on the job list
- DYNAMOHUB-1225: Syncs now correctly rotate snapshots when all
tasks are completely successful
- DYNAMOHUB-1227: Schedule deletion no longer times out
- DYNAMOHUB-1230: Recall resulting in "deletion of remote objects
failed" no longer causes job failures
- DYNAMOHUB-1248: Complex sync operations no longer cause errors in
dynamo.tasks.filter_snapshot_results
- DYNAMOHUB-1249: Fixed race condition of task completing too
```

```
quickly to report the results from redis
- DYNAMOHUB-1259: Syncs now correctly tracks the amount of input
files

Release notes - Ngenea Hub - 1.17.3: 2023-02-09
===============================================

Improvement
-----------

- Ability to configure RabbitMQ max_message_size (DYNAMOHUB-1221)

Bug
---

- Migration failure due to "no such file or directory" from ngenea
(DYNAMOHUB-1285)
- Reverse stub failure due to TypeError (DYNAMOHUB-1294)
- Migration failure due to "no such file or directory" when checking
ctime (DYNAMOHUB-1302)

Release notes - Ngenea Hub - 1.17.0: 2022-12-07
===============================================

Improvement
-----------

- Shareable URLs that navigate to a specific position in the
filebrowser tree (DYNAMOHUB-650)
- Search filters that require a timestamp input now present a date-
picker UI (DYNAMOHUB-652)
- Improve search filters interface to display only relevant
operators (DYNAMOHUB-653)
- New configuration UI for global settings (DYNAMOHUB-696)
- Support for placing the web interface behind a reverse proxy \
(e.g. SSL termination\) and customising the base URL (DYNAMOHUB-813)
- Remember filter setting for "Hide successful jobs with no
processed files" between pages with job lists (DYNAMOHUB-906)
- Subscribed workflows in the schedule UI now display in
alphabetical order (DYNAMOHUB-909)
- Allow the resubmit button to be used on cancelled jobs
(DYNAMOHUB-938)
- Improve websocket connection robustness by reusing the same
connection between pages (DYNAMOHUB-982)
- Improve advice on transparent recalls regarding space and quote
limitiations (DYNAMOHUB-998)
- Improve messaging in the file browser view when only hidden files
are in a directory. (DYNAMOHUB-1000)
- UI Interface for linking sites to datastores (DYNAMOHUB-1008)
- Support symlinks in default workflows and tasks (DYNAMOHUB-1018)
- Prevent administrative task \(cleanup\_old\_events\) from running
multiple instances at the same time and improve performance
(DYNAMOHUB-1038)
- Add support to synchronise empty directories  via site-sync
```

```
(DYNAMOHUB-1090)
- Refreshing the job details page will keep any currently applied
filters (DYNAMOHUB-1096)
- Provide support for ngenea --sync-metadata option for syncing
metadata without re-transmitting data (DYNAMOHUB-1101)
- Improve job filtering by setting sensible defaults for job states
(DYNAMOHUB-1104)
- Improve memory usage when refreshing large jobs (DYNAMOHUB-1130)
- Improved logging when encountering large celery payloads: "message
size XXX is larger than configured max size YYY" (DYNAMOHUB-1132)

Bug
---

- "Cannot read properties of null" errors when opening jobs
(DYNAMOHUB-700)
- Console constantly outputting web socket connection errors
(DYNAMOHUB-701)
- Fixed an issue where ngenea workers would unnecessarily set
bandwidth controls when no changes have been made. (DYNAMOHUB-855)
- Fixes an issue where duplicate items appear in the file browser
when they are created on the file system (DYNAMOHUB-1002)
- Fixes an issue where some unicode code points in a path could
cause the snapdiff discovery task to fail (DYNAMOHUB-1028)
- Could not filter tasks in a job with a large number of tasks
(DYNAMOHUB-1040)
- Gracefully handle errors if a file is deleted during a migration
(DYNAMOHUB-1059)
- Clean-up stale internal snapdiff related data \(ObjectEvents\)
from failed snapdiff jobs (DYNAMOHUB-1094)
- Discrepancy between hwdev and hwdev03 if a mv is run one one site
while a rm is run on the other (DYNAMOHUB-1100)
- Fixes an issue in the Job summary filters where the job type
filter dropdown could render incorrectly (DYNAMOHUB-1108)
- Fixes an issue where passing a symlink to a migrate step would
cause an error (DYNAMOHUB-1110)
- Fixes an issue where filtering search results on sites would reset
unexpectedly  (DYNAMOHUB-1112)
- Fixes an issue in remove\_location\_xattrs\_for\_moved workflow
step,  which causes an error if the file is deleted during
processing (DYNAMOHUB-1117)
- Fixes an issue where tasks in a large job could be marked as
failed due to a race condition with the lost message detection logic
(DYNAMOHUB-1120)
- Fixes an issue in the migrate step where an erroneous message
"duplicate file skipped" could be logged as a file error is the file
changed multiple times in quick succession   (DYNAMOHUB-1131)
- Upgrade of worker threw error likely in trying to modify worker
config (DYNAMOHUB-1149)
- Fixes an issue that prevented users re-logging in after logging
out in the same browser session (DYNAMOHUB-1159)
- Fixes an issue where files that should be excluded by the
check\_sync\_state step were still being processed by subsequent
steps in a workflow  (DYNAMOHUB-1169)
```

```
Release notes - Ngenea Hub - 1.16.0: 2022-11-08
===============================================

Improvement
-----------

- Display summarized item counts when deleteing, enabling and
disabling. (DYNAMOHUB-1003)
- Remove site health details from the site details page.
(DYNAMOHUB-1011)
- First and last page buttons are added to table pagination.
(DYNAMOHUB-619)
- Job filter displays site name instead of site id in the filter.
(DYNAMOHUB-654)
- Job filter owner list is calculated faster on the backend.
(DYNAMOHUB-713)
- Fixed the issue with changing url when redirects to 404 not-found
page (DYNAMOHUB-805)
- Make job stats endpoint return statistics about last 7 days
(DYNAMOHUB-950)
- Gray out resubmit button for schedules jobs. (DYNAMOHUB-983)
- Improve site health details (DYNAMOHUB-985)
- Fixed font so that 0 and 'o' , 1 and l won't be mixed up.
(DYNAMOHUB-994)


Bugfixes
--------

- Fixed displaying gray screen when there is no site health data.
(DYNAMOHUB-1006)
- Error when migrating tasks from before version 1.15 on demand
(DYNAMOHUB-1113)
- A nicer API response is sent to the users when filesets can't be
retrieved due to celery timeout error. (DYNAMOHUB-929)
- Fixed the issue related to updating the file browser when the
filename contains emojis. (DYNAMOHUB-988)


Features
--------

- Confirm workflow modal summarizes items if there are too many
items. (DYNAMOHUB-826)
- Support cold failover on deployments where ngenea hub is deployed
on a pixstor cluster. (DYNAMOHUB-966)


Documentation
-------------

- Improved example policy in the transparent recall docs, which
resolves issues with deferred deletes (DYNAMOHUB-997)
```

```
- Updated advice on transparent recall policy

  Due to limitations of GPFS, users must choose between a policy
which supports paths with whitespace or paths with single quotes.
(DYNAMOHUB-998)


Release notes - Ngenea Hub - 1.15.1: 2022-09-29
===============================================

Bugfixes
--------

- Fixed a bug where Jobs created before 1.15.0 could not be viewed
or cancelled. (DYNAMOHUB-1113)
- Fixed a bug where a bi-directional site-sync could fail if
directory moves are present (DYNAMOHUB-1116)


Release notes - Ngenea Hub - 1.15.0: 2022-09-27
===============================================

Improvement
-----------

- Improve DB locking to make the UI more responsive while large jobs
are refreshing (DYNAMOHUB-1024)
- Optimise event comparison for Bidirectional site sync
(DYNAMOHUB-1035)
- Remove unneeded list of spawned tasks from result payload of
filter_snapshot_results (DYNAMOHUB-1037)
- Add a runtime field to jobs that use snapdiff:
"snapdiff_rotate_on_error". If set to True, this will rotate a
snapdiff forwards even if there are failed files. (DYNAMOHUB-1045)
- Subscribed workflows are editable on the UI. (DYNAMOHUB-1049)
- Don't overwrite remote objects for default sync workflows.
(DYNAMOHUB-1063)
- Change snapdiff based workflows, such as site sync, to always
rotate snapshots even when errors occur. (DYNAMOHUB-1077)
- Job files API endpoint is paginated. (DYNAMOHUB-916)
- Job files modal on the job details page uses paginated file
endpoint now. (DYNAMOHUB-917)
- Apply cron expression to add and update schedule ui.
(DYNAMOHUB-941)
- Ensure that the file list for snapdiff based jobs matches the
reported number of files (DYNAMOHUB-976)
- Users can not delete running jobs by using the API (DYNAMOHUB-977)
- Prevent users from deleting pending and started jobs.
(DYNAMOHUB-978)
- Display dash(-) for empty node fields. (DYNAMOHUB-986)


Bugfixes
--------
```

```
- Ensure snapshots are rolled-back when a job using the snapdiff
discovery is cancelled (DYNAMOHUB-1027)
- Don't cleanup events for active, unscheduled syncs (DYNAMOHUB-1029)
- Ensure that cancelled bidirectional site sync jobs don't have
tasks stuck as PENDING (DYNAMOHUB-1039)
- Job cancel hangs when the job has a lot of tasks (DYNAMOHUB-1047)
- Dispaly jobtypes in alphabetical order (DYNAMOHUB-933)
- Stat paths task results recorded as an error (DYNAMOHUB-942)
- Ensure that files which are skipped due to checking sync state are
reported as skipped in the UI Job details page (DYNAMOHUB-943)
- No discovery selection for schedules is fixed. (DYNAMOHUB-967)
- Make show_noop parameter work only for successful jobs & make the
default setting to True (DYNAMOHUB-999)


Features
--------

- Adds 'aborted' state which could not be processed but may be
retried later. (DYNAMOHUB-1067)
- Allow user to delete jobs (DYNAMOHUB-618)
- New default workflow for deleting individual files (DYNAMOHUB-893)
- Allow users to enable and disable multiple schedules from the
administrative ui. (DYNAMOHUB-940)

Release notes - Ngenea Hub - 1.14.2: 2022-08-12
===============================================

Bugfixes
--------

- Optimised snapdiff event processing for initial sync runs
(DYNAMOHUB-1014)

Release notes - Ngenea Hub - 1.14.1: 2022-07-27
===============================================

Bugfixes
--------

- Ensure snapdiff soft-lock is removed in the event of an error
(DYNAMOHUB-975)

Release notes - Ngenea Hub - 1.14.0: 2022-07-15
===============================================

Highlights
----------

- It's now possible to cancel running or stuck jobs via the Jobs
Details page.
- Introducing a new and improved workflow: bi-directional site-sync.
A schedule-only workflow that allows 2 sites to stay in sync.
```

```
- New heath status page - showing the status of ngenea hub and it's
workers.
- It's now possible to have site-wide include and exclude lists.

Improvement
-----------

- Autocomplete selection for task types is added to filter tasks
modal. (DYNAMOHUB-522)
- Search filters are displayed as an autocomplete input box
(DYNAMOHUB-546)
- Site selection for search UI is enabled. (DYNAMOHUB-859)
- Sidebar toggler icon is animated (DYNAMOHUB-874)
- Jobs can be filtered by completion time. (DYNAMOHUB-887)
- Path picker component is added for schedule management.
(DYNAMOHUB-889)
- Limit bidirectional_snapdiff discovery schedules to one
bidirectional_sync subscribed workflow (DYNAMOHUB-894)
- Tasks are sorted by their start time by default. (DYNAMOHUB-895)
- bidirectional_snapdiff discovery schedules can be created in the
UI (DYNAMOHUB-899)
- Only allow one bi-directional sync run at a time in a schedule
(DYNAMOHUB-901)
- Jobs can be cancelled on the UI (DYNAMOHUB-911)
- New tasks are introduced for retrieving search metadata fields
(DYNAMOHUB-546)
- Support for different fileset and pool name format in Analytics
2.X (DYNAMOHUB-797)
- Move snapdiff lock file location to the .rotate directory
(DYNAMOHUB-931)


Bugfixes
--------

- Fixed users deactivating themselves. (DYNAMOHUB-578)
- Halting the search issue is fixed when the search filters are
changed (DYNAMOHUB-647)
- Fixed jobs filter not syncing with recent jobs issue.
(DYNAMOHUB-821)
- Possible web socket connection flaws are resolved. (DYNAMOHUB-885)
- Fix intermittent transparent recall hangs (DYNAMOHUB-902)
- RPM creates a blank config file on new install (DYNAMOHUB-947)
- Skip moves which have already been applied (DYNAMOHUB-926)


Features
--------

- Added user configurable port to access hub (DYNAMOHUB-198)
- Support for site-global includes and excludes (DYNAMOHUB-613)
- Removed HubSchedule model from DB (DYNAMOHUB-769)
- Added validation to prevent call to workflow/schedule when no work
is done (DYNAMOHUB-798)
```

```
- Added endpoint for both site specific and hub wide health status
reporting (DYNAMOHUB-838)
- Added Site Health Feature (DYNAMOHUB-840)
- Display cron schedule in human readable format (DYNAMOHUB-868)
- Snapdiff task results are now tracked within the hub database
(DYNAMOHUB-870)
- Background task to clean up old sync events (DYNAMOHUB-871)
- Workflows can now be set for use only within a schedule
(DYNAMOHUB-873)
- New workflow added for two way syncing between sites within
schedules (DYNAMOHUB-877)


Documentation
-------------

- Documentation on how to set up site sync (DYNAMOHUB-574)
- Search set-up documentation (DYNAMOHUB-781)
- Document using ngclient to interact with feature flags
(DYNAMOHUB-843)
- Fixed worker service name in upgrade guide (DYNAMOHUB-882)
- Documentation on how to set up bidirectional site sync
(DYNAMOHUB-908)
- Troubleshooting guide (DYNAMOHUB-948)


Release notes - Ngenea Hub - 1.13.0: 2022-06-13
===============================================


Bugfixes
--------

- Timeout is handled on login. (DYNAMOHUB-348)
- If the token is expired, redirect to original target page after
login. (DYNAMOHUB-611)
- Removed non-deterministic behaviour while displaying the search
bar. (DYNAMOHUB-646)
- Long paths are wrapped to a new line in job details page.
(DYNAMOHUB-659)
- Missing ngclient sub-command man pages (DYNAMOHUB-692)
- Fixed Api-Key issues in Resubmit API (DYNAMOHUB-703)
- Added new celery beat that runs every one hour to update the state
from STARTED to FAILURE for inactive tasks (DYNAMOHUB-704)
- Single search web socket connection is managed against working
intermittently. (DYNAMOHUB-729)
- Show jobs with no processed files filter is configurable for all
job tables. (DYNAMOHUB-744)
- Ensures that a snapshot rotation task is created when a job is
resolved regardless of the task chain (DYNAMOHUB-747)
- Ensured that a clearer message is provided when a GPFS error
occurs on worker for a stat (DYNAMOHUB-761)
- Recent job counts are corrected for both API and web socket
consumer. (DYNAMOHUB-783)
```

```
- Fixed schedule job resulting in pending state for snapdiff task
(DYNAMOHUB-790)
- Fixed the issue which chases when sumitting bandwidth form with no
changes. (DYNAMOHUB-800)
- Bug is fixed when managed_paths is null for a schedule.
(DYNAMOHUB-817)
- Removed extra file batch field (DYNAMOHUB-852)
- Sort actions in alphabetical order (DYNAMOHUB-752)
- Fixed issue on UUID mismatch using ngrecall overwrite-recall by
comparing local and remote UUID (DYNAMOHUB-464)
- Added interface type dummy to virtual drivers list to exclude it
from physical interface (DYNAMOHUB-757)
- Correctly caught GPFS errors to prevent task from returning stack
trace. (DYNAMOHUB-761)
- Directory moves in move_paths_on_gpfs now move all files along
with the directory themselves (DYNAMOHUB-782)
- Removed the strict format for snapshots within the snapdiff
tracking file (DYANMOHUB-699)


Features
--------

- Feature added for limiting fields in metadata returned
(DYNAMOHUB-376)
- feature added for search REST API to provide more_results key if
elasticsearch backend has more results than max_results
(DYNAMOHUB-550)
- Added support for overriding includes and excludes at runtime
(DYNAMOHUB-640)
- Added reverse_stub in default workflows (DYNAMOHUB-681)
- Added recursive flag for move_paths_on_gpfs task in docs
workflow_steps.md (DYNAMOHUB-685)
- Feature added for chunk_size to chunk paths in snapdiff tasks
(DYNAMOHUB-711)
- Feature added for Id based endpoint for IpAddress Apis GET, DELETE
and PATCH (DYNAMOHUB-712)
- Added support for pixstor_search backend in hub (DYNAMOHUB-738)
- Add Datastore managemant and IP address management feature.
(DYNAMOHUB-774)
- Updated sync_preference type to choices for site_sync workflow
(DYNAMOHUB-787)
- NgeneaHub UI supports having new themes now. (DYNAMOHUB-842)
- Add list of available metadata fields to results of Hub searches
performed against Pixstor Search (DYNAMOHUB-734)
- Add mechanism to return specified metadata fields only, in Hub
searches against Pixstor Search (DYNAMOHUB-736)

Documentation
-------------

- Upgrade guide documentation (DYNAMOHUB-687)
- Add site-specific configurations to the docs (DYNAMOHUB-739)
- Documentation for default workflows (DYNAMOHUB-741)
- Document how to use search metadata field limiting (DYNAMOHUB-756)


Removals
--------

- Removed the site field for IP address objects as they were once
depreciated (DYNAMOHUB-657)


Improvement
-----------

- Job runtime field is introduced and becomes sortable on the UI.
(DYNAMOHUB-714)
- feature improvement done in stats to add extended attributes as a
dict (DYNAMOHUB-201)
- feature improvement added for allowing directory deletes with
kwargs (DYNAMOHUB-345)
- Add support for `--endpoint` flag on recall and reverse stub tasks
(DYNAMOHUB-361)
- Workaround for stale file handles when performing reverse stub
(DYNAMOHUB-439)
- Add support for worker auto-scaling (DYNAMOHUB-461)
- Record exact ngenea command in task results (DYNAMOHUB-462)
- Adjusted recursive task to allow for multiple operations based of
generic rules (DYNAMOHUB-476)
- feature improvement added for failure scenario for
remove_location_xattrs_moved (DYNAMOHUB-509)
- feature improvement added for snapdiff tasks to allow only single
path else raise an exception with error message (DYNAMOHUB-515)
- Add Ngenea server and client as rpm package dependencies for
installing the ngenea worker, and require at least version 1.15.
(DYNAMOHUB-694)
- Ensured that ngenea binary stderr capture is not blocked by
reporting the status of a job (DYNAMOHUB-765)


Release notes - Ngenea Hub - 1.12.0: 2022-05-20
===============================================

Improvement
-----------

- Hash is used instead of content hash for cache busting.
(DYNAMOHUB-743)
- Fixed not displaying error or failure state jobs with no
operations (DYNAMOHUB-745)
- File browser performance improvement for file status updates.
(DYNAMOHUB-766)
- Log PID instead of process name within the celery logs.
(DYNAMOHUB-784)
- List actions in alphabetical order. (DYNAMOHUB-752)
- Allow deleting of /ipaddresses/{ipaddr}/ if an ip address is
```

```
associated with multiple datastores. (DYNAMOHUB-712)
- Add new reverse stub workflow as part of deployment.
(DYNAMOHUB-681)
- Provide mechanism to limit fields of search metadata returned.
(DYNAMOHUB-376)

Features
--------

- Feature added for limiting metadata fields in search_analytics
(DYNAMOHUB-376)


Bugfixes
--------

- Fixed migration bug with mis-matching schedule IDs during
migration. (DYNAMOHUB-824)
- UI now handles managed_paths being null within a schedule.
(DYNAMOHUB-817)
- Ensured rotate task do not remain stuck in a PENDING state.
(DYNAMOHUB-747)
- Users can now show no-op jobs across all job tables.
(DYNAMOHUB-744)
- Search no longer works intermittently when previous search is
stopped. (DYNAMOHUB-729)
- Searching by gpfs.filesetname now works correctly. (DYNAMOHUB-727)
- When task gets killed, it is no longer marked as "STARTED" in
celery. (DYNAMOHUB-704)
- Jobs can now be resubmited using an API Key. (DYNAMOHUB-703)
- Search bar now always shows when you first log into UI.
(DYNAMOHUB-646)
- Fixed migrate task issue when migrating same filepath to multi
targets (DYNAMOHUB-726)


Release notes - Ngenea Hub - 1.11.0: 2022-05-09
===============================================

Improvement
-----------

- Descriptive messages for all tables (DYNAMOHUB-541)
- Default search filter is changed to core.pathname (DYNAMOHUB-545)
- Workflow details can be inspected on job details page.
(DYNAMOHUB-648)
- Table row highlighting is improved visually. (DYNAMOHUB-656)
- Make number of files diplay dialog nicer (DYNAMOHUB-661)
- Elasticsearch url management for sites (DYNAMOHUB-693)
- Added a message when there are more searched results to be
displayed (DYNAMOHUB-697)
- Task details preview is disabled when task details JSON is too
large. (DYNAMOHUB-719)
- file_size_gb field of sites became managable from UI.
```

```
(DYNAMOHUB-740)
- Update UI for new discovery schedules, to which multiple workflows
can be subscribed. (DYNAMOHUB-772)
- Feature added in search_analytics to hint users if there are
more_results than max_results in elasticsearch backend
(DYNAMOHUB-550)
- Added flag to recall and reverse_stub tasks for gid and uid
(DYNAMOHUB-614)
- Added elasticsearch credentials for analytics search backend
(DYNAMOHUB-689)
- Support for AP-Analytics 2.X search backend (DYNAMOHUB-760)

Bugfixes
--------

- Fix upgrade generating spurious systemd services (DYNAMOHUB-359)
- Added support for CredentialsJSON in GCS ngenea configs
(DYNAMOHUB-655)
- Adjusted the start date of all tasks to return UTC times for
consistency (DYNAMOHUB-663)


Release notes - Ngenea Hub - 1.10.0: 2022-04-01
===============================================

Improvement
-----------

- Added user friendly log viewer (DYNAMOHUB-436)
- feature improvement added for filter_results to return clear
results for the user (DYNAMOHUB-491)
- Workflow configuration is visible in job details and job list
pages (DYNAMOHUB-626)
- Table optional fields are managed by switches in Show/Hide column
dialog. (DYNAMOHUB-631)
- Converted runtime value into human readable value (DYNAMOHUB-644)
- Resolved UI Typescript compile issues & unit test issues
(DYNAMOHUB-665)
- Tab view for Administration, SiteDetails, ScheduleDetails pages
(DYNAMOHUB-666)
- Refactoring on the file browser (DYNAMOHUB-667)
- Refactoring forms on the UI (DYNAMOHUB-668)
- Refactoring tables & sticky table controls (DYNAMOHUB-669)
- UI test coverage is increased. (DYNAMOHUB-670)
- Refactoring on websocket serialization (DYNAMOHUB-680)
- Bandwidth controls are hidden behind a feature flag.
(DYNAMOHUB-750)



Features
--------

- Support for setting search-related configurations via the REST API
```

```
(DYNAMOHUB-474)
- feature added for supporting default filter rules for snapdiff and
recursive discovery tasks (DYNAMOHUB-533)
- API endpoint for setting speed for site (DYNAMOHUB-568)
- Support for setting the IP address for a DataStore (DYNAMOHUB-569)
- Feature added for auto update mechanism of cloud hosted storage
target ips for s3,azure,gcs (DYNAMOHUB-570)
- UI for setting bandwidth controls (DYNAMOHUB-571)
- Updated docs with endpoint flag for migrate tasks (DYNAMOHUB-580)
- Snapdiff discovery tasks now only rotate their snapshot on no
errors in other tasks. (DYNAMOHUB-630)
- Ability to scan for worker nodes via the REST API (DYNAMOHUB-638)
- Feature added to allow directory move in move_paths_on_gpfs
(DYNAMOHUB-542)
- Feature improvement added endpoint flag for migrate task
(DYNAMOHUB-580)
- Adjusted the parsing method to new json output for ngenea 1.15
(DYNAMOHUB-606)
- Added worker task for rotating snapshots outside of the snapdiff
task (DYNAMOHUB-630)


Bugfixes
--------

- Display workflow name as the tooltip text (DYNAMOHUB-554)
- fixed issue job fails while other tasks are running by refreshing
the job when there are running/pending remaining tasks
(DYNAMOHUB-577)
- Fixed clicking to search result path for hidden items
(DYNAMOHUB-683)
- Fix inconsistencies between PixStor Search and Analyitcs search
backends (DYNAMOHUB-556)
- Fixed ngenea error reporting by adding proper status logs
(DYNAMOHUB-624)
- fixed issue for moving files between different filesets
(DYNAMOHUB542)


Deprecations and Removals
-------------------------

- Setting IP address for a site is deprecated as of release 1.9 and
will be removed in 1.10 (DYNAMOHUB-569)


Release notes - Ngenea Hub - 1.9.0: 2022-02-23
==============================================

Features
--------

- Model for object stores (S3, GCS, etc.) (DYNAMOHUB-207)
- SiteLink model for connecting sites to datastores (DYNAMOHUB-224)
- Worker node model and monitoring (DYNAMOHUB-563)
- Feature added for configuring file batch size in site model
(DYNAMOHUB-572)
- Job started time, completed time, runtime fields are accessible on
the UI. (DYNAMOHUB-573)
- Modifications done for file count for all tasks to report job
statistics (DYNAMOHUB-587)
- Added endpoint to job API to cancel all on-going tasks
(DYNAMOHUB-71)
- changes added for chunk size in recursive action (DYNAMOHUB-472)

Improvement
-----------

- Allow user to search job id from the job table (DYNAMOHUB-411)
- Complex queries can be made with the search bar (DYNAMOHUB-547)
- changed tooltip to display workflow name (DYNAMOHUB-554)
- Update frontend depedancy: follow-redirects to address
CVE-2022-0536 (DYNAMOHUB-637)


Bugfixes
--------

- Job filter preferences are remembered on the jobs page
(DYNAMOHUB-427)
- Display correct page size after applying the filters.
(DYNAMOHUB-531)
- Jobs page table glitch is fixed. (DYNAMOHUB-632)


Release notes - Ngenea Hub - 1.8.0: 2022-02-08
==============================================

Improvement
-----------

- Usability improvements on task detail modal (DYNAMOHUB-420)
- Automatically create hub environment auth file if it does not
exist (DYNAMOHUB-576)


Features
--------

- Added UI element for "choices" field type (DYNAMOGUB-551)
- Added ngenea locking mode support for default workflow operations
(DYNAMOHUB-537)
- feature added for migrating empty directories (DYNAMOHUB-490)
- Added flag to recall, reverse_stub and migrate tasks for
controlling ngenea locking levels (DYNAMOHUB-537)
- changes added for chunk size in recursive action (DYNAMOHUB-472)
```

```
Bugfixes
--------

- fixed misleading error messages on login page (DYNAMOHUB-348)
- changed return code of /api/file/workflow endpoint from 200 to 201
(DYNAMOHUB-488)
- Informative messages when the worker is shutdown (DYNAMOHUB-514)
- Usability improvements on tables and schedule pages (DYNAMOHUB-517)
- Ensure debug mode isn't enabled in production (DYNAMOHUB-518)
- fixed the warning issue raised for timezone in task started
(DYNAMOHUB-519)
- Removed default option as recursive set for discovery for /api/
file/workflow (DYNAMOHUB-534)
- Added jobid in signatures for recursive discovery task
(DYNAMOHUB-543)
- Ensured that the detault value of a field is respected in the UI
(DYNAMOHUB-558)
- Ensure secure file permissions on hub auth configuration file
(DYNAMOHUB-561)
- Ensured that the workflow API route has full validation on JSON
fields (DYNAMOHUB-562)
- More detailed messages for file browser when the worker is not
available (DYNAMOHUB-566)
- No results returned for date range search (DYNAMOHUB-489)
- Number of search results differs for path with trailing slash
(DYNAMOHUB-500)
- added try/except block for get_xattrs for gpfs.FileHeat key error
(DYNAMOHUB-525)
- Fixed an issue where if a file was skipped by a step in a
workflow, they were not being processed by a migrate or recall step
later in the workflow. (DYNAMOHUB-540)


Release notes - Ngenea Hub - 1.7.0: 2022-01-24
==============================================

Improvement
-----------

- feature improvement added choices as field type (DYNAMOHUB-154)
- Record and report errors when performing searches (DYNAMOHUB-365)
- Search UI has now filtering option. Before submitting a search,
users can add more filters to narrow down the search. (DYNAMOHUB-374)
- Time range selection for job filtering (DYNAMOHUB-403)
- Multiselect functionality on the job state field for job filtering
is added. (DYNAMOHUB-449)
- Owner selection while filtering jobs (DYNAMOHUB-457)
- All alerts are manually closed now. (DYNAMOHUB-458)
- Job state and job type selection is swapped visually on the job
filtering dialog (DYNAMOHUB-468)
- Replaced "In progress" label with "Started" for jobs
(DYNAMOHUB-470)
- Performance improvement on job table and auto refresh toggle for
web socket live updates (DYNAMOHUB-480)
```

```
- feature improvement added for snapdiff tasks numfiles
(DYNAMOHUB-510)

Features
--------

- feature improvement added for runtime fields to have default
values for choices type (DYNAMOHUB-153)
- feature added for choices runtime field to support list of objects
(DYNAMOHUB-154)
- Added support for providing file states for non-discovery
workflows (DYNAMOHUB-344)
- Allowed the use of multiple generic rules within the recursive
discovery task (DYNAMOHUB-476)
- API endpoint for feature flags (DYNAMOHUB-482)
- Search UI is a configurable feature now. (DYNAMOHUB-483)
- ngclient sub-commands for listing and setting feature flags
(DYNAMOHUB-484)
- Feature API is used for enabling and disabling the usage of UI
elements. (DYNAMOHUB-485)
- Tasks for moving files and directories within cloud storage
(DYNAMOHUB-355)


Bugfixes
--------

- Number of search results per site is 200 results by default
(DYNAMOHUB-374)
- fixed issue by adding fields in job model instance for resubmit
tasks (DYNAMOHUB-465)
- Performance improvements on the Jobs page (DYNAMOHUB-466)
- A timeout is added for failed search operations. If no results are
found in 100 seconds, the error will show up and the search will
halt. (DYNAMOHUB-498)
- Search results can be sorted by metadata fields. (DYNAMOHUB-557)
- added exception for permission errors (DYNAMOHUB-345)
- Policy error in snapdiff (DYNAMOHUB-450)
- fixed issue raised when submitting a workflow with multiple
filters in recursive task (DYNAMOHUB-476)


Release notes - Ngenea Hub - 1.6.0: 2021-12-21
==============================================

Improvement
-----------

- return extended attributes in file API (DYNAMOHUB-201)
- Add support for `--endpoint` flag on recall and reverse stub tasks
(DYNAMOHUB-361)
- Support for workflows without a discovery task, through the API
(DYNAMOHUB-402)
- limit the number of hub celery worker threads to 2 (DYNAMOHUB-405)
- Include error type when reporting task errors (DYNAMOHUB-418)
- Apply dynamo.tasks.remove_location_xattrs_for_moved to send and
snapdiff workflows to better handle moving files (DYNAMOHUB-422)
- Improve handling of file system xattr "stale nfs handle" message
scenarios (DYNAMOHUB-439)
- Disable validation of non-included fields when patching workflows
via api (DYNAMOHUB-456)

Features
--------

- Set graceful timeout on stat requests, default to 10 seconds.
(DYNAMOHUB-338)
- JWTs are created by using RS256 algorithm (DYNAMOHUB-353)
- Use search result to jump to directory in file browser
(DYNAMOHUB-380)
- Expose JWK set used in token verification via API endpoint
(DYNAMOHUB-430)
- Use Hub authentication for Grafana access (DYNAMOHUB-431)
- Initial framework and metrics for Grafana dashboads (DYNAMOHUB-432)
- ngclient command to import and export custom workflows
(DYNAMOHUB-434)
- NgeneaHub UI management of scheduled workflows (DYNAMOHUB-435)
- Added task for checking the existence of files in remote storage
(DYNAMOHUB-421)
- Task to remove remote location xattrs if a premigrated file was
moved (DYNAMOHUB-422)


Bugfixes
--------

- Migration path not correct when upgrading from 1.3.0
(DYNAMOHUB-397)
- Fix incorrect PDF Documention download link (DYNAMOHUB-440)
- Fixed issue with task start time formatting (DYNAMOHUB-441)
- Custom workflow steps can get wiped on upgrade (DYNAMOHUB-442)
- Fix incorrect download link for ngenea client (DYNAMOHUB-444)
- Mark tasks as completed when skipped (DYNAMOHUB-477)
- Fix date string parsing error in update_task celery task
(DYNAMOHUB-508)
- Don't require ArcaPix policy driver to be executable
(DYNAMOHUB-419)
- Check for no paths passed to recall and reverse_stub tasks
(DYNAMOHUB-438)
- fixed the status of snapdiff tasks processed files to created
instead of created and deleted (DYNAMOHUB-492)
- Fix issue with parsing ngenea config file when the [General]
section is present (DYNAMOHUB-506)
- Ensured that all chained tasks parse results correctly
(DYNAMOHUB-511)


Release notes - Ngenea Hub - 1.5.0: 2021-11-22
```

```
==============================================

Features
--------

- Workflows can be enabled/disabled. Also, workflows can be
available to use via the API only. (DYNAMOHUB-167)
- File browser can be filtered by filetype, size, change date,
accessed date. (DYNAMOHUB-196)
- Job list filtering options are extended. Jobs can be filtered by
site and path prefix information too. (DYNAMOHUB-299)
- Endpoint to submit search requests (DYNAMOHUB-319)
- Endpoint for retrieving search results (DYNAMOHUB-320)
- Submit async search tasks and store results (DYNAMOHUB-321)
- Periodically remove old search results (DYNAMOHUB-328)
- Support for search backend configurations (DYNAMOHUB-329)
- Added the option to provide the django secret key via the
environment (DYNAMOHUB-342)
- Include available metadata fields with search results
(DYNAMOHUB-382)
- Add job completion and run time to metadata (DYNAMOHUB-63)
- Added support for ngenea-hub to delete GPFS files (DYNAMOHUB-164)
- Added support for ngrecall --skip-check-hash in workflows
(DYNAMOHUB-176)
- Snapdiff discovery task (DYNAMOHUB-245)
- Task to enumerate independent filesets on a site (DYNAMOHUB-251)
- AP-Analytics based search backend task (DYNAMOHUB-321)
- Accept configuration settings for search backend tasks
(DYNAMOHUB-329)
- Compatibility with ngenea 1.12 (DYNAMOHUB-333)
- Added argument "delete_remote_xattrs" to the move_paths task to
remove remote location metadata (DYNAMOHUB-346)
- Provide available metadata fields with search results
(DYNAMOHUB-382)
- Size-aware file list chunking in recursive_action (DYNAMOHUB-53)
- Track size of files being processed for reporting (DYNAMOHUB-63)

Bugfixes
--------

- Set task status to ERROR if any file failed (DYNAMOHUB-161)
- Fixed job processed and failed files incorrectly returning empty
lists (DYNAMOHUB-264)
- Fix transparent recall policy to handle paths with whitespace
(DYNAMOHUB-324)
- Retry on error when waiting for transparent recall (DYNAMOHUB-336)
- Don't treat individual file failures as a task failure
(DYNAMOHUB-161)
- Don't treat ngenea warnings as errors (DYNAMOHUB-186)
- Disable task late ack to try to mitigate consumer timeouts
(DYNAMOHUB-291)
- Friendlier error response when Elasticsearch is unavailable
(DYNAMOHUB-364)
```

```
Improved Documentation
----------------------

- Clarify transparent recall documentation (DYNAMOHUB-324)


Release notes - Ngenea Hub - 1.4.0: 2021-10-15
==============================================

Features
--------

- Consolidate the 3 ngenea-worker systemd services to a single
service for easier management. (DYNAMOHUB-149)


Release notes - Ngenea Hub - 1.3.0: 2021-09-22
==============================================

Features
--------

- “live” changes to file status (DYNAMOHUB-105)
- Transparent Recall logging (DYNAMOHUB-100)
- Expose Groups model via REST API (DYNAMOHUB-277)
- Option to show/hide hidden objects in the filebrowser
(DYNAMOHUB-262)
- UI for setting IP addresses against Site (DYNAMOHUB-232)
- Extend site model to store it's IP addresses (DYNAMOHUB-229)
- ClientKey UI (DYNAMOHUB-143)
- PATCH support for /workflow/ endpoint (DYNAMOHUB-285)
- Spinners on UI components waiting for API response (DYNAMOHUB-281)
- Having visible alerts or redirects according to the websocket
warnings (DYNAMOHUB-276)
- Websocket request should wait for response before sending another
request (DYNAMOHUB-272)
- Reconsider websockets broadcast approach (DYNAMOHUB-271)
- Revamp 404 page (DYNAMOHUB-252)
- Allow the client-key to access all routes excluding user and
client-key (DYNAMOHUB-243)
- Control number of worker threads (DYNAMOHUB-237)
- Change colours for paginated table (DYNAMOHUB-218)
- Make action dropdown wider (DYNAMOHUB-212)
- Make task table sortable (DYNAMOHUB-202)
- Execute the production code via gunicorn (DYNAMOHUB-129)
- Task for checking the current filesets residing on a site
(DYNAMOHUB-251)
- Site worker snapdiff celery discovery task (DYNAMOHUB-245)
- Ability to submit tasks to an existing job (DYNAMOHUB-215)
- Ngenea Hub remote client (DYNAMOHUB-213)


Bugfixes
```

```
--------

- Workers can timeout on waiting for response from rabbitmq
(DYNAMOHUB-291)
- Fails to open directories with strange file names (DYNAMOHUB-288)
- Swagger logs exception (DYNAMOHUB-283)
- Prevent resubmitting a job multiple times in quick succession
(DYNAMOHUB-274)
- "Failure" job filter only shows failures for one site
(DYNAMOHUB-273)
- Actions button visible on the user profile page (DYNAMOHUB-270)
- Unable to deactivate users (DYNAMOHUB-269)
- Update user profile "save" button is always enabled (DYNAMOHUB-268)
- Inaccurate "last login" info (DYNAMOHUB-267)
- Sort order of jobs (DYNAMOHUB-266)
- Browser tries to show contents of a non-directory (DYNAMOHUB-260)
- Fix frontend issue when submitting large number of files to a
workflow (DYNAMOHUB-259)
- Clicking on "Owner" for a clientkey gives a 404 (DYNAMOHUB-253)
- Include ngeneahub-frontend in ngeneahub-images RPM (DYNAMOHUB-242)
- Resubmitting a job doesn't use the same discovery method
(DYNAMOHUB-239)
- Some overall job stats don't update until the dir walk is complete
(DYNAMOHUB-210)
- Files without a file extension are listed as both directory and
folder (DYNAMOHUB-200)

Release notes - Ngenea Hub - 1.2.0: 2021-08-11
==============================================

Features
--------

- Update RabbitMQ to 3.9 (DYNAMOHUB-223)
- paginate tasks in  /api/jobs/<id> (DYNAMOHUB-204)
- Group transparent recall tasks (DYNAMOHUB-216)
- Ability to submit non-recursive tasks (DYNAMOHUB-214)

Bugfixes
--------

- Rest API can not be authenticated with an access token
(DYNAMOHUB-227)
- Fix client side pagination for task list in job details
(DYNAMOHUB-219)
- List of files is job details is empty (DYNAMOHUB-211)
- Using a boolean as a workflow step parameter fails (DYNAMOHUB-208)

Release notes - Ngenea Hub - 1.1.0: 2021-06-19
==============================================

Features
--------
```

```
- Send data between different sites via the UI (DYNAMOHUB-124)
- Rework ngenea output parsing (DYNAMOHUB-115)
- Rebuild UI using React (DYNAMOHUB-103)
- Improve stat task to make directory handling explicit
(DYNAMOHUB-188)
- step argument: skip-check-hash (DYNAMOHUB-176)
- step argument: overwrite-remote (DYNAMOHUB-175)
- step argument: overwrite-local (DYNAMOHUB-174)
- step argument: stub-size (DYNAMOHUB-173)
- React 404 should not change URL (DYNAMOHUB-166)
- Support file delete workflow (DYNAMOHUB-164)
- Recent jobs list should show the most recent jobs, nevermind what
state they are in (DYNAMOHUB-147)
- Job reporting scaling improvements (DYNAMOHUB-84)

Bugfixes
--------

- `api/users/` endpoint throwing errror when a username containing
"." exists (DYNAMOHUB-187)
- "failed to acquire a DMAPI lock EXCL immediately; keeping
trying..." is treated as an error (DYNAMOHUB-186)
- Job reporting not providing correct counts (DYNAMOHUB-185)
- Resubmit button should not be enabled when not usable
(DYNAMOHUB-184)
- Using UI to trigger workflow fails (DYNAMOHUB-177)
- Attempting to post to /api/file/workflow with only a token causes
500 (DYNAMOHUB-171)
- Bug when a directory is in intermediate selection state and closed
(DYNAMOHUB-170)

Release notes - Ngenea Hub - 1.0.4: 2021-06-10
==============================================

Features
--------

- Allow overriding the site on a per-step basis (DYNAMOHUB-151)
- Submission time arguments to workflow support (DYNAMOHUB-150)
- Validate runtime fields (DYNAMOHUB-136)
- Show verbose error information on failure (DYNAMOHUB-156)
- New task: reverse stub (DYNAMOHUB-155)
- Report task types with more meaningful names (DYNAMOHUB-152)

Bugfixes
--------

- Fix bug where sub-directories can fail to be processed
(DYNAMOHUB-148)
- recursive_action failures report as success (DYNAMOHUB-142)

Release notes - Ngenea Hub - 1.0.3: 2021-05-19
==============================================
Features
--------

- Refactor existing ngenea hub tasks to message passing format
(DYNAMOHUB-140)
- Support static arguments to a step (DYNAMOHUB-137)
- Provide existing actions as default workflows (DYNAMOHUB-131)
- enable easy access to dbshell (DYNAMOHUB-134)
- Workflows are now defined dynamically (DYNAMOHUB-132)
- Allow users to create (API) client keys (DYNAMOHUB-141)

Bugfixes
--------

- Fixed an issue with submitting workflows with Client Key
(DYNAMOHUB-144)
- Fixed warnings generated from auto field (DYNAMOHUB-133)

Release notes - Ngenea Hub - 1.0.2: 2021-04-26
==============================================

Features
--------

- Expose JWT Login API endpoints (DYNAMOHUB-90)
- Add Swagger API Documentation (DYNAMOHUB-94)
- Validate that site names do not end with the suffix we use to
identify queue types (DYNAMOHUB-85)
- Allow middle-click/right-click to open navbar links in new tabs
(DYNAMOHUB-83)
- Ship ngeneahub cli tool as a venv with docker-compose included
(DYNAMOHUB-81)
- Improve performance of job page by doing pagination server-side
(DYNAMOHUB-56)

Release notes - Ngenea Hub - 1.0.1: 2021-04-01
==============================================

Bugfixes
--------

- Symlinks cause file browser to fail (DYNAMOHUB-86)
- "Creation time" should be "ctime" - "Last changed
time" (DYNAMOHUB-78)
- Files and folders with newlines in their names fails, with console
error (DYNAMOHUB-55)

Features
--------

- Standalone UI providing interfaces for core Ngenea/Dynamo
workflows (DYNAMOHUB-1)
- Positioning and content of UI buttons (DYNAMOHUB-38)
- Show percentage completion of job based on # of files transferred
```

```
vs remaining (DYNAMOHUB-34)
- Remove redundant environment settings (DYNAMOHUB-82)
- Show a user view page when clicking on a user link (DYNAMOHUB-79)
- Pop-up confirmation of job actions (DYNAMOHUB-57)
- Support for premigrate task (DYNAMOHUB-50)
- Allow viewing directories from different sites (DYNAMOHUB-12)
```

License

Ngenea Hub is licensed under the ArcaPix EULA: https://www.arcapix.com/licenses/EULA.txt

```
ArcaPix EULA
January 2021
http://www.arcapix.com/licenses/EULA.txt
Copyright 2021 ArcaPix Limited

This end user licence agreement (EULA) is a legal agreement between
you (the entity or individual who is using the software) and us (as
applicable either Pixit Media Limited, company number 07298805 or
ArcaStream Ltd, company number 08346283) in respect of (as
applicable):

* the computer software described in the order form, proof of
entitlement (POE), invoice or other document linking to this EULA
(in each case as issued by or agreed in writing with us), or which
otherwise incorporates or is governed by this EULA and the data
supplied with that software (collectively, Software);

* the application programming interface described in the order form,
proof of entitlement (POE), invoice or other document linking to
this EULA (in each case as issued by or agreed in writing with us),
or which otherwise incorporates or is governed by this EULA and the
data supplied with that application programming interface, in each
case whether provided on a standalone basis or alongside the
Software (collectively, API) and

* printed materials and electronic documents associated with that
Software or API (Documents).

The Software, API and Documents are collectively the "Work" and such
term includes each of the Software, API and Documents as applicable.

We license use of the Work to you on the basis of this EULA. We do
not sell the Work to you. We remain the owners of the Work at all
times. By using the Work, or otherwise by clicking "accept" or
otherwise indicating acceptance of this EULA, you confirm you accept
the terms of this EULA. If you do not accept this EULA you may not
use the Work.

Any services we provide, including but not limited to maintenance,
support, development and hosting, will be under separate terms of
```

business, but any software we provide to you incidentally in the course of those services, will also be governed by this EULA unless expressly stated that other licence terms will apply.

You should print a copy of this EULA for future reference.

1 Grant and scope of licence

1.1 In consideration of payment by you of the agreed licence fee and you agreeing to abide by the terms of this EULA, we grant to you a non-exclusive, non-transferable licence to use the Work on the terms of this EULA for the duration of your subscription. Your subscription will only be valid during the period for which you have a valid POE from us to use the Software and when your subscription expires this EULA will automatically terminate without the need for notice. Any termination of this EULA will also terminate your subscription and your POE will be invalidated.

1.2 You may download, install and use the Software for your own internal business purposes only on the systems, either physical or virtual detailed in the accompanying POE as identified in your purchase order (if applicable) or otherwise approved by us or our authorised representatives.

1.3 You may not use the Work for the purposes of making its functionality available to third parties as a service, whether directly or indirectly, without our express written agreement.

1.4 You may access and use the API solely for the purposes of

1.4.1 internally developing applications which communicate and interoperate with software or systems detailed in (and for the purposes detailed in) the accompanying POE as identified in your purchase order (if applicable) or otherwise approved by us or our authorised representatives; and

1.4.2 making calls to the systems or software permitted under clause 1.4.1, subject to any limits detailed in the accompanying POE as identified in your purchase order (if applicable) or otherwise agreed with us or our authorised representatives.

1.5 Your display and use information received through the API or data derived from that information is in each case subject to any limits detailed in the accompanying POE as identified in your purchase order (if applicable) or otherwise agreed with us or our authorised representatives).

1.6 You may use any Documents in support of the use permitted under condition 1.2 and make copies of the Documents as are reasonably necessary for their lawful use.

1.7 This EULA does not grant you permission to use the trade names, trademarks, service marks, or product names of us or our contributors or licensors, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of any "licence" files.

2 Restrictions

2.1 Except as expressly set out in this EULA or as permitted by any local law, you undertake:

2.1.1 not to copy the Work except where such copying is incidental to normal use of the Software, or where it is strictly necessary for the purpose of back-up or operational security;

2.1.2 not to rent, lease, sub-license, loan, translate, merge, adapt, vary or modify the Work;

2.1.3 not to make alterations to, or modifications of, the whole or any part of the Work, nor permit the Work or any part of them to be combined with, or become incorporated in, any other programs or other documentation as applicable, other than as expressly permitted in writing by us;

2.1.4 not to disassemble, decompile, reverse-engineer or create derivative works based on the whole or any part of the Software or API (except as expressly permitted by us in writing or clearly provided for within the functionality of the Software or any accompanying API we provide) nor attempt to do any such thing except to the extent that (by virtue of section 296A of the Copyright, Designs and Patents Act 1988) such actions cannot be prohibited because they are essential for the purpose of achieving inter-operability of the Software or API with another software program, and provided that the information obtained by you during such activities:

2.1.4.1 is used only for the purpose of achieving inter-operability of the Software or API with another software program; and

2.1.4.2 is not unnecessarily disclosed or communicated without our prior written consent to any third party; and

2.1.4.3 is not used to create any software which is substantially similar to the Software or API;

2.1.5 to keep all copies of the Work secure and to maintain accurate and up-to-date records of the number and locations of all copies of the Work;

2.1.6 to supervise and control use of the Work and ensure that the Work are only used by your employees (or such other individuals or entities are you may be expressly permitted in writing by us to allow to access or use the Work) in accordance with the terms of this EULA;

2.1.7 to include our copyright notice on and any "licence" text files in all entire and partial copies you make of the Work on any

```
medium, however you may not use any component parts of the Work
outside of or separately from the Work;

2.1.8 not to provide or otherwise make available the Work in whole
or in part (including but not limited to program listings, object
and source program listings, object code and source code), in any
form to any person other than your employees without prior written
consent from us; and

2.1.9 to comply with all applicable technology control or export
laws and regulations.

2.2 Without prejudice to the restrictions in this EULA on copying,
modifying or creating derivative works from the Work, where you (or
someone on your behalf) creates (solely or in conjunction with
others, and whether in object or source code form) any software or
other work which is based on or derived from the Work (Derivative
Work) in breach of this EULA or otherwise, then in consideration of
the sum of 1 GBP (receipt and sufficiency of which you acknowledge),
you hereby:

2.2.1 assign to us (by way of present assignment of future rights)
all intellectual property rights in such Derivative Work and waive
(and shall procure a waiver of) all moral rights arising under the
Copyright, Designs and Patents Act 1988 in relation to the
Derivative Work and, so far as is legally possible, any broadly
equivalent rights that may exist in any territory of the world; and

2.2.2 In the event that any rights in such Derivative Work are not
assigned to us pursuant to clause 2.2.1, you hereby grant to us an
exclusive, royalty-free, worldwide, transferrable, irrevocable,
perpetual licence (together with the right to grant sub-licences) to
use in any manner as we determine, any such Derivative Work.

2.3 For the avoidance of doubt, for the purposes of clause 2.2,
Derivative Work shall not include works which merely link or bind by
name an existing third party application to the interfaces of the
Software or API but does include works which are created to
integrate with, or to be processed using, the interface of the
Software or any API which we provide.

2.4 You agree not to (by your act or omission) do, or permit to be
done, any act that will or may weaken, damage or be detrimental to
the Work or any of our intellectual property rights or our or any of
our contributors or licensors' rights in such, or seek to register
any rights in the Work or any part of it or seek to commence
litigation against any third party in respect of any intellectual
property infringement in relation to the Work or any part of it.

3 Intellectual property rights

3.1 You acknowledge that all intellectual property rights in the
Work anywhere in the world belong to us or our licensors or
contributors, that rights in the Work are licensed (not sold) to
```

```
you, and that you have no rights in, or to, the Work other than the
right to use them in accordance with the terms of this EULA.

3.2 You acknowledge (unless explicitly agreed in writing by us) that
you have no right to have access to the Software in source code form.

4 Liability

4.1 You acknowledge that the Work has not been developed to meet
your individual requirements, including any particular cybersecurity
requirements you might be subject to under law or otherwise, and
that it is therefore your responsibility to ensure that the
facilities and functions of the Software and API as described in the
Documents meet your requirements.

4.2 We only supply the Work for internal use by your business, and
you agree not to use the Work for any other purposes unless
expressly permitted in writing by us.

4.3 We shall not in any circumstances whatever be liable to you,
whether in contract, tort (including negligence), breach of
statutory duty, or otherwise, arising under or in connection with
this EULA for:

4.3.1 loss of profits, sales, business, or revenue;

4.3.2 business interruption;

4.3.3 loss of anticipated savings;

4.3.4 loss or corruption of data or information or any loss arising
from misconfiguration or incorrect implementation or use of any API;

4.3.5 loss of business opportunity, goodwill or reputation;

where any of the losses set out in condition 4.3.1 to condition
4.3.5 are direct or indirect; or

4.3.6 any special, indirect or consequential loss, damage, charges
or expenses.

4.4 Other than the losses set out in condition 4.3 (for which we are
not liable), our maximum aggregate liability under or in connection
with this EULA whether in contract, tort (including negligence) or
otherwise, shall in all circumstances not exceed a sum equal to the
Licence Fee paid in the 12 months prior to the event first giving
rise to any liability. This maximum cap does not apply to condition
4.5.

4.5 Nothing in this EULA shall limit or exclude our liability for:

4.5.1 death or personal injury resulting from our negligence;

4.5.2 fraud or fraudulent misrepresentation;
```

4.5.3 any other liability that cannot be excluded or limited by English law.

4.6 Save as required by applicable law or agreed to in writing, we provide the Work on an "AS IS" basis, without conditions, warranties, representations or other terms of any kind, either express or implied (and any such implied conditions, warranties, representations or other terms, whether implied by statute, common law or otherwise, are excluded to the fullest extent permitted by law), including, without limitation, any conditions, warranties, representations or other terms relating to title, non-infringement, merchantability, or fitness for a particular purpose. You are solely responsible for determining the appropriateness of using the Work and for any configuration or interface necessary for you to effectively use the Work and assume any risks associated with your exercise of permissions under this EULA.

4.7 Without prejudice to clause 4.6, where the API interacts with any software or system which is not provided by us, we are not responsible and shall have no liability in any way for such software or system.

5 Termination

5.1 We may terminate this EULA immediately by written notice to you if you commit a breach of this EULA which you fail to remedy (if remediable) within 14 days after the service of written notice requiring you to do so. Without prejudice to our rights under this clause 5.1, your rights under this EULA will terminate automatically without the need for notice if you commit a material breach of any of the terms of this EULA.

5.2 On termination for any reason:

5.2.1 all rights granted to you under this EULA shall cease;

5.2.2 you must immediately cease all activities authorised by this EULA; and

5.2.3 you must immediately and permanently delete or remove the Work from all computer equipment in your possession, and immediately destroy or return to us (at our option) all copies of the Work then in your possession, custody or control and, in the case of destruction, certify to us that you have done so.

6 Communications between us

6.1 We may update the terms of this EULA at any time on notice to you in accordance with this condition 6. Your continued use of the Work following the deemed receipt and service of the notice under condition 6.3 shall constitute your acceptance to the terms of this EULA, as varied. If you do not wish to accept the terms of the EULA (as varied) you must immediately stop using and accessing the Work on the deemed receipt and service of the notice.

6.2 If we have to contact you, we will do so by email or by pre-paid post to the address you provided in accordance with your order for or registration of the Work.

6.3 Note that any notice:

6.3.1 given by us to you will be deemed received and properly served 24 hours after it is first posted on our website, 24 hours after an email is sent, or three days after the date of posting of any letter; and

6.3.2 given by you to us will be deemed received and properly served 24 hours after an email is sent, or three days after the date of posting of any letter.

6.4 In proving the service of any notice, it will be sufficient to prove, in the case of posting on our website, that the website was generally accessible to the public for a period of 24 hours after the first posting of the notice; in the case of a letter, that such letter was properly addressed, stamped and placed in the post to the address of the recipient given for these purposes; and, in the case of an email, that such email was sent to the email address of the recipient given for these purposes.

7 Events outside our control

7.1 We will not be liable or responsible for any failure to perform, or delay in performance of, any of our obligations under this EULA that is caused by an Event Outside Our Control. An Event Outside Our Control is defined below in condition 7.2.

7.2 An Event Outside Our Control means any act or event beyond our reasonable control, including without limitation failure of public or private telecommunications networks.

7.3 If an Event Outside Our Control takes place that affects the performance of our obligations under this EULA:

7.3.1 our obligations under this EULA will be suspended and the time for performance of our obligations will be extended for the duration of the Event Outside Our Control; and

7.3.2 we will use our reasonable endeavours to find a solution by which our obligations under this EULA may be performed despite the Event Outside Our Control.

8 Third Party Software

8.1 Any part or component of the Software which has been contributed or created by any third party (including any open-source software) and which is not owned by us (Third Party Software) shall be deemed to be incorporated within the Software for the purposes of this EULA (except where expressly provided to the contrary) and use of the Third Party Software shall be subject to (and you shall comply with) such additional terms as relate to such Third Party Software from time to time (Third Party Additional Terms), and such Third Party Additional terms shall take precedence over this EULA in relation to such Third Party Software. You shall indemnify and hold us harmless against any loss or damage which we may suffer or incur as a result of your breach of any Third Party Additional Terms howsoever arising, and we may treat your breach of any Third Party Additional Terms as a material breach of this EULA.

8.2 For the avoidance of doubt, the performance of, and any issues caused by or arising from, any Third Party Software shall be considered an Event Outside Our Control and (without prejudice to the provisions of this EULA in relation to warranties regarding the Software generally) all Third Party Software is provided on an "AS IS" basis and without conditions, warranties, representations or other terms of any kind, either express or implied (and any such implied conditions, warranties, representations or other terms, whether implied by statute, common law or otherwise, are excluded to the fullest extent permitted by law), including, without limitation, any conditions, warranties, representations or other terms relating to title, non-infringement, merchantability, or fitness for a particular purpose.

9 Other important terms

9.1 We may transfer our rights and obligations under this EULA to another organisation, but this will not affect your rights or our obligations under this EULA.

9.2 You may only transfer your rights or your obligations under this EULA to another person if we agree in writing.

9.3 This EULA and any document expressly referred to in it constitutes the entire agreement between us and supersedes and extinguishes all previous agreements, promises, assurances, warranties, representations and understandings between us, whether written or oral, relating to its subject matter. You agree that you shall have no remedies in respect of any statement, representation, assurance or warranty (whether made innocently or negligently) that is not set out in this EULA or any document expressly referred to in it. You agree that you shall have no claim for innocent or negligent misrepresentation or negligent misstatement based on any statement in this EULA or any document expressly referred to in it.

9.4 If we fail to insist that you perform any of your obligations under this EULA, or if we do not enforce our rights against you, or if we delay in doing so, that will not mean that we have waived our rights against you and will not mean that you do not have to comply with those obligations. If we do waive a default by you, we will only do so in writing signed by us, and that will not mean that we will automatically waive any later default by you.

```
9.5 Each of the conditions of this EULA operates separately. If any
court or competent authority decides that any of them are unlawful
or unenforceable, the remaining conditions will remain in full force
and effect.

9.6 This EULA, its subject matter and its formation (and any non-
contractual disputes or claims) are governed by English law. We both
irrevocably agree to the exclusive jurisdiction of the courts of
England and Wales.
```