PanAPI Reference Guide

RESTful management of multi-tenant VDURA storage infrastructure

Documentation for PanFS 12.0.0

© 2002–2026 VDURA Inc. All rights reserved. VDURA, the VDURA logo, VDURA Data Platform, VeLO, VPOD, DirectFlow, PanView, PanMove, and Multi-Level Erasure Coding (MLEC) are trademarks or registered trademarks of VDURA, Inc. in the U.S. and/or other countries. All other trademarks, registered trademarks, trade names, company names, and service marks are the respective properties of their holders.

Parts of this product are derived from Berkeley Database software, licensed from Sleepycat Software.

Parts of this product are derived from Berkeley BSD systems, licensed from the University of California. The BDB software is also BSD licensed, and carries the following copyrights:

- Copyright © 1990–2009 Oracle. All rights reserved.
- Copyright © 1990, 1993, 1994, 1995 The Regents of the University of California. All rights reserved.
- Copyright © 1995, 1996 The President and Fellows of Harvard University. All rights reserved.
- Copyright © 2000–2005 INRIA, France Telecom

Parts of this product are derived from gd Version 1.8.4, licensed by Boutell.Com, Inc.

Copyright © 1982, 1986, 1991, 1993 The Regents of the University of California. All rights reserved.

© UNIX System Laboratories, Inc.

All or some portions of this file are derived from material licensed to the University of California by American Telephone and Telegraph Co. or Unix System Laboratories, Inc. and are reproduced herein with the permission of UNIX System Laboratories, Inc.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Disclaimer

VDURA has carefully checked the information in this document and believes it to be accurate. However, VDURA assumes no responsibility for any inaccuracies that this document may contain. In no event will VDURA be liable for direct, indirect, special, exemplary, incidental, or consequential damages resulting from any defect or omission in this document, even if advised of the possibility of such damages.

Any alterations or inconsistent replications of this document cannot be considered accurate and are void. VDURA may not be able to support issues related to altered documents that are not approved by subject matter experts.

Document version 12.0.0-2026-11-12-11:48:34

1. Introduction 1
1.1 What PanAPI Is 1
1.2 Authentication Schemes 1
1.3 Request and Response Conventions 1
1.4 Who May Call What 2
1.5 Before You Call an Endpoint 5
1.6 How This Guide Is Organized 5

2. Getting Started 6
2.1 Before You Start 6
2.2 Bring Up a Realm 7
2.3 Onboard a Tenant 9
2.4 Give Clients Access to a Tenant 13
2.5 Enable and Use S3 13
2.6 Delete a Tenant 16
2.7 Collect Logs for Support 16
2.8 Before You Call This: Dependencies at a Glance 17

3. Endpoint Summary 21
4. Authentication 29
5. Realm and Platform 49
5.1 About 49
5.2 Realm 50

6. Tenants 69
7. Tenant Users and Roles 92
7.1 Tenant Users 92
7.2 Tenant User Roles 103

8. PanFS Users 107
9. Service Accounts 120
10. Volumes 139
11. StorageSets 160
12. Subnets and Attachments 175
12.1 Subnets 175
12.2 Host Attachments 201

13. Nodes and Hardware 215
13.1 Nodes 215
13.2 Hardware 226

14. S3 Gateway 234
14.1 Tenant S3 Configuration 234
14.2 Realm-Wide S3 Administration 247
14.3 S3 Buckets 254
14.4 S3 Gateway Instances 262

15. Observability 280
15.1 Metrics and Logs Behind These Endpoints 280
15.2 Endpoint Reference 283

Appendix A. Error Responses 310
Appendix B. Schema Index 312

1. Introduction

This chapter covers What PanAPI Is, Authentication Schemes, Request and Response Conventions, Who May Call What, Before You Call an Endpoint, and How This Guide Is Organized.

1.1 What PanAPI Is

PanAPI is the RESTful management API for VDURA storage infrastructure: tenants, users, volumes, nodes, hardware, StorageSets, subnets, and the S3 gateway. It is the single backend service both `panctl` (the command-line client) and the VDURA web UI (Vantage) talk to; neither is more authoritative than the other, and changes made through one are immediately visible through the other. See the VDURA PanCTL User Guide's introduction for that CLI-side perspective.

All endpoints are namespaced under the base path `/api/v1`. The realm also serves an interactive Swagger UI at `/docs` and the raw machine-readable specification at `/openapi.json`; this guide describes PanAPI version `12.0.0`. A realm serves its own interactive reference at `/docs`; use it to see the exact behavior of the realm you administer.

1.2 Authentication Schemes

Every call needs an `Authorization: Bearer <JWT>` header, except `POST /auth/login`, `POST /auth/refresh`, and `GET /api/v1/about` (see Chapter 5). Two schemes can supply that header, and only one may be active on a given request. A request carrying both credentials is rejected with 401:

- **`PasswordLogin`**: sign in with a tenant username and password. The username carries the tenant it belongs to, `<tenant-name>/<user>`. Leave the prefix off for the platform tenant, so `admin` signs in as the platform administrator. Tenant users are not fully supported in this release (see Chapter 7).
- **`BearerJWT`**: present a JWT access token directly, such as one returned by `POST /auth/tenant-switch` or a long-lived API token from `POST /auth/tokens`.

The full authentication endpoint reference, including tenant switching and API token issuance, is Chapter 4.

1.3 Request and Response Conventions

- **Content types.** Most request and response bodies are `application/json`. The two token endpoints (`POST /auth/login`, `POST /auth/refresh`) take `application/x-www-form-urlencoded` instead, matching the OAuth2 password-grant convention.

- **Success responses.** A successful call returns either a JSON body describing the result (with a `2xx` status) or `204 No Content` for actions and deletes that return nothing.
- **Error responses.** Every error response is a JSON object with a `detail` field: a plain string for most errors (`{"detail": "Not authenticated"}`), or, for `422 Validation Error` , a list of per-field `ValidationError` objects describing exactly which field failed and why. See Appendix A for the common error shapes shared across every endpoint in this guide. The endpoint sections below list the status codes their spec declares; the common codes apply to every endpoint whether or not they are listed.
- **Tenant scoping.** Most resources belong to a tenant. A tenant user's calls are implicitly scoped to their own tenant; the platform tenant (Platform Admin) can typically pass an optional `tenant_id` query parameter or `all_tenants` on list endpoints to read across tenants. A tenant user's writes stay in their own tenant, while a Platform Admin can write into any tenant through the `/tenants/{tenant_id}/...` paths.

1.4 Who May Call What

Access depends on the caller's role, and some calls are open only to the platform tenant. The prose elsewhere in this guide follows the table below where the two differ.

Task	Who may do it	Anything else returns
Update a tenant (`PATCH /tenants/{tenant_id}`)	Platform Admin only	`403` for a Tenant Admin
Change a tenant's StorageSet quota	`platform-admin` and `storage-admin` , for any tenant	`403` for a Tenant Admin
Create a tenant's subnet (`POST /tenants/{tenant_id}/subnets`)	A Platform Admin, calling the endpoint directly without switching into the tenant	`403` from a switched-in session

Task	Who may do it	Anything else returns
Call `mount-info` and the attachment endpoints	Platform roles only, not switched into a tenant	`403` for a Tenant Admin, and for `mount-info` through the tenant switcher
Query logs and metrics through the observability proxy	Platform roles only (`observability-realm:view`)	`403`
Start, download, and delete a realm log collection	`platform-admin` . Downloading needs its own permission, `realm-logs:download` because the archive holds every node's logs and system databases.	`403`
List and read realm log collections	`platform-admin` , `platform-read-only` , and `storage-admin`	`403`
Create or delete a volume (`POST /volumes` , `DELETE /volumes/{volume_id}`)	A caller whose token is for the tenant: a Platform Admin through the tenant switcher, or a tenant admin	`404` for a Platform Admin who has not switched in because the volume would go to the platform tenant

Task	Who may do it	Anything else returns
Write into a tenant other than your own	A Platform Admin, through the `/tenants/{tenant_id}/...` paths (S3, users, PanFS users) or the tenant switcher	Tenant users write only in their own tenant
Set `dns.mode` to `panfs` on a tenant	The platform tenant only	`400` for any other tenant
Assign the roles `platform-support` or `s3-gateway`	Nobody; both are reserved	`400`

Roles are defined per tenant, so an unknown role in a role-assignment request also returns `400`. The S3 aggregator is off by default and must be turned on.

Reserved roles. Two built-in roles exist only for the platform's own use, and no user can be given either one; trying to assign them returns `400`. `platform-support` is the role a token holds while a Platform Admin acts inside a tenant through the tenant switcher. It has nothing to do with VDURA support staff. It is why, on a switched-in call, an audit or token record shows `platform-support` as the acting principal (`is_delegated` is true) and names the real caller in the delegation fields, and why a switched-in token cannot switch again. `s3-gateway` is the identity the S3 gateway itself uses. It can appear in a role listing, but it is never given to a customer user.

The tenant switcher, `POST /auth/tenant-switch`, exchanges a Platform Admin token for a short-lived token scoped to one tenant (30 minutes to 12 hours). Calls made with it act in that tenant. Keep the original token for calls that must not be switched in, such as `mount-info` and subnet creation.

1.5 Before You Call an Endpoint

Many calls only work after other resources exist. Chapter 2 walks through the main scenarios end to end: bringing up a realm, onboarding a tenant, giving clients access, enabling S3, deleting a tenant, and collecting logs for support. Endpoints that depend on earlier steps also carry a **Before you call this** note under their description, naming what must already exist and the error returned if it does not. Section 2.8 collects those notes in one table.

1.6 How This Guide Is Organized

Chapter 2 is a getting-started walkthrough of the main scenarios, in the order the calls must be made. Chapter 3 is a single master index of every endpoint in the guide, for a full-text scan or search without opening each chapter in turn. Chapters 4 through 15 each cover one area of the API, grouped the same way the specification itself tags its endpoints: Authentication, Realm and Platform, Tenants, Tenant Users, PanFS Users, Service Accounts, Volumes, StorageSets, Subnets and Attachments, Nodes and Hardware, the S3 Gateway, and Observability. Each of those chapters opens with its own **Endpoint Summary** table (method, path, one-line summary) that doubles as that chapter's index, followed by the full reference for every endpoint: parameters, request body fields, and response shapes. Appendix A covers the error conventions shared across all endpoints; Appendix B is an alphabetical index of every schema used anywhere in this guide.

2. Getting Started

The reference chapters describe each endpoint on its own. Many calls only work after other resources exist, so this chapter shows how the calls fit together, covering what to create, in what order, and what to wait for. Each scenario lists its steps in the order that works. Run in another order, a step fails with the error noted beside it. Appendix A lists the common error shapes.

Every endpoint named here links to its full reference. Most of these steps are Platform Admin calls; the step says when a different caller is needed.

2.1 Before You Start

1. Sign in with `POST /api/v1/auth/login` and send the returned token as `Authorization: Bearer <JWT>` on every other call (see Chapter 1.2).
2. Check the realm with `GET /api/v1/realm`. The field `panfs_mounted` must be `true` before you create a tenant or a volume. While it is `false`, those calls return `503`. The API runs on the realm president, so after a change of president call the new president; the old address stops answering.
3. Keep the dependency order in mind. A **StorageSet** holds capacity; a **tenant** gets a quota in a StorageSet and a **subnet**; **volumes**, **users**, and **S3** resources live inside the tenant. Nothing in the second group works before the first group exists.
4. **Use the right token for each call.** Most calls here use your own Platform Admin token. Two kinds of call differ. `POST /api/v1/volumes` creates the volume in the tenant of the token, so call `POST /api/v1/auth/tenant-switch` first and use the token it returns for the volume call. A Platform Admin who calls without switching gets `404` because the volume would go to the platform tenant. `mount-info` is the reverse; it returns `403` through the tenant switcher, so call it with your own token. Calls that carry the tenant in the URL (S3, users, and PanFS users) work either way. Keep your original token while you use the switched one.

Every request body in this chapter is JSON, so send it with `Content-Type: application/json`. `<realm>` is the address of the realm president. For example, sign in and then create a StorageSet:

```
curl -X POST "https://<realm>/api/v1/auth/login" \
  -d "username=admin" -d "password=<password>"

curl -X POST "https://<realm>/api/v1/storagesets" \
  -H "Authorization: Bearer <token>" \
  -H "Content-Type: application/json" \
  -d '{"name": "Set 1", "spares_requested": 2}'
```

The request bodies below use placeholders such as `<tenant-id>` for values you supply. Each body matches the schema of its endpoint.

2.2 Bring Up a Realm

Run these once, as a Platform Admin, before any tenant exists.

1. **Create a StorageSet.** `POST /api/v1/storagesets` with a `name` (unique in the realm) and, optionally, `spares_requested`. The response carries the StorageSet ID.

   Request body for `POST /api/v1/storagesets`:

   ```
   {
     "spares_requested": 2,
     "name": "Set 1"
   }
   ```

2. **Find the nodes you can add.** `GET /api/v1/nodes` lists every node. Add only nodes that are unassigned (a StorageSet ID of 0) and whose role is `vhost`, `vosd`, or `snode`. Other roles, such as `qnode`, are refused. On a realm that runs on virtual hosts, list the `vhost` nodes; their VPODs join with them.

3. **Add the storage.** `POST /api/v1/storagesets/{storageset_id}/actions` with the `add_storage` action, a `mode` (`add_spare` to expand capacity), and the `dataserials` to add. Always list the nodes; a call without `dataserials` adds nothing, yet still returns `204`. The call returns `204` at once, but the storage is absorbed asynchronously, which takes about 15 minutes. A vhost adds its capacity later because each of its VPODs joins after it installs and boots; until the first one joins, the StorageSet reports the `activating` condition. Poll `GET /api/v1/storagesets/{storageset_id}` until `state.status` is `ok`. One node is rarely enough; a volume needs several online disks (OSDs) for its RAID layout, 4 for `raid6+` (the default) and 3 for `raid5+` and `raid10+`. With too few, volume create returns `409` "Not enough online disks in StorageSet to create volume". A StorageSet built from one vhost has 3 VPODs, which is too few for `raid6+`.

   Request body for `POST /api/v1/storagesets/{storageset_id}/actions`:

   ```
   {
     "action": "add_storage",
     "params": {
       "mode": "add_spare",
       "dataserials": [
         "<dataserial-1>",
         "<dataserial-2>"
       ],
       "spares_requested": 2
     }
   }
   ```

4. **Wait for the filesystem.** After the StorageSet status is `ok`, poll `GET /api/v1/realm` until `panfs_mounted` is `true`. The filesystem usually mounts about 30 seconds after the status is `ok`. Until then, tenant and volume create return `503`.

5. **Choose the network segment type.** `PUT /api/v1/subnets/segment-type` sets VLAN or VXLAN for the whole realm. Set it before the first subnet is created. It cannot change while subnets exist on a different type (`409`). Read it back with `GET /api/v1/subnets/segment-type`.

   Request body for `PUT /api/v1/subnets/segment-type`:

   ```
   {
     "segment_type": "vxlan"
   }
   ```

The realm is ready for tenants when a StorageSet reports status `ok`, `panfs_mounted` is `true`, and the segment type is set.

2.3 Onboard a Tenant

Steps 1 to 4 are Platform Admin calls. Run them in this order.

1. **Create the tenant.** `POST /api/v1/tenants` with `name` (3 to 10 alphanumeric characters), `display_name`, `email`, and `auth_settings`. It returns `503` while `panfs_mounted` is `false`. The tenant's `status` follows its subnet: `pending` means it has no subnet, `configuring` means it has a subnet but no node address on it yet, and `active` means at least one node address has been issued. Check it with `GET /api/v1/tenants/{tenant_id}`.

   Request body for `POST /api/v1/tenants`:

   ```
   {
     "name": "demo",
     "display_name": "Demo Tenant",
     "email": "admin@demo.example.com",
     "auth_settings": {
       "auth_mode": "local"
     }
   }
   ```

2. **Give the tenant capacity.** `PUT /api/v1/tenants/{tenant_id}/storagesets/{storageset_id}` with a `quota_bytes`. Without this, creating a volume for the tenant returns `404`. Only `platform-admin` and `storage-admin` can set it, and a Tenant Admin gets `403`. A quota on a StorageSet with no free space returns `400` ("... exceeds available 0 bytes"), so an empty StorageSet cannot be given to a tenant.

   Request body for `PUT /api/v1/tenants/{tenant_id}/storagesets/{storageset_id}`:

   ```
   {
     "quota_bytes": 10000000000
   }
   ```

3. **Give the tenant a subnet.** `POST /api/v1/tenants/{tenant_id}/subnets` with a `name`, `ip_range_base`, `ip_range_prefix`, and optionally a `segment_id`. If you leave it out, the realm assigns an unused one; if you give one, it must be unused on the realm. Call it directly as a Platform Admin. Do not switch into the tenant first. A switched-in session returns `403`.

   Request body for `POST /api/v1/tenants/{tenant_id}/subnets`:

```
{
  "name": "subnet-demo",
  "ip_range_base": "192.168.0.0",
  "ip_range_prefix": 16,
  "gateway": "192.168.0.1",
  "segment_id": 4201
}
```

4. **Wait for the subnet to be usable.** You need two things, and they are not the same. Poll `GET /api/v1/tenants/{tenant_id}` until `status` is `active`, which is enough to create volumes; before that, volume create returns `409` ("The tenant holds no issued subnet address, so a volume created for it could not be reached. Create a subnet for the tenant and let one of its nodes report an address into it first"). Then poll `GET /api/v1/tenants/{tenant_id}/subnets/{subnet_id}/mount-info` until `rmlist` is not empty, which is what mounting needs. `rmlist` lists only replica Directors with an IPv4 address, so it can still be empty when `directors` is not. Both normally arrive within seconds of creating the subnet: `active` typically 2 to 7 seconds after, and `rmlist` under a second later. If the tenant stays `configuring` for minutes, check that the Directors are online. `active` means an address was issued, not that traffic flows, so confirm with a mount.

5. **Optional: create a tenant user and give it a role.** Tenant users are not fully supported in this release (see Chapter 7), so you can skip this step and use the tenant switcher in step 6. `POST /api/v1/tenants/{tenant_id}/users` with a `username` and `password`, then `PATCH /api/v1/tenants/{tenant_id}/users/{user_id}/roles` with `["admin"]`. Roles are defined per tenant; list them with `GET /api/v1/tenants/roles`. An unknown or reserved role (`platform-support`, `s3-gateway`) returns `400`. The API supports tenant users, though Vantage does not yet offer full management of them. The user signs in as `<tenant-name>/<user>` through `POST /api/v1/auth/login`.

   Request body for `POST /api/v1/tenants/{tenant_id}/users`:

   ```
   {
     "username": "demoadmin",
     "password": "change-me-now",
     "email": "demoadmin@demo.example.com",
     "auth_settings": {
       "auth_mode": "local"
     }
   }
   ```

   Request body for `PATCH /api/v1/tenants/{tenant_id}/users/{user_id}/roles`:

   ```
   {
     "roles": [
       "admin"
     ]
   }
   ```

6. **Create volumes.** Use a token for the tenant. Switch into it with `POST /api/v1/auth/tenant-switch` as a Platform Admin, which needs no tenant user, or sign in as the tenant admin if you created one in step 5. Then call `POST /api/v1/volumes` with a `pathname` and a `storageset_id`. Always send `hard_quota_bytes` with a value greater than 0; a tenant volume without one returns `400`. The call returns `503` while `panfs_mounted` is `false`, and `504` if no answer arrived in time (the volume may still appear, so list volumes before retrying).

   Request body for `POST /api/v1/auth/tenant-switch`:

   ```
   {
     "tenant_id": "<tenant-id>",
     "ttl_minutes": 60
   }
   ```

   Request body for `POST /api/v1/volumes`:

   ```
   {
     "pathname": "/vol1",
     "storageset_id": 1,
     "hard_quota_bytes": 10000000000
   }
   ```

2.4 Give Clients Access to a Tenant

A client host reaches a tenant's volumes over the tenant subnet. A Platform Admin who is not switched into the tenant performs both steps. A Tenant Admin gets `403` on `mount-info` and on every attachment endpoint, and `mount-info` also returns `403` through the tenant switcher.

1. **Attach the host.** `POST /api/v1/tenants/{tenant_id}/subnets/{subnet_id}/attachments` with the host's `underlay_ip`, and optionally a `hostname` and an `overlay_ip` on the tenant network. The response carries the `overlay_ip`, `overlay_prefix`, and `mtu` the host must apply to its tenant interface. The `mtu` is never 0; set the host's interface to exactly that value.

   Request body for `POST /api/v1/tenants/{tenant_id}/subnets/{subnet_id}/attachments`:

   ```
   {
     "underlay_ip": "10.128.9.0",
     "hostname": "client-0"
   }
   ```

2. **Mount with the realm addresses.** `GET /api/v1/tenants/{tenant_id}/subnets/{subnet_id}/mount-info` returns `rmlist`, the addresses to pass to the mount command, joined with `;`, and `directors`, the Directors that hold an address on the subnet. Use `rmlist` for the mount; it is empty until a realm node has reported an address. Give the addresses to the administrator of the client.

To remove a host, call `DELETE /api/v1/tenants/{tenant_id}/subnets/{subnet_id}/attachments/{attachment_id}`.

2.5 Enable and Use S3

S3 needs a realm switch, a tenant configuration, a running gateway instance, and a PanFS user for the credentials. Run the steps in this order. The S3, PanFS user, and bucket calls carry the tenant in the URL, so a Platform Admin can call them directly or through the tenant switcher.

1. **Create the volume S3 will serve.** The tenant's `default_prefix` must name an existing volume of the tenant (or a directory inside one), so create it first (see 2.3, step 6, which needs a token for the tenant).

2. **Enable S3 for the realm.** `PUT /api/v1/s3/enabled` with `enabled` set to `true`. Enabling the tenant before the realm returns `409`.

   Request body for `PUT /api/v1/s3/enabled`:

   ```
   {
     "enabled": true
   }
   ```

3. **Create the tenant S3 configuration.** `PUT /api/v1/tenants/{tenant_id}/s3`. No field is required, so `{}` returns `201`.

   Request body for `PUT /api/v1/tenants/{tenant_id}/s3`:

   ```
   {}
   ```

4. **Turn S3 on for the tenant.** `PATCH /api/v1/tenants/{tenant_id}/s3` with `enabled` set to `true` and the `default_prefix`. The response is `200` and lists `restart_required`, the changes that running gateways pick up only after a restart.

   Request body for `PATCH /api/v1/tenants/{tenant_id}/s3`:

   ```
   {
     "enabled": true,
     "default_prefix": "/vol1"
   }
   ```

5. **Create a PanFS user for the S3 credentials.** `POST /api/v1/tenants/{tenant_id}/fs-users` with a `name`. The user's `access_key` defaults to the name. Read the secret with `GET /api/v1/tenants/{tenant_id}/fs-users/{fs_user_id}/s3/secret`.

   Request body for `POST /api/v1/tenants/{tenant_id}/fs-users`:

   ```
   {
     "name": "alice",
     "uid": 1000,
     "gid": 1000
   }
   ```

6. **Create a gateway instance.** `POST /api/v1/tenants/{tenant_id}/s3/instances` with a five-character `instance_id` and the `dataserial` of the node that hosts it (from `GET /api/v1/nodes`). The tenant needs a subnet, and gets at most one instance per node (`409` otherwise).

   Request body for `POST /api/v1/tenants/{tenant_id}/s3/instances`:

   ```
   {
     "instance_id": "abc12",
     "dataserial": "<dataserial>"
   }
   ```

7. **Start it.** `POST /api/v1/tenants/{tenant_id}/s3/instances/{iid}/start` returns `202`: The desired state is recorded, not reached. Poll `GET /api/v1/tenants/{tenant_id}/s3/instances/{iid}` until `observed` is `running`.

8. **Create a bucket.** `POST /api/v1/tenants/{tenant_id}/s3/buckets` with a `name` and an `owner`, which is the access key of the PanFS user from step 5. Bucket calls return `503` until an instance is running, and a `503` for "no running gateway" does not clear on retry. A `504` means the bucket may exist, so list buckets before retrying.

   Request body for `POST /api/v1/tenants/{tenant_id}/s3/buckets`:

   ```
   {
     "name": "my-bucket",
     "owner": "alice"
   }
   ```

9. **Connect a client.** Read the instance's address from the `endpoint` field of `GET /api/v1/tenants/{tenant_id}/s3/instances/{iid}`, for example `https://192.168.79.51:3000`. Each instance has its own address; the field is null until the hosting node holds an address on the tenant subnet. The port is 3000 for every instance. Connect over HTTPS from a host attached to the tenant subnet (see 2.4), and sign requests with the access key and secret from step 5. By default (`realm-generated`) the gateway certificate is signed by the realm's internal certificate authority, which is not exposed to customers, so a client cannot verify it. To give clients a certificate they can verify, set `tls.mode` to `tenant-provided` and supply your own certificate and authority.

2.6 Delete a Tenant

Deleting a tenant does not cascade. The call returns `409` until the tenant's S3 instances, volumes, and subnets are removed, in that order.

1. **Remove the S3 instances.** `POST /api/v1/tenants/{tenant_id}/s3/instances/delete-all` waits (up to its `timeout`) for the instances to be gone. Skip this when the tenant never used S3. A leftover S3 instance also makes the tenant delete return `409`.
2. **Delete the tenant's volumes.** Find the IDs with `GET /api/v1/volumes` and its `tenant_id` filter, which a Platform Admin can call without switching. Deleting needs a token for the tenant, as creating does; switch in with `POST /api/v1/auth/tenant-switch` or sign in as the tenant admin, then call `DELETE /api/v1/volumes/{volume_id}` for each volume (`204`). Without the switcher a Platform Admin gets `404` "Volume not found".
3. **Delete the tenant's subnets.** `DELETE /api/v1/tenants/{tenant_id}/subnets/{subnet_id}` for each subnet.
4. **Delete the tenant.** `DELETE /api/v1/tenants/{tenant_id}`.

2.7 Collect Logs for Support

A log collection gathers logs, system databases, and corefiles from the realm's nodes into one archive. Starting, downloading, and deleting a collection are Platform Admin calls (see Chapter 1.4).

1. **Start the collection.** `POST /api/v1/realm/log-collections` with no body collects everything from every node. To limit it, set `scope` (`all`, `repset`, `directors`, or `storage`), `hosts`, or `storagesets`. The call answers `202` with the new collection and a `Location` URL. Only one collection runs at a time, so a second start returns `409`.
2. **Wait for it to end.** Poll `GET /api/v1/realm/log-collections/{collection_id}` until `status` is `succeeded` or `failed`. A run in which some hosts failed still succeeds, and `failed_hosts` lists them. On failure, `error` says why.
3. **Download the archive.** `GET /api/v1/realm/log-collections/{collection_id}/archive` returns a tar file with one `.tgz` per node. It is often hundreds of megabytes, so write it to disk as it arrives and do not hold it in memory. The collection's `archive_sha256` is also sent as the `ETag`, so check the finished file against it. To resume an interrupted download, send `Range` and `If-Range` as described under the endpoint.
4. **Clean up.** `DELETE /api/v1/realm/log-collections/{collection_id}` stops a collection that is still running, cleans the nodes, and removes the archive.

Old archives are cleaned when a new run starts or when a collection is deleted, not on a timer. By default, the newest 3 runs and 24 hours are kept.

An archive holds configuration and state from every node, so handle it as sensitive and keep it inside support channels. Archives stay on the Director that produced them, and they do not follow the API if it moves to another Director.

2.8 Before You Call This: Dependencies at a Glance

Each endpoint below needs something to exist or be true first. The same note appears under the endpoint in its reference chapter.

Endpoint	Needs first	Otherwise
`POST /api/v1/tenants`	The realm filesystem is mounted (`panfs_mounted` is `true` on `GET /api/v1/realm`).	`503` while it is not mounted.
`PUT /api/v1/tenants/{tenant_id}/storagesets/{storageset_id}`	The tenant and the StorageSet exist, and the StorageSet has capacity. Only `platform-admin` and `storage-admin` may call it.	`403` for any other role; `400` for a quota that exceeds the space available in the StorageSet, so an empty StorageSet cannot be given to a tenant. A tenant with no quota in a StorageSet cannot create volumes in it: volume create returns `404` .
`POST /api/v1/tenants/{tenant_id}/subnets`	The realm segment type is set (`PUT /api/v1/subnets/segment-type`), any `segment_id` you give is unused on the realm (leave it out and the realm assigns one), and the call comes from a platform role directly, not from a session switched into the tenant.	`409` when the segment ID is held by another subnet or the realm has subnets on a different segment type; `403` from a switched-in session.
`GET /api/v1/tenants/{tenant_id}/subnets/{subnet_id}/mount-info`	The subnet exists and a realm node has reported an address into it. Call it with a Platform Admin token that is not switched into a tenant.	`rmlist` stays empty until then (an empty `directors` array with an empty `rmlist` is a legitimate answer); `403` for a Tenant Admin and through the tenant switcher; `503` when the realm manager cannot be reached.

Endpoint	Needs first	Otherwise
`POST /api/v1/tenants/{tenant_id}/subnets/{subnet_id}/attachments`	The subnet exists. Platform roles only.	`404` when the subnet does not exist; `403` for a Tenant Admin; `503` when the realm manager cannot be reached.
`POST /api/v1/storagesets/{storageset_id}/actions`	For `add_storage`: list the nodes in `dataserials`, each unassigned and with a role of `vhost`, `vosd`, or `snode`.	A call without `dataserials` adds nothing yet returns `204`; a role such as `qnode` is refused. The call returns `204` before the storage is usable, so poll `GET /api/v1/storagesets/{storageset_id}` until `state.status` is `ok`.
`POST /api/v1/volumes`	A token for the tenant (`POST /api/v1/auth/tenant-switch` as a Platform Admin, or a tenant admin); the realm filesystem is mounted; the tenant has a quota in the StorageSet; the tenant has a subnet with an issued address; the StorageSet has enough online disks for the RAID layout; and `hard_quota_bytes` is greater than 0.	`404` for a Platform Admin who has not switched into the tenant (the volume would go to the platform tenant), and also without the StorageSet quota; `503` while not mounted; `409` ("The tenant holds no issued subnet address, so a volume created for it could not be reached. Create a subnet for the tenant and let one of its nodes report an address into it first") before the tenant is `active`; `409` "Not enough online disks in StorageSet to create volume" when the StorageSet has too few online OSDs for the RAID layout (4 for `raid6+`, 3 for `raid5+` and `raid10+`); `400` without a positive `hard_quota_bytes`. A `504` means the volume may still appear, so list volumes before retrying.

Endpoint	Needs first	Otherwise
`DELETE /api/v1/volumes/{volume_id}`	A token for the tenant that owns the volume (`POST /api/v1/auth/tenant-switch` as a Platform Admin, or a tenant admin).	`404` "Volume not found" for a Platform Admin who has not switched into the tenant, the same as for `GET /api/v1/volumes/{volume_id}` . Find the IDs with `GET /volumes?tenant_id={tenant_id}` , which works without switching.
`PATCH /api/v1/tenants/{tenant_id}/users/{user_id}/roles`	Every role in the request is defined for the tenant (`GET /api/v1/tenants/roles`).	`400` for an unknown or reserved role (`platform-support` , `s3-gateway`).
`PATCH /api/v1/tenants/{tenant_id}/s3`	S3 is enabled for the realm (`PUT /api/v1/s3/enabled`) and the tenant S3 configuration exists. To set `enabled` to `true` , `default_prefix` must name an existing volume of the tenant.	`409` when the realm is not enabled. The call is refused when the prefix does not name an existing volume.
`POST /api/v1/tenants/{tenant_id}/s3/instances`	S3 is enabled for the tenant, the tenant has a subnet, and the tenant has no instance on that node yet.	`409` when the tenant already has an instance on that node.
`POST /api/v1/tenants/{tenant_id}/s3/instances/{iid}/start`	The instance exists.	A `202` means the desired state is recorded, not reached. Poll `observed` on `GET /api/v1/tenants/{tenant_id}/s3/instances/{iid}` until it is `running` .
`GET /api/v1/tenants/{tenant_id}/s3/buckets`	A running S3 gateway instance for the tenant.	`503` , never an empty list. It does not clear on retry until an instance is started.
`POST /api/v1/tenants/{tenant_id}/s3/buckets`	A running S3 gateway instance, and an `owner` that is the access key of an existing PanFS user in the tenant.	`503` with no running instance, which does not clear on retry. A `504` means the bucket may exist, so list buckets before retrying.

Endpoint	Needs first	Otherwise
`DELETE /api/v1/tenants/{tenant_id}/s3/buckets/{name}`	A running S3 gateway instance for the tenant.	`503` with no running instance; `409` for a non-empty bucket. A `504` means the bucket may have been deleted.
`DELETE /api/v1/tenants/{tenant_id}`	Nothing left in the tenant: delete its S3 instances (`POST /api/v1/tenants/{tenant_id}/s3/instances/delete-all`), then its volumes, then its subnets.	`409` for any leftover resource, including an S3 gateway instance.
`DELETE /api/v1/tenants/{tenant_id}/s3/buckets/{name}/policy`	A running S3 gateway instance for the tenant.	`503` with no running instance. A `504` means the policy may have been removed; the call is safe to repeat.
`POST /api/v1/realm/log-collections`	No other collection running.	`409` while one runs, including a `collect_logs` started on the Director outside the API; `503` if `collect_logs` is not installed or the collection directory is unusable.
`GET /api/v1/realm/log-collections/{collection_id}/archive`	A collection whose `status` is `succeeded` .	`409` while it is still running; `404` if it failed or its archive was removed.

3. Endpoint Summary

Every endpoint in this guide, one row each, grouped and ordered the same way as Chapters 4 through 15 below. Each path links directly to that endpoint's full reference.

Chapter 4. Authentication

Path	Summary	Method
`/api/v1/auth/change-password`	Change own password	`POST`
`/api/v1/auth/login`	Login	`POST`
`/api/v1/auth/me`	Current user identity	`GET`
`/api/v1/auth/refresh`	Refresh tokens	`POST`
`/api/v1/auth/tenant-switch`	Switch into a tenant (Platform Admin only)	`POST`
`/api/v1/auth/tokens`	List your API tokens	`GET`
`/api/v1/auth/tokens`	Issue an API token	`POST`
`/api/v1/auth/tokens/{token_id}`	Read one of your API tokens	`GET`
`/api/v1/auth/tokens/{token_id}`	Revoke an API token	`DELETE`

Chapter 5. Realm and Platform

Path	Summary	Method
`/api/v1/about`	About	`GET`
`/api/v1/realm`	Get realm	`GET`
`/api/v1/realm/actions`	Perform realm action	`POST`
`/api/v1/realm/log-collections`	List realm log collections	`GET`
`/api/v1/realm/log-collections`	Start a realm log collection	`POST`
`/api/v1/realm/log-collections/{collection_id}`	Get a realm log collection	`GET`
`/api/v1/realm/log-collections/{collection_id}`	Delete a realm log collection	`DELETE`

Path	Summary	Method
`/api/v1/realm/log-collections/{collection_id}/archive`	Download a realm log collection archive	`GET`

Chapter 6. Tenants

Path	Summary	Method
`/api/v1/tenants`	List tenants	`GET`
`/api/v1/tenants`	Create tenant	`POST`
`/api/v1/tenants/archived`	List archived tenants	`GET`
`/api/v1/tenants/roles`	List roles available on the current tenant	`GET`
`/api/v1/tenants/{tenant_id}`	Get tenant info	`GET`
`/api/v1/tenants/{tenant_id}`	Update tenant	`PATCH`
`/api/v1/tenants/{tenant_id}`	Delete tenant	`DELETE`
`/api/v1/tenants/{tenant_id}/storagesets`	List tenant storageset quota allocations	`GET`
`/api/v1/tenants/{tenant_id}/storagesets/{storageset_id}`	Get tenant storageset	`GET`
`/api/v1/tenants/{tenant_id}/storagesets/{storageset_id}`	Assign or resize tenant storageset	`PUT`
`/api/v1/tenants/{tenant_id}/storagesets/{storageset_id}`	Unassign tenant storageset	`DELETE`

Chapter 7. Tenant Users and Roles

Path	Summary	Method
`/api/v1/tenants/{tenant_id}/users`	List tenant users	`GET`
`/api/v1/tenants/{tenant_id}/users`	Create tenant user	`POST`
`/api/v1/tenants/{tenant_id}/users/{user_id}`	Get tenant user	`GET`
`/api/v1/tenants/{tenant_id}/users/{user_id}`	Update tenant user	`PATCH`
`/api/v1/tenants/{tenant_id}/users/{user_id}`	Delete tenant user	`DELETE`

Path	Summary	Method
`/api/v1/tenants/{tenant_id}/users/{user_id}/roles`	List tenant user roles	`GET`
`/api/v1/tenants/{tenant_id}/users/{user_id}/roles`	Set tenant user roles	`PATCH`

Chapter 8. PanFS Users

Path	Summary	Method
`/api/v1/tenants/{tenant_id}/fs-users`	List PanFS users	`GET`
`/api/v1/tenants/{tenant_id}/fs-users`	Create PanFS user	`POST`
`/api/v1/tenants/{tenant_id}/fs-users/{fs_user_id}`	Get PanFS user	`GET`
`/api/v1/tenants/{tenant_id}/fs-users/{fs_user_id}`	Update PanFS user	`PATCH`
`/api/v1/tenants/{tenant_id}/fs-users/{fs_user_id}`	Delete PanFS user	`DELETE`
`/api/v1/tenants/{tenant_id}/fs-users/{fs_user_id}/s3/secret`	Reveal a PanFS user S3 secret	`GET`
`/api/v1/tenants/{tenant_id}/fs-users/{fs_user_id}/s3/secret/rotate`	Rotate a PanFS user S3 secret	`POST`

Chapter 9. Service Accounts

Path	Summary	Method
`/api/v1/service-accounts`	List service accounts	`GET`
`/api/v1/service-accounts`	Create a service account	`POST`
`/api/v1/service-accounts/{service_account_id}`	Get a service account	`GET`
`/api/v1/service-accounts/{service_account_id}`	Update a service account	`PATCH`
`/api/v1/service-accounts/{service_account_id}`	Delete a service account	`DELETE`

Path	Summary	Method
`/api/v1/service-accounts/{service_account_id}/roles`	List service account roles	`GET`
`/api/v1/service-accounts/{service_account_id}/roles`	Set service account roles	`PATCH`
`/api/v1/service-accounts/{service_account_id}/tokens`	List service account keys	`GET`
`/api/v1/service-accounts/{service_account_id}/tokens`	Issue a service account key	`POST`
`/api/v1/service-accounts/{service_account_id}/tokens/{token_id}`	Revoke a service account key	`DELETE`

Chapter 10. Volumes

Path	Summary	Method
`/api/v1/volumes`	List volumes	`GET`
`/api/v1/volumes`	Create a new volume	`POST`
`/api/v1/volumes/{volume_id}`	Get volume info	`GET`
`/api/v1/volumes/{volume_id}`	Update volume	`PATCH`
`/api/v1/volumes/{volume_id}`	Delete volume	`DELETE`
`/api/v1/volumes/{volume_id}/actions`	Perform action on volume	`POST`
`/api/v1/volumes/{volume_id}/space`	Get volume space	`GET`

Chapter 11. StorageSets

Path	Summary	Method
`/api/v1/storagesets`	List storagesets	`GET`
`/api/v1/storagesets`	Create StorageSet	`POST`
`/api/v1/storagesets/{storageset_id}`	Get StorageSet details	`GET`

Path	Summary	Method
`/api/v1/storagesets/{storageset_id}`	Update StorageSet	`PATCH`
`/api/v1/storagesets/{storageset_id}`	Delete StorageSet	`DELETE`
`/api/v1/storagesets/{storageset_id}/actions`	Perform StorageSet Action	`POST`
`/api/v1/storagesets/{storageset_id}/space`	Get StorageSet space	`GET`

Chapter 12. Subnets and Attachments

Path	Summary	Method
`/api/v1/subnets`	List every subnet in the realm	`GET`
`/api/v1/subnets/segment-type`	Get the network segment type new subnets are placed on	`GET`
`/api/v1/subnets/segment-type`	Set the network segment type new subnets are placed on	`PUT`
`/api/v1/tenants/{tenant_id}/subnets`	List a tenant's subnets	`GET`
`/api/v1/tenants/{tenant_id}/subnets`	Create a new subnet	`POST`
`/api/v1/tenants/{tenant_id}/subnets/{subnet_id}`	Get subnet info	`GET`
`/api/v1/tenants/{tenant_id}/subnets/{subnet_id}`	Update subnet	`PATCH`
`/api/v1/tenants/{tenant_id}/subnets/{subnet_id}`	Delete subnet	`DELETE`
`/api/v1/tenants/{tenant_id}/subnets/{subnet_id}/mount-info`	Get the mount targets on a subnet	`GET`
`/api/v1/tenants/{tenant_id}/subnets/{subnet_id}/attachments`	List the subnet's host attachments	`GET`
`/api/v1/tenants/{tenant_id}/subnets/{subnet_id}/attachments`	Attach a host to the subnet's tenant network	`POST`
`/api/v1/tenants/{tenant_id}/subnets/{subnet_id}/attachments/{attachment_id}`	Get attachment detail	`GET`

Path	Summary	Method
`/api/v1/tenants/{tenant_id}/subnets/{subnet_id}/attachments/{attachment_id}`	Detach a host from the subnet	`DELETE`

Chapter 13. Nodes and Hardware

Path	Summary	Method
`/api/v1/nodes`	List nodes	`GET`
`/api/v1/nodes/{dataserial}`	Get node info	`GET`
`/api/v1/nodes/{dataserial}/actions`	Perform action on node	`POST`
`/api/v1/nodes/{dataserial}/drain`	Get OSD drain status	`GET`
`/api/v1/nodes/{dataserial}/drain`	Start or stop OSD drain	`POST`
`/api/v1/hardware`	List hardware models	`GET`
`/api/v1/hardware/{dataserial}/actions`	Perform an action on a hardware component	`POST`
`/api/v1/hardware/{model_id}`	Get hardware model info	`GET`

Chapter 14. S3 Gateway

Path	Summary	Method
`/api/v1/tenants/{tenant_id}/s3`	Get S3 config	`GET`
`/api/v1/tenants/{tenant_id}/s3`	Create S3 config	`PUT`
`/api/v1/tenants/{tenant_id}/s3`	Update S3 config	`PATCH`
`/api/v1/tenants/{tenant_id}/s3`	Delete S3 config	`DELETE`
`/api/v1/tenants/{tenant_id}/s3/tls/renew`	Renew the realm-generated S3 edge cert	`POST`
`/api/v1/s3/enabled`	Get realm-wide S3 enable flag	`GET`
`/api/v1/s3/enabled`	Set realm-wide S3 enable flag	`PUT`
`/api/v1/s3/prefix`	Get realm-wide S3 prefix path	`GET`
`/api/v1/s3/prefix`	Set realm-wide S3 prefix path	`PUT`

Path	Summary	Method
`/api/v1/tenants/{tenant_id}/s3/buckets`	List S3 buckets	`GET`
`/api/v1/tenants/{tenant_id}/s3/buckets`	Create S3 bucket	`POST`
`/api/v1/tenants/{tenant_id}/s3/buckets/{name}`	Delete S3 bucket	`DELETE`
`/api/v1/tenants/{tenant_id}/s3/buckets/{name}/policy`	Delete S3 bucket policy	`DELETE`
`/api/v1/tenants/{tenant_id}/s3/instances`	List a tenant's S3 gateway instances	`GET`
`/api/v1/tenants/{tenant_id}/s3/instances`	Create an S3 gateway instance	`POST`
`/api/v1/tenants/{tenant_id}/s3/instances/delete-all`	Delete every instance of the tenant and wait for teardown	`POST`
`/api/v1/tenants/{tenant_id}/s3/instances/start-all`	Start every instance of the tenant	`POST`
`/api/v1/tenants/{tenant_id}/s3/instances/stop-all`	Stop every instance of the tenant	`POST`
`/api/v1/tenants/{tenant_id}/s3/instances/{iid}`	Show an S3 gateway instance (bare-iid fan-out or specific node)	`GET`
`/api/v1/tenants/{tenant_id}/s3/instances/{iid}`	Delete an S3 gateway instance (fan-out if node omitted; requires confirm)	`DELETE`
`/api/v1/tenants/{tenant_id}/s3/instances/{iid}/start`	Start an S3 gateway instance (fan-out if node omitted)	`POST`
`/api/v1/tenants/{tenant_id}/s3/instances/{iid}/stop`	Stop an S3 gateway instance (fan-out if node omitted)	`POST`

Chapter 15. Observability

Path	Summary	Method
`/api/v1/observability/alerts`	Alerts snapshot (keyed set)	`GET`
`/api/v1/observability/alerts/stream`	Alerts SSE stream (added/resolved)	`GET`
`/api/v1/observability/audit`	Audit trail snapshot (last N)	`GET`
`/api/v1/observability/audit/stream`	Audit SSE stream (append deltas)	`GET`

Path	Summary	Method
`/api/v1/observability/audit/{request_id}`	Audit record with failure diagnostics	`GET`
`/api/v1/observability/events`	PanFS events snapshot (last N)	`GET`
`/api/v1/observability/events/stream`	PanFS events SSE stream (append deltas)	`GET`
`/api/v1/observability/logs`	Logs snapshot (last N)	`GET`
`/api/v1/observability/logs/stream`	Logs SSE stream (append deltas)	`GET`
`/api/v1/observability/logs/{logsql_endpoint}`	VictoriaLogs /select/logsql proxy	`GET`
`/api/v1/observability/metrics/query`	Prometheus instant query proxy	`GET`
`/api/v1/observability/metrics/query_range`	Prometheus range query proxy	`GET`
`/api/v1/observability/metrics/stream`	Grouped metric SSE stream (append deltas for several keys)	`GET`
`/api/v1/observability/metrics/targets`	List metrics targets	`GET`
`/api/v1/observability/metrics/{key}`	Metric snapshot (range window)	`GET`
`/api/v1/observability/metrics/{key}/stream`	Metric SSE stream (append deltas)	`GET`
`/api/v1/observability/storagesets/status`	Storage-set alert status	`GET`

4. Authentication

Session login, identity, password changes, Platform Admin tenant switching, and long-lived API tokens for non-interactive callers (CI, CSI drivers, monitoring collectors).

Endpoint Summary

Path	Summary	Method
`/api/v1/auth/change-password`	Change own password	`POST`
`/api/v1/auth/login`	Login	`POST`
`/api/v1/auth/me`	Current user identity	`GET`
`/api/v1/auth/refresh`	Refresh tokens	`POST`
`/api/v1/auth/tenant-switch`	Switch into a tenant (Platform Admin only)	`POST`
`/api/v1/auth/tokens`	List your API tokens	`GET`
`/api/v1/auth/tokens`	Issue an API token	`POST`
`/api/v1/auth/tokens/{token_id}`	Read one of your API tokens	`GET`
`/api/v1/auth/tokens/{token_id}`	Revoke an API token	`DELETE`

`POST /api/v1/auth/change-password`

Change own password

Change the signed-in user's own password, verifying the current one. Acts on the caller's identity from the token. Not available for accounts managed by an external identity provider. For administrative resets, use `PATCH /tenants/{tenant_id}/users/{user_id}`.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Request Body (`application/json` , required)

Schema: `ChangePasswordRequest`

Self-service change-password payload. Carries no user/tenant ID - the caller comes from the token.

Field	Type	Required	Default	Example	Description
`current_password`	string (password)	yes	n/a		The caller's current password, verified server-side.
`new_password`	string (password)	yes	n/a		New password. Must contain at least 8 characters with an upper-case letter, a lower-case letter, a digit, and a special character, and must not exceed 72 bytes.

Example

```
{
   "current_password" : "Curr3nt!pw",
   "new_password" : "N3w!password"
}
```

Responses

Status	Description	Schema
204	No content. Password changed successfully.	-

Status	Description	Schema
400	The payload is well-formed but conflicts with the stored account state. `detail` carries a human-readable explanation of which rule was broken.	-
401	Unauthorized. The Bearer token is missing, invalid, or expired.	-
403	Password change is not available for this account (managed by an external identity provider).	-
404	The caller's account no longer exists (e.g. it was deleted after the token was issued).	-
422	The new password is malformed - too short, too long, or missing a required character class. Reported in the standard validation-error shape, pointing at the offending field.	-

`POST /api/v1/auth/login`

Login

Authenticate with username and password and receive a JWT access and refresh token pair. The username names the tenant it belongs to, `<tenant-name>/<user>` (for example `fintech/alice`); an unprefixed name belongs to the platform tenant.

Authentication: not required.

Request Body (`application/x-www-form-urlencoded` , required)

Schema: `LoginForm`

Credentials for `POST /auth/login` , where the username carries the tenant.

Any other form field an OAuth2 client sends is accepted and ignored.

Field	Type	Required	Default	Example	Description
`password`	string (password)	yes	n/a		The user password.
`username`	string	yes	n/a		Username, tenant-qualified as `<tenant-name>/<user>` (for example `fintech/alice`). An unprefixed name belongs to the platform tenant.

Example

```
{
   "password" : "Secret1!",
   "username" : "admin"
}
```

Responses

Status	Description	Schema
200	Token pair issued successfully.	`TokenResponse`
401	Unauthorized. Wrong password, unknown user, or the target tenant could not be resolved. Every case answers with the same body to avoid user and tenant enumeration.	-
422	Validation Error	`HTTPValidationError`

Response Body (200, `TokenResponse`)

Field	Type	Required	Default	Example	Description
`access_token`	string	yes	n/a		Short-lived JWT access token.
`expires_in`	integer	yes	n/a	`3600`	Access token lifetime in seconds (60 minutes by default).
`refresh_token`	string	yes	n/a		Long-lived JWT refresh token (7 days by default).
`token_type`	literal `bearer`	no	`bearer`		

`GET /api/v1/auth/me`

Current user identity

Returns the authenticated user's identity, the effective tenant, and what the caller may do there: the roles they hold and every permission those roles grant, as `resource:action`. The tenant is resolved from the `tid` claim in the JWT, and roles and permissions are read from that tenant's policy - the same one the server enforces against. While a tenant switch is active, `tenant_id` is the tenant being acted in, the access reported is the one held there, and the delegation fields describe the switch - enough for the client to show an "acting as" banner.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Responses

Status	Description	Schema
200	User identity, effective tenant, and the access held in it.	`MeResponse`
401	Unauthorized. The Bearer token is missing, invalid, or expired.	-

Response Body (200, `MeResponse`)

Field	Type	Required	Default	Example	Description
`actor_tenant_id`	string (uuid), nullable	no	`null`	`00000000-0000-0000-0000-000000000000`	Home tenant of the acting user; set only while a tenant switch is active.
`actor_user_id`	string (uuid), nullable	no	`null`	`00000000-0000-0000-0000-000000000001`	The user that asked for the switch; set only while a tenant switch is active.

Field	Type	Required	Default	Example	Description
`is_delegated`	boolean	no	`false`		True while the caller acts in another tenant through a tenant switch. `user_id` is then the platform-support principal, and the delegation fields name the real caller.
`is_platform_admin`	boolean	yes	n/a		True if the authenticated user is the built-in Platform Admin account acting in its own tenant. Informational only - it grants nothing that the platform policy does not already grant. Always false while a tenant switch is active.
`permissions`	array of string	no	n/a	`["volumes:list","volumes:create","users:view"]`	Every permission the caller's roles grant, as `resource:action`, deduplicated and in policy order. This is the effective set the server enforces against, so a client can gate a control on membership alone without knowing which role supplied it.
`roles`	array of string	no	n/a	`["admin"]`	Roles the caller holds in the effective tenant. Empty when the caller is bound to no role there - a role defined by another tenant grants nothing here. The full catalog, with the permissions of every role the tenant defines, is at `GET /tenants/roles`.

Field	Type	Required	Default	Example	Description
`tenant_id`	string (uuid)	yes	n/a	`123e4567-e89b-12d3-a456-426614174000`	Effective tenant UUID. Resolved from the `tid` claim in the JWT.
`user_id`	string (uuid)	yes	n/a	`123e4567-e89b-12d3-a456-426614174000`	The authenticated user identifier (JWT `sub` claim).

Example


```
{
   "is_delegated" : false,
   "is_platform_admin" : false,
   "permissions" : [
      "tenants:view",
      "volumes:list",
      "volumes:view",
      "users:list",
      "users:view",
      "users:update",
      "users:delete",
      "observability:list",
      "observability:view",
      "roles:view",
      "roles:update"
   ],
   "roles" : [
      "admin"
   ],
   "tenant_id" : "123e4567-e89b-12d3-a456-426614174000",
   "user_id" : "123e4567-e89b-12d3-a456-426614174000"
}
```

`POST /api/v1/auth/refresh`

Refresh tokens

Exchange a valid refresh token for a new JWT access and refresh token pair.

Authentication: not required.

Request Body (`application/x-www-form-urlencoded` , required)

Schema: `Body_refresh_api_v1_auth_refresh_post`

Field	Type	Required	Default	Example	Description
`refresh_token`	string	yes	n/a		

Responses

Status	Description	Schema
200	New token pair issued successfully.	`TokenResponse`
401	Unauthorized. The refresh token is invalid, expired, or is not a refresh token.	-
422	Validation Error	`HTTPValidationError`

Response Body (200, `TokenResponse`)

Field	Type	Required	Default	Example	Description
`access_token`	string	yes	n/a		Short-lived JWT access token.
`expires_in`	integer	yes	n/a	`3600`	Access token lifetime in seconds (60 minutes by default).
`refresh_token`	string	yes	n/a		Long-lived JWT refresh token (7 days by default).
`token_type`	literal `bearer`	no	`bearer`		

`POST /api/v1/auth/tenant-switch`

Switch into a tenant (Platform Admin only)

Exchange your current access token for a short-lived token scoped to the requested tenant. Only callers with `tenants:switch` can use this endpoint. The delegated token includes `tid` (target tenant) and `act` (original caller), has no refresh token, and does not change stored RBAC policy.

In Swagger UI, paste the returned `access_token` into `BearerJWT`; logging out of it ends the switch.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Request Body (`application/json` , required)

Schema: `TenantSwitchRequest`

Platform Admin request to exchange the current access token for a target-tenant one.

Field	Type	Required	Default	Example	Description
`tenant_id`	string (uuid)	yes	n/a	`e97b94d6-9cc1-4325-a267-1064ae538ccd`	Target tenant to act in.
`ttl_minutes`	integer, nullable	no	`null`	`60` , `480`	Requested lifetime in minutes. Clamped server-side to the supported range (30 minutes - 12 hours); the granted lifetime is returned in `expires_in` .

Example

```
{
   "tenant_id" : "e97b94d6-9cc1-4325-a267-1064ae538ccd",
   "ttl_minutes" : 60
}
```

Responses

Status	Description	Schema
200	Token scoped to the target tenant.	`DelegatedTokenResponse`
401	Unauthorized. The Bearer token is missing, invalid, or expired.	-
403	Forbidden. The caller lacks `tenants:switch` in its effective tenant policy. Only the **platform-admin** role holds it - so a switched-in token, which holds the target **platform-support** role, cannot switch again.	-
404	The target tenant does not exist.	-
409	The target tenant is the caller's own tenant, so there is nothing to switch into.	-
422	Validation Error	`HTTPValidationError`

Response Body (200, `DelegatedTokenResponse`)

Field	Type	Required	Default	Example	Description
`access_token`	string	yes	n/a		Short-lived JWT scoped to the target tenant.
`expires_in`	integer	yes	n/a	`3600`	Granted token lifetime in seconds.
`tenant_id`	string (uuid)	yes	n/a		Target tenant the token acts in (`tid` claim).
`token_type`	literal `bearer`	no	`bearer`		

`GET /api/v1/auth/tokens`

List your API tokens

Return the caller's own API tokens, newest first. Carries no secret and no digest - the token itself is unrecoverable after issuance. Revoked and expired tokens stay listed, so a revocation stays visible; neither holds a slot or its name any more.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`active_only`	query	boolean	no	Return only records whose `state` is `active` .

Responses

Status	Description	Schema
200	The caller's tokens.	`ApiTokenListResponse`
401	Unauthorized. The Bearer token is missing, invalid, or expired.	-
422	Validation Error	`HTTPValidationError`

Response Body (200, `ApiTokenListResponse`)

Field	Type	Required	Default	Example	Description
`tokens`	array of `ApiTokenResponse` object	yes	n/a		Token metadata, newest first.
`total`	integer	yes	n/a		Number of tokens returned.

`POST /api/v1/auth/tokens`

Issue an API token

Issue a long-lived credential for a non-interactive caller - a CLI, a CSI driver, a CI job, a monitoring collector. The token authenticates as you, carrying the roles you hold now and never exceeding what you hold at the time of each request.

The token is returned once, in this response, and cannot be retrieved again.

Requires an interactive session: an API token cannot issue another, and a tenant switch cannot leave one behind.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Request Body (`application/json` , required)

Schema: `ApiTokenCreateRequest`

Request to issue a long-lived API token for the caller.

Field	Type	Required	Default	Example	Description
`description`	string, nullable	no	`null`		What this token is for. Optional; `name` remains the handle an operator identifies it by.
`expires_in_days`	integer	yes	n/a	`30` , `90` , `180` , `365`	Lifetime in days. Required, and must name one of the offered values (30, 90, 180, 365), so a long lifetime is always a deliberate choice.
`name`	string	yes	n/a		Name for this token, unique among the caller's usable tokens. An operator identifies a token by it. A revoked or expired token releases its name, so a rotated credential can keep the name it had.

Example

```
{
   "description" : "Nightly backup runner",
   "expires_in_days" : 90,
   "name" : "nightly-backup@ci-runner-04"
}
```

Responses

Status	Description	Schema
201	Token issued. The only response that contains the secret.	`ApiTokenCreatedResponse`
400	Bad request.	-
401	Unauthorized. The Bearer token is missing, invalid, or expired.	-
403	Forbidden. The caller may not issue a token here: the credential presented is itself an API token, the session is a tenant switch, the role is not permitted to create tokens, or the request came from a browser application.	-
409	Conflict. The caller already has a usable token with this name.	-
422	Validation Error	`HTTPValidationError`

Response Body (201, `ApiTokenCreatedResponse`)

Field	Type	Required	Default	Example	Description
`created_at`	string (date-time)	yes	n/a		When the secret was created. Never changes.
`created_by`	`Principal` object	yes	n/a		Who issued the token.
`description`	string, nullable	no	`null`		What this token is for, if a description was given.
`expires_at`	string (date-time)	yes	n/a		Absolute expiry.

Field	Type	Required	Default	Example	Description
`id`	string	yes	n/a		Public token identifier. Appears in audit events; not a secret.
`name`	string	yes	n/a		The handle given at issuance, unique among the owner's usable tokens.
`revoked_at`	string (date-time), nullable	no	`null`		When the token was revoked, if it was. The only lifecycle fact the record stores.
`revoked_by`	`Principal` object, nullable	no	`null`		Who revoked the token, if it was.
`roles`	array of string, nullable	no	`null`		The role ceiling held in the record: the owner's own roles as at issuance. Not chosen - a token carries what its owner held, and never more than they hold now. Null for a service account key, which carries no ceiling: it resolves whatever the account holds at the time of each request.
`state`	`ApiTokenState` (`active` , `disabled`)	yes	n/a		Whether the token still authenticates. Computed per response, never stored.
`state_reason`	`ApiTokenStateReason` (`expired` , `revoked`), nullable	no	`null`		Why the token is disabled. Null while it is active. Revocation outranks expiry.

Field	Type	Required	Default	Example	Description
`token`	string	yes	n/a	`vdt1_6XuU1pyDS0abcDEF12345w_9fK3mZq7Xw2Lb1cD_hT8n...7bQ2xM`	The token. Returned once and never retrievable again; a lost token must be replaced.
`warning`	string	no	`Store this token now. It cannot be retrieved again.`		Reminder that the secret is shown exactly once.

`GET /api/v1/auth/tokens/{token_id}`

Read one of your API tokens

Metadata for one token you own. A token belonging to another user answers the same as one that does not exist, so the endpoint cannot be used to probe another inventory.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`token_id`	path	string	yes	Public token identifier, as returned at issuance.

Responses

Status	Description	Schema
200	Token metadata.	`ApiTokenResponse`
401	Unauthorized. The Bearer token is missing, invalid, or expired.	-
404	Not found. No such token, or it belongs to another user - both answer the same so an inventory cannot be probed.	-
422	Validation Error	`HTTPValidationError`

Response Body (200, `ApiTokenResponse`)

Field	Type	Required	Default	Example	Description
`created_at`	string (date-time)	yes	n/a		When the secret was created. Never changes.
`created_by`	`Principal` object	yes	n/a		Who issued the token.
`description`	string, nullable	no	`null`		What this token is for, if a description was given.

Field	Type	Required	Default	Example	Description
`expires_at`	string (date-time)	yes	n/a		Absolute expiry.
`id`	string	yes	n/a		Public token identifier. Appears in audit events; not a secret.
`name`	string	yes	n/a		The handle given at issuance, unique among the owner's usable tokens.
`revoked_at`	string (date-time), nullable	no	`null`		When the token was revoked, if it was. The only lifecycle fact the record stores.
`revoked_by`	`Principal` object, nullable	no	`null`		Who revoked the token, if it was.
`roles`	array of string, nullable	no	`null`		The role ceiling held in the record: the owner's own roles as at issuance. Not chosen - a token carries what its owner held, and never more than they hold now. Null for a service account key, which carries no ceiling: it resolves whatever the account holds at the time of each request.
`state`	`ApiTokenState` (`active`, `disabled`)	yes	n/a		Whether the token still authenticates. Computed per response, never stored.

Field	Type	Required	Default	Example	Description
`state_reason`	`ApiTokenStateReason` (`expired`, `revoked`), nullable	no	`null`		Why the token is disabled. Null while it is active. Revocation outranks expiry.

`DELETE /api/v1/auth/tokens/{token_id}`

Revoke an API token

Revoke one token you own. It stops authenticating on the very next request - the record is read on every request, so there is no cache to wait out.

Idempotent: revoking an already-revoked token succeeds and keeps the first revocation on record. The record is retained as a tombstone, so a later use of the token reports as revoked rather than as unknown.

Unlike issuance, this is open to a token-authenticated caller, so a compromised client can always burn its own credential.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`token_id`	path	string	yes	Public token identifier, as returned at issuance.

Responses

Status	Description	Schema
204	No content. The token no longer authenticates.	-
401	Unauthorized. The Bearer token is missing, invalid, or expired.	-
404	Not found. No such token, or it belongs to another user - both answer the same so an inventory cannot be probed.	-
422	Validation Error	`HTTPValidationError`

5. Realm and Platform

This chapter covers About and Realm.

5.1 About

Endpoint Summary

Path	Summary	Method
`/api/v1/about`	About	`GET`

`GET /api/v1/about`

About

Return service name, version, and PanFS version.

Nothing sits between the route and its two sources: there is no store to adapt and no invariant to guard. The version file is read per request, so the endpoint follows the realm software without a restart.

Authentication: not required.

Responses

Status	Description	Schema
200	Successful Response	`AboutResponse`

Response Body (200, `AboutResponse`)

Field	Type	Required	Default	Example	Description
`name`	string	yes	n/a	`PanAPI`	Service name
`panfs_version`	string, nullable	no	`null`	`12.0.0`	PanFS realm software version read from the local filesystem; null when unavailable
`version`	string	yes	n/a	`1.2.3`	Service version, derived from the package metadata at build time

5.2 Realm

Endpoint Summary

Path	Summary	Method
`/api/v1/realm`	Get realm	`GET`
`/api/v1/realm/actions`	Perform realm action	`POST`
`/api/v1/realm/log-collections`	List realm log collections	`GET`

Path	Summary	Method
`/api/v1/realm/log-collections`	Start a realm log collection	`POST`
`/api/v1/realm/log-collections/{collection_id}`	Get a realm log collection	`GET`
`/api/v1/realm/log-collections/{collection_id}`	Delete a realm log collection	`DELETE`
`/api/v1/realm/log-collections/{collection_id}/archive`	Download a realm log collection archive	`GET`

`GET /api/v1/realm`

Get realm

Returns the current realm.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Responses

Status	Description	Schema
200	Successful response containing the realm	`RealmResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
500	Internal server error.	-

Response Body (200, `RealmResponse`)

Field	Type	Required	Default	Example	Description
`realm`	`Realm` object	yes	n/a		Realm details
`server_tzone`	string	yes	n/a	`America/Los_Angeles` , `UTC`	IANA timezone name of the API server's host (e.g. 'America/Los_Angeles').

Example


```
{
   "realm" : {
      "boot_epoch_seconds" : 0,
      "clientmode_current" : "balanced",
      "clientmode_next" : "balanced",
      "data_generation" : 6,
      "heartbeat_failure_grace_seconds" : 30,
      "heartbeat_period_seconds" : 30,
      "id" : "a538deadbeef0001889abcdef1234567",
      "locked" : false,
      "log_sizing" : "12g",
      "mem_sizing_current" : 1,
      "mem_sizing_next" : 1,
      "mia_timer" : 360,
      "name" : "exp-aic-97-5",
      "panfs_mounted" : true,
      "president_flags" : [
         "settled_storage",
         "settled_mgr"
      ],
      "state" : "online",
      "state_flags" : [],
      "volume_layout_type_default" : "raid6+",
      "volume_max_vols_realm" : 1200,
      "volume_recovery_priority_default" : 50
   },
   "server_tzone" : "America/Los_Angeles"
}
```

`POST /api/v1/realm/actions`

Perform realm action

Performs a realm-scoped action. Supported actions: “rehome_root” - reassign the realm root volume to a different StorageSet; “reboot” - reboot the realm; “online” - bring the realm online; “offline” - take the realm offline (mode: “paused” for graceful drain, “unavailable” to make volumes inaccessible without draining); “lock” - prevent new nodes from joining via PanDHCP; “unlock” - re-enable PanDHCP joins. Lock and unlock are idempotent; “move_service” - relocate a service (volume service, PM, ...) to a target Director Node (service_type “primary” performs a live failover, “backup” moves the standby).

Authentication: required (`PasswordLogin` or `BearerJWT`).

Request Body (`application/json` , required)

One of the following, selected by `action` :

`action: lock` - `RealmLockActionRequest`

Field	Type	Required	Default	Example	Description
`action`	literal `lock`	yes	n/a		The action to perform on the realm.
`params`	null (not used)	no	n/a		

`action: move_service` - `RealmMoveServiceActionRequest`

Field	Type	Required	Default	Example	Description
`action`	literal `move_service`	yes	n/a		The action to perform on the realm.
`params`	`RealmMoveServiceActionParams` object	yes	n/a		Parameters identifying the service to move and its destination node.

action: offline - `RealmOfflineActionRequest`

Field	Type	Required	Default	Example	Description
`action`	literal `offline`	yes	n/a		The action to perform on the realm.
`params`	`RealmOfflineActionParams` object	no	n/a		Parameters for taking the realm offline.

action: online - `RealmOnlineActionRequest`

Field	Type	Required	Default	Example	Description
`action`	literal `online`	yes	n/a		The action to perform on the realm.
`params`	null (not used)	no	n/a		

action: reboot - `RealmRebootActionRequest`

Field	Type	Required	Default	Example	Description
`action`	literal `reboot`	yes	n/a		The action to perform on the realm.
`params`	null (not used)	no	n/a		

action: rehome_root - `RealmRehomeRootActionRequest`

Field	Type	Required	Default	Example	Description
`action`	literal `rehome_root`	yes	n/a		The action to perform on the realm.
`params`	`RealmRehomeRootActionParams` object	yes	n/a		Parameters required to reassign the realm root volume to a different StorageSet.

action: unlock - `RealmUnlockActionRequest`

Field	Type	Required	Default	Example	Description
`action`	literal `unlock`	yes	n/a		The action to perform on the realm.
`params`	null (not used)	no	n/a		

Responses

Status	Description	Schema
204	Action performed successfully on the realm. No content is returned.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
409	Conflict. Action conflicts with current system state.	-
422	Validation error.	-
500	Internal server error.	-

`GET /api/v1/realm/log-collections`

List realm log collections

Lists running and finished collections, newest first.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Responses

Status	Description	Schema
200	The collections, newest first.	`LogCollectionListResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
500	Internal server error.	-

Response Body (200, `LogCollectionListResponse`)

Field	Type	Required	Default	Example	Description
`collections`	array of `LogCollectionResponse` object	yes	n/a		Running and finished collections, newest first.

`POST /api/v1/realm/log-collections`

Start a realm log collection

Starts collecting logs, system databases, and corefiles from the realm's nodes with `collect_logs`. No body, or an empty one, collects everything from every node (`collect_logs all`). The body selects nodes (`scope` , `hosts` , `storagesets`) and leaves content out (`include_cores` , `include_databases` , `include_sed` , `run_diagnostics` , `age`). Poll the `Location` URL until `status` is `succeeded` or `failed` . One collection runs at a time; a second start gets 409, as does one while `collect_logs` runs on this Director outside the API. The server keeps finished collections for a limited time and number, so download the archive soon.

Before you call this. Needs: No other collection running. Otherwise: `409` while one runs, including a `collect_logs` started on the Director outside the API; `503` if `collect_logs` is not installed or the collection directory is unusable.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Request Body (`application/json` , optional)

Schema: `LogCollectionCreateRequest`

What to collect, and from which nodes. An empty body collects everything from every node.

Field	Type	Required	Default	Example	Description
`age`	`LogAgeRequest` object, nullable	no	`null`		Skip rotated `/var/log` files and corefiles older than this. Default: no age limit.
`hosts`	array of string	no	n/a		Nodes to collect from, by IPv4 address, serial, or name (e.g. `VCH-1,1`). Adds to `scope` .
`include_cores`	boolean	no	`true`		Collect corefiles and executables. Stack files always are.
`include_databases`	boolean	no	`true`		Collect the system databases (sanitised copies).

Field	Type	Required	Default	Example	Description
`include_sed`	boolean	no	`true`		Collect the self-encrypting drive (SED) event log, `/pan/var/lib/pan_sed/event.log` (ASU nodes only).
`node_timeout_seconds`	integer, nullable	no	`null`		Seconds each node may take to collect, and to copy back (collect_logs `timeout`). Default 900; 0 means no limit. The server also limits the whole run.
`run_diagnostics`	boolean	no	`true`		Run the diagnostic tools on each node before collecting.
`scope`	`LogCollectionScope` (`all`, `repset`, `directors`, `storage`), nullable	no	`null`		A whole class of nodes: `all`, `repset` (the Directors in the replication set), `directors` or `storage`. Defaults to `all` when neither `hosts` nor `storagesets` is given. `all` cannot be combined with them, because it already covers every node.
`storagesets`	array of string	no	n/a		StorageSets whose nodes to collect from, by name (e.g. `Set 1`). Adds to `scope`.
`wait_for_savecore`	boolean	no	`false`		Wait for a running savecore to finish first.

Responses

Status	Description	Schema
202	Accepted. The collection runs in the background. Poll the `Location` URL until `status` is `succeeded` or `failed`, then download the archive from `archive_url`.	`LogCollectionResponse`

Status	Description	Schema
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
409	Conflict. A collection is already running, and only one runs at a time. That includes a `collect_logs` started on this Director outside the API, by hand or left by an API that was killed.	-
422	Validation error.	-
500	Internal server error.	-
503	Service unavailable. Nothing was started. Either collect_logs is not installed on this Director, or the collection directory is unusable or short of free space.	-

Response Body (202, `LogCollectionResponse`)

Field	Type	Required	Default	Example	Description
`archive_sha256`	string, nullable	no	`null`	`0f26932a2db4c86847bbf2b28e8f38ef8e9fd1b802e9e228471c13cd82db208c`	SHA-256 of the archive, in hex. The download sends it as its `ETag`.
`archive_size_bytes`	integer, nullable	no	`null`		Size of the archive, once it is ready.
`archive_url`	string, nullable	yes	n/a		Where to download the archive, once it is ready.
`duration_seconds`	number, nullable	yes	n/a		Seconds from start to end, once the run has ended.
`error`	string, nullable	no	`null`		Why the run failed.

Field	Type	Required	Default	Example	Description
`failed_hosts`	array of string	no	n/a		Addresses of the `hosts` that failed. A run with some failed hosts still succeeds.
`finished_at`	string (date-time), nullable	no	`null`		When the run ended (UTC); the archive name carries this time.
`host_count`	integer, nullable	no	`null`		Nodes the run collects from, once it has listed them.
`hosts`	array of `CollectedHostResponse` object	no	n/a		Every node the run collected from, once it has ended.
`id`	string (uuid)	yes	n/a		Identifier of the run.
`options`	`LogCollectionOptionsResponse` object	yes	n/a		What the run collects, as requested.
`realm`	`RealmRefResponse` object, nullable	no	`null`		The realm, when the RM could be read.
`requested_by`	string, nullable	no	`null`	`platform/admin`	Who started the run: `<tenant>/<principal>`.
`started_at`	string (date-time)	yes	n/a		When the run started (UTC).

Field	Type	Required	Default	Example	Description
`status`	`LogCollectionStatus` (`running`, `succeeded`, `failed`)	yes	n/a		`running` until the run ends, then `succeeded` (an archive is ready) or `failed`.

`GET /api/v1/realm/log-collections/{collection_id}`

Get a realm log collection

Returns one collection: its status, node count, failed nodes, and, once succeeded, the archive size and `archive_url`.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`collection_id`	path	string (uuid)	yes	

Responses

Status	Description	Schema
200	The collection.	`LogCollectionResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Not found. No collection has this ID.	-
422	Validation error.	-
500	Internal server error.	-

Response Body (200, `LogCollectionResponse`)

Field	Type	Required	Default	Example	Description
`archive_sha256`	string, nullable	no	`null`	`0f26932a2db4c86847bbf2b28e8f38ef8e9fd1b802e9e228471c13cd82db208c`	SHA-256 of the archive, in hex. The download sends it as its `ETag`.

Field	Type	Required	Default	Example	Description
`archive_size_bytes`	integer, nullable	no	`null`		Size of the archive, once it is ready.
`archive_url`	string, nullable	yes	n/a		Where to download the archive, once it is ready.
`duration_seconds`	number, nullable	yes	n/a		Seconds from start to end, once the run has ended.
`error`	string, nullable	no	`null`		Why the run failed.
`failed_hosts`	array of string	no	n/a		Addresses of the `hosts` that failed. A run with some failed hosts still succeeds.
`finished_at`	string (date-time), nullable	no	`null`		When the run ended (UTC); the archive name carries this time.
`host_count`	integer, nullable	no	`null`		Nodes the run collects from, once it has listed them.
`hosts`	array of `CollectedHostResponse` object	no	n/a		Every node the run collected from, once it has ended.
`id`	string (uuid)	yes	n/a		Identifier of the run.
`options`	`LogCollectionOptionsResponse` object	yes	n/a		What the run collects, as requested.
`realm`	`RealmRefResponse` object, nullable	no	`null`		The realm, when the RM could be read.
`requested_by`	string, nullable	no	`null`	`platform/admin`	Who started the run: `<tenant>/<principal>`.

Field	Type	Required	Default	Example	Description
`started_at`	string (date-time)	yes	n/a		When the run started (UTC).
`status`	`LogCollectionStatus` (`running`, `succeeded`, `failed`)	yes	n/a		`running` until the run ends, then `succeeded` (an archive is ready) or `failed`.

`DELETE /api/v1/realm/log-collections/{collection_id}`

Delete a realm log collection

Removes a collection and its archive. A running collection is stopped first, and its leftovers are cleaned off the nodes.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`collection_id`	path	string (uuid)	yes	

Responses

Status	Description	Schema
204	Deleted. A running collection was stopped and its leftovers cleaned off the nodes.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Not found. No collection has this ID.	-
422	Validation error.	-
500	Internal server error.	-
503	Service unavailable. The collection was stopped and removed from the list. Its files could not be removed from disk; the API log has the details.	-

`GET /api/v1/realm/log-collections/{collection_id}/archive`

Download a realm log collection archive

Downloads the archive of a succeeded collection as `application/x-tar`. It holds one `.tgz` per node. Downloading needs its own permission, `realm-logs:download`: the archive holds every node's logs and system databases.

To resume an interrupted download, send `Range: bytes=<bytes you have>-` with `If-Range: <ETag>`, and append only a `206` whose `Content-Range` starts where you asked. A `200` means the archive changed: start again. `HEAD` on this URL gives the size and `ETag` without the body.

Before you call this. Needs: A collection whose `status` is `succeeded`. Otherwise: `409` while it is still running; `404` if it failed or its archive was removed.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`collection_id`	path	string (uuid)	yes	

Responses

Status	Description	Schema
200	The archive: a tar with one `<node ip>/logs_<time>.tgz` per node, plus `manifest.json`, `collect_logs_cmdline.txt` and `ip_map.txt`. Check it with `sha256sum` against the `ETag`.	string
206	Partial content: the bytes the `Range` header asked for. Append them only when `Content-Range` starts at the byte you asked for.	string
400	Bad request: the `Range` header is malformed. The body is a plain-text explanation, not JSON; its wording depends on the server version.	string
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-

Status	Description	Schema
404	Not found. No such collection, or it has no archive: it failed, or the archive was removed.	-
409	Conflict. The collection is still running; its archive is not ready yet.	-
416	Range not satisfiable: the `Range` starts at or past the end of the archive. A partial download of that size is already complete, or stale. The body is plain text, not JSON.	string
422	Validation error.	-
500	Internal server error.	-

6. Tenants

Creating, listing, and configuring the tenants that partition a realm, including each tenant's StorageSet quota allocations.

Endpoint Summary

Path	Summary	Method
`/api/v1/tenants`	List tenants	`GET`
`/api/v1/tenants`	Create tenant	`POST`
`/api/v1/tenants/archived`	List archived tenants	`GET`
`/api/v1/tenants/roles`	List roles available on the current tenant	`GET`
`/api/v1/tenants/{tenant_id}`	Get tenant info	`GET`
`/api/v1/tenants/{tenant_id}`	Update tenant	`PATCH`
`/api/v1/tenants/{tenant_id}`	Delete tenant	`DELETE`
`/api/v1/tenants/{tenant_id}/storagesets`	List tenant storageset quota allocations	`GET`
`/api/v1/tenants/{tenant_id}/storagesets/{storageset_id}`	Get tenant storageset	`GET`
`/api/v1/tenants/{tenant_id}/storagesets/{storageset_id}`	Assign or resize tenant storageset	`PUT`
`/api/v1/tenants/{tenant_id}/storagesets/{storageset_id}`	Unassign tenant storageset	`DELETE`

`GET /api/v1/tenants`

List tenants

Returns a list of all tenants, each carrying the network status the realm manager reports for it. Any platform role that grants `tenants:list` may call it.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Responses

Status	Description	Schema
200	Successful response containing the list of tenants	`ListTenantResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
500	Internal server error.	-

Response Body (200, `ListTenantResponse`)

Field	Type	Required	Default	Example	Description
`tenants`	array of `TenantSummarySchema` object	no	n/a		Array of tenant summary objects. Empty array if no tenants found
`total`	integer	yes	n/a	`0` , `5` , `50`	Total count of tenants in this response.

Example


```
{
   "tenants" : [
      {
         "created_at" : "2026-03-02T12:00:00Z",
         "description" : "FinTech tenant",
         "id" : "e97b94d6-9cc1-4325-a267-1064ae538ccd",
         "name" : "fintech",
         "status" : "active",
         "updated_at" : "2026-03-02T12:00:00Z"
      }
   ],
   "total" : 1
}
```

`POST /api/v1/tenants`

Create tenant

Creates a new tenant with the specified configuration and returns its ID. Platform Admin only.

Before you call this. Needs: The realm filesystem is mounted (`panfs_mounted` is `true` on `GET /api/v1/realm`). Otherwise: `503` while it is not mounted.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Request Body (`application/json` , required)

Schema: `TenantCreateRequest`

Request payload for creating a tenant.

Field	Type	Required	Default	Example	Description
`auth_settings`	`LocalAuthSettings` object or `PamAuthSettings` object	yes	n/a	`{"auth_mode":"local"}` , `{"auth_mode":"pam","service":"login"}`	Tenant authentication settings. Discriminated on `auth_mode` .
`description`	string, nullable	no	`null`	`FinTech department tenant` , `Engineering team workspace`	Tenant description.
`display_name`	string	yes	n/a	`FinTech` , `Engineering` , `Data Science`	Human-readable tenant name. Must be 1-255 printable characters.

Field	Type	Required	Default	Example	Description
`email`	string	yes	n/a	`admin@fintech.example.com`, `ops@vdura.example.com`	Tenant contact email address.
`name`	string	yes	n/a	`fintech`, `eng`, `datasci`	Tenant name, used for volume mounting and as the prefix of a tenant-qualified username. Must be 3-10 alphanumeric characters and unique within the system. Stored in lower case. `platform` is reserved. Immutable after creation.

Responses

Status	Description	Schema
201	Tenant created successfully	`TenantCreateResponse`
400	Bad request. Invalid parameters or validation error.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
409	Conflict. This realm serves its clients over InfiniBand, which cannot carry the VLAN or VXLAN encapsulation multi-tenancy needs, so no tenant can be created on it.	-
422	Validation error. Returned for invalid request body.	-
500	Internal server error.	-
503	Service unavailable. A backend dependency the request needs is unreachable. The request was well formed and changed nothing, so the same call is worth retrying once the dependency is back; which one failed is recorded server-side, not in the response.	-

Response Body (201, `TenantCreateResponse`)

Field	Type	Required	Default	Example	Description
`id`	string (uuid)	yes	n/a	`e97b94d6-9cc1-4325-a267-1064ae538ccd`	Unique tenant identifier (UUID) assigned by the system

Example

```
{
   "id" : "e97b94d6-9cc1-4325-a267-1064ae538ccd"
}
```

`GET /api/v1/tenants/archived`

List archived tenants

Returns the deleted tenants kept in the platform archive, newest deletion first. Platform Admin only. An archived tenant exists in neither the storage fabric nor the configuration store: the entry is all that is left of it, and it is never removed. Its display name and contact email have no other surviving copy.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Responses

Status	Description	Schema
200	Successful response containing the deleted tenants held in the platform archive.	`ListArchivedTenantResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
500	Internal server error.	-

Response Body (200, `ListArchivedTenantResponse`)

Field	Type	Required	Default	Example	Description
`tenants`	array of `ArchivedTenantSchema` object	no	n/a		Array of archived tenants, newest deletion first. Empty array if none were ever deleted.
`total`	integer	yes	n/a	`0` , `5` , `50`	Total count of archived tenants in this response.

Example


```
{
   "tenants" : [
      {
         "created_at" : "2026-01-10T08:00:00Z",
         "deleted_at" : "2026-04-02T11:20:00Z",
         "description" : "Decommissioned engineering tenant",
         "display_name" : "Engineering (retired)",
         "email" : "eng-owner@vdura.example.com",
         "id" : "3f2b1c0d-5e4a-4b8c-9d7e-6a5b4c3d2e1f",
         "name" : "oldeng",
         "status" : "archived",
         "updated_at" : "2026-04-02T11:15:00Z"
      }
   ],
   "total" : 1
}
```

`GET /api/v1/tenants/roles`

List roles available on the current tenant

Returns the roles available on the caller's current tenant - resolved from the auth context, with no tenant_id path parameter. The response is a `roles` array with a `total`; each entry carries the role `id` and the `permissions` map of resource to the actions that role may perform on it. Roles are read from the current tenant's RBAC policy, so Platform and Tenant contexts each return the role set defined and enforced on their own tenant.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Responses

Status	Description	Schema
200	Roles available on the current tenant retrieved successfully.	`TenantRolesResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant RBAC policy roles not found.	-
500	Internal server error.	-

Response Body (200, `TenantRolesResponse`)

Field	Type	Required	Default	Example	Description
`roles`	array of `TenantRole` object	no	n/a		Array of roles the tenant policy defines. Empty array if the policy defines none.
`total`	integer	yes	n/a	`2` , `5`	Total count of roles in this response.

Example


```
{
   "roles" : [
      {
         "id" : "admin",
         "permissions" : {
            "roles" : [
               "view",
               "update"
            ],
            "tenants" : [
               "view"
            ],
            "users" : [
               "list",
               "view",
               "create",
               "update",
               "delete"
            ],
            "volumes" : [
               "list",
               "view",
               "create",
               "update",
               "delete",
               "action"
            ]
         },
         "reserved" : false
      },
      {
         "id" : "s3-gateway",
         "permissions" : {
            "fs-users" : [
               "read-s3-secret",
               "list"
            ]
```

```
        },
        "reserved" : true
      }
   ],
   "total" : 2
}
```

`GET /api/v1/tenants/{tenant_id}`

Get tenant info

Returns details for a specific tenant by its ID, including the network status the realm manager reports for it. The platform tenant may view any tenant, and a Tenant Admin may view only their own tenant.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)

Responses

Status	Description	Schema
200	Successfully retrieved detailed information about the tenant.	`GetTenantResponse`
400	Bad request. Invalid tenant_id parameter.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant not found.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (200, `GetTenantResponse`)

Field	Type	Required	Default	Example	Description
`tenant`	`Tenant` object	yes	n/a		Tenant entity with all available details.

`PATCH /api/v1/tenants/{tenant_id}`

Update tenant

Updates a tenant using PATCH semantics. Only the fields provided in the request are modified. Platform Admin only.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)

Request Body (`application/json` , required)

Schema: `TenantPatchRequest`

Request payload for updating a tenant via PATCH.

Allows partial updates. At least one field must be provided. Patchable fields: `display_name` , `email` , `description` , `auth_settings` . Only a Platform Admin can update a tenant; the tenant's own Admin role can view it but not change it. `name` is the immutable handle and is not patchable.

Field	Type	Required	Default	Example	Description
`auth_settings`	`LocalAuthSettings` object or `PamAuthSettings` object, nullable	no	`null`	`{"auth_mode":"local"}` , `{"auth_mode":"pam","service":"login"}`	Updated tenant authentication settings.
`description`	string, nullable	no	`null`	`Updated tenant description` , `New department focus`	Updated tenant description.

Field	Type	Required	Default	Example	Description
`display_name`	string, nullable	no	`null`	`FinTech Renamed`, `Engineering 2.0`	Updated human-readable tenant name. Must be 1-255 printable characters.
`email`	string, nullable	no	`null`	`admin@fintech.example.com`	Updated tenant contact email address.

Responses

Status	Description	Schema
204	Tenant updated successfully.	-
400	Bad request. Invalid tenant_id or custom validation error.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant not found.	-
422	Validation error. Returned for invalid request body (all fields are optional).	-
500	Internal server error.	-

`DELETE /api/v1/tenants/{tenant_id}`

Delete tenant

Deletes the specified tenant by its ID, removing its records from both the storage fabric and the configuration store and leaving an entry in the platform archive. Platform Admin only. Safe to repeat: a delete interrupted part-way is resumed by sending it again.

Before you call this. Needs: Nothing left in the tenant: delete its S3 instances (`POST /api/v1/tenants/{tenant_id}/s3/instances/delete-all`), then its volumes, then its subnets. Otherwise: `409` for any leftover resource, including an S3 gateway instance.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)

Responses

Status	Description	Schema
204	Tenant deleted successfully.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant not found.	-
409	Conflict. The tenant cannot be deleted as it currently stands: it still owns volumes or subnets, a delete of it is already running, or it is the immutable PanAPI Platform Tenant. The `detail` says which; the example shows the shape, not the alternatives.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

`GET /api/v1/tenants/{tenant_id}/storagesets`

List tenant storageset quota allocations

Lists the storageset quota allocations assigned to a tenant. Callers may only list their own tenant.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)

Responses

Status	Description	Schema
200	Successful response containing the tenant's storageset quota allocations.	`ListTenantStorageSetResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant not found.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (200, `ListTenantStorageSetResponse`)

Field	Type	Required	Default	Example	Description
`storagesets`	array of `StorageSetAssignment` object	no	n/a		Array of storageset assignments (ID + quota + name). Empty array if none are assigned.
`total`	integer	yes	n/a	`0` , `2` , `5`	Total count of storageset assignments in this response.

Example


```
{
   "storagesets" : [
      {
         "quota_bytes" : 2199023255552,
         "storageset_id" : 1,
         "storageset_name" : "Set 1"
      },
      {
         "quota_bytes" : 5497558138880,
         "storageset_id" : 3,
         "storageset_name" : "Set 3"
      }
   ],
   "total" : 2
}
```

`GET /api/v1/tenants/{tenant_id}/storagesets/{storageset_id}`

Get tenant storageset

Returns a single storageset quota allocation for a tenant. Callers may only access their own tenant.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)
`storageset_id`	path	integer	yes	StorageSet identifier

Responses

Status	Description	Schema
200	Successfully retrieved the tenant storageset quota allocation.	`GetTenantStorageSetResponse`
400	Bad request. Invalid tenant_id parameter.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant or storageset not found.	-
422	Validation error.	-
500	Internal server error.	-

Response Body (200, `GetTenantStorageSetResponse`)

Field	Type	Required	Default	Example	Description
`storageset`	`StorageSetAssignment` object	yes	n/a		Storageset assignment (ID + quota + name).

Example


```
{
   "storageset" : {
      "quota_bytes" : 2199023255552,
      "storageset_id" : 1,
      "storageset_name" : "Set 1"
   }
}
```

`PUT /api/v1/tenants/{tenant_id}/storagesets/{storageset_id}`

Assign or resize tenant storageset

Upserts a storageset quota allocation for a tenant: assigns the storageset with the given quota if it is not yet assigned, or replaces the quota if it already is. The combined cross-tenant quota must fit within the storageset capacity. Only `platform-admin` and `storage-admin` may change tenant storageset quotas, for any tenant; a Tenant Admin gets 403.

Before you call this. Needs: The tenant and the StorageSet exist, and the StorageSet has capacity. Only `platform-admin` and `storage-admin` may call it. Otherwise: `403` for any other role; `400` for a quota that exceeds the space available in the StorageSet, so an empty StorageSet cannot be given to a tenant. A tenant with no quota in a StorageSet cannot create volumes in it: volume create returns `404`.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)
`storageset_id`	path	integer	yes	StorageSet identifier

Request Body (`application/json` , required)

Schema: `TenantStorageSetQuotaRequest`

Request body for assigning or resizing a tenant storageset quota.

Field	Type	Required	Default	Example	Description
`quota_bytes`	integer	yes	n/a	`2199023255552` , `5497558138880`	Maximum storage quota in bytes that the tenant can consume within this storageset.

Example

```
{
   "quota_bytes" : 2199023255552
}
```

Responses

Status	Description	Schema
200	Storageset assigned to the tenant, or its quota updated. Returns the resulting allocation.	`GetTenantStorageSetResponse`
400	Bad request. The requested quota exceeds the storageset capacity.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant or storageset not found.	-
422	Validation error. Returned for an invalid request body or path parameter.	-
500	Internal server error.	-

Response Body (200, `GetTenantStorageSetResponse`)

Field	Type	Required	Default	Example	Description
`storageset`	`StorageSetAssignment` object	yes	n/a		Storageset assignment (ID + quota + name).

Example


```
{
   "storageset" : {
      "quota_bytes" : 2199023255552,
      "storageset_id" : 1,
      "storageset_name" : "Set 1"
   }
}
```

`DELETE /api/v1/tenants/{tenant_id}/storagesets/{storageset_id}`

Unassign tenant storageset

Unassigns a storageset from a tenant. Callers may only modify their own tenant.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)
`storageset_id`	path	integer	yes	StorageSet identifier

Responses

Status	Description	Schema
204	Storageset unassigned from the tenant successfully. No content is returned.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant or storageset not found.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

7. Tenant Users and Roles

Tenant users are not fully supported in this release. The endpoints in this chapter work, and a tenant user can sign in as `<tenant-name>/<user>`, but Vantage manages users only in the Platform area, and a tenant has no sign-in of its own there. To administer a tenant, use a Platform Admin and the tenant switcher (`POST /auth/tenant-switch`) instead.

7.1 Tenant Users

Endpoint Summary

Path	Summary	Method
`/api/v1/tenants/{tenant_id}/users`	List tenant users	`GET`
`/api/v1/tenants/{tenant_id}/users`	Create tenant user	`POST`
`/api/v1/tenants/{tenant_id}/users/{user_id}`	Get tenant user	`GET`
`/api/v1/tenants/{tenant_id}/users/{user_id}`	Update tenant user	`PATCH`
`/api/v1/tenants/{tenant_id}/users/{user_id}`	Delete tenant user	`DELETE`

`GET /api/v1/tenants/{tenant_id}/users`

List tenant users

Return all users that belong to the specified tenant. The response includes a `total` count field. Tenant Admins can only list users within their own tenant.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID).

Responses

Status	Description	Schema
200	Tenant users retrieved successfully.	`UserListResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (200, `UserListResponse`)

Field	Type	Required	Default	Example	Description
`total`	integer	yes	n/a	`0` , `5` , `50`	Total count of Tenant Users in this response.
`users`	array of `User` object	no	n/a		Array of Tenant User objects. Empty array if no users found.

Example


```
{
   "total" : 1,
   "users" : [
      {
         "auth_settings" : {
            "auth_mode" : "local"
         },
         "created_at" : "2026-01-01T00:00:00Z",
         "display_name" : "Alice Anderson",
         "email" : "alice@fintech.example.com",
         "id" : "a1b2c3d4-e5f6-7890-abcd-ef1234567890",
         "updated_at" : "2026-01-01T00:00:00Z",
         "username" : "alice"
      }
   ]
}
```

`POST /api/v1/tenants/{tenant_id}/users`

Create tenant user

Create a new tenant user within the specified tenant. The name must be unique within the tenant. The password is hashed server-side; the plaintext value is never stored.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID).

Request Body (`application/json` , required)

Schema: `UserCreateRequest`

Request payload for creating a Tenant User.

The password is hashed server-side before storage; the plaintext value is never persisted. The name must be unique within the tenant.

Field	Type	Required	Default	Example	Description
`auth_settings`	`LocalAuthSettings` object or `PamAuthSettings` object	yes	n/a	`{"auth_mode":"local"}` , `{"auth_mode":"pam","service":"login"}`	Authentication backend for this user, discriminated on `auth_mode` .
`display_name`	string, nullable	no	`null`	`Alice Anderson`	Human-readable full name. Optional; `username` remains the unique handle.
`email`	string, nullable	no	`null`	`alice@fintech.example.com`	Optional contact email for the Tenant User.

Field	Type	Required	Default	Example	Description
`password`	string (password)	yes	n/a		Tenant User password (min 8 characters). Hashed server-side before storage.
`username`	string	yes	n/a	`alice`	Tenant User username. Must be unique within the tenant. Whitespace, control, and invisible characters are not allowed.

Example

```
{
   "auth_settings" : {
      "auth_mode" : "local"
   },
   "display_name" : "Alice Anderson",
   "email" : "alice@fintech.example.com",
   "password" : "secure-password",
   "username" : "alice"
}
```

Responses

Status	Description	Schema
201	Tenant user created successfully.	`UserCreateResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
409	Conflict. A user with this username already exists in the target tenant.	-
422	Validation error.	-
500	Internal server error.	-

Response Body (201, `UserCreateResponse`)

Field	Type	Required	Default	Example	Description
`id`	string (uuid)	yes	n/a	`a1b2c3d4-e5f6-7890-abcd-ef1234567890`	Unique Tenant User identifier (UUID) assigned by the system.

Example

```
{
   "id" : "a1b2c3d4-e5f6-7890-abcd-ef1234567890"
}
```

`GET /api/v1/tenants/{tenant_id}/users/{user_id}`

Get tenant user

Retrieve a specific tenant user by ID. Returns 404 if the user does not exist or belongs to a different tenant.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID).
`user_id`	path	string (uuid)	yes	Tenant user identifier (UUID).

Responses

Status	Description	Schema
200	Tenant user retrieved successfully.	`GetUserResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant user not found.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (200, `GetUserResponse`)

Field	Type	Required	Default	Example	Description
`user`	`User` object	yes	n/a		Tenant User entity with all available details.

Example


```
{
   "user" : {
      "auth_settings" : {
         "auth_mode" : "local"
      },
      "created_at" : "2026-01-01T00:00:00Z",
      "display_name" : "Alice Anderson",
      "email" : "alice@fintech.example.com",
      "id" : "a1b2c3d4-e5f6-7890-abcd-ef1234567890",
      "updated_at" : "2026-01-01T00:00:00Z",
      "username" : "alice"
   }
}
```

`PATCH /api/v1/tenants/{tenant_id}/users/{user_id}`

Update tenant user

Apply a partial update to the specified tenant user: set a new password, change the authentication backend (`auth_settings`), update the `email` and/or the `display_name` . At least one field must be supplied. This is an administrative operation - the caller does not need to supply the current password. Returns 400 if a password change is requested for a user whose resulting auth mode is not local. Returns 404 if the user does not exist or belongs to a different tenant.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID).
`user_id`	path	string (uuid)	yes	Tenant user identifier (UUID).

Request Body (`application/json` , required)

Schema: `UserPatchRequest`

Request payload for updating a Tenant User.

Allows partial updates of `new_password` , `auth_settings` , `email` and/or `display_name` ; at least one must be supplied. Administrative operation - the caller does not need to supply the current password. Intended for Tenant Admin or Platform Admin use only.

Field	Type	Required	Default	Example	Description
`auth_settings`	`LocalAuthSettings` object or `PamAuthSettings` object, nullable	no	`null`	`{"auth_mode":"local"}` , `{"auth_mode":"pam","service":"login"}`	Updated authentication backend for this user, discriminated on `auth_mode` .
`display_name`	string, nullable	no	`null`	`Alice Anderson`	Updated human-readable full name for the Tenant User.

Field	Type	Required	Default	Example	Description
`email`	string, nullable	no	`null`	`alice@fintech.example.com`	Updated contact email for the Tenant User.
`new_password`	string (password), nullable	no	`null`		New password for the Tenant User (min 8 characters). Hashed server-side before storage.

Responses

Status	Description	Schema
204	Password updated successfully.	-
400	Bad request. A password change was requested for a user whose auth mode is not 'local'.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant user not found.	-
422	Validation error. Password does not meet minimum length requirement.	-
500	Internal server error.	-

`DELETE /api/v1/tenants/{tenant_id}/users/{user_id}`

Delete tenant user

Permanently delete a tenant user by ID, together with everything that would still authenticate as it: the API tokens it holds and its role bindings. Returns 404 if the user does not exist or belongs to a different tenant. Returns 409 for the Platform Admin, which cannot be deleted by any caller. Tenant Admins can only delete users within their own tenant.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID).
`user_id`	path	string (uuid)	yes	Tenant user identifier (UUID).

Responses

Status	Description	Schema
204	Tenant user deleted successfully.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant user not found.	-
409	Conflict. The user is protected and cannot be deleted through the API.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

7.2 Tenant User Roles

Endpoint Summary

Path	Summary	Method
`/api/v1/tenants/{tenant_id}/users/{user_id}/roles`	List tenant user roles	`GET`
`/api/v1/tenants/{tenant_id}/users/{user_id}/roles`	Set tenant user roles	`PATCH`

`GET /api/v1/tenants/{tenant_id}/users/{user_id}/roles`

List tenant user roles

Returns the RBAC roles assigned to a tenant user.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID).
`user_id`	path	string (uuid)	yes	Tenant user identifier (UUID).

Responses

Status	Description	Schema
200	User roles retrieved successfully.	`RolesResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant user not found.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (200, `RolesResponse`)

Field	Type	Required	Default	Example	Description
`roles`	array of string	yes	n/a	`["admin"]`	Roles currently bound to the principal.

`PATCH /api/v1/tenants/{tenant_id}/users/{user_id}/roles`

Set tenant user roles

Replaces the RBAC roles assigned to a tenant user. The roles of a protected user cannot be changed through the API, and the request is refused with 409.

Before you call this. Needs: Every role in the request is defined for the tenant (`GET /api/v1/tenants/roles`). Otherwise: `400` for an unknown or reserved role (`platform-support` , `s3-gateway`).

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID).
`user_id`	path	string (uuid)	yes	Tenant user identifier (UUID).

Request Body (`application/json` , required)

Schema: `SetRolesRequest`

Request payload for replacing the full role set of a Tenant User.

Field	Type	Required	Default	Example	Description
`roles`	array of string	yes	n/a	`["admin"]`	Roles to assign to the user; replaces the current set. A role the target tenant does not define, or a reserved one, is rejected with HTTP 400.

Responses

Status	Description	Schema
200	User roles retrieved successfully.	`RolesResponse`
400	Bad request. A requested role is not defined by the tenant's RBAC policy.	-

Status	Description	Schema
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant user not found.	-
409	Conflict. The request conflicts with the current state of the resource.	-
422	Validation error.	-
500	Internal server error.	-

Response Body (200, `RolesResponse`)

Field	Type	Required	Default	Example	Description
`roles`	array of string	yes	n/a	`["admin"]`	Roles currently bound to the principal.

8. PanFS Users

Per-tenant PanFS identities, with POSIX uid/gid, used for S3 object ownership, distinct from the tenant users who sign in to PanAPI itself.

Endpoint Summary

Path	Summary	Method
`/api/v1/tenants/{tenant_id}/fs-users`	List PanFS users	`GET`
`/api/v1/tenants/{tenant_id}/fs-users`	Create PanFS user	`POST`
`/api/v1/tenants/{tenant_id}/fs-users/{fs_user_id}`	Get PanFS user	`GET`
`/api/v1/tenants/{tenant_id}/fs-users/{fs_user_id}`	Update PanFS user	`PATCH`
`/api/v1/tenants/{tenant_id}/fs-users/{fs_user_id}`	Delete PanFS user	`DELETE`
`/api/v1/tenants/{tenant_id}/fs-users/{fs_user_id}/s3/secret`	Reveal a PanFS user S3 secret	`GET`
`/api/v1/tenants/{tenant_id}/fs-users/{fs_user_id}/s3/secret/rotate`	Rotate a PanFS user S3 secret	`POST`

`GET /api/v1/tenants/{tenant_id}/fs-users`

List PanFS users

Every PanFS user in the tenant, sorted by name. No secrets.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID).

Responses

Status	Description	Schema
200	PanFS users listed.	`FsUserListResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (200, `FsUserListResponse`)

Field	Type	Required	Default	Example	Description
`fs_users`	array of `FsUser` object	yes	n/a		
`total`	integer	yes	n/a		How many users the listing carries.

`POST /api/v1/tenants/{tenant_id}/fs-users`

Create PanFS user

Create a PanFS user in the tenant. The name must be unique within the tenant, and so must the S3 access key, which defaults to the name. The S3 secret is minted here and is **not** returned: read it through the reveal route, which gates on a stronger permission.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID).

Request Body (`application/json` , required)

Schema: `FsUserCreateRequest`

Request payload for creating a PanFS user.

Field	Type	Required	Default	Example	Description
`access_key`	string, nullable	no	`null`		S3 access key. Defaults to name, which is why both carry the same charset rule.
`gid`	integer, nullable	no	`null`		POSIX group ID. Same defaulting and bounds as uid.
`name`	string	yes	n/a		Unique handle within the tenant.
`profile`	`S3Profile` (`disabled` , `user` , `privileged`)	no	`user`		S3 access profile.

Field	Type	Required	Default	Example	Description
`uid`	integer, nullable	no	`null`		POSIX user ID. Omit to take the tenant's configured default, or nobody (65534) when it has none or when that default is not a usable ID, 0 included. The resolved value is written onto the record, so a later change to the tenant default does not move this identity. 0 is not accepted: it would name root. Nor is anything above 4294967294, the last ID a 32-bit uid_t can hold.

Example

```
{
   "gid" : 1000,
   "name" : "alice",
   "uid" : 1000
}
```

Responses

Status	Description	Schema
201	PanFS user created.	`FsUserResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant not found.	-
409	name or access_key already in use, or reserved by the S3 config.	-
422	Validation error.	-
500	Internal server error.	-

Response Body (201, `FsUserResponse`)

Field	Type	Required	Default	Example	Description
`fs_user`	`FsUser` object	yes	n/a		A PanFS user as published.

`GET /api/v1/tenants/{tenant_id}/fs-users/{fs_user_id}`

Get PanFS user

Fetch one PanFS user.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID).
`fs_user_id`	path	string (uuid)	yes	

Responses

Status	Description	Schema
200	PanFS user returned.	`FsUserResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	No such PanFS user in this tenant.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (200, `FsUserResponse`)

Field	Type	Required	Default	Example	Description
`fs_user`	`FsUser` object	yes	n/a		A PanFS user as published.

`PATCH /api/v1/tenants/{tenant_id}/fs-users/{fs_user_id}`

Update PanFS user

Partial update of uid, gid, profile and/or access_key. The secret cannot be changed here - rotation is its own operation, because it is destructive.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID).
`fs_user_id`	path	string (uuid)	yes	

Request Body (`application/json` , required)

Schema: `FsUserPatchRequest`

Request payload for updating a PanFS user. The secret is not among the fields.

Field	Type	Required	Default	Example	Description
`access_key`	string, nullable	no	`null`		Updated S3 access key.
`gid`	integer, nullable	no	`null`		Updated POSIX group ID.
`profile`	`S3Profile` (`disabled` , `user` , `privileged`), nullable	no	`null`		Updated S3 access profile.
`uid`	integer, nullable	no	`null`		Updated POSIX user ID.

Responses

Status	Description	Schema
200	PanFS user updated.	`FsUserResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	No such PanFS user in this tenant.	-
409	access_key already in use, or reserved by the S3 config.	-
422	Validation error.	-
500	Internal server error.	-

Response Body (200, `FsUserResponse`)

Field	Type	Required	Default	Example	Description
`fs_user`	`FsUser` object	yes	n/a		A PanFS user as published.

`DELETE /api/v1/tenants/{tenant_id}/fs-users/{fs_user_id}`

Delete PanFS user

Idempotent: deleting a user that is already gone succeeds, so a retry resumes.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID).
`fs_user_id`	path	string (uuid)	yes	

Responses

Status	Description	Schema
204	PanFS user deleted.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

`GET /api/v1/tenants/{tenant_id}/fs-users/{fs_user_id}/s3/secret`

Reveal a PanFS user S3 secret

Return the S3 secret. Gated on `fs-users:read-s3-secret` rather than `fs-users:view`, because it hands over a credential rather than describing one.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID).
`fs_user_id`	path	string (uuid)	yes	

Responses

Status	Description	Schema
200	Secret returned.	`FsUserSecretResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	No such PanFS user, or the user has no S3 credential.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (200, `FsUserSecretResponse`)

Field	Type	Required	Default	Example	Description
`access_key`	string	yes	n/a		The access key the secret belongs to.
`secret_key`	string	yes	n/a		The S3 secret key.

Example

```
{
   "access_key" : "alice",
   "secret_key" : "..."
}
```

`POST /api/v1/tenants/{tenant_id}/fs-users/{fs_user_id}/s3/secret/rotate`

Rotate a PanFS user S3 secret

Mint a new secret and return it once. The access key is left standing: it is what bucket owner fields and audit lines name. Permitted regardless of profile, so an operator can prepare a disabled user before re-enabling it.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID).
`fs_user_id`	path	string (uuid)	yes	

Responses

Status	Description	Schema
200	Secret rotated.	`FsUserSecretResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	No such PanFS user in this tenant.	-
409	The user has no S3 credential to rotate.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (200, `FsUserSecretResponse`)

Field	Type	Required	Default	Example	Description
`access_key`	string	yes	n/a		The access key the secret belongs to.
`secret_key`	string	yes	n/a		The S3 secret key.

Example

```
{
   "access_key" : "alice",
   "secret_key" : "..."
}
```

9. Service Accounts

Non-interactive identities with their own roles and API keys, for automation that should not authenticate as a specific human tenant user.

Endpoint Summary

Path	Summary	Method
`/api/v1/service-accounts`	List service accounts	`GET`
`/api/v1/service-accounts`	Create a service account	`POST`
`/api/v1/service-accounts/{service_account_id}`	Get a service account	`GET`
`/api/v1/service-accounts/{service_account_id}`	Update a service account	`PATCH`
`/api/v1/service-accounts/{service_account_id}`	Delete a service account	`DELETE`
`/api/v1/service-accounts/{service_account_id}/roles`	List service account roles	`GET`
`/api/v1/service-accounts/{service_account_id}/roles`	Set service account roles	`PATCH`
`/api/v1/service-accounts/{service_account_id}/tokens`	List service account keys	`GET`
`/api/v1/service-accounts/{service_account_id}/tokens`	Issue a service account key	`POST`
`/api/v1/service-accounts/{service_account_id}/tokens/{token_id}`	Revoke a service account key	`DELETE`

`GET /api/v1/service-accounts`

List service accounts

Return the service accounts of the caller's own tenant, newest first.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Responses

Status	Description	Schema
200	The service accounts.	`ServiceAccountListResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
500	Internal server error.	-

Response Body (200, `ServiceAccountListResponse`)

Field	Type	Required	Default	Example	Description
`service_accounts`	array of `ServiceAccountResponse` object	yes	n/a		The accounts.
`total`	integer	yes	n/a		How many accounts are listed.

`POST /api/v1/service-accounts`

Create a service account

Create a non-interactive principal in your own tenant. It holds no password and cannot sign in; it acts only through the keys you issue it, and only within the roles you bind to it. A new account starts with no roles, so it can do nothing until you grant some.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Request Body (`application/json` , required)

Schema: `ServiceAccountCreateRequest`

Request to create a service account.

Field	Type	Required	Default	Example	Description
`description`	string, nullable	no	`null`		What this account is for. Newlines and tabs are allowed; other control characters are not.
`display_name`	string, nullable	no	`null`		Human-readable label. Control and invisible characters are not allowed.
`name`	string	yes	n/a		Unique name within the tenant, 4-30 characters of lowercase letters, digits, and hyphens, starting with a letter. Identifies the account wherever it appears; it cannot be changed afterwards.

Example

```
{
   "display_name" : "Nightly CI",
   "name" : "ci-nightly"
}
```

Responses

Status	Description	Schema
201	Service account created.	`ServiceAccountResponse`
400	Bad request.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
409	Conflict. The request conflicts with the current state of the resource.	-
422	Validation error.	-

Response Body (201, `ServiceAccountResponse`)

Field	Type	Required	Default	Example	Description
`created_at`	string (date-time)	yes	n/a		When the account was created.
`created_by`	string (uuid)	yes	n/a		The user that created it.
`description`	string, nullable	no	`null`		What this account is for.
`display_name`	string, nullable	no	`null`		Human-readable label.
`id`	string (uuid)	yes	n/a		Identifier of the service account.
`name`	string	yes	n/a		Unique name within the tenant, 4-30 characters of lowercase letters, digits, and hyphens, starting with a letter. Identifies the account wherever it appears; it cannot be changed afterwards.

Field	Type	Required	Default	Example	Description
`updated_at`	string (date-time)	yes	n/a		When the account was last changed.

`GET /api/v1/service-accounts/{service_account_id}`

Get a service account

Return one service account of the caller's own tenant.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`service_account_id`	path	string (uuid)	yes	Identifier of the service account.

Responses

Status	Description	Schema
200	The service account.	`ServiceAccountResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Not found. The requested resource does not exist.	-
422	Validation Error	`HTTPValidationError`

Response Body (200, `ServiceAccountResponse`)

Field	Type	Required	Default	Example	Description
`created_at`	string (date-time)	yes	n/a		When the account was created.
`created_by`	string (uuid)	yes	n/a		The user that created it.
`description`	string, nullable	no	`null`		What this account is for.
`display_name`	string, nullable	no	`null`		Human-readable label.

Field	Type	Required	Default	Example	Description
`id`	string (uuid)	yes	n/a		Identifier of the service account.
`name`	string	yes	n/a		Unique name within the tenant, 4-30 characters of lowercase letters, digits, and hyphens, starting with a letter. Identifies the account wherever it appears; it cannot be changed afterwards.
`updated_at`	string (date-time)	yes	n/a		When the account was last changed.

`PATCH /api/v1/service-accounts/{service_account_id}`

Update a service account

Apply a partial update. The name is fixed at creation: it is how the account is recognised wherever it appears, and a rename would leave every earlier record naming something that no longer exists.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`service_account_id`	path	string (uuid)	yes	Identifier of the service account.

Request Body (`application/json` , required)

Schema: `ServiceAccountUpdateRequest`

Request to update a service account. Only the fields present are applied.

Field	Type	Required	Default	Example	Description
`description`	string, nullable	no	`null`		What this account is for. Newlines and tabs are allowed; other control characters are not.
`display_name`	string, nullable	no	`null`		Human-readable label. Control and invisible characters are not allowed.

Example

```
{
   "description" : "Nightly backup runner"
}
```

Responses

Status	Description	Schema
200	The updated service account.	`ServiceAccountResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Not found. The requested resource does not exist.	-
422	Validation error.	-

Response Body (200, `ServiceAccountResponse`)

Field	Type	Required	Default	Example	Description
`created_at`	string (date-time)	yes	n/a		When the account was created.
`created_by`	string (uuid)	yes	n/a		The user that created it.
`description`	string, nullable	no	`null`		What this account is for.
`display_name`	string, nullable	no	`null`		Human-readable label.
`id`	string (uuid)	yes	n/a		Identifier of the service account.
`name`	string	yes	n/a		Unique name within the tenant, 4-30 characters of lowercase letters, digits, and hyphens, starting with a letter. Identifies the account wherever it appears; it cannot be changed afterwards.
`updated_at`	string (date-time)	yes	n/a		When the account was last changed.

`DELETE /api/v1/service-accounts/{service_account_id}`

Delete a service account

Delete the account, every key it holds, and every role bound to it. The keys stop authenticating on the next request. Repeating the call finishes an interrupted delete.

A protected account cannot be deleted through the API, and the request is refused with 409.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`service_account_id`	path	string (uuid)	yes	Identifier of the service account.

Responses

Status	Description	Schema
204	Deleted.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Not found. The requested resource does not exist.	-
409	Conflict. The request conflicts with the current state of the resource.	-
422	Validation Error	`HTTPValidationError`

`GET /api/v1/service-accounts/{service_account_id}/roles`

List service account roles

Return the roles bound to the account. Every key it holds resolves exactly these.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`service_account_id`	path	string (uuid)	yes	Identifier of the service account.

Responses

Status	Description	Schema
200	The roles.	`RolesResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Not found. The requested resource does not exist.	-
422	Validation Error	`HTTPValidationError`

Response Body (200, `RolesResponse`)

Field	Type	Required	Default	Example	Description
`roles`	array of string	yes	n/a	`["admin"]`	Roles currently bound to the principal.

`PATCH /api/v1/service-accounts/{service_account_id}/roles`

Set service account roles

Replace the roles bound to the account. Every key it holds resolves the new set from the next request onwards, whenever it was issued. An empty list leaves the account with no permissions at all, which is how it is switched off without destroying anything.

The roles of a protected account cannot be changed through the API, and the request is refused with 409.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`service_account_id`	path	string (uuid)	yes	Identifier of the service account.

Request Body (`application/json` , required)

Schema: `SetRolesRequest`

Request payload for replacing the full role set of a Tenant User.

Field	Type	Required	Default	Example	Description
`roles`	array of string	yes	n/a	`["admin"]`	Roles to assign to the user; replaces the current set. A role the target tenant does not define, or a reserved one, is rejected with HTTP 400.

Responses

Status	Description	Schema
200	The roles now bound.	`RolesResponse`
400	Bad request. A requested role is not defined by the tenant's RBAC policy.	-

Status	Description	Schema
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Not found. The requested resource does not exist.	-
409	Conflict. The request conflicts with the current state of the resource.	-
422	Validation error.	-

Response Body (200, `RolesResponse`)

Field	Type	Required	Default	Example	Description
`roles`	array of string	yes	n/a	`["admin"]`	Roles currently bound to the principal.

`GET /api/v1/service-accounts/{service_account_id}/tokens`

List service account keys

Return the keys the account holds, newest first. Carries no secret - a key is unrecoverable after issuance. Revoked and expired keys stay listed.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`service_account_id`	path	string (uuid)	yes	Identifier of the service account.
`active_only`	query	boolean	no	Return only records whose `state` is `active` .

Responses

Status	Description	Schema
200	The keys.	`ApiTokenListResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Not found. The requested resource does not exist.	-
422	Validation Error	`HTTPValidationError`

Response Body (200, `ApiTokenListResponse`)

Field	Type	Required	Default	Example	Description
`tokens`	array of `ApiTokenResponse` object	yes	n/a		Token metadata, newest first.
`total`	integer	yes	n/a		Number of tokens returned.

`POST /api/v1/service-accounts/{service_account_id}/tokens`

Issue a service account key

Issue a long-lived credential that authenticates as the service account. It resolves whatever roles the account holds at the time of each request - every key of one account carries the same permissions, whenever it was issued.

The key is returned once, in this response, and cannot be retrieved again.

Requires an interactive session: a key cannot issue another, and a tenant switch cannot leave one behind.

Refused with 409 for a built-in platform account (for example the S3 gateway). Its credentials are managed by the platform, so no key can be issued for it.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`service_account_id`	path	string (uuid)	yes	Identifier of the service account.

Request Body (`application/json` , required)

Schema: `ApiTokenCreateRequest`

Request to issue a long-lived API token for the caller.

Field	Type	Required	Default	Example	Description
`description`	string, nullable	no	`null`		What this token is for. Optional; `name` remains the handle an operator identifies it by.
`expires_in_days`	integer	yes	n/a	`30` , `90` , `180` , `365`	Lifetime in days. Required, and must name one of the offered values (30, 90, 180, 365), so a long lifetime is always a deliberate choice.

Field	Type	Required	Default	Example	Description
`name`	string	yes	n/a		Name for this token, unique among the caller's usable tokens. An operator identifies a token by it. A revoked or expired token releases its name, so a rotated credential can keep the name it had.

Example

```
{
   "description" : "Nightly backup runner",
   "expires_in_days" : 90,
   "name" : "nightly-backup@ci-runner-04"
}
```

Responses

Status	Description	Schema
201	Key issued. The only response that contains the secret.	`ApiTokenCreatedResponse`
400	Bad request.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Not found. The requested resource does not exist.	-
409	Conflict. The request conflicts with the current state of the resource.	-
422	Validation error.	-

Response Body (201, `ApiTokenCreatedResponse`)

Field	Type	Required	Default	Example	Description
`created_at`	string (date-time)	yes	n/a		When the secret was created. Never changes.
`created_by`	`Principal` object	yes	n/a		Who issued the token.
`description`	string, nullable	no	`null`		What this token is for, if a description was given.
`expires_at`	string (date-time)	yes	n/a		Absolute expiry.
`id`	string	yes	n/a		Public token identifier. Appears in audit events; not a secret.
`name`	string	yes	n/a		The handle given at issuance, unique among the owner's usable tokens.
`revoked_at`	string (date-time), nullable	no	`null`		When the token was revoked, if it was. The only lifecycle fact the record stores.
`revoked_by`	`Principal` object, nullable	no	`null`		Who revoked the token, if it was.

Field	Type	Required	Default	Example	Description
`roles`	array of string, nullable	no	`null`		The role ceiling held in the record: the owner's own roles as at issuance. Not chosen - a token carries what its owner held, and never more than they hold now. Null for a service account key, which carries no ceiling: it resolves whatever the account holds at the time of each request.
`state`	`ApiTokenState` (`active` , `disabled`)	yes	n/a		Whether the token still authenticates. Computed per response, never stored.
`state_reason`	`ApiTokenStateReason` (`expired` , `revoked`), nullable	no	`null`		Why the token is disabled. Null while it is active. Revocation outranks expiry.
`token`	string	yes	n/a	`vdt1_6XuU1pyDS0abcDEF12345w_9fK3mZq7Xw2Lb1cD_hT8n...7bQ2xM`	The token. Returned once and never retrievable again; a lost token must be replaced.
`warning`	string	no	`Store this token now. It cannot be retrieved again.`		Reminder that the secret is shown exactly once.

`DELETE /api/v1/service-accounts/{service_account_id}/tokens/{token_id}`

Revoke a service account key

Revoke one key. It stops authenticating on the next request. The record is kept as a tombstone, so a client still presenting it is told the key was revoked rather than that it never existed.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`service_account_id`	path	string (uuid)	yes	Identifier of the service account.
`token_id`	path	string	yes	Public identifier of the key.

Responses

Status	Description	Schema
204	Revoked.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Not found. The requested resource does not exist.	-
422	Validation Error	`HTTPValidationError`

10. Volumes

Creating, updating, and deleting volumes, reading their space usage, and volume-level actions (rename, recover).

Endpoint Summary

Path	Summary	Method
`/api/v1/volumes`	List volumes	`GET`
`/api/v1/volumes`	Create a new volume	`POST`
`/api/v1/volumes/{volume_id}`	Get volume info	`GET`
`/api/v1/volumes/{volume_id}`	Update volume	`PATCH`
`/api/v1/volumes/{volume_id}`	Delete volume	`DELETE`
`/api/v1/volumes/{volume_id}/actions`	Perform action on volume	`POST`
`/api/v1/volumes/{volume_id}/space`	Get volume space	`GET`

`GET /api/v1/volumes`

List volumes

Returns a list of volumes accessible to the authenticated user.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	query	string (uuid), nullable	no	Platform tenant only. Read the volumes of another tenant instead of your own.
`all_tenants`	query	boolean	no	Platform tenant only. Read the volumes of every tenant.

Responses

Status	Description	Schema
200	Successful response containing the list of volumes	`VolumeListResponse`
400	Bad request. Invalid parameters or validation error.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (200, `VolumeListResponse`)

Field	Type	Required	Default	Example	Description
`total`	integer	yes	n/a	`0` , `2` , `150`	Total count of volumes available (may exceed returned array size if paginated).

Field	Type	Required	Default	Example	Description
`volumes`	array of `Volume` object	no	n/a		Array of volume objects matching the query. Empty array if no volumes found

Example


```
{
   "total" : 2,
   "volumes" : [
      {
         "efsa_mode" : "retry",
         "hard_quota_bytes" : 549755813888,
         "id" : 1,
         "object_id" : "I-xD0200000000000005-xG82c418af-xU00004a75726a656e",
         "pathname" : "/",
         "quotas_policy" : "enable",
         "raid_type" : "raid6+",
         "recovery_priority" : 1,
         "soft_quota_bytes" : 524288000000,
         "status" : "online",
         "storageset_id" : 1,
         "tenant_id" : "00000000-0000-0000-0000-000000000001",
         "volume_service_id" : "0x0400000000000005"
      },
      {
         "efsa_mode" : "retry",
         "id" : 2,
         "object_id" : "I-xD0200000000000006-xGc4f7a724-xU00004a75726a656e",
         "pathname" : "/home",
         "quotas_policy" : "disable",
         "raid_type" : "raid6+",
         "recovery_priority" : 50,
         "status" : "online",
         "storageset_id" : 1,
         "tenant_id" : "e97b94d6-9cc1-4325-a267-1064ae538ccd",
         "volume_service_id" : "0x0400000000000006"
      }
   ]
}
```

`POST /api/v1/volumes`

Create a new volume

Creates a new volume and returns its ID.

Before you call this. Needs: A token for the tenant (`POST /api/v1/auth/tenant-switch` as a Platform Admin, or a tenant admin); the realm filesystem is mounted; the tenant has a quota in the StorageSet; the tenant has a subnet with an issued address; the StorageSet has enough online disks for the RAID layout; and `hard_quota_bytes` is greater than 0. Otherwise: `404` for a Platform Admin who has not switched into the tenant (the volume would go to the platform tenant), and also without the StorageSet quota; `503` while not mounted; `409` ("The tenant holds no issued subnet address, so a volume created for it could not be reached. Create a subnet for the tenant and let one of its nodes report an address into it first") before the tenant is `active` ; `409` "Not enough online disks in StorageSet to create volume" when the StorageSet has too few online OSDs for the RAID layout (4 for `raid6+` , 3 for `raid5+` and `raid10+`); `400` without a positive `hard_quota_bytes` . A `504` means the volume may still appear, so list volumes before retrying.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`ignore_limits`	query	boolean	no	Platform tenant only. Act without the limits the tenant's storageset assignment imposes: no assignment need exist, no quota ceiling applies, and free space is not checked, as before tenancy. For a create, the target storageset must still exist in the realm.

Request Body (`application/json` , required)

Schema: `VolumeCreateRequest`

Request payload for creating a volume.

Mirrors the CLI “volume create” command. Requires a volume name and a StorageSet ID. Optional parameters use system defaults when omitted.

Field	Type	Required	Default	Example	Description
`efsa_mode`	`EfsaMode` (`retry` , `file-unavailable` , `default`)	no	`default`	`retry` , `file-unavailable` , `default`	EFSA mode for the volume. Defines behavior during partial storage unavailability. Supported: retry, file-unavailable, default. If not specified, system default (retry) is applied. Only for RAID 5+, 6+, 10+ volumes.
`group`	string, nullable	no	`null`	`0` , `100`	Owner GID of the volume, as a numeric string (e.g. “0”, “100”). Defaults to root group (gid=0) if not specified. String name resolution and Windows SID are not supported.
`group_permissions`	`VolumePermission` (`none` , `read-only` , `write-only` , `execute-only` , `read-write` , `read-execute` , `write-execute` , `all`), nullable	no	`null`	`read-only` , `read-execute`	Filesystem permissions for the owner group on the root of the volume. Allowed values: none, read-only, write-only, execute-only, read-write, read-execute, write-execute, all.

Field	Type	Required	Default	Example	Description
`hard_quota_bytes`	integer, nullable	no	`null`	`54975581388800`, `null`	Hard quota in bytes. Prevents additional writes when limit is reached. A volume in a tenant requires a value greater than 0; a request without one returns 400.
`max_width`	integer, nullable	no	`null`	`12`, `20`	Maximum number of VPODs to stripe data across. Minimum depends on RAID layout: 2 for RAID 5+, 3 for RAID 6+, 2 for RAID 10+. If not specified, the system allows striping across the StorageSet.
`other_permissions`	`VolumePermission` (`none`, `read-only`, `write-only`, `execute-only`, `read-write`, `read-execute`, `write-execute`, `all`), nullable	no	`null`	`none`, `read-only`	Filesystem permissions for others on the root of the volume. Allowed values: none, read-only, write-only, execute-only, read-write, read-execute, write-execute, all.
`pathname`	string	yes	n/a	`/vol2`, `/data`	Required unique mount path of the volume. Must start with / and be unique within the system.

Field	Type	Required	Default	Example	Description
`raid_type`	`VolumeLayout` (`raid6+`, `raid5+`, `raid10+`, `raid5`, `raid10`)	no	`raid6+`	`raid6+`, `raid5+`, `raid10+`, `raid5`, `raid10`	RAID layout for the volume. Default: raid6+. Cannot be changed after creation.
`recovery_priority`	integer	no	`50`	`0`, `1`, `50`, `100`	Recovery priority (1-100) of the volume. Lower values indicate higher recovery priority, where 1 represents the highest priority. If not specified the system default (50) is applied.
`rg_depth`	integer, nullable	no	`null`	`4`, `16`	RAID group depth (number of stripes per RAID parity group). Only for RAID 5+, 6+. If not specified, a system default value is applied.
`rg_width`	integer, nullable	no	`null`	`8`, `12`	RAID group width (VPODs to stripe over per single RAID group, 3-20). Only for RAID 5+, 6+. If not specified, a system default value is applied.
`soft_quota_bytes`	integer, nullable	no	`null`	`549755813888`, `null`	Soft quota limit in bytes. Triggers quota notifications when the threshold is exceeded. Use 0 or null for unlimited.
`storageset_id`	integer	yes	n/a	`1`, `2`, `5`	Identifier of the StorageSet where the volume will be created. Required if multiple StorageSets exist.

Field	Type	Required	Default	Example	Description
`stripe_unit_bytes`	integer, nullable	no	`null`	`262144`, `1048576`	Specifies the number of bytes written to a VPOD before advancing to the next VPOD in the stripe. Must be a multiple of 16 KB (min: 16 KB, max: 4 MB). If not specified, a system default value is applied.
`user`	string, nullable	no	`null`	`0`, `1001`	Owner UID of the volume, as a numeric string (e.g. “0”, “1001”). Defaults to root (uid=0) if not specified. String name resolution and Windows SID are not supported.
`user_permissions`	`VolumePermission` (`none`, `read-only`, `write-only`, `execute-only`, `read-write`, `read-execute`, `write-execute`, `all`), nullable	no	`null`	`read-write`, `all`	Filesystem permissions for the owner user on the root of the volume. Allowed values: none, read-only, write-only, execute-only, read-write, read-execute, write-execute, all.

Field	Type	Required	Default	Example	Description
`volume_service_id`	string, nullable	no	`null`	`0x0400000000000005`	Identifier of the volume service that will manage this volume, in hexadecimal format (e.g., "0x0400000000000005"). This allows control over how volume management load is distributed across Directors. If not specified, the system automatically assigns the volume to an existing service.

Example

```
{
   "efsa_mode" : "retry",
   "pathname" : "/vol2",
   "raid_type" : "raid6+",
   "recovery_priority" : 50,
   "storageset_id" : 1
}
```

Responses

Status	Description	Schema
201	Volume created successfully	`VolumeCreateResponse`
400	Bad request. Custom validation error or invalid parameters, including a hard_quota_bytes too small to hold the new volume. The smallest quota that fits depends on the realm.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant or storageset not found.	-

Status	Description	Schema
409	Conflict. The volume pathname already exists, or the owning tenant holds no issued subnet address. A named tenant is served over its own network, so the realm manager refuses to create a volume for a tenant no client could reach it through: give the tenant a subnet and let one of its nodes report an address into it first.	-
422	Validation error. Returned for invalid request body.	-
500	Internal server error.	-
503	Service unavailable. A backend dependency the request needs is unreachable. The request was well formed and changed nothing, so the same call is worth retrying once the dependency is back; which one failed is recorded server-side, not in the response.	-
504	Gateway timeout. The storage backend did not answer in time. It cannot abort a create in progress, so the volume may exist, and may appear only later: list the volumes before retrying the create.	-

Response Body (201, `VolumeCreateResponse`)

Field	Type	Required	Default	Example	Description
`id`	integer	yes	n/a	`1`, `2`, `42`	Unique volume identifier assigned by RM

Example

```
{
   "id" : 1
}
```

`GET /api/v1/volumes/{volume_id}`

Get volume info

Returns details for a specific volume by its volume_id.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`volume_id`	path	integer	yes	ID of the volume
`tenant_id`	query	string (uuid), nullable	no	Platform tenant only. Read a volume of another tenant instead of your own. There is no all-tenants form here: a volume ID already names exactly one volume, so the only question is whose volume the caller may reach. Reads only - a write always stays in the caller's own tenant.

Responses

Status	Description	Schema
200	Successfully retrieved detailed information about the volume.	`VolumeGetResponse`
400	Bad request. Invalid volume_id parameter.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Volume not found.	-
422	Validation error for the volume_id path parameter.	-
500	Internal server error.	-

Response Body (200, `VolumeGetResponse`)

Field	Type	Required	Default	Example	Description
`volume`	`Volume` object	yes	n/a		Volume entity with all available details.

Example


```
{
   "volume" : {
      "efsa_mode" : "retry",
      "hard_quota_bytes" : 549755813888,
      "id" : 1,
      "max_width" : 12,
      "object_id" : "I-xD0200000000000005-xG82c418af-xU00004a75726a656e",
      "pathname" : "/",
      "quotas_policy" : "disable",
      "raid_type" : "raid6+",
      "recovery_priority" : 1,
      "rg_depth" : 4,
      "rg_width" : 8,
      "soft_quota_bytes" : 524288000000,
      "status" : "online",
      "storageset_id" : 1,
      "stripe_unit_bytes" : 262144,
      "tenant_id" : "00000000-0000-0000-0000-000000000001",
      "volume_service_id" : "0x0400000000000005"
   }
}
```

`PATCH /api/v1/volumes/{volume_id}`

Update volume

Updates an existing volume using PATCH semantics. Only the fields provided in the request are modified.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`volume_id`	path	integer	yes	ID of the volume
`ignore_limits`	query	boolean	no	Platform tenant only. Act without the limits the tenant's storageset assignment imposes: no assignment need exist, no quota ceiling applies, and free space is not checked, as before tenancy. For a create, the target storageset must still exist in the realm.

Request Body (`application/json` , required)

Schema: `VolumePatchRequest`

Request payload for updating an existing volume via PATCH.

Field	Type	Required	Default	Example	Description
`access_permission`	`VolumeAccessMode` (`read-only` , `read-write`), nullable	no	`null`	`read-only` , `read-write`	Volume access permission mode. Set to read-only for administrative purposes
`compression`	`VolumeCompressionMode` (`enable` , `disable`), nullable	no	`null`	`enable` , `disable`	Enable or disable compression for the volume. Affects new data only

Field	Type	Required	Default	Example	Description
`efsa_mode`	`EfsaMode` (`retry` , `file-unavailable` , `default`), nullable	no	`null`	`retry` , `file-unavailable` , `default`	EFSA mode for the volume. Defines behavior during partial storage unavailability. Supported: retry, file-unavailable, default. If not specified, system default (retry) is applied. Only for RAID 5+, 6+, 10+ volumes.
`hard_quota_bytes`	integer, nullable	no	`null`	`1099511627776` , `0`	Hard quota in bytes. Prevents additional writes when limit is reached. A volume in a tenant requires a value greater than 0; a request without one returns 400.
`recovery_priority`	integer, nullable	no	`null`	`0` , `1` , `50` , `100`	Recovery priority (1-100) of the volume. Lower values indicate higher recovery priority, where 1 represents the highest priority. If not specified the system default (50) is applied.
`soft_quota_bytes`	integer, nullable	no	`null`	`54975581388` , `0`	Soft quota limit in bytes. Triggers quota notifications when the threshold is exceeded. Use 0 for unlimited.
`state`	`VolumeStateOption` (`online` , `offline` , `paused`), nullable	no	`null`	`online` , `offline` , `paused`	Target operational state for the volume. Use to bring volume online/offline

Example


```
{
   "access_permission" : "read-write",
   "efsa_mode" : "file-unavailable",
   "hard_quota_bytes" : 1099511627776,
   "recovery_priority" : 75,
   "soft_quota_bytes" : 549755813888,
   "state" : "online"
}
```


Responses

Status	Description	Schema
204	Volume updated successfully.	-
400	Bad request. Invalid volume_id or custom validation error.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant or storageset not found.	-
409	Conflict. Pathname already exists or update conflicts with system state.	-
422	Validation error. Returned for invalid request body (all fields are optional).	-
500	Internal server error.	-

`DELETE /api/v1/volumes/{volume_id}`

Delete volume

Deletes the specified volume by its volume_id. The volume record is removed from the realm first, then its PanFS directory is removed. If the directory removal fails the volume is already unregistered and will not appear in listings; the orphaned directory at the reported path must be removed manually.

Before you call this. Needs: A token for the tenant that owns the volume (`POST /api/v1/auth/tenant-switch` as a Platform Admin, or a tenant admin). Otherwise: `404` "Volume not found" for a Platform Admin who has not switched into the tenant, the same as for `GET /api/v1/volumes/{volume_id}` . Find the IDs with `GET /volumes?tenant_id={tenant_id}` , which works without switching.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`volume_id`	path	integer	yes	ID of the volume

Responses

Status	Description	Schema
204	Volume deleted successfully.	-
400	Bad request. Invalid volume_id parameter.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Volume not found.	-
409	Volume backs an enabled tenant's S3 prefix and cannot be deleted.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

`POST /api/v1/volumes/{volume_id}/actions`

Perform action on volume

Performs a specified action (rename or recover).

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`volume_id`	path	integer	yes	ID of the volume

Request Body (`application/json` , required)

One of the following, selected by `action` :

`action: recover` - `VolumeRecoverActionRequest`

Field	Type	Required	Default	Example	Description
`action`	literal `recover`	yes	n/a		The action to perform on the volume.
`params`	`VolumeRecoverActionParams` object	yes	n/a		Parameters required to perform the recover action.

`action: rename` - `VolumeRenameActionRequest`

Field	Type	Required	Default	Example	Description
`action`	literal `rename`	yes	n/a		The action to perform on the volume.
`params`	`VolumeRenameActionParams` object	yes	n/a		Parameters required to perform the rename action.

Responses

Status	Description	Schema
204	Action performed successfully on the volume. No content is returned.	-
400	Bad request. Invalid volume_id or invalid action parameters.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Volume not found.	-
409	Conflict. Action conflicts with current system state.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

`GET /api/v1/volumes/{volume_id}/space`

Get volume space

Returns space usage for a specific volume: used bytes, hard/soft quotas, and free capacity.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`volume_id`	path	integer	yes	ID of the volume
`tenant_id`	query	string (uuid), nullable	no	Platform tenant only. Read a volume of another tenant instead of your own. There is no all-tenants form here: a volume ID already names exactly one volume, so the only question is whose volume the caller may reach. Reads only - a write always stays in the caller's own tenant.

Responses

Status	Description	Schema
200	Successful response containing space information for the volume.	`VolumeSpaceResponse`
400	Bad request. Validation error (e.g., invalid volume ID).	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Volume not found.	-
422	Validation error. Returned for invalid path parameters.	-
500	Internal server error.	-

Response Body (200, `VolumeSpaceResponse`)

Field	Type	Required	Default	Example	Description
`free_bytes`	integer	yes	n/a	`107374182400`, `0`	Remaining capacity within the hard quota (hard_quota_bytes - used_bytes).
`hard_quota_bytes`	integer, nullable	yes	n/a	`214748364800`, `549755813888`	Hard quota limit in bytes. This is the total capacity allocated to the volume.
`soft_quota_bytes`	integer, nullable	no	`null`	`193273528320`, `null`	Soft quota limit in bytes. Advisory threshold that triggers warnings. Null if not configured.
`used_bytes`	integer	yes	n/a	`107374182400`, `0`	Bytes consumed by this volume on disk.

Example

```
{
   "free_bytes" : 107374182400,
   "hard_quota_bytes" : 214748364800,
   "soft_quota_bytes" : 193273528320,
   "used_bytes" : 107374182400
}
```

11. StorageSets

The StorageSets that volumes are created within, their space usage, and StorageSet-level actions.

Endpoint Summary

Path	Summary	Method
`/api/v1/storagesets`	List storagesets	`GET`
`/api/v1/storagesets`	Create StorageSet	`POST`
`/api/v1/storagesets/{storageset_id}`	Get StorageSet details	`GET`
`/api/v1/storagesets/{storageset_id}`	Update StorageSet	`PATCH`
`/api/v1/storagesets/{storageset_id}`	Delete StorageSet	`DELETE`
`/api/v1/storagesets/{storageset_id}/actions`	Perform StorageSet Action	`POST`
`/api/v1/storagesets/{storageset_id}/space`	Get StorageSet space	`GET`

`GET /api/v1/storagesets`

List storagesets

Returns a list of storagesets accessible to the authenticated user.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Responses

Status	Description	Schema
200	Successful response containing the list of storagesets	`StorageSetListResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
500	Internal server error.	-

Response Body (200, `StorageSetListResponse`)

Field	Type	Required	Default	Example	Description
`storagesets`	array of `StorageSet` object	no	n/a		Array of StorageSet objects
`total`	integer	yes	n/a	`1` , `5` , `10`	Total count of StorageSets in this response.

Example


```
{
   "storagesets" : [
      {
         "capacity_balancing" : true,
         "encrypted" : false,
         "fips_mode" : false,
         "id" : 1,
         "mlec" : false,
         "name" : "Set 1",
         "nodes_not_responding" : 0,
         "nodes_offline" : 0,
         "nodes_online" : 6,
         "nodes_reconstructing" : 0,
         "nodes_total" : 6,
         "spares_available" : 2,
         "spares_requested" : 2,
         "state" : {
            "conditions" : [],
            "status" : "ok"
         },
         "topology_aware_placement" : false,
         "ure_default" : false,
         "vhosts_total" : 6
      }
   ],
   "total" : 1
}
```

`POST /api/v1/storagesets`

Create StorageSet

Creates a new StorageSet with the specified configuration.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Request Body (`application/json` , required)

Schema: `StorageSetCreateRequest`

Request payload for creating a StorageSet.

Requires a unique StorageSet name. Optional parameters use system defaults when omitted.

Field	Type	Required	Default	Example	Description
`flags`	array of `StorageSetFlag` (`ure_default` , `capacity_balancing_disabled` , `encrypted` , `fips_mode` , `topology_aware_placement` , `mlec`)	no	n/a	`["mlec"]` , `[]`	Flags to set during StorageSet creation.
`name`	string	yes	n/a	`Set 2` , `Production Set`	Unique display name of the StorageSet. Must be unique within the realm.
`spares_requested`	integer	no	`2`	`2` , `4` , `6`	Number of VPODs to be reserved for reconstruction within the new StorageSet. Default: 2.

Example


```
{
   "flags" : [
      "mlec"
   ],
   "name" : "Set 1",
   "spares_requested" : 4
}
```


Responses

Status	Description	Schema
201	StorageSet created successfully	`StorageSetCreateResponse`
400	Bad request. Invalid parameters or validation error.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
409	Conflict. StorageSet name already exists.	-
422	Validation error. Returned for invalid request body.	-
500	Internal server error.	-

Response Body (201, `StorageSetCreateResponse`)

Field	Type	Required	Default	Example	Description
`id`	integer	yes	n/a	`1`, `2`, `3`	Unique StorageSet identifier assigned by the system

Example


```
{
   "id" : 3
}
```

`GET /api/v1/storagesets/{storageset_id}`

Get StorageSet details

Returns details for a specific StorageSet by its ID.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`storageset_id`	path	integer	yes	StorageSet identifier

Responses

Status	Description	Schema
200	Successfully retrieved detailed information about the StorageSet.	`StorageSetGetResponse`
400	Bad request. Invalid storageset_id parameter.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	StorageSet not found.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (200, `StorageSetGetResponse`)

Field	Type	Required	Default	Example	Description
`storageset`	`StorageSet` object	yes	n/a		StorageSet entity with all available details.

Example


```
{
   "storageset" : {
      "capacity_balancing" : true,
      "encrypted" : false,
      "fips_mode" : false,
      "id" : 1,
      "mlec" : false,
      "name" : "Set 1",
      "nodes_not_responding" : 0,
      "nodes_offline" : 0,
      "nodes_online" : 6,
      "nodes_reconstructing" : 0,
      "nodes_total" : 6,
      "spares_available" : 2,
      "spares_requested" : 2,
      "state" : {
         "conditions" : [],
         "status" : "ok"
      },
      "topology_aware_placement" : false,
      "ure_default" : false,
      "vhosts_total" : 6
   }
}
```

`PATCH /api/v1/storagesets/{storageset_id}`

Update StorageSet

Updates an existing StorageSet using PATCH semantics. Only the fields provided in the request are modified.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`storageset_id`	path	integer	yes	StorageSet identifier

Request Body (`application/json` , required)

Schema: `StorageSetPatchRequest`

Request payload for updating an existing StorageSet via PATCH.

Field	Type	Required	Default	Example	Description
`capacity_balancing_disabled`	boolean, nullable	no	`null`	`true` , `false`	True to disable capacity balancing (set flag), False to enable it (clear flag).
`name`	string, nullable	no	`null`	`Set 3` , `Production Set`	Updated StorageSet name. Must be unique within the realm.
`spares_requested`	integer, nullable	no	`null`	`1` , `2` , `4`	Optional updated number of VPOD spares reserved for reconstruction. If provided, must be at least 1.

Example

```
{
   "capacity_balancing_disabled" : false,
   "name" : "Set 3",
   "spares_requested" : 3
}
```

Responses

Status	Description	Schema
204	StorageSet updated successfully.	-
400	Bad request. Invalid storageset_id or custom validation error.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	StorageSet not found.	-
409	Conflict. StorageSet update conflicts with system state.	-
422	Validation error. Returned for invalid request body or Path parameter.	-
500	Internal server error.	-

`DELETE /api/v1/storagesets/{storageset_id}`

Delete StorageSet

Deletes a StorageSet by its ID.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`storageset_id`	path	integer	yes	StorageSet identifier

Responses

Status	Description	Schema
204	StorageSet successfully deleted.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	StorageSet not found.	-
409	StorageSet cannot be deleted due to a conflict (e.g. it still contains volumes).	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

`POST /api/v1/storagesets/{storageset_id}/actions`

Perform StorageSet Action

Performs an action on a StorageSet. Currently supports “add_storage” action to add unassigned Storage Nodes (dataserials). The add_storage action supports “add_spare” mode to expand capacity with optional spare adjustment, or “migration” mode to add storage for draining another enclosure. A vhost adds its capacity later, not at once. Each of its VPODs joins the StorageSet after it installs and boots, which takes tens of minutes. Until the first VPOD joins, the StorageSet reports the “activating” condition.

Before you call this. Needs: For `add_storage` : list the nodes in `dataserials` , each unassigned and with a role of `vhost` , `vosd` , or `snode` . Otherwise: A call without `dataserials` adds nothing yet returns `204` ; a role such as `qnode` is refused. The call returns `204` before the storage is usable, so poll `GET /api/v1/storagesets/{storageset_id}` until `state.status` is `ok` .

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`storageset_id`	path	integer	yes	StorageSet identifier

Request Body (`application/json` , required)

Schema: `StorageSetAddStorageActionRequest`

Request to perform an add_storage action on a StorageSet.

Field	Type	Required	Default	Example	Description
`action`	literal `add_storage`	yes	n/a		The action to perform on the storageset.
`params`	`StorageSetAddStorageParams` object	yes	n/a		Parameters required to perform the add_storage action.

Responses

Status	Description	Schema
204	Action completed successfully.	-
400	Bad Request. Dataserials belong to non-storage nodes.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	StorageSet not found.	-
409	Conflict. Operation conflicts with current system state.	-
422	Validation error. Returned for invalid request body or Path parameter.	-
500	Internal server error.	-

`GET /api/v1/storagesets/{storageset_id}/space`

Get StorageSet space

Returns detailed space usage for a specific StorageSet: raw, usable, and reserved capacity buckets (each with total/free/used), plus volume/node counts, capacity-balancing state, and volume hard quota.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`storageset_id`	path	integer	yes	StorageSet identifier

Responses

Status	Description	Schema
200	Successful response containing space information for the StorageSet.	`StorageSetSpaceResponse`
400	Bad request. Validation error (e.g., invalid StorageSet ID).	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	StorageSet not found.	-
422	Validation error. Returned for invalid path parameters.	-
500	Internal server error.	-

Response Body (200, `StorageSetSpaceResponse`)

Field	Type	Required	Default	Example	Description
`capacity_balancing_active`	boolean	yes	n/a	`true` , `false`	Whether automatic capacity balancing is currently active for this StorageSet.

Field	Type	Required	Default	Example	Description
`node_count`	integer	yes	n/a	`12`, `6`	Number of OSD nodes (storage blades) in this StorageSet.
`raw`	`StorageSetSpaceCapacity` object	yes	n/a		Overall physical capacity across all buckets in this StorageSet.
`reserved`	`StorageSetSpaceCapacity` object	yes	n/a		Reconstruction reserve capacity. Held for spare rebuilds; not available for regular data storage.
`usable`	`StorageSetSpaceCapacity` object	yes	n/a		Capacity available for data storage (usable bucket).
`volume_hard_quota`	integer	yes	n/a	`524288000`, `0`	Sum of hard quotas for all volumes in this StorageSet, in bytes. A create is reflected immediately; after a delete the RM releases the quota asynchronously, so this figure can briefly overstate the true sum.

Example


```
{
   "capacity_balancing_active" : true,
   "node_count" : 6,
   "raw" : {
      "free" : 333884440576,
      "total" : 333900152832,
      "used" : 15712256
   },
   "reserved" : {
      "free" : 55650041856,
      "total" : 55650041856,
      "used" : 0
   },
   "usable" : {
      "free" : 278234398720,
      "total" : 278250110976,
      "used" : 15712256
   },
   "volume_hard_quota" : 524288000
}
```

12. Subnets and Attachments

This chapter covers Subnets and Host Attachments.

12.1 Subnets

Endpoint Summary

Path	Summary	Method
`/api/v1/subnets`	List every subnet in the realm	`GET`
`/api/v1/subnets/segment-type`	Get the network segment type new subnets are placed on	`GET`
`/api/v1/subnets/segment-type`	Set the network segment type new subnets are placed on	`PUT`
`/api/v1/tenants/{tenant_id}/subnets`	List a tenant's subnets	`GET`
`/api/v1/tenants/{tenant_id}/subnets`	Create a new subnet	`POST`
`/api/v1/tenants/{tenant_id}/subnets/{subnet_id}`	Get subnet info	`GET`
`/api/v1/tenants/{tenant_id}/subnets/{subnet_id}`	Update subnet	`PATCH`
`/api/v1/tenants/{tenant_id}/subnets/{subnet_id}`	Delete subnet	`DELETE`
`/api/v1/tenants/{tenant_id}/subnets/{subnet_id}/mount-info`	Get the mount targets on a subnet	`GET`

`GET /api/v1/subnets`

List every subnet in the realm

Returns every subnet the realm holds, whatever tenant owns it. To read one tenant instead, name it in the path: GET /tenants/{tenant_id}/subnets.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Responses

Status	Description	Schema
200	Successful response containing the list of subnets.	`SubnetListResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
500	Internal server error.	-

Response Body (200, `SubnetListResponse`)

Field	Type	Required	Default	Example	Description
`subnets`	array of `Subnet` object	no	n/a		Array of subnet objects. Empty array if none are found.
`total`	integer	yes	n/a	`0` , `1` , `12`	Total count of subnets returned.

Example


```
{
   "subnets" : [
      {
         "addr_count" : 12,
         "addr_external" : false,
         "created_at" : "2026-06-29T12:00:00Z",
         "gateway" : "192.168.50.1",
         "id" : "36a8995f-026b-47e1-88bd-7fb9e7239995",
         "ip_range_base" : "192.168.50.0",
         "ip_range_prefix" : 24,
         "mtu" : 1450,
         "name" : "storage-net",
         "range_end" : "192.168.50.200",
         "range_start" : "192.168.50.10",
         "segment_id" : 10,
         "segment_type" : "vlan",
         "tenant_id" : "00000000-0000-0000-0000-000000000001",
         "tunnel_udp_port" : 0,
         "updated_at" : "2026-06-29T12:30:00Z"
      }
   ],
   "total" : 1
}
```

`GET /api/v1/subnets/segment-type`

Get the network segment type new subnets are placed on

Returns the realm-wide network segment type new subnets are placed on.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Responses

Status	Description	Schema
200	The network segment type new subnets are placed on.	`RealmSegmentTypeResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
500	Internal server error.	-
503	Service unavailable. A backend dependency the request needs is unreachable. The request was well formed and changed nothing, so the same call is worth retrying once the dependency is back; which one failed is recorded server-side, not in the response.	-

Response Body (200, `RealmSegmentTypeResponse`)

Field	Type	Required	Default	Example	Description
`segment_type`	`SubnetSegmentType` (`none` , `vlan` , `vxlan`)	yes	n/a	`vxlan` , `vlan` , `none`	The segment type a new subnet is placed on. `vlan` carries a VLAN ID in `segment_id` and `vxlan` a VNI. `none` is the realm naming no segment at all, which is a state to report rather than an error: a create on such a realm is refused, with a reason.

Example

```
{
   "segment_type" : "vxlan"
}
```

`PUT /api/v1/subnets/segment-type`

Set the network segment type new subnets are placed on

Sets the realm-wide network segment type. Refused while any tenant has a subnet.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Request Body (`application/json` , required)

Schema: `RealmSegmentTypeRequest`

Set the network segment type new subnets are placed on.

Field	Type	Required	Default	Example	Description
`segment_type`	one of: `vlan` , `vxlan`	yes	n/a	`vlan` , `vxlan`	The segment type new subnets are placed on.

Example

```
{
   "segment_type" : "vlan"
}
```

Responses

Status	Description	Schema
204	Success. No content is returned.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
409	Conflict. The request conflicts with the current state of the resource.	-
422	Validation error.	-
500	Internal server error.	-

Status	Description	Schema
503	Service unavailable. A backend dependency the request needs is unreachable. The request was well formed and changed nothing, so the same call is worth retrying once the dependency is back; which one failed is recorded server-side, not in the response.	-

`GET /api/v1/tenants/{tenant_id}/subnets`

List a tenant's subnets

Returns the subnets the tenant named in the path owns.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID).

Responses

Status	Description	Schema
200	Successful response containing the list of subnets.	`SubnetListResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant not found.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (200, `SubnetListResponse`)

Field	Type	Required	Default	Example	Description
`subnets`	array of `Subnet` object	no	n/a		Array of subnet objects. Empty array if none are found.
`total`	integer	yes	n/a	`0` , `1` , `12`	Total count of subnets returned.

Example


```
{
   "subnets" : [
      {
         "addr_count" : 12,
         "addr_external" : false,
         "created_at" : "2026-06-29T12:00:00Z",
         "gateway" : "192.168.50.1",
         "id" : "36a8995f-026b-47e1-88bd-7fb9e7239995",
         "ip_range_base" : "192.168.50.0",
         "ip_range_prefix" : 24,
         "mtu" : 1450,
         "name" : "storage-net",
         "range_end" : "192.168.50.200",
         "range_start" : "192.168.50.10",
         "segment_id" : 10,
         "segment_type" : "vlan",
         "tenant_id" : "00000000-0000-0000-0000-000000000001",
         "tunnel_udp_port" : 0,
         "updated_at" : "2026-06-29T12:30:00Z"
      }
   ],
   "total" : 1
}
```

`POST /api/v1/tenants/{tenant_id}/subnets`

Create a new subnet

Creates a subnet owned by the tenant named in the path. Two things answer 404 here, with different detail text: a tenant the realm does not hold (“Owning tenant not found”), and a tenant the caller may not reach (“Tenant not found”), which is refused before the handler so the code cannot be used to establish which tenants exist. The platform tenant itself cannot own a subnet: it is given one with the realm and no other, so a create addressed to it is answered 409. The window must hold at least as many addresses as the realm primary pool, the node ipranges. A node takes the window address at its own position in that pool. A smaller window is answered 400, however few nodes the realm has.

Before you call this. Needs: The realm segment type is set (`PUT /api/v1/subnets/segment-type`), any `segment_id` you give is unused on the realm (leave it out and the realm assigns one), and the call comes from a platform role directly, not from a session switched into the tenant. Otherwise: `409` when the segment ID is held by another subnet or the realm has subnets on a different segment type; `403` from a switched-in session.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID).

Request Body (`application/json` , required)

Schema: `SubnetCreateRequest`

Request payload for creating a subnet.

Field	Type	Required	Default	Example	Description
`addr_external`	boolean	no	`false`	`false` , `true`	Selects the address-allocation model. False leaves PanFS managing the addresses, true hands that to an outside allocator. Fixed at create and cannot be changed afterwards.

Field	Type	Required	Default	Example	Description
`gateway`	string, nullable	no	`null`	`192.168.50.1`, `null`	Optional default gateway address, in the same family as ip_range_base.
`ip_range_base`	string	yes	n/a	`192.168.50.0`, `fd00:dead:beef::`	Base IP address of the subnet range (IPv4 or IPv6).
`ip_range_prefix`	integer	yes	n/a	`24`, `64`	CIDR prefix length. 1-32 for IPv4, 1-128 for IPv6.
`mtu`	integer	no	`0`	`0`, `1450`, `8950`	The MTU of the subnet's tenant interface on every realm node and every attached host. 0 selects the default, which is the realm's primary network MTU less the encapsulation overhead (50 bytes on vxlan, 4 bytes on vlan): 1450 on vxlan over a 1500-byte primary network. Any other value, from 68 to 9216, has to leave that overhead within the primary network MTU. A larger one is refused with 400 naming the largest value that fits.
`name`	string	yes	n/a	`storage-net`, `tenant_a-vlan10`	Subnet name, unique per tenant. 3-31 chars from [A-Za-z0-9_-].

Field	Type	Required	Default	Example	Description
`range_end`	string, nullable	no	`null`	`192.168.50.200`, `null`	Upper bound of the usable-address window, which is the range the realm allocates its own node addresses from. Supply it together with range_start, or omit both and let the realm derive the window, as range_start describes.
`range_start`	string, nullable	no	`null`	`192.168.50.10`, `null`	Lower bound of the usable-address window, which is the range the realm allocates its own node addresses from. The window must hold at least as many addresses as the realm primary pool. A host attached from outside the realm is placed outside this window. Supply it together with range_end, or omit both and the realm derives the window from the first usable host, as deep as its primary pool, so the subnet as stored reports a window this request did not carry. The realm refuses with 400 when it cannot read that pool, and then the window has to be named. Nothing is derived for a subnet whose addresses an outside allocator owns.

Field	Type	Required	Default	Example	Description
`segment_id`	integer	no	`0`	`0`, `10`, `100100`	Network segment identifier: a VLAN ID when the realm runs 802.1Q and a VNI when it runs VXLAN. If you leave it out, the realm assigns an unused identifier; 0 leaves it unassigned. Otherwise the bound here is the wider of the two spaces, 1-16777215. The realm manager owns the narrower range that follows the realm segment type, and the uniqueness of the identifier across the realm. Fixed at create and cannot be changed afterwards.
`tunnel_udp_port`	integer	no	`0`	`0`, `4789`	UDP port carrying a tunnelled segment, which is meaningful when the realm runs VXLAN. 0 leaves it unset and the realm manager applies its own default. Otherwise the bound is 1-65535. Fixed at create and cannot be changed afterwards.

Example

```
{
   "addr_external" : false,
   "gateway" : "192.168.50.1",
   "ip_range_base" : "192.168.50.0",
   "ip_range_prefix" : 24,
   "mtu" : 1450,
   "name" : "storage-net",
   "range_end" : "192.168.50.200",
   "range_start" : "192.168.50.10",
   "segment_id" : 10,
   "tunnel_udp_port" : 0
}
```

Responses

Status	Description	Schema
201	Subnet created successfully.	`SubnetCreateResponse`
400	Bad request. The realm manager refused the subnet: a gateway outside the range, a window the stored CIDR does not hold, a window smaller than the realm primary pool, no window named on a realm whose primary pool cannot be read to derive one, an unknown allocation flag, the default tenant as owner, a segment identifier offered while the realm names no segment type, or a segment identifier outside the space that type defines. An mtu larger than the realm's primary network MTU less the segment's encapsulation overhead is refused here too, before the realm manager sees it, with the largest value that fits named in the detail. Input the request schema can judge for itself is refused before the realm manager sees it and comes back as 422.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant not found.	-

Status	Description	Schema
409	Conflict. The name is already taken, the segment identifier is already held by another subnet, or the realm already carries subnets on a different network segment type. The name is unique within the owning tenant, while the segment identifier is unique across the whole realm, because the identifier space belongs to the physical interface every tenant on a node shares. One realm carries one segment type, so a create lands on the type the realm already holds. A create addressed to the platform tenant answers this too: it owns its subnet with the realm and is given no other. A create also answers this when the realm could not record an address for the new subnet on every node, because every node takes one address on every tenant subnet and the realm caps how many it holds in all.	-
422	Validation error. The request schema settles the name charset and length, the address family, the prefix, the MTU bounds, the segment-identifier and encapsulation-port bounds, and every usable-window rule a single request can judge on its own. A cross-field refusal reports `body` as its location and names the offending field in the message. `segment_type` and `segment_flags` land here too, because the realm owns them and the create schema forbids fields it does not declare.	-
500	Internal server error.	-
503	Service unavailable. A backend dependency the request needs is unreachable. The request was well formed and changed nothing, so the same call is worth retrying once the dependency is back; which one failed is recorded server-side, not in the response.	-

Response Body (201, `SubnetCreateResponse`)

Field	Type	Required	Default	Example	Description
`id`	string (uuid)	yes	n/a		Server-assigned unique identifier of the created subnet.

Example


```
{
   "id" : "36a8995f-026b-47e1-88bd-7fb9e7239995"
}
```

`GET /api/v1/tenants/{tenant_id}/subnets/{subnet_id}`

Get subnet info

Returns details for a specific subnet by its UUID. A subnet outside the tenant named in the path is answered 404, so the code cannot be used to establish that one exists.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID).
`subnet_id`	path	string (uuid)	yes	UUID of the subnet

Responses

Status	Description	Schema
200	Successfully retrieved detailed information about the subnet.	`SubnetGetResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Subnet not found.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (200, `SubnetGetResponse`)

Field	Type	Required	Default	Example	Description
`subnet`	`Subnet` object	yes	n/a		Subnet entity with all available details.

Example


```
{
   "subnet" : {
      "addr_count" : 12,
      "addr_external" : false,
      "created_at" : "2026-06-29T12:00:00Z",
      "gateway" : "192.168.50.1",
      "id" : "36a8995f-026b-47e1-88bd-7fb9e7239995",
      "ip_range_base" : "192.168.50.0",
      "ip_range_prefix" : 24,
      "mtu" : 1450,
      "name" : "storage-net",
      "range_end" : "192.168.50.200",
      "range_start" : "192.168.50.10",
      "segment_id" : 10,
      "segment_type" : "vlan",
      "tenant_id" : "00000000-0000-0000-0000-000000000001",
      "tunnel_udp_port" : 0,
      "updated_at" : "2026-06-29T12:30:00Z"
   }
}
```

`PATCH /api/v1/tenants/{tenant_id}/subnets/{subnet_id}`

Update subnet

Updates an existing subnet using PATCH semantics. Only the fields provided in the request are modified, and a field provided as null is cleared. An empty patch is rejected with HTTP 422. The subnet must belong to the tenant named in the path: one owned by another tenant is answered 404.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID).
`subnet_id`	path	string (uuid)	yes	UUID of the subnet

Request Body (`application/json` , required)

Schema: `SubnetPatchRequest`

Request payload for updating a subnet via PATCH (partial update).

Only the supplied fields are modified, and a patch carrying no field at all is refused here, so it answers 422 with the same envelope every other patch schema uses. A field supplied as null is cleared, which is how a gateway is removed and how a usable window is withdrawn by clearing both of its endpoints. `name` and `mtu` carry no meaning when cleared, so an explicit null is refused for them.

`addr_external` is absent by design: the flag is fixed at create, and `extra='forbid'` refuses a request that carries it. The four network segment values are absent for the same reason, and a patch naming any of them is refused with that field in the error location.

Field	Type	Required	Default	Example	Description
`gateway`	string, nullable	no	`null`	`192.168.50.254` , `null`	New default gateway address (IPv4 or IPv6). Null clears the gateway.

Field	Type	Required	Default	Example	Description
`mtu`	integer, nullable	no	`null`	`1450`, `8950`	New MTU. The MTU of the subnet's tenant interface on every realm node and every attached host. 0 selects the default, which is the realm's primary network MTU less the encapsulation overhead (50 bytes on vxlan, 4 bytes on vlan): 1450 on vxlan over a 1500-byte primary network. Any other value, from 68 to 9216, has to leave that overhead within the primary network MTU. A larger one is refused with 400 naming the largest value that fits.
`name`	string, nullable	no	`null`		
`range_end`	string, nullable	no	`null`	`192.168.50.220`, `null`	New upper bound of the usable-address window. Patchable on its own. Null clears it, and clearing both bounds together has the realm derive the window anew, as range_start describes.

Field	Type	Required	Default	Example	Description
`range_start`	string, nullable	no	`null`	`192.168.50.20`, `null`	New lower bound of the usable-address window. Patchable on its own, and the realm manager validates the resulting pair against the subnet CIDR. Null clears it. Clearing both bounds together has the realm derive the window anew, on the same terms a create naming none gets one, and is refused with 409 while a subnet the realm addresses itself has an attached host.

Example

```
{
   "gateway" : "192.168.50.254",
   "mtu" : 1500,
   "name" : "storage-net-2",
   "range_end" : "192.168.50.220",
   "range_start" : "192.168.50.20"
}
```

Responses

Status	Description	Schema
204	Subnet updated successfully.	-

Status	Description	Schema
400	Bad request. The realm manager refused the result of moving one window endpoint against the record it stores, including a window left smaller than the realm primary pool, or a window cleared on a realm whose primary pool cannot be read to derive a new one. An mtu larger than the realm's primary network MTU less the subnet's encapsulation overhead is refused before the realm manager sees it, with the largest value that fits named in the detail. A patch the request schema can judge for itself comes back as 422.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Subnet not found.	-
409	Conflict. The name is already taken, the window overlaps another subnet of the same tenant, or it would strand an address already bound to the subnet. The platform tenant's own subnet answers this too: it comes with the realm. A write of the window or the gateway answers this when the subnet it leaves behind would reserve an address one of the subnet's attached hosts already holds. Clearing range_start and range_end has the realm derive the window anew, which cannot be known in advance, so it answers this whenever the subnet has an attached host. A subnet whose addresses an outside allocator owns reserves nothing for the realm, so a write to it never answers this for an attached host.	-
422	Validation error. A patch carrying no field at all, a field that is fixed at create, a null on a field that cannot be cleared, or a window whose own two endpoints contradict each other. `addr_external` and the four network segment values land here because the patch schema forbids fields it does not declare.	-
500	Internal server error.	-
503	Service unavailable. A backend dependency the request needs is unreachable. The request was well formed and changed nothing, so the same call is worth retrying once the dependency is back; which one failed is recorded server-side, not in the response.	-

`DELETE /api/v1/tenants/{tenant_id}/subnets/{subnet_id}`

Delete subnet

Deletes the specified subnet by its UUID. The realm manager removes every address bound to the subnet in the same transaction, so bound addresses do not refuse the delete. The subnet must belong to the tenant named in the path: one owned by another tenant is answered 404.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID).
`subnet_id`	path	string (uuid)	yes	UUID of the subnet

Responses

Status	Description	Schema
204	Subnet deleted successfully.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Subnet not found.	-
409	Conflict. This is the last subnet of a tenant that still owns volumes, whose addresses it carries. Delete those volumes, or create another subnet for the tenant, first. The platform tenant's own subnet answers this too: it comes with the realm.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

`GET /api/v1/tenants/{tenant_id}/subnets/{subnet_id}/mount-info`

Get the mount targets on a subnet

Returns the realm Directors holding an address on this subnet, together with the address list a client puts in its mount options as rmlist=(). A client inside a tenant network has no route to the primary network and no access to the realm CLI, so this call is where the addresses it mounts against come from. The volume name is not a parameter of this call, so no assembled mount URL is returned: GET /volumes already names the volumes.

Before you call this. Needs: The subnet exists and a realm node has reported an address into it. Call it with a Platform Admin token that is not switched into a tenant. Otherwise: `rmlist` stays empty until then (an empty `directors` array with an empty `rmlist` is a legitimate answer); `403` for a Tenant Admin and through the tenant switcher; `503` when the realm manager cannot be reached.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID).
`subnet_id`	path	string (uuid)	yes	UUID of the subnet

Responses

Status	Description	Schema
200	Successfully retrieved the realm Directors a client on this subnet can mount to. An empty `directors` array with an empty `rmlist` is a legitimate answer: the subnet exists and the realm holds no address on it yet, which is one of the states a subnet is in. A realm provisioning operation records those addresses. An empty `rmlist` beside a populated `directors` array is legitimate too, and it says the recorded addresses are ones `mount_panfs` cannot bootstrap against.	`MountInfoResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Subnet not found.	-

Status	Description	Schema
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-
503	Service unavailable. The realm manager could not be reached, or it kept changing its address table for long enough that no consistent snapshot could be read.	-

Response Body (200, `MountInfoResponse`)

Field	Type	Required	Default	Example	Description
`directors`	array of `DirectorEntry` object	no	n/a		The realm Directors holding an address on this subnet, nodes carrying a realm-database replica first and then ordered by address. Every such Director appears here, in either address family. An empty array means the subnet exists and carries no address records yet.
`rmlist`	string	yes	n/a	`10.20.0.11;10.20.0.12`	The addresses for mount_panfs, joined with ";" in the order above and wrapped by the caller as rmlist=(). It narrows the Directors array to replica nodes, which is what the option means and what keeps it under the five addresses mount_panfs accepts, and to IPv4, which is all that grammar can express. Empty when no replica node holds an IPv4 address here, even when Directors is not empty.

Field	Type	Required	Default	Example	Description
`subnet_id`	string (uuid)	yes	n/a		UUID of the subnet the addresses belong to.
`tenant_id`	string (uuid)	yes	n/a		UUID of the tenant owning the subnet.

Example


```
{
   "directors" : [
      {
         "dataserial" : "SN-D01",
         "ip" : "10.20.0.11",
         "repset" : true,
         "status" : "online"
      },
      {
         "dataserial" : "SN-D02",
         "ip" : "10.20.0.12",
         "repset" : true,
         "status" : "online"
      },
      {
         "dataserial" : "SN-D03",
         "ip" : "10.20.0.13",
         "repset" : false,
         "status" : "online"
      }
   ],
   "rmlist" : "10.20.0.11;10.20.0.12",
   "subnet_id" : "36a8995f-026b-47e1-88bd-7fb9e7239995",
   "tenant_id" : "e97b94d6-9cc1-4325-a267-1064ae538ccd"
}
```

12.2 Host Attachments

Attaching and detaching individual hosts to a subnet's tenant network.

Endpoint Summary

Path	Summary	Method
`/api/v1/tenants/{tenant_id}/subnets/{subnet_id}/attachments`	List the subnet's host attachments	`GET`
`/api/v1/tenants/{tenant_id}/subnets/{subnet_id}/attachments`	Attach a host to the subnet's tenant network	`POST`
`/api/v1/tenants/{tenant_id}/subnets/{subnet_id}/attachments/{attachment_id}`	Get attachment detail	`GET`
`/api/v1/tenants/{tenant_id}/subnets/{subnet_id}/attachments/{attachment_id}`	Detach a host from the subnet	`DELETE`

`GET /api/v1/tenants/{tenant_id}/subnets/{subnet_id}/attachments`

List the subnet's host attachments

Returns the hosts attached to the subnet. Supply the underlay_ip query parameter to narrow the listing to one host, which is how a host that knows only its own address finds its record in a single call.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID).
`subnet_id`	path	string (uuid)	yes	UUID of the subnet the host attaches to
`underlay_ip`	query	string, nullable	no	Narrow the listing to the host with this address on the realm underlay.

Responses

Status	Description	Schema
200	Successful response containing the subnet's attachments. Narrowed to a single host when the `underlay_ip` query parameter is given, which is how a host that knows only its own address finds its record in one call.	`AttachmentListResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Not found. No such subnet or no such attachment, and equally a subnet or an attachment owned by another tenant. The two are one answer on purpose, so a response code cannot be used to establish that a record exists where the caller cannot see it.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-
503	Service unavailable. The attachment store is unreachable, so the record could neither be read nor written. The request itself was well formed, which is why this is a 5xx.	-

Response Body (200, `AttachmentListResponse`)

Field	Type	Required	Default	Example	Description
`attachments`	array of `Attachment` object	no	n/a		Array of attachment objects. Empty array if none are found.
`total`	integer	yes	n/a	`0`, `1`, `12`	Total count of attachments returned.

Example


```
{
   "attachments" : [
      {
         "hostname" : "client-0",
         "id" : "8f14e45f-ceea-467a-9575-5c4bcbdcd2f1",
         "joined_at" : "2026-08-28T12:00:00Z",
         "mtu" : 1450,
         "overlay_ip" : "192.168.50.100",
         "overlay_prefix" : 24,
         "segment_id" : 4257216,
         "segment_type" : "vxlan",
         "subnet_id" : "36a8995f-026b-47e1-88bd-7fb9e7239995",
         "tenant_id" : "00000000-0000-0000-0000-000000000001",
         "tunnel" : {
            "mac" : "92:b2:99:59:f3:fa",
            "udp_port" : 4789,
            "vteps" : [
               "10.128.1.0",
               "10.128.1.1",
               "10.128.1.2"
            ]
         },
         "underlay_ip" : "10.128.9.0"
      }
   ],
   "total" : 1
}
```

`POST /api/v1/tenants/{tenant_id}/subnets/{subnet_id}/attachments`

Attach a host to the subnet's tenant network

Records the host and allocates its address on the tenant network. The response carries the whole record, so the host builds its interface from it with no further call. A host already attached to this subnet answers 200 with the stored record and an unchanged address.

Before you call this. Needs: The subnet exists. Platform roles only. Otherwise: `404` when the subnet does not exist; `403` for a Tenant Admin; `503` when the realm manager cannot be reached.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID).
`subnet_id`	path	string (uuid)	yes	UUID of the subnet the host attaches to

Request Body (`application/json` , required)

Schema: `AttachmentCreateRequest`

Request body for attaching one host to a subnet.

Field	Type	Required	Default	Example	Description
`hostname`	string	no	(empty string)	`client-0`	Informational host name, carried for the operator and never interpreted.
`overlay_ip`	string, nullable	no	`null`	`192.168.50.100`	Requested address on the tenant network. Must sit inside the subnet, outside the realm node pool and be unallocated. Omit to let the backend allocate one. Applied on first attach only, so a repeated create never moves a host.

Field	Type	Required	Default	Example	Description
`underlay_ip`	string	yes	n/a	`10.128.9.0`	The host's address on the realm underlay. Identifies the attachment inside its subnet, and on a tunnelled segment it is the endpoint the realm sends the host traffic to.

Example

```
{
   "hostname" : "client-0",
   "overlay_ip" : "192.168.50.100",
   "underlay_ip" : "10.128.9.0"
}
```

Responses

Status	Description	Schema
200	The host is attached already, and the stored record is returned unchanged. A repeated create never moves a host, so an address named in the request is not applied to a record that exists: changing a host address is a delete followed by a create.	-
201	Host attached. The response carries the whole record, so the host builds its interface from it with no further call. `tunnel` is null on a segment the fabric carries itself.	`Attachment`
400	Bad request. The named address sits outside the subnet or, on a subnet the realm addresses itself, inside the range_start-range_end window the realm allocates its own node addresses from or on the gateway the subnet's router holds. It is equally a request that named no address at all on a subnet whose addresses an outside allocator owns, where the realm has none to give. Input the request schema can judge for itself is refused before this and comes back as 422.	-
401	Unauthorized. Authentication required.	-

Status	Description	Schema
403	Forbidden. Insufficient permissions.	-
404	Not found. No such subnet or no such attachment, and equally a subnet or an attachment owned by another tenant. The two are one answer on purpose, so a response code cannot be used to establish that a record exists where the caller cannot see it.	-
409	Conflict. The named address is already allocated in this subnet, the subnet has no free address left outside the window the realm allocates its own node addresses from, or it declares no window at all, so any usable address may be a node's. It may also already hold the maximum number of attachments.	-
422	Validation error. The request schema settles that each address parses and that the body carries no field it does not declare. The link-layer address, the segment identifier, and the realm node list are assigned by the backend, so naming any of them lands here.	-
500	Internal server error.	-
503	Service unavailable. The attachment store is unreachable, so the record could neither be read nor written. The request itself was well formed, which is why this is a 5xx.	-

Response Body (201, `Attachment`)

Field	Type	Required	Default	Example	Description
`hostname`	string	no	(empty string)		Host name as supplied at create. Informational.
`id`	string (uuid)	yes	n/a		Unique identifier of the attachment.
`joined_at`	string (date-time)	yes	n/a		First attach time (UTC).

Field	Type	Required	Default	Example	Description
`mtu`	integer	yes	n/a		The MTU the host sets on its tenant interface, which is the MTU the realm nodes run the subnet at: the subnet's declared MTU, or the realm's primary network MTU less the encapsulation overhead when the subnet declares 0. Never 0.
`overlay_ip`	string	yes	n/a		The host's address on the tenant network.
`overlay_prefix`	integer	yes	n/a		CIDR prefix length of the subnet, which the host applies to its own address.
`segment_id`	integer	yes	n/a		The subnet VLAN ID under one segment type and its VNI under the other. 0 means unassigned.
`segment_type`	`SubnetSegmentType` (`none` , `vlan` , `vxlan`)	yes	n/a		Kind of network segment the subnet is placed on, as the realm has it configured.
`subnet_id`	string (uuid)	yes	n/a		UUID of the subnet the host is attached to.
`tenant_id`	string (uuid)	yes	n/a		UUID of the tenant owning the subnet.
`tunnel`	`AttachmentTunnel` object, nullable	no	`null`		Tunnel recipe, present only when the segment is tunnelled and null otherwise.

Field	Type	Required	Default	Example	Description
`underlay_ip`	string	yes	n/a	`10.128.9.0`	The host's address on the realm underlay. Identifies the attachment inside its subnet, and on a tunnelled segment it is the endpoint the realm sends the host traffic to.

Example


```
{
   "hostname" : "client-0",
   "id" : "8f14e45f-ceea-467a-9575-5c4bcbdcd2f1",
   "joined_at" : "2026-08-28T12:00:00Z",
   "mtu" : 1450,
   "overlay_ip" : "192.168.50.100",
   "overlay_prefix" : 24,
   "segment_id" : 4257216,
   "segment_type" : "vxlan",
   "subnet_id" : "36a8995f-026b-47e1-88bd-7fb9e7239995",
   "tenant_id" : "00000000-0000-0000-0000-000000000001",
   "tunnel" : {
      "mac" : "92:b2:99:59:f3:fa",
      "udp_port" : 4789,
      "vteps" : [
         "10.128.1.0",
         "10.128.1.1",
         "10.128.1.2"
      ]
   },
   "underlay_ip" : "10.128.9.0"
}
```

`GET /api/v1/tenants/{tenant_id}/subnets/{subnet_id}/attachments/{attachment_id}`

Get attachment detail

Returns one attachment of the subnet by its UUID.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID).
`subnet_id`	path	string (uuid)	yes	UUID of the subnet the host attaches to
`attachment_id`	path	string (uuid)	yes	UUID of the attachment

Responses

Status	Description	Schema
200	Successfully retrieved detailed information about the attachment.	`Attachment`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Not found. No such subnet or no such attachment, and equally a subnet or an attachment owned by another tenant. The two are one answer on purpose, so a response code cannot be used to establish that a record exists where the caller cannot see it.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-
503	Service unavailable. The attachment store is unreachable, so the record could neither be read nor written. The request itself was well formed, which is why this is a 5xx.	-

Response Body (200, `Attachment`)

Field	Type	Required	Default	Example	Description
`hostname`	string	no	(empty string)		Host name as supplied at create. Informational.
`id`	string (uuid)	yes	n/a		Unique identifier of the attachment.
`joined_at`	string (date-time)	yes	n/a		First attach time (UTC).
`mtu`	integer	yes	n/a		The MTU the host sets on its tenant interface, which is the MTU the realm nodes run the subnet at: the subnet's declared MTU, or the realm's primary network MTU less the encapsulation overhead when the subnet declares 0. Never 0.
`overlay_ip`	string	yes	n/a		The host's address on the tenant network.
`overlay_prefix`	integer	yes	n/a		CIDR prefix length of the subnet, which the host applies to its own address.
`segment_id`	integer	yes	n/a		The subnet VLAN ID under one segment type and its VNI under the other. 0 means unassigned.
`segment_type`	`SubnetSegmentType` (`none`, `vlan`, `vxlan`)	yes	n/a		Kind of network segment the subnet is placed on, as the realm has it configured.
`subnet_id`	string (uuid)	yes	n/a		UUID of the subnet the host is attached to.

Field	Type	Required	Default	Example	Description
`tenant_id`	string (uuid)	yes	n/a		UUID of the tenant owning the subnet.
`tunnel`	`AttachmentTunnel` object, nullable	no	`null`		Tunnel recipe, present only when the segment is tunnelled and null otherwise.
`underlay_ip`	string	yes	n/a	`10.128.9.0`	The host's address on the realm underlay. Identifies the attachment inside its subnet, and on a tunnelled segment it is the endpoint the realm sends the host traffic to.

Example


```
{
   "hostname" : "client-0",
   "id" : "8f14e45f-ceea-467a-9575-5c4bcbdcd2f1",
   "joined_at" : "2026-08-28T12:00:00Z",
   "mtu" : 1450,
   "overlay_ip" : "192.168.50.100",
   "overlay_prefix" : 24,
   "segment_id" : 4257216,
   "segment_type" : "vxlan",
   "subnet_id" : "36a8995f-026b-47e1-88bd-7fb9e7239995",
   "tenant_id" : "00000000-0000-0000-0000-000000000001",
   "tunnel" : {
      "mac" : "92:b2:99:59:f3:fa",
      "udp_port" : 4789,
      "vteps" : [
         "10.128.1.0",
         "10.128.1.1",
         "10.128.1.2"
      ]
   },
   "underlay_ip" : "10.128.9.0"
}
```

`DELETE /api/v1/tenants/{tenant_id}/subnets/{subnet_id}/attachments/{attachment_id}`

Detach a host from the subnet

Removes the attachment record and frees the address it held. The call programs nothing, so there is no outcome to report: the realm withdraws the forwarding state on its own, after which the host is unreachable on the tenant network even if it keeps its interface.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID).
`subnet_id`	path	string (uuid)	yes	UUID of the subnet the host attaches to
`attachment_id`	path	string (uuid)	yes	UUID of the attachment

Responses

Status	Description	Schema
204	Host detached. The call removes the record and programs nothing, so there is no outcome to report. The realm withdraws the forwarding state on its own.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Not found. No such subnet or no such attachment, and equally a subnet or an attachment owned by another tenant. The two are one answer on purpose, so a response code cannot be used to establish that a record exists where the caller cannot see it.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-
503	Service unavailable. The attachment store is unreachable, so the record could neither be read nor written. The request itself was well formed, which is why this is a 5xx.	-

13. Nodes and Hardware

This chapter covers Nodes and Hardware.

13.1 Nodes

Endpoint Summary

Path	Summary	Method
`/api/v1/nodes`	List nodes	`GET`
`/api/v1/nodes/{dataserial}`	Get node info	`GET`
`/api/v1/nodes/{dataserial}/actions`	Perform action on node	`POST`
`/api/v1/nodes/{dataserial}/drain`	Get OSD drain status	`GET`
`/api/v1/nodes/{dataserial}/drain`	Start or stop OSD drain	`POST`

`GET /api/v1/nodes`

List nodes

Returns a list of nodes accessible to the authenticated user.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Responses

Status	Description	Schema
200	Successful response containing the list of nodes	`NodeListResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
500	Internal server error.	-

Response Body (200, `NodeListResponse`)

Field	Type	Required	Default	Example	Description
`nodes`	array of `Node` object	no	n/a		Array of node objects matching the query. Empty array if no nodes found
`total`	integer	yes	n/a	`0` , `2` , `150`	Total count of nodes available (may exceed returned array size if paginated).

Example


```
{
   "nodes" : [
      {
         "capacity_bytes" : 55650025472,
         "dataserial" : "e9ff4075d18233",
         "flags" : [
            "repset",
            "activated"
         ],
         "hw_serial" : "3e7e0000310fff001",
         "ip_address" : "10.97.18.200",
         "name" : "ASD-1,1",
         "role" : "any_director",
         "shelf_id" : "3e7e0001402001001",
         "status" : "online",
         "storageset_id" : 1,
         "vhost_id" : "bf806fa0c17feb"
      }
   ],
   "total" : 1
}
```

`GET /api/v1/nodes/{dataserial}`

Get node info

Returns details for a specific node by its dataserial identifier.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`dataserial`	path	string	yes	dataserial to retrieve

Responses

Status	Description	Schema
200	Successfully retrieved detailed information about the node.	`NodeGetResponse`
400	Bad request. Invalid dataserial parameter.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Node not found.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (200, `NodeGetResponse`)

Field	Type	Required	Default	Example	Description
`node`	`Node` object	yes	n/a		Node entity with all available details.

Example


```
{
   "node" : {
      "capacity_bytes" : 55650025472,
      "dataserial" : "e9ff4075d18233",
      "flags" : [
         "repset",
         "activated"
      ],
      "hw_serial" : "3e7e0000310fff001",
      "ip_address" : "10.97.18.200",
      "name" : "ASD-1,1",
      "role" : "any_director",
      "shelf_id" : "3e7e0001402001001",
      "status" : "online",
      "storageset_id" : 1,
      "vhost_id" : "bf806fa0c17feb"
   }
}
```

`POST /api/v1/nodes/{dataserial}/actions`

Perform action on node

Performs a specified action on a node (see request body for supported actions).

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`dataserial`	path	string	yes	Node dataserial to perform action on

Request Body (`application/json` , required)

One of the following, selected by `action` :

`action: halt` - `NodeHaltActionRequest`

Field	Type	Required	Default	Example	Description
`action`	literal `halt`	yes	n/a		The action to perform on the node.
`params`	null (not used)	no	n/a		

`action: reboot` - `NodeRebootActionRequest`

Field	Type	Required	Default	Example	Description
`action`	literal `reboot`	yes	n/a		The action to perform on the node.
`params`	null (not used)	no	n/a		

Responses

Status	Description	Schema
204	Action performed successfully on the node. No content is returned.	-

Status	Description	Schema
400	Bad request. Invalid dataserial or invalid action parameters.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Node not found.	-
409	Conflict. Action conflicts with current system state.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

`GET /api/v1/nodes/{dataserial}/drain`

Get OSD drain status

Returns drain progress for a node: byte and object counts, rate, state, percent complete, and ETA.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`dataserial`	path	string	yes	Node dataserial to query drain status for

Responses

Status	Description	Schema
200	Successfully retrieved drain progress for the node.	`NodeDrainStatusResponse`
400	Bad request. Invalid dataserial parameter.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Node not found.	-
409	Conflict. The node is not currently draining, so there is no drain progress to report.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (200, `NodeDrainStatusResponse`)

Field	Type	Required	Default	Example	Description
`amount_error`	integer	yes	n/a		Bytes that could not be migrated.
`amount_moved`	integer	yes	n/a		Bytes already migrated off the node.

Field	Type	Required	Default	Example	Description
`amount_remaining`	integer	yes	n/a		Bytes still to be migrated off the node.
`eta_seconds`	integer, nullable	yes	n/a	`51`, `0`, `null`	Estimated seconds to completion at the current rate, or `null` when the rate is zero.
`objects_error`	integer	yes	n/a		Objects that could not be migrated.
`objects_moved`	integer	yes	n/a		Objects already migrated off the node.
`objects_remaining`	integer	yes	n/a		Objects still to be migrated off the node.
`percent_complete`	number, nullable	yes	n/a	`50`, `100`, `null`	Percentage of bytes migrated, or `null` when there is nothing to drain.
`pm_id`	integer	yes	n/a		Identifier of the performance manager (PM) reporting this drain.
`rate_bytes_per_sec`	integer	yes	n/a		Current data-movement rate for the drain, in bytes per second.
`state`	`NodeDrainState` (`draining`, `not_draining`, `suspended`, `unknown`)	yes	n/a		High-level state of the drain operation.

Example


```
{
   "amount_error" : 0,
   "amount_moved" : 5368709120,
   "amount_remaining" : 5368709120,
   "eta_seconds" : 51,
   "objects_error" : 0,
   "objects_moved" : 1024,
   "objects_remaining" : 1024,
   "percent_complete" : 50,
   "pm_id" : 4,
   "rate_bytes_per_sec" : 104857600,
   "state" : "draining"
}
```

`POST /api/v1/nodes/{dataserial}/drain`

Start or stop OSD drain

Starts or stops draining data off a Storage Node (OSD). Draining migrates all data off the node onto other nodes in the same StorageSet.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`dataserial`	path	string	yes	Node dataserial to drain

Request Body (`application/json` , required)

One of the following, selected by `action` :

`action: start_drain` - `NodeStartDrainRequest`

Field	Type	Required	Default	Example	Description
`action`	literal `start_drain`	yes	n/a		The drain action to perform.
`params`	null (not used)	no	n/a		

`action: stop_drain` - `NodeStopDrainRequest`

Field	Type	Required	Default	Example	Description
`action`	literal `stop_drain`	yes	n/a		The drain action to perform.
`params`	null (not used)	no	n/a		

Responses

Status	Description	Schema
204	Drain action performed successfully on the node. No content is returned.	-
400	Bad request. Invalid dataserial or invalid action parameters.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Node not found.	-
409	Conflict. Action conflicts with current system state.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

13.2 Hardware

Endpoint Summary

Path	Summary	Method
`/api/v1/hardware`	List hardware models	`GET`
`/api/v1/hardware/{dataserial}/actions`	Perform an action on a hardware component	`POST`
`/api/v1/hardware/{model_id}`	Get hardware model info	`GET`

`GET /api/v1/hardware`

List hardware models

Returns the hardware models present in the current realm.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`refresh`	query	boolean	no	Bypass the cache: re-read the catalog from disk and refresh the cached entries.

Responses

Status	Description	Schema
200	Successful response containing the list of hardware models	`HardwareListResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (200, `HardwareListResponse`)

Field	Type	Required	Default	Example	Description
`hardware`	array of `Hardware` object	no	n/a		Array of hardware models present in the current realm. Empty array if none are found.
`total`	integer	yes	n/a	`0` , `1` , `3`	Total count of hardware models present in the current realm.

Example


```
{
   "hardware" : [
      {
         "arch" : 64,
         "compatibility_type" : "bravo",
         "component_type" : "node",
         "cpu" : "Intel Xeon E5-1630 v4 3.7GHz",
         "generation" : "100",
         "hdinfo" : "(1) 512GB SSD",
         "id" : "301002001",
         "introduced" : "7.0.0",
         "memory" : "96GB",
         "name" : "ASD-1-100-96-80",
         "network" : "2x200GbE",
         "platform" : "freebsd_142_amd64",
         "powerdraw" : 400,
         "product_name" : "Bravo Director Node 100",
         "product_type" : "Bravo Director Node",
         "role" : "director"
      }
   ],
   "total" : 1
}
```

`POST /api/v1/hardware/{dataserial}/actions`

Perform an action on a hardware component

Performs a component-scoped action, selected by the request body. `blink_node_led` turns on a node's identify ("locate") LED so the component can be found physically; `stop_node_led` turns it off. `blink_drive_led` / `stop_drive_led` do the same for a single drive, identified by `params.drive_number`. `dataserial` is always the hosting node/blade data serial (for a drive, the node that hosts it). A `drive_number` past the node's last drive answers 404. Platform Admin only.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`dataserial`	path	string	yes	Data serial of the hardware component to act on

Request Body (`application/json` , required)

One of the following, selected by `action` :

`action: blink_drive_led` - `HardwareBlinkDriveLedActionRequest`

Field	Type	Required	Default	Example	Description
`action`	literal `blink_drive_led`	yes	n/a		The action to perform on the component.
`params`	`HardwareDriveLedActionParams` object	yes	n/a		Identifies the drive on the hosting node.

`action: blink_node_led` - `HardwareBlinkNodeLedActionRequest`

Field	Type	Required	Default	Example	Description
`action`	literal `blink_node_led`	yes	n/a		The action to perform on the component.

Field	Type	Required	Default	Example	Description
`params`	null (not used)	no	n/a		

`action: stop_drive_led` - `HardwareStopDriveLedActionRequest`

Field	Type	Required	Default	Example	Description
`action`	literal `stop_drive_led`	yes	n/a		The action to perform on the component.
`params`	`HardwareDriveLedActionParams` object	yes	n/a		Identifies the drive on the hosting node.

`action: stop_node_led` - `HardwareStopNodeLedActionRequest`

Field	Type	Required	Default	Example	Description
`action`	literal `stop_node_led`	yes	n/a		The action to perform on the component.
`params`	null (not used)	no	n/a		

Responses

Status	Description	Schema
204	Action performed successfully on the hardware component. No content is returned.	-
400	Bad request. Invalid component ID or invalid action parameters.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Not found. The requested resource does not exist.	-
409	Conflict. Action conflicts with current system state.	-

Status	Description	Schema
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

`GET /api/v1/hardware/{model_id}`

Get hardware model info

Returns details for a hardware model present in the current realm, by its catalog ID.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`model_id`	path	string	yes	Catalog ID of the hardware model to retrieve
`refresh`	query	boolean	no	Bypass the cache: re-read the catalog from disk and refresh the cached entry.

Responses

Status	Description	Schema
200	Successfully retrieved detailed information about the hardware model.	`HardwareGetResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Hardware model not found.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (200, `HardwareGetResponse`)

Field	Type	Required	Default	Example	Description
`hardware`	`Hardware` object	yes	n/a		Hardware model with all available details.

Example


```
{
   "hardware" : {
      "arch" : 64,
      "compatibility_type" : "bravo",
      "component_type" : "node",
      "cpu" : "Intel Xeon E5-1630 v4 3.7GHz",
      "generation" : "100",
      "hdinfo" : "(1) 512GB SSD",
      "id" : "301002001",
      "introduced" : "7.0.0",
      "memory" : "96GB",
      "name" : "ASD-1-100-96-80",
      "network" : "2x200GbE",
      "platform" : "freebsd_142_amd64",
      "powerdraw" : 400,
      "product_name" : "Bravo Director Node 100",
      "product_type" : "Bravo Director Node",
      "role" : "director"
   }
}
```

14. S3 Gateway

VDURA's S3 gateway exposes tenant volumes over the S3 protocol. Configuration is split across four areas: per-tenant gateway config, realm-wide S3 administration (Platform Admin only), buckets, and the gateway instances themselves.

14.1 Tenant S3 Configuration

Endpoint Summary

Path	Summary	Method
`/api/v1/tenants/{tenant_id}/s3`	Get S3 config	`GET`
`/api/v1/tenants/{tenant_id}/s3`	Create S3 config	`PUT`
`/api/v1/tenants/{tenant_id}/s3`	Update S3 config	`PATCH`
`/api/v1/tenants/{tenant_id}/s3`	Delete S3 config	`DELETE`
`/api/v1/tenants/{tenant_id}/s3/tls/renew`	Renew the realm-generated S3 edge cert	`POST`

`GET /api/v1/tenants/{tenant_id}/s3`

Get S3 config

Returns the S3 gateway configuration for the specified tenant.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)

Responses

Status	Description	Schema
200	Successfully retrieved the S3 configuration for the tenant.	`GetS3ConfigResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	S3 configuration not found for this tenant.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (200, `GetS3ConfigResponse`)

Field	Type	Required	Default	Example	Description
`s3_config`	`S3ConfigSchema` object	yes	n/a		Tenant S3 gateway configuration.

Example


```
{
   "s3_config" : {
      "default_dir_mode" : "755",
      "default_group" : "gid:65534",
      "default_obj_mode" : "644",
      "default_owner" : "uid:65534",
      "default_prefix" : "/vol1/t1",
      "enabled" : true,
      "tls" : {}
   }
}
```

`PUT /api/v1/tenants/{tenant_id}/s3`

Create S3 config

Creates the S3 gateway configuration for the specified tenant. 409 if a config already exists; callers needing replace semantics must DELETE first.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)

Request Body (`application/json` , required)

Schema: `S3ConfigCreateRequest`

Request payload for creating a tenant's S3 configuration.

Every field carries a default, so `PUT /s3 {}` is a complete request: it creates the disabled config with the empty prefix and the POSIX defaults a tenant starts with. Those defaults live here rather than in a factory beside the service, because this is the only place that can publish them - an operator reading the API document sees what an omitted field will be, and there is no second copy to drift from it.

`access_key` , `secret_key` and `control_plane` have no field here at all;
:mod: `app.use_cases.create_s3_config` mints all three.

Field	Type	Required	Default	Example	Description
`access_log`	boolean	no	`true`		Whether the S3 access log is enabled for this tenant (the S3 gateway's –access-log audit log). Defaults to true.
`aggregator`	`S3AggregatorConfigSchema` object	no	n/a		Tenant-level S3 aggregator policy (opt-in via `enabled=true`).
`debug_log`	boolean	no	`false`		Whether the S3 debug request log is enabled for this tenant. Defaults to false.

Field	Type	Required	Default	Example	Description
`default_dir_mode`	string	no	`755`	`755`, `0755`	Default directory permission mode in octal (e.g. 755).
`default_group`	string	no	`gid:65534`	`gid:65534`, `gid:1000`	Default POSIX group for objects (e.g. gid:65534). An fs-user created without a gid takes it. gid:65534 (nogroup) is the unset value; an empty string is not accepted.
`default_obj_mode`	string	no	`644`	`644`, `0644`	Default object/file permission mode in octal (e.g. 644).
`default_owner`	string	no	`uid:65534`	`uid:65534`, `uid:1000`	Default POSIX owner for objects (e.g. uid:65534). An fs-user created without a uid takes it. uid:65534 (nobody) is the unset value; an empty string is not accepted.
`default_prefix`	string	no	(empty string)	`/vol1`, `/vol1/data`	Tenant-relative path served by S3 (e.g. /vol1 or /vol1/data). Must name an existing volume of the tenant (or a directory inside one); stored resolved into the tenant volume namespace. Optional: omit it to create the config before the tenant has a volume to serve. The empty value skips volume validation and cannot be enabled until a real prefix is set by PATCH.
`dns`	`S3DnsConfigSchema` object	no	n/a		DNS/named-access policy (default mode=off).

Field	Type	Required	Default	Example	Description
`enabled`	boolean	no	`false`		Whether S3 access is enabled for this tenant.
`metrics_enabled`	boolean	no	`true`		Whether S3 metrics are collected for this tenant (the S3 gateway's StatsD counters to the node-local Telegraf listener). Defaults to true.
`tls`	`S3TlsConfigSchema` object	no	n/a		TLS configuration for the S3 gateway.

Example

```
{
   "aggregator" : {
      "enabled" : false
   },
   "default_dir_mode" : "755",
   "default_group" : "gid:65534",
   "default_obj_mode" : "644",
   "default_owner" : "uid:65534",
   "default_prefix" : "/vol1/t1",
   "enabled" : true,
   "tls" : {}
}
```

Responses

Status	Description	Schema
201	S3 configuration created successfully.	-
400	Bad request: the S3 configuration is invalid for the requested state - enabling without a `default_prefix`, TLS material that fails validation for the resulting `tls.mode`, or a tenant short name and realm DNS zone too long to name a realm-generated edge cert.	-

Status	Description	Schema
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant not found.	-
409	S3 configuration already exists for this tenant.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-
503	The realm key service could not be reached to sign the S3 edge cert; retry.	-

`PATCH /api/v1/tenants/{tenant_id}/s3`

Update S3 config

Partially updates the S3 gateway configuration for the specified tenant. The response names the fields of this request that already-running gateway instances will not pick up until they are restarted - empty when every change is in effect now, which includes a tenant with no started instance.

Before you call this. Needs: S3 is enabled for the realm (`PUT /api/v1/s3/enabled`) and the tenant S3 configuration exists. To set `enabled` to `true` , `default_prefix` must name an existing volume of the tenant. Otherwise: `409` when the realm is not enabled. The call is refused when the prefix does not name an existing volume.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)

Request Body (`application/json` , required)

Schema: `S3ConfigPatchRequest`

Request payload for partially updating a tenant's S3 configuration.

At least one field must be provided. Null values are not allowed - omit fields that should not change.

Field	Type	Required	Default	Example	Description
`access_log`	boolean, nullable	no	`null`		Whether the S3 access log is enabled for this tenant.
`aggregator`	`S3AggregatorConfigSchema` object, nullable	no	`null`		Tenant-level S3 aggregator policy (opt-in via `enabled=true`).
`debug_log`	boolean, nullable	no	`null`		Whether the S3 debug request log is enabled for this tenant.

Field	Type	Required	Default	Example	Description
`default_dir_mode`	string, nullable	no	`null`		Default directory permission mode in octal.
`default_group`	string, nullable	no	`null`		Default POSIX group for objects. To unset it, send gid:65534 (nogroup), the value a new config starts with; an empty string is not accepted.
`default_obj_mode`	string, nullable	no	`null`		Default object/file permission mode in octal.
`default_owner`	string, nullable	no	`null`		Default POSIX owner for objects. To unset it, send uid:65534 (nobody), the value a new config starts with; an empty string is not accepted.
`default_prefix`	string, nullable	no	`null`		Tenant-relative path served by S3. Must name an existing volume of the tenant (or a directory inside one); immutable while the tenant has S3 enabled.
`dns`	`S3DnsConfigSchema` object, nullable	no	`null`		DNS/named-access policy (mode off/panfs/external).
`enabled`	boolean, nullable	no	`null`		Whether S3 access is enabled for this tenant.
`metrics_enabled`	boolean, nullable	no	`null`		Whether S3 metrics are collected for this tenant.
`tls`	`S3TlsConfigSchema` object, nullable	no	`null`		TLS configuration for the S3 gateway.

Example

```
{
   "default_prefix" : "/vol1/t1",
   "enabled" : false
}
```

Responses

Status	Description	Schema
200	S3 configuration updated successfully. The body names the fields of this request that already-running gateway instances will not pick up until they are restarted; it is empty when every change is in effect now.	`S3ConfigUpdateResponse`
400	Bad request: the S3 configuration is invalid for the requested state - enabling without a `default_prefix`, TLS material that fails validation for the resulting `tls.mode`, or a tenant short name and realm DNS zone too long to name a realm-generated edge cert.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	S3 configuration not found for this tenant.	-
409	The request conflicts with the tenant S3 state: disabling while an instance is desired started, changing `default_prefix` while enabled, or deleting the config while any instance record exists.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-
503	The realm key service could not be reached to sign the S3 edge cert; retry.	-

Response Body (200, `S3ConfigUpdateResponse`)

Field	Type	Required	Default	Example	Description
`restart_required`	array of string	yes	n/a		Names of S3 config fields this request changed whose new value reaches already-running S3 gateway instances only after those instances are restarted. Empty when every change is in effect now, when the request changed nothing, or when the tenant has no started instance. The set of names can change between releases.

Example

```
{
   "restart_required" : [
      "access_log",
      "debug_log"
   ]
}
```

`DELETE /api/v1/tenants/{tenant_id}/s3`

Delete S3 config

Deletes the S3 gateway configuration for the specified tenant.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)

Responses

Status	Description	Schema
204	S3 configuration deleted successfully. No content in response body.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	S3 configuration not found for this tenant.	-
409	The request conflicts with the tenant S3 state: disabling while an instance is desired started, changing `default_prefix` while enabled, or deleting the config while any instance record exists.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

`POST /api/v1/tenants/{tenant_id}/s3/tls/renew`

Renew the realm-generated S3 edge cert

Forces a fresh mint of the tenant S3 edge certificate and stores it with the tenant's S3 configuration. Manual operator action: renewal is **not** automated, so an operator runs this before the stored cert lapses. Valid only when `tls.mode = realm-generated` (the realm owns that cert); a `tenant-provided` cert is renewed by uploading a new one via PATCH /s3 and this returns 409. The minted SANs follow the tenant's current `aggregator.enabled` state. Renewal disrupts nothing: a running gateway re-reads the cert and key without restarting. **Authorization:** gated on `s3-config:update`, like every other write to the tenant S3 config; the tenant admin and platform-admin both hold it.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)

Responses

Status	Description	Schema
204	S3 edge cert renewed and stored. No content in response body.	-
400	The tenant short name and realm DNS zone are too long to name the edge cert.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	S3 configuration not found for this tenant.	-
409	Tenant uses tls.mode=tenant-provided; upload a new cert instead.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-
503	The realm key service could not be reached to sign the S3 edge cert; retry.	-

14.2 Realm-Wide S3 Administration

Endpoint Summary

Path	Summary	Method
`/api/v1/s3/enabled`	Get realm-wide S3 enable flag	`GET`
`/api/v1/s3/enabled`	Set realm-wide S3 enable flag	`PUT`
`/api/v1/s3/prefix`	Get realm-wide S3 prefix path	`GET`
`/api/v1/s3/prefix`	Set realm-wide S3 prefix path	`PUT`

`GET /api/v1/s3/enabled`

Get realm-wide S3 enable flag

Returns the realm-wide S3 enable flag. Distinct from the per-tenant `s3 enabled` toggle on `/api/v1/tenants/{id}/s3/`; this flag gates provisioning of any new tenant-level S3 state. A fresh realm reports `enabled=false` until the operator opts in via PUT.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Responses

Status	Description	Schema
200	Current realm-wide enable flag.	`S3EnabledResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
500	Internal server error.	-

Response Body (200, `S3EnabledResponse`)

Field	Type	Required	Default	Example	Description
`enabled`	boolean	yes	n/a		Realm-wide S3 enable flag.
`tenants_with_s3_enabled`	array of string, nullable	no	`null`		Tenant short names that still have tenant-level S3 enabled after the realm-wide disable. Absent on get and on enable.
`warning`	string, nullable	no	`null`		Warning surfaced when realm-disable leaves tenants with S3 enabled.

Example


```
{
   "enabled" : false,
   "tenants_with_s3_enabled" : [
      "tn1"
   ],
   "warning" : "1 tenant(s) still have S3 enabled at the tenant level"
}
```

`PUT /api/v1/s3/enabled`

Set realm-wide S3 enable flag

Sets the realm-wide S3 enable flag. A successful write takes effect on every node: turning S3 off stops every running S3 gateway instance; turning it back on re-applies per-tenant intent and brings the gateways back up. Per-tenant `enabled` records are preserved across both transitions. Disabling while at least one tenant has tenant-level S3 enabled is permitted but surfaces a warning listing the still-enabled tenants in the response.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Request Body (`application/json` , required)

Schema: `S3EnabledRequest`

Request body for PUT /s3/enabled.

Field	Type	Required	Default	Example	Description
`enabled`	boolean	yes	n/a		Realm-wide S3 enable flag.

Example

```
{
   "enabled" : true
}
```

Responses

Status	Description	Schema
200	Realm-wide enable flag updated.	`S3EnabledResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (200, `S3EnabledResponse`)

Field	Type	Required	Default	Example	Description
`enabled`	boolean	yes	n/a		Realm-wide S3 enable flag.
`tenants_with_s3_enabled`	array of string, nullable	no	`null`		Tenant short names that still have tenant-level S3 enabled after the realm-wide disable. Absent on get and on enable.
`warning`	string, nullable	no	`null`		Warning surfaced when realm-disable leaves tenants with S3 enabled.

Example


```
{
   "enabled" : false,
   "tenants_with_s3_enabled" : [
      "tn1"
   ],
   "warning" : "1 tenant(s) still have S3 enabled at the tenant level"
}
```

`GET /api/v1/s3/prefix`

Get realm-wide S3 prefix path

Returns the realm-wide S3 prefix path that anchors per-tenant prefixes under a common root. Distinct from the per-tenant `default_prefix` carried on `/api/v1/tenants/{id}/s3/`. A fresh realm reports the empty string until the operator sets a value.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Responses

Status	Description	Schema
200	Current realm-wide prefix.	`S3PrefixResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
500	Internal server error.	-

Response Body (200, `S3PrefixResponse`)

Field	Type	Required	Default	Example	Description
`prefix`	string	yes	n/a		Current realm-wide path prefix.

Example

```
{
   "prefix" : "/realm-root"
}
```

`PUT /api/v1/s3/prefix`

Set realm-wide S3 prefix path

Sets the realm-wide S3 prefix path. Non-empty values MUST begin with "/" and MUST NOT carry trailing whitespace. An empty string clears the prefix.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Request Body (`application/json` , required)

Schema: `S3PrefixRequest`

Request body for PUT /s3/prefix.

Field	Type	Required	Default	Example	Description
`prefix`	string	yes	n/a		Realm-wide POSIX-style path prefix. Empty string means "no realm-wide prefix"; otherwise the value MUST begin with "/" and MUST NOT contain trailing whitespace.

Example

```
{
   "prefix" : "/realm-root"
}
```

Responses

Status	Description	Schema
200	Realm-wide prefix updated.	`S3PrefixResponse`
400	Invalid prefix (missing leading "/" or trailing whitespace).	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-

Status	Description	Schema
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (200, `S3PrefixResponse`)

Field	Type	Required	Default	Example	Description
`prefix`	string	yes	n/a		Current realm-wide path prefix.

Example

```
{
   "prefix" : "/realm-root"
}
```

14.3 S3 Buckets

Endpoint Summary

Path	Summary	Method
`/api/v1/tenants/{tenant_id}/s3/buckets`	List S3 buckets	`GET`
`/api/v1/tenants/{tenant_id}/s3/buckets`	Create S3 bucket	`POST`
`/api/v1/tenants/{tenant_id}/s3/buckets/{name}`	Delete S3 bucket	`DELETE`
`/api/v1/tenants/{tenant_id}/s3/buckets/{name}/policy`	Delete S3 bucket policy	`DELETE`

`GET /api/v1/tenants/{tenant_id}/s3/buckets`

List S3 buckets

List the tenant's S3 buckets, as a running gateway reports them.

The gateway is the only bucket catalogue, so a tenant with no running instance cannot be listed: that is a 503, never an empty list.

Before you call this. Needs: A running S3 gateway instance for the tenant. Otherwise: `503`, never an empty list. It does not clear on retry until an instance is started.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)

Responses

Status	Description	Schema
200	The buckets a running S3 gateway holds for the tenant.	`ListS3BucketsResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant not found.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-
503	No running S3 instance could answer, and nothing was performed. The response detail names which case applies and the panctl command that resolves it.	-

Response Body (200, `ListS3BucketsResponse`)

Field	Type	Required	Default	Example	Description
`s3_buckets`	array of `S3BucketSchema` object	yes	n/a		List of S3 buckets.

`POST /api/v1/tenants/{tenant_id}/s3/buckets`

Create S3 bucket

Create an S3 bucket on a running gateway, owned by the PanFS user the request names.

Before you call this. Needs: A running S3 gateway instance, and an `owner` that is the access key of an existing PanFS user in the tenant. Otherwise: `503` with no running instance, which does not clear on retry. A `504` means the bucket may exist, so list buckets before retrying.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)

Request Body (`application/json` , required)

Schema: `S3BucketCreateRequest`

Request payload for creating an S3 bucket.

Field	Type	Required	Default	Example	Description
`name`	string	yes	n/a		Bucket name.
`owner`	string	yes	n/a		S3 access_key of the PanFS user the bucket belongs to. Required, and must name a PanFS user in this tenant: the gateway resolves the owner through its IAM backend, so a value that is not a PanFS user cannot be resolved there at all.

Example

```
{
   "name" : "my-bucket",
   "owner" : "alice"
}
```

Responses

Status	Description	Schema
201	S3 bucket created on a running gateway.	-
400	The S3 gateway refused the operation - most often a bucket name outside the gateway-enforced AWS rules (3-63 characters, lowercase, DNS-shaped, not an IP address).	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant not found.	-
409	S3 bucket already exists.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-
503	No running S3 instance could answer, and nothing was performed. The response detail names which case applies and the panctl command that resolves it.	-
504	No running S3 instance confirmed the create, and the bucket may exist. List the buckets to check.	-

`DELETE /api/v1/tenants/{tenant_id}/s3/buckets/{name}`

Delete S3 bucket

Delete an S3 bucket from a running gateway.

A non-empty bucket returns 409 and stays listed. A bucket the gateway does not hold is a 404, unless an earlier attempt of this same request may have deleted it.

Authorization is the `s3-buckets:delete` verb plus caller-tenant scoping, like every other operation on this surface. Deletion is NOT conditioned on the caller having created the bucket.

Before you call this. Needs: A running S3 gateway instance for the tenant. Otherwise: `503` with no running instance; `409` for a non-empty bucket. A `504` means the bucket may have been deleted.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)
`name`	path	string	yes	S3 bucket name

Responses

Status	Description	Schema
204	S3 bucket deleted from a running gateway.	-
400	The S3 gateway refused the operation - most often a bucket name outside the gateway-enforced AWS rules (3-63 characters, lowercase, DNS-shaped, not an IP address).	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant not found, or S3 bucket not found on the gateway.	-
409	S3 bucket still holds objects; empty it over the S3 API before deleting.	-
422	Validation Error	`HTTPValidationError`

Status	Description	Schema
500	Internal server error.	-
503	No running S3 instance could answer, and nothing was performed. The response detail names which case applies and the panctl command that resolves it.	-
504	No running S3 instance confirmed the delete, and the bucket may have been deleted. List the buckets to check.	-

`DELETE /api/v1/tenants/{tenant_id}/s3/buckets/{name}/policy`

Delete S3 bucket policy

Remove an S3 bucket's policy on a running gateway.

The recovery for a bucket whose policy locks its own owner out. Once a bucket has a policy it is the only authority on the S3 API, so an owner whose policy omits `s3:DeleteBucketPolicy` cannot remove it there; the gateway's admin route, reached through the node admin service, does not evaluate the policy. The policy is removed, not replaced: the bucket returns to the gateway's default access, under which its owner has full control again. A bucket with no policy answers 204 too (idempotent, as AWS).

The policy is a sub-resource of the bucket, not a fourth bucket operation: gateway-side bucket state lives under `/{name}/` and takes the `s3-buckets` action matching its verb - here `delete`, held by the same callers that may delete the bucket. Caller-tenant scoping is the same as for delete. The audit target is the bucket name, from the path.

Before you call this. Needs: A running S3 gateway instance for the tenant. Otherwise: `503` with no running instance. A `504` means the policy may have been removed; the call is safe to repeat.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)
`name`	path	string	yes	S3 bucket name

Responses

Status	Description	Schema
204	The bucket policy was removed from a running gateway, or the bucket had none.	-
400	The S3 gateway refused the operation - most often a bucket name outside the gateway-enforced AWS rules (3-63 characters, lowercase, DNS-shaped, not an IP address).	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-

Status	Description	Schema
404	Tenant not found, or S3 bucket not found on the gateway.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-
503	No running S3 instance could answer, and nothing was performed. The response detail names which case applies and the panctl command that resolves it.	-
504	No running S3 instance confirmed the policy delete, and the policy may have been removed. The operation is idempotent: repeat the request.	-

14.4 S3 Gateway Instances

Endpoint Summary

Path	Summary	Method
`/api/v1/tenants/{tenant_id}/s3/instances`	List a tenant's S3 gateway instances	`GET`
`/api/v1/tenants/{tenant_id}/s3/instances`	Create an S3 gateway instance	`POST`
`/api/v1/tenants/{tenant_id}/s3/instances/delete-all`	Delete every instance of the tenant and wait for teardown	`POST`
`/api/v1/tenants/{tenant_id}/s3/instances/start-all`	Start every instance of the tenant	`POST`
`/api/v1/tenants/{tenant_id}/s3/instances/stop-all`	Stop every instance of the tenant	`POST`
`/api/v1/tenants/{tenant_id}/s3/instances/{iid}`	Show an S3 gateway instance (bare-iid fan-out or specific node)	`GET`
`/api/v1/tenants/{tenant_id}/s3/instances/{iid}`	Delete an S3 gateway instance (fan-out if node omitted; requires confirm)	`DELETE`
`/api/v1/tenants/{tenant_id}/s3/instances/{iid}/start`	Start an S3 gateway instance (fan-out if node omitted)	`POST`
`/api/v1/tenants/{tenant_id}/s3/instances/{iid}/stop`	Stop an S3 gateway instance (fan-out if node omitted)	`POST`

`GET /api/v1/tenants/{tenant_id}/s3/instances`

List a tenant's S3 gateway instances

Return every instance of the tenant with its observed/health merged in.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)

Responses

Status	Description	Schema
200	List of merged desired + observed instance views.	`S3InstanceListResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant not found.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (200, `S3InstanceListResponse`)

Field	Type	Required	Default	Example	Description
`instances`	array of `S3InstanceViewSchema` object	yes	n/a		

`POST /api/v1/tenants/{tenant_id}/s3/instances`

Create an S3 gateway instance

Writes the desired-state record for a new S3 gateway instance placed on the specified node. The instance starts in desired=created; use the start endpoint to bring it online. At most one instance per tenant per node, and the tenant must have a subnet: the gateway binds inside the tenant network namespace, which a tenant with no subnet does not have.

Before you call this. Needs: S3 is enabled for the tenant, the tenant has a subnet, and the tenant has no instance on that node yet. Otherwise: `409` when the tenant already has an instance on that node.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)

Request Body (`application/json` , required)

Schema: `S3InstanceCreateRequest`

Create a new S3 gateway instance for a tenant on a specific node.

Field	Type	Required	Default	Example	Description
`dataserial`	string	yes	n/a		Target node's dataserial (14 hex chars) as returned by `GET /nodes` and consumed by every `/nodes/{dataserial}` endpoint (e.g. `08823f71a1eecc`). The stable placement identity - unaffected by node renames.
`instance_id`	string	yes	n/a		Instance identifier: exactly 5 alphanumeric characters (D10).

Example

```
{
   "dataserial" : "08823f71a1eecc",
   "instance_id" : "abc12"
}
```

Responses

Status	Description	Schema
201	Instance created.	`S3InstanceCreateResponse`
400	Invalid iid format or unknown dataserial.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant not found.	-
409	Instance already exists at (iid, dataserial), the tenant already has an instance on that node, or the tenant has no subnet.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (201, `S3InstanceCreateResponse`)

Field	Type	Required	Default	Example	Description
`instance`	`S3InstanceSchema` object	yes	n/a		Desired-state view of an S3 gateway instance.

`POST /api/v1/tenants/{tenant_id}/s3/instances/delete-all`

Delete every instance of the tenant and wait for teardown

Fan out `desired=deleting` to every (iid, data_serial) owned by the tenant and block until the owning node has removed each instance record, which is the terminal state, or until `timeout` seconds elapse. Intended as the companion to the `DELETE /api/v1/tenants/{id}` guard: an operator runs delete-all first, waits for it to succeed, then deletes the tenant.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)
`timeout`	query	integer	no	Upper bound in seconds on the blocking wait for instance teardown.

Responses

Status	Description	Schema
200	Every instance torn down; no rows remain. results lists what each pass recorded on each node (202 once recorded), so a node retried by a later pass appears more than once.	`S3InstanceFanoutResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant not found.	-
409	Timed out waiting for teardown; see error body.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (200, `S3InstanceFanoutResponse`)

Field	Type	Required	Default	Example	Description
`aggregate_status`	integer	yes	n/a		Aggregate HTTP status: the rows' shared success code (202 for a recorded state change, 200 for a read) if every node succeeded, the shared error code if every node failed with the same code, otherwise 207. delete-all answers 200 once every instance is gone, and start-all/stop-all answer 200 with no results for a tenant with no instances.
`confirmed`	boolean	no	`true`		For delete fan-out, true once the caller has confirmed the destructive operation. When false, the request body enumerates matches without deleting anything.
`results`	array of `S3InstancePerNodeResult` object	yes	n/a		

`POST /api/v1/tenants/{tenant_id}/s3/instances/start-all`

Start every instance of the tenant

Records the new desired state of every instance of the tenant and returns without waiting for them to reach it. Poll GET /tenants/{tenant_id}/s3/instances for the observed state.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)

Responses

Status	Description	Schema
200	The tenant has no instances; nothing was recorded.	-
202	Recorded on every instance.	`S3InstanceFanoutResponse`
207	Partial success; see per-node results.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant not found.	-
409	Every target refused: S3 is disabled for the tenant, the target's node has its S3 service stopped, or the instance is being deleted (desired=deleting).	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (202, `S3InstanceFanoutResponse`)

Field	Type	Required	Default	Example	Description
`aggregate_status`	integer	yes	n/a		Aggregate HTTP status: the rows' shared success code (202 for a recorded state change, 200 for a read) if every node succeeded, the shared error code if every node failed with the same code, otherwise 207. delete-all answers 200 once every instance is gone, and start-all/stop-all answer 200 with no results for a tenant with no instances.
`confirmed`	boolean	no	`true`		For delete fan-out, true once the caller has confirmed the destructive operation. When false, the request body enumerates matches without deleting anything.
`results`	array of `S3InstancePerNodeResult` object	yes	n/a		

`POST /api/v1/tenants/{tenant_id}/s3/instances/stop-all`

Stop every instance of the tenant

Records the new desired state of every instance of the tenant and returns without waiting for them to reach it. Poll GET /tenants/{tenant_id}/s3/instances for the observed state.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)

Responses

Status	Description	Schema
200	The tenant has no instances; nothing was recorded.	-
202	Recorded on every instance.	`S3InstanceFanoutResponse`
207	Partial success; see per-node results.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	Tenant not found.	-
409	Every instance is being deleted (desired=deleting) and cannot be stopped.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (202, `S3InstanceFanoutResponse`)

Field	Type	Required	Default	Example	Description
`aggregate_status`	integer	yes	n/a		Aggregate HTTP status: the rows' shared success code (202 for a recorded state change, 200 for a read) if every node succeeded, the shared error code if every node failed with the same code, otherwise 207. delete-all answers 200 once every instance is gone, and start-all/stop-all answer 200 with no results for a tenant with no instances.
`confirmed`	boolean	no	`true`		For delete fan-out, true once the caller has confirmed the destructive operation. When false, the request body enumerates matches without deleting anything.
`results`	array of `S3InstancePerNodeResult` object	yes	n/a		

`GET /api/v1/tenants/{tenant_id}/s3/instances/{iid}`

Show an S3 gateway instance (bare-iid fan-out or specific node)

Return one or many matching rows as a per-node result set.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)
`iid`	path	string	yes	Instance identifier: exactly 5 alphanumeric characters.
`dataserial`	query	string, nullable	no	Optional target node dataserial; without it, fan out over every matching (iid, data_serial).

Responses

Status	Description	Schema
200	One entry in results per matching node.	`S3InstanceFanoutResponse`
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	No matching instance for this iid under the tenant.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (200, `S3InstanceFanoutResponse`)

Field	Type	Required	Default	Example	Description
`aggregate_status`	integer	yes	n/a		Aggregate HTTP status: the rows' shared success code (202 for a recorded state change, 200 for a read) if every node succeeded, the shared error code if every node failed with the same code, otherwise 207. delete-all answers 200 once every instance is gone, and start-all/stop-all answer 200 with no results for a tenant with no instances.
`confirmed`	boolean	no	`true`		For delete fan-out, true once the caller has confirmed the destructive operation. When false, the request body enumerates matches without deleting anything.
`results`	array of `S3InstancePerNodeResult` object	yes	n/a		

`DELETE /api/v1/tenants/{tenant_id}/s3/instances/{iid}`

Delete an S3 gateway instance (fan-out if node omitted; requires confirm)

If dataserial is omitted and more than one node hosts a matching iid, confirm=true is required; otherwise the API returns 409 and a preview body listing the matches, so the caller can decide whether to proceed.

Records the delete and returns without waiting for the instance to be removed. Poll GET /tenants/{tenant_id}/s3/instances/{iid} until it answers 404: the instance is gone. If the delete named a dataserial, poll with the same dataserial: without it the GET still answers for the placements on other nodes.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)
`iid`	path	string	yes	Instance identifier: exactly 5 alphanumeric characters.
`dataserial`	query	string, nullable	no	Optional target node dataserial.
`confirm`	query	boolean	no	Required (true) when dataserial is omitted and multiple instances match - prevents destructive fan-out without explicit opt-in.

Responses

Status	Description	Schema
202	Recorded on every target.	`S3InstanceFanoutResponse`
207	Partial success; see per-node results.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	No matching instance.	-
409	Destructive fan-out without confirm=true; body lists matches.	-

Status	Description	Schema
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (202, `S3InstanceFanoutResponse`)

Field	Type	Required	Default	Example	Description
`aggregate_status`	integer	yes	n/a		Aggregate HTTP status: the rows' shared success code (202 for a recorded state change, 200 for a read) if every node succeeded, the shared error code if every node failed with the same code, otherwise 207. delete-all answers 200 once every instance is gone, and start-all/stop-all answer 200 with no results for a tenant with no instances.
`confirmed`	boolean	no	`true`		For delete fan-out, true once the caller has confirmed the destructive operation. When false, the request body enumerates matches without deleting anything.
`results`	array of `S3InstancePerNodeResult` object	yes	n/a		

`POST /api/v1/tenants/{tenant_id}/s3/instances/{iid}/start`

Start an S3 gateway instance (fan-out if node omitted)

Records the new desired state and returns without waiting for the instance to reach it. Poll GET /tenants/{tenant_id}/s3/instances/{iid} for the observed state.

Before you call this. Needs: The instance exists. Otherwise: A `202` means the desired state is recorded, not reached. Poll `observed` on `GET /api/v1/tenants/{tenant_id}/s3/instances/{iid}` until it is `running`.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)
`iid`	path	string	yes	Instance identifier: exactly 5 alphanumeric characters.
`dataserial`	query	string, nullable	no	Optional target node dataserial.

Responses

Status	Description	Schema
202	Recorded on every target.	`S3InstanceFanoutResponse`
207	Mixed success; see per-node results.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	No matching instance.	-
409	Every target refused: S3 is disabled for the tenant, the target's node has its S3 service stopped, or the instance is being deleted (desired=deleting).	-
422	Validation Error	`HTTPValidationError`

Status	Description	Schema
500	Internal server error.	-

Response Body (202, `S3InstanceFanoutResponse`)

Field	Type	Required	Default	Example	Description
`aggregate_status`	integer	yes	n/a		Aggregate HTTP status: the rows' shared success code (202 for a recorded state change, 200 for a read) if every node succeeded, the shared error code if every node failed with the same code, otherwise 207. delete-all answers 200 once every instance is gone, and start-all/stop-all answer 200 with no results for a tenant with no instances.
`confirmed`	boolean	no	`true`		For delete fan-out, true once the caller has confirmed the destructive operation. When false, the request body enumerates matches without deleting anything.
`results`	array of `S3InstancePerNodeResult` object	yes	n/a		

`POST /api/v1/tenants/{tenant_id}/s3/instances/{iid}/stop`

Stop an S3 gateway instance (fan-out if node omitted)

Records the new desired state and returns without waiting for the instance to reach it. Poll GET /tenants/{tenant_id}/s3/instances/{iid} for the observed state.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	path	string (uuid)	yes	Tenant identifier (UUID)
`iid`	path	string	yes	Instance identifier: exactly 5 alphanumeric characters.
`dataserial`	query	string, nullable	no	Optional target node dataserial.

Responses

Status	Description	Schema
202	Recorded on every target.	`S3InstanceFanoutResponse`
207	Mixed success; see per-node results.	-
401	Unauthorized. Authentication required.	-
403	Forbidden. Insufficient permissions.	-
404	No matching instance.	-
409	Every target is being deleted (desired=deleting) and cannot be stopped.	-
422	Validation Error	`HTTPValidationError`
500	Internal server error.	-

Response Body (202, `S3InstanceFanoutResponse`)

Field	Type	Required	Default	Example	Description
`aggregate_status`	integer	yes	n/a		Aggregate HTTP status: the rows' shared success code (202 for a recorded state change, 200 for a read) if every node succeeded, the shared error code if every node failed with the same code, otherwise 207. delete-all answers 200 once every instance is gone, and start-all/stop-all answer 200 with no results for a tenant with no instances.
`confirmed`	boolean	no	`true`		For delete fan-out, true once the caller has confirmed the destructive operation. When false, the request body enumerates matches without deleting anything.
`results`	array of `S3InstancePerNodeResult` object	yes	n/a		

15. Observability

Alerts, audit trail, PanFS events, logs, and metrics, each as a point-in-time snapshot endpoint and a matching Server-Sent Events (SSE) stream endpoint for live updates.

15.1 Metrics and Logs Behind These Endpoints

The endpoints in this chapter are windows onto the data stores of the VDURA observability stack, not a separate data model. Knowing what those stores hold, and how it is named, is what makes the query endpoints useful. This section summarizes it; the full catalog of metric names, labels, alert definitions, and dashboards is in the VDURA PanFS Metrics Reference Guide.

Where each endpoint group reads from

Endpoint group	Backing store	How it reads
`metrics/query`, `metrics/query_range`	VictoriaMetrics (Prometheus-compatible, MetricsQL)	Forwarded to the VictoriaMetrics `/api/v1/query` and `/api/v1/query_range` endpoints. Only the documented parameters are accepted; any other name returns 422.
`metrics/{key}`, `metrics/stream`, `metrics/{key}/stream`, `metrics/targets`	VictoriaMetrics	Snapshots and Server-Sent Events streams for named metric keys, and node target discovery in Prometheus HTTP Service Discovery format.
`logs`, `logs/stream`, `logs/{logsql_endpoint}`	VictoriaLogs (LogsQL)	The proxy forwards to `/select/logsql/{endpoint}` on the VictoriaLogs query instance on the Director Node.
`audit`, `audit/stream`, `audit/{request_id}`, `events`, `events/stream`	VictoriaLogs	The server owns the LogsQL query, so clients pass no query text. Audit records are deduplicated by `request_id` and PanFS events by `event_id`.
`alerts`, `alerts/stream`, `storagesets/status`	Alert series in the metrics database	Active alerts evaluated by vmalert and projected from the `ALERTS` series.

Metric naming families

Four naming families coexist in the metrics database. Recognizing the family tells you where a metric comes from and how to read it.

Prefix	Source and reading rules
`magent_local_*`, `magent_shelf_*`	The PanFS management agent statistics tree, flattened as `magent_local_<hw, sw, or sys>_<subsystem>_<stat>`. Array statistics get a numeric suffix, and a string statistic becomes a series with the value 1 and the text in a `value` label.
`panfs_*`	Realm-level inventory and events from the management database. The sample value is a state code and the content is in the labels. Each family has a matching `<family>_scrape_success` gauge.
`cpu_`, `mem_`, `disk_`, `diskio_`, `net_`, `netstat_`, `nstat_`, `ethtool_`, `processes_`, `procstat_`, `system_`	Host operating system metrics from Telegraf input plugins, named `<measurement>_<field>`. Standard Linux semantics apply, and counters need `rate()` or `increase()`.
`ALERTS`, `ALERTS_FOR_STATE`, `observability_node_type`	The vmalert output, and the declaration of each node's role.

Common labels

Nearly every series carries five labels that identify the emitting node, so any metric can be sliced by node or node role without a join. Realm-level `panfs_*` inventory metrics describe realm objects rather than the emitting host and carry only `node_name`.

Label	Meaning
`host`	Linux hostname of the node.
`host_id`	Stable node UUID assigned by the node identity service.
`ip_address`	Management IP address of the node.
`node_name`	The PanFS realm name of the node, in enclosure,slot form, such as `VCH-1,1`.
`node_type`	The node role: `director_l` for a Director, `vpod` for a virtual Storage Node hosting one OSD.

Reading values correctly

- **Counters** (host disk and network totals, File Manager operation counts) increase monotonically; use `rate()` or `increase()`.
- **Gauges** are read directly.
- **Info metrics** have a constant sample value and carry their content in labels; read them with `last_over_time()` or group by the label, never with `rate()`.
- **Interval gauges** (the `measure_` group under `magent_local_sw_iopath_`) already represent the last collection interval; sum them across nodes for a realm figure, and do not apply `rate()` to them.
- Management agent and realm metrics arrive every 60 seconds. The VDURA PanFS Metrics Reference Guide lists the interval for each metric family in section 1.2.

Example queries

Pass any of these as the `query` parameter of `GET /api/v1/observability/metrics/query`:

```
curl -sG "https://<realm-address>/api/v1/observability/metrics/query" \
  -H "Authorization: Bearer $TOKEN" \
  --data-urlencode 'query=count by (node_type) (system_uptime)'
```

Question	Expression
How many nodes of each role are reporting?	`count by (node_type) (system_uptime)`
Which Director holds the realm president role?	`panfs_president == 1`
Is the master management server reachable?	`min(panfs_master_mserver_up)`
How many realm alerts are active, by severity?	`sum by (severity) (panfs_alert_occurrences)`
How many alerts are firing, by severity?	`count by (severity) (ALERTS{alertstate="firing"})`
How long has each alert been active?	`time() - ALERTS_FOR_STATE`
Which network interfaces have the link down?	`panfs_nic{uplink_down="true"}`

Logs and the audit trail

The audit and event endpoints read structured log lines from VictoriaLogs. The fields of an `AuditRecord` are the fields of the API audit log (`/var/log/panapi-audit.log`): `client_ip`, `method`, `request_id`, `result`, `status_code`, `action`, `resource`, `target_id`, `tenant_id`, and `user_id`. Every log line also

carries stream identity fields such as `_time`, `_msg`, `node_name`, `host_id`, and `filename` (or `unit` for journald entries). To go beyond what the audit endpoints offer, query the logs directly through the proxy with LogsQL. For example, to list recent failed API operations:

```
curl -sG "https://<realm-address>/api/v1/observability/logs/query" \
  -H "Authorization: Bearer $TOKEN" \
  --data-urlencode 'query=filename:="/var/log/panapi-audit.log" result:=failed' \
  --data-urlencode 'limit=20'
```

15.2 Endpoint Reference

Endpoint Summary

Path	Summary	Method
`/api/v1/observability/alerts`	Alerts snapshot (keyed set)	`GET`
`/api/v1/observability/alerts/stream`	Alerts SSE stream (added/resolved)	`GET`
`/api/v1/observability/audit`	Audit trail snapshot (last N)	`GET`
`/api/v1/observability/audit/stream`	Audit SSE stream (append deltas)	`GET`
`/api/v1/observability/audit/{request_id}`	Audit record with failure diagnostics	`GET`
`/api/v1/observability/events`	PanFS events snapshot (last N)	`GET`
`/api/v1/observability/events/stream`	PanFS events SSE stream (append deltas)	`GET`
`/api/v1/observability/logs`	Logs snapshot (last N)	`GET`
`/api/v1/observability/logs/stream`	Logs SSE stream (append deltas)	`GET`
`/api/v1/observability/logs/{logsql_endpoint}`	VictoriaLogs /select/logsql proxy	`GET`
`/api/v1/observability/metrics/query`	Prometheus instant query proxy	`GET`
`/api/v1/observability/metrics/query_range`	Prometheus range query proxy	`GET`
`/api/v1/observability/metrics/stream`	Grouped metric SSE stream (append deltas for several keys)	`GET`
`/api/v1/observability/metrics/targets`	List metrics targets	`GET`
`/api/v1/observability/metrics/{key}`	Metric snapshot (range window)	`GET`

Path	Summary	Method
`/api/v1/observability/metrics/{key}/stream`	Metric SSE stream (append deltas)	`GET`
`/api/v1/observability/storagesets/status`	Storage-set alert status	`GET`

`GET /api/v1/observability/alerts`

Alerts snapshot (keyed set)

The current alert set of the caller's own tenant, most severe first. A platform caller widens to one tenant with `tenant_id`, or to every tenant with `all_tenants=true`; its own tenant is the realm's telemetry, so the default read is the realm's health rather than the tenants' records. A tenant-filtered read returns only records that carry a tenant, so during a rolling upgrade a node that has not taken the new configuration yet produces records no tenant-filtered read returns. That fails closed: it is a completeness limit, never a disclosure.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	query	string (uuid), nullable	no	Platform tenant only. Read the observability records of another tenant instead of your own.
`all_tenants`	query	boolean	no	Platform tenant only. Read the observability records of every tenant. Omitted, a read covers the caller's own tenant alone - for the platform tenant that is the realm's own telemetry, which carries the platform tenant ID, and not the records tenants own.

Responses

Status	Description	Schema
200	Successful Response	array
422	Validation Error	`HTTPValidationError`

`GET /api/v1/observability/alerts/stream`

Alerts SSE stream (added/resolved)

Deltas for the same feed the alerts snapshot reads, taking the same `tenant_id` scope. The stream ends itself with one terminal `end` event whose `reason` is `credential_expired` (refresh the token and reconnect), `credential_revoked` (sign in again), `permission_revoked` (the caller no longer holds the grant), `lifetime_expired` (the maximum stream lifetime; reconnect) or `recheck_unavailable` (the credential could not be re-checked several times running; reconnect with backoff). The checks run on stream ticks, which on a quiet feed happen every 20 seconds, so an end can lag by about that.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	query	string (uuid), nullable	no	Platform tenant only. Read the observability records of another tenant instead of your own.
`all_tenants`	query	boolean	no	Platform tenant only. Read the observability records of every tenant. Omitted, a read covers the caller's own tenant alone - for the platform tenant that is the realm's own telemetry, which carries the platform tenant ID, and not the records tenants own.

Responses

Status	Description	Schema
200	Server-Sent Events stream (`text/event-stream`).	-
422	Validation Error	`HTTPValidationError`

`GET /api/v1/observability/audit`

Audit trail snapshot (last N)

Newest API audit records, newest first, one row per request. The server owns the LogsQL query, so clients pass no query text. Records are deduplicated by their `request_id` - one record per API call. A tenant caller reads its own tenant's records; and so does a platform caller by default; a platform caller widens with `tenant_id` or `all_tenants=true`. `user_id` reads one user's records; the filters compose. `only` keeps just the records recorded under the named actions, and `skip` drops them - `skip=list,view` hides the read-only bulk, leaving the records that changed something, while `only=create,delete` reads the lifecycle alone. `since` and `until` bound the record time inclusively (RFC 3339; a value with no timezone reads as UTC); an inverted range matches nothing. A tenant-filtered read returns only records that carry a tenant, so during a rolling upgrade a node that has not taken the new configuration yet produces records no tenant-filtered read returns. That fails closed: it is a completeness limit, never a disclosure.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`user_id`	query	string (uuid), nullable	no	Read one user's audit records, including what they did while switched into another tenant. Omit it to read every record, whoever made the request. Applied upstream, so the `limit` is spent on the records asked for; composes with `tenant_id`.
`skip`	query	string, nullable	no	Comma-separated action names to drop, e.g. `skip=list,view`. Any action name is accepted; one matching no records excludes nothing. A record carrying no action survives every exclusion.
`only`	query	string, nullable	no	Comma-separated action names to keep, e.g. `only=create,delete`. Any action name is accepted; a record carrying no action is dropped. Composes with `skip`, which wins where they overlap.

Name	In	Type	Required	Description
`since`	query	string (date-time), nullable	no	Oldest record time to return (inclusive, RFC 3339; no timezone reads as UTC).
`until`	query	string (date-time), nullable	no	Newest record time to return (inclusive, RFC 3339; no timezone reads as UTC).
`limit`	query	integer	no	Maximum records to return; unbounded above, so ask only for what you will render.
`tenant_id`	query	string (uuid), nullable	no	Platform tenant only. Read the observability records of another tenant instead of your own.
`all_tenants`	query	boolean	no	Platform tenant only. Read the observability records of every tenant. Omitted, a read covers the caller's own tenant alone - for the platform tenant that is the realm's own telemetry, which carries the platform tenant ID, and not the records tenants own.

Responses

Status	Description	Schema
200	Successful Response	array
422	Validation Error	`HTTPValidationError`

`GET /api/v1/observability/audit/stream`

Audit SSE stream (append deltas)

Deltas for the same feed the audit snapshot reads, taking the same `tenant_id`, `user_id`, `skip` and `only` filters - pass the values the snapshot was taken with, so the stream continues it. `since` / `until` have no stream counterpart: a stream starts at its baseline and appends from there. The stream ends itself with one terminal `end` event whose `reason` is `credential_expired` (refresh the token and reconnect), `credential_revoked` (sign in again), `permission_revoked` (the caller no longer holds the grant), `lifetime_expired` (the maximum stream lifetime; reconnect) or `recheck_unavailable` (the credential could not be re-checked several times running; reconnect with backoff). The checks run on stream ticks, which on a quiet feed happen every 20 seconds, so an end can lag by about that.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`user_id`	query	string (uuid), nullable	no	Read one user's audit records, including what they did while switched into another tenant. Omit it to read every record, whoever made the request. Applied upstream, so the `limit` is spent on the records asked for; composes with `tenant_id`.
`skip`	query	string, nullable	no	Comma-separated action names to drop, as on the audit snapshot.
`only`	query	string, nullable	no	Comma-separated action names to keep, as on the audit snapshot.
`tenant_id`	query	string (uuid), nullable	no	Platform tenant only. Read the observability records of another tenant instead of your own.
`all_tenants`	query	boolean	no	Platform tenant only. Read the observability records of every tenant. Omitted, a read covers the caller's own tenant alone - for the platform tenant that is the realm's own telemetry, which carries the platform tenant ID, and not the records tenants own.

Responses

Status	Description	Schema
200	Server-Sent Events stream (`text/event-stream`).	-
422	Validation Error	`HTTPValidationError`

`GET /api/v1/observability/audit/{request_id}`

Audit record with failure diagnostics

One audit record plus the correlated pan-api log detail for that request. `diagnostics` is parsed from the correlated ERROR line when one is found, and is always present, empty when no correlated lines exist. 404 for an unknown or aged-out ID, and for a tenant caller also for a record of another tenant or one carrying no tenant, so a tenant cannot probe which IDs exist. There is no all-tenants form here, because a request ID already names one record: a platform caller reads another tenant's record by naming it with `tenant_id`, and gets the same 404 otherwise.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`request_id`	path	string	yes	The audit record ID, as returned by the audit snapshot.
`tenant_id`	query	string (uuid), nullable	no	Platform tenant only. Read an audit record of another tenant instead of your own. There is no all-tenants form here: a request ID already names exactly one record, so the only question is whose record the caller may reach.

Responses

Status	Description	Schema
200	Successful Response	`AuditRecordDetail`
422	Validation Error	`HTTPValidationError`

Response Body (200, `AuditRecordDetail`)

Field	Type	Required	Default	Example	Description
`action`	string	no	(empty string)		Action performed on that resource, e.g. `create` .

Field	Type	Required	Default	Example	Description
`auth_token_id`	string	no	(empty string)		Id of the API token that authenticated the request; empty for a session.
`client_ip`	string	no	(empty string)		Client IP address of the request.
`diagnostics`	`AuditDiagnostics` object	no	n/a		Correlated pan-api log detail for this request.
`duration_ms`	number	no	`0`		Request duration in milliseconds.
`id`	string	yes	n/a		The record's own identifier, which is its `request_id`.
`method`	string	yes	n/a		HTTP method of the audited request.
`principal_id`	string	no	(empty string)		Who acted, as `<tenant id>/<principal id>`. Under a tenant switch this is the person who switched, in their own tenant. Empty when the request was not authenticated.
`principal_name`	string	no	(empty string)		`<tenant name>/<name>`, or the bare name when the tenant name was not known.
`principal_type`	string	no	(empty string)		`user` | `service-account`.
`query`	string	no	(empty string)		Query string of the audited request.
`reason`	any, nullable	no	`null`		Reason for a denied or failed result, if recorded.
`request_body`	any, nullable	no	`null`		Request body, secret-redacted at write time.

Field	Type	Required	Default	Example	Description
`request_id`	string	yes	n/a		Audit request ID; the identity used for deduplication.
`resource`	string	no	(empty string)		Resource the request acted on, e.g. `volumes`.
`result`	string	yes	n/a		Outcome of the request: `success` | `denied` | `failed`.
`severity`	string	yes	n/a		Audit-significance severity: `INFO` | `WARNING`.
`status_code`	integer, nullable	no	`null`		HTTP response status code.
`target_id`	string	no	(empty string)		Id of the specific instance acted on, if any.
`tenant_id`	string	no	(empty string)		The tenant the credential acts in: the caller's own, or under a tenant switch the tenant switched into. It differs from the tenant half of `principal_id` only under a switch.
`time`	string	yes	n/a		When the request was audited (RFC 3339, from VictoriaLogs `_time`).
`url`	string	yes	n/a		URL path of the audited request.
`user_agent`	string	no	(empty string)		Client `User-Agent` header of the request.

`GET /api/v1/observability/events`

PanFS events snapshot (last N)

Newest PanFS management events, newest first, one row per event. The server owns the query, so clients pass no query text. Records are deduplicated by their PanFS `event_id`, which the feed repeats whenever the events file is re-read, or by their `request_id` when they come from the audit trail and carry no `event_id`. A tenant caller reads its own tenant's events, and so does a platform caller by default; a platform caller widens to one tenant's with `tenant_id`. A tenant-filtered read returns only records that carry a tenant, so during a rolling upgrade a node that has not taken the new configuration yet produces records no tenant-filtered read returns. That fails closed: it is a completeness limit, never a disclosure.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`limit`	query	integer	no	
`tenant_id`	query	string (uuid), nullable	no	Platform tenant only. Read the observability records of another tenant instead of your own.
`all_tenants`	query	boolean	no	Platform tenant only. Read the observability records of every tenant. Omitted, a read covers the caller's own tenant alone - for the platform tenant that is the realm's own telemetry, which carries the platform tenant ID, and not the records tenants own.

Responses

Status	Description	Schema
200	Successful Response	array
422	Validation Error	`HTTPValidationError`

`GET /api/v1/observability/events/stream`

PanFS events SSE stream (append deltas)

Deltas for the same feed the snapshot reads, taking the same `tenant_id` filter. Pass the value the snapshot was taken with, so the stream continues it. The stream ends itself with one terminal `end` event whose `reason` is `credential_expired` (refresh the token and reconnect), `credential_revoked` (sign in again), `permission_revoked` (the caller no longer holds the grant), `lifetime_expired` (the maximum stream lifetime; reconnect) or `recheck_unavailable` (the credential could not be re-checked several times running; reconnect with backoff). The checks run on stream ticks, which on a quiet feed happen every 20 seconds, so an end can lag by about that.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`tenant_id`	query	string (uuid), nullable	no	Platform tenant only. Read the observability records of another tenant instead of your own.
`all_tenants`	query	boolean	no	Platform tenant only. Read the observability records of every tenant. Omitted, a read covers the caller's own tenant alone - for the platform tenant that is the realm's own telemetry, which carries the platform tenant ID, and not the records tenants own.

Responses

Status	Description	Schema
200	Server-Sent Events stream (`text/event-stream`).	-
422	Validation Error	`HTTPValidationError`

`GET /api/v1/observability/logs`

Logs snapshot (last N)

Newest log records matching a LogsQL `query`. A query the backend cannot parse answers 400 with `Invalid query:` and the backend's own diagnosis, which never echoes the query itself. A backend that is down answers 503.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`query`	query	string	no	
`limit`	query	integer	no	

Responses

Status	Description	Schema
200	Successful Response	-
422	Validation Error	`HTTPValidationError`

`GET /api/v1/observability/logs/stream`

Logs SSE stream (append deltas)

Deltas for the same feed the logs snapshot reads. A query the backend cannot parse answers 400 with `Invalid query:` and the backend's own diagnosis, which never echoes the query itself. A backend that is down answers 503. The stream ends itself with one terminal `end` event whose `reason` is `credential_expired` (refresh the token and reconnect), `credential_revoked` (sign in again), `permission_revoked` (the caller no longer holds the grant), `lifetime_expired` (the maximum stream lifetime; reconnect) or `recheck_unavailable` (the credential could not be re-checked several times running; reconnect with backoff). The checks run on stream ticks, which on a quiet feed happen every 20 seconds, so an end can lag by about that.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`query`	query	string	no	

Responses

Status	Description	Schema
200	Server-Sent Events stream (`text/event-stream`).	-
422	Validation Error	`HTTPValidationError`

`GET /api/v1/observability/logs/{logsql_endpoint}`

VictoriaLogs /select/logsql proxy

Proxies to VictoriaLogs `/select/logsql/{endpoint}` on the local `vlselect` instance (co-located on the Director Node). All query parameters are forwarded as-is (GET only). See VictoriaLogs querying docs for supported endpoints and their parameters.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`logsql_endpoint`	path	`VictoriaLogsLogsqlEndpoint` (`query` , `tail` , `hits` , `facets` , `stats_query` , `stats_query_range` , `stream_ids` , `streams` , `stream_field_names` , `stream_field_values` , `field_names` , `field_values`)	yes	VictoriaLogs `/select/logsql/{endpoint}` path segment. See querying docs.
`query`	query	string, nullable	no	LogsQL selector (upstream `query`). Used by `query` , `tail` , `hits` , `facets` , `stats_query` , `stats_query_range` , `stream_ids` , `streams` , `stream_field_names` , `stream_field_values` , `field_names` , `field_values` .

Name	In	Type	Required	Description
`start`	query	string, nullable	no	Time range start (Victoria timestamp format). `query` (optional window), `hits`, `facets`, `stats_query_range`, `stream_ids`, `streams`, `stream_field_names`, `stream_field_values`, `field_names`, `field_values`.
`end`	query	string, nullable	no	Time range end, exclusive `[start, end)`. Same endpoints as `start` where applicable.
`step`	query	string, nullable	no	Histogram bucket width. `hits`, `stats_query_range` (e.g. `1h`).
`time`	query	string, nullable	no	Evaluation instant for `stats_query` (not used by `stats_query_range` range API).
`limit`	query	string, nullable	no	Cap rows or facet/field results. `query`, `facets`, `stream_ids`, `streams`, `field_values`, etc., per VictoriaLogs docs.
`offset`	query	string, nullable	no	Pagination skip on `query`; timezone offset on `hits` / `stats_query_range`; delivery delay on `tail` - same parameter name, semantics per upstream endpoint.
`timeout`	query	string, nullable	no	Max query duration (e.g. `4.2s`). Primarily `query`.
`format`	query	string, nullable	no	Response shape, e.g. `csv` for `query`.
`start_offset`	query	string, nullable	no	Historical lookback before live tail (e.g. `1h`). `tail` only.
`refresh_interval`	query	string, nullable	no	Poll interval for new logs. `tail` only.
`fields_limit`	query	string, nullable	no	Max number of `fields` groups on `hits`.
`ignore_pipes`	query	string, nullable	no	Set to `1` to ignore LogsQL pipes where upstream supports it (`hits`, `facets`, ...).
`max_values_per_field`	query	string, nullable	no	`facets`: cap distinct values per field.

Name	In	Type	Required	Description
`max_value_len`	query	string, nullable	no	`facets` : max length of field values to consider.
`keep_const_fields`	query	string, nullable	no	`facets` : set to `1` to include fields with a single constant value.
`field`	query	string, nullable	no	Field name for grouping or lookup. `hits` (repeatable in raw URL), `stream_field_values` , `field_values` . For multiple `field=` values use repeated query keys (OpenAPI shows a single value).
`filter`	query	string, nullable	no	Substring filter on names or values. `stream_field_names` , `stream_field_values` , `field_names` , `field_values` .
`timezone`	query	string, nullable	no	Bucket alignment / timezone for `hits` and `stats_query_range` when documented upstream.
`extra_filters`	query	string, nullable	no	Extra LogsQL filters applied by VictoriaLogs to all HTTP querying APIs (JSON or filter text).
`extra_stream_filters`	query	string, nullable	no	Extra stream filters applied by VictoriaLogs (JSON or stream filter text).
`hidden_fields_filters`	query	string, nullable	no	Hide specific log fields during execution (VictoriaLogs "hidden fields" API).

Responses

Status	Description	Schema
200	Successful response or upstream error body (status may vary).	-
403	Forbidden. Insufficient permissions.	-
422	Unknown `logsql_endpoint` path segment (not a VictoriaLogs logsql endpoint).	-
503	Logs backend unreachable or request timed out.	-

`GET /api/v1/observability/metrics/query`

Prometheus instant query proxy

Forwards only Prometheus instant-query parameters (`query` , `time` , `timeout` , `step`) to the Victoria Metrics `/api/v1/query` endpoint; other query parameter names are rejected. Upstream status, body, and end-to-end headers are forwarded; hop-by-hop headers (RFC 7230) and any names listed in `Connection` are stripped. `Content-Length` and `Content-Encoding` are omitted because the response body is returned already decoded. If `Content-Type` is missing on a 200 response, `application/json` is assumed; other status codes are not given a default type. Query parameters are validated with `PrometheusInstantQueryParams` ; extra names yield 422.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`query`	query	string, nullable	no	Prometheus expression (MetricsQL / PromQL-compatible instant query). Evaluated by the upstream VictoriaMetrics `/api/v1/query` endpoint.
`time`	query	string, nullable	no	Evaluation time as Unix timestamp or RFC3339 (Prometheus `time`).
`timeout`	query	string, nullable	no	Maximum evaluation timeout (Prometheus `timeout`), e.g. `5s` .
`step`	query	string, nullable	no	Optional resolution step (VictoriaMetrics compatibility). Range queries use `/api/v1/query_range` on the upstream; this route proxies instant queries only.

Responses

Status	Description	Schema
200	Successful query or upstream error body (status may vary).	-
403	Forbidden. Insufficient permissions.	-

Status	Description	Schema
422	Unknown query parameter names (not in the allow-listed instant-query model).	-
503	Metrics backend unreachable or request timed out.	-

`GET /api/v1/observability/metrics/query_range`

Prometheus range query proxy

Forwards Prometheus range-query parameters (`query` , `start` , `end` , `step` , `timeout`) to the Victoria Metrics `/api/v1/query_range` endpoint; other query parameter names are rejected. Upstream status, body, and end-to-end headers are forwarded; hop-by-hop headers (RFC 7230) and any names listed in `Connection` are stripped. `Content-Length` and `Content-Encoding` are omitted because the response body is returned already decoded. If `Content-Type` is missing on a 200 response, `application/json` is assumed; other status codes are not given a default type. Query parameters are validated with `PrometheusRangeQueryParams` ; extra names yield 422.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`query`	query	string, nullable	no	Prometheus expression (MetricsQL / PromQL-compatible range query). Evaluated by the upstream VictoriaMetrics `/api/v1/query_range` endpoint.
`start`	query	string, nullable	no	Range start as Unix timestamp or RFC3339 (Prometheus `start`).
`end`	query	string, nullable	no	Range end as Unix timestamp or RFC3339 (Prometheus `end`).
`step`	query	string, nullable	no	Resolution step width as duration string (e.g. `60s` , `5m`) or float seconds.
`timeout`	query	string, nullable	no	Maximum evaluation timeout (Prometheus `timeout`), e.g. `5s` .

Responses

Status	Description	Schema
200	Successful query or upstream error body (status may vary).	-
403	Forbidden. Insufficient permissions.	-

Status	Description	Schema
422	Unknown query parameter names (not in the allow-listed range-query model).	-
503	Metrics backend unreachable or request timed out.	-

`GET /api/v1/observability/metrics/stream`

Grouped metric SSE stream (append deltas for several keys)

Streams `append` deltas for every key in `k` over a single connection. Each event carries `{"key": "<metric key>", "samples": [...]}` so the client can route it to the right chart. Snapshots stay per-key (`GET /observability/metrics/{key}`).

Grouping keeps concurrent SSE connections low: browsers cap HTTP/1.1 connections at six per host, so one stream per chart would starve ordinary REST calls. Each key is still subscribed as its own broker feed, so clients requesting overlapping sets share one upstream poller per common key.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`step`	query	integer	no	Resolution step (seconds); rounded up to a supported tier.
`k`	query	string	yes	Comma-separated metric keys, e.g. `disk_throughput,disk_iops` .

Responses

Status	Description	Schema
200	Server-Sent Events stream (`text/event-stream`).	-
404	One or more keys in `k` are not registered metrics.	-
422	Empty `k` , or more keys than the per-stream cap.	-

`GET /api/v1/observability/metrics/targets`

List metrics targets

Returns a list of cluster nodes grouped by their type and state in Prometheus HTTP Service Discovery format. Each target is `<node management IP>:<port>` where the port is the observability agent (telegraf) Prometheus exporter serving `/metrics` on that node - 9273 by default. `__meta_node_state` is the PanFS node state, not the health of the scrape endpoint. Nodes that do not run the observability agent are still listed.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Responses

Status	Description	Schema
200	Successful retrieval of metrics targets.	array
400	Bad request due to invalid parameters or service error.	-
401	Unauthorized.	-
403	Forbidden. Insufficient permissions.	-
404	Requested metrics target or resource not found.	-
500	Internal server error.	-

`GET /api/v1/observability/metrics/{key}`

Metric snapshot (range window)

Range-query snapshot for a named metric over the lookback window.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`key`	path	string	yes	
`window`	query	integer	no	Lookback window (seconds).
`step`	query	integer	no	Resolution step (seconds); rounded up to a supported tier.

Responses

Status	Description	Schema
200	Successful Response	-
422	Validation Error	`HTTPValidationError`

`GET /api/v1/observability/metrics/{key}/stream`

Metric SSE stream (append deltas)

SSE: `append` events as new metric samples arrive (deltas only).

Authentication: required (`PasswordLogin` or `BearerJWT`).

Parameters

Name	In	Type	Required	Description
`key`	path	string	yes	
`step`	query	integer	no	Resolution step (seconds); rounded up to a supported tier.

Responses

Status	Description	Schema
200	Server-Sent Events stream (`text/event-stream`).	-
422	Validation Error	`HTTPValidationError`

`GET /api/v1/observability/storagesets/status`

Storage-set alert status

Active-alert presence per StorageSet, resolved from one query over all PanFS alerts. Only sets carrying at least one critical or warning alert are listed - an absent set means no alerts. Callers combine this with the set's node/spare counts to render its status badge.

Authentication: required (`PasswordLogin` or `BearerJWT`).

Responses

Status	Description	Schema
200	Successful Response	array

Appendix A. Error Responses

The status codes below recur across nearly every endpoint in this guide with the same meaning each time; endpoint sections list the codes their spec declares, so the codes in this table apply to every endpoint whether or not they are listed there.

Status	Meaning	Details
`202`	Accepted.	The request was accepted and the desired state recorded, but the change has not happened yet. Poll the resource, such as an S3 gateway instance's `observed` field, to see when it takes effect.
`207`	Multi-Status.	Returned by calls that act on several targets at once, such as the S3 instance calls that fan out across nodes, when some succeeded and some did not. The body lists a result for each target.
`400`	Bad request.	The request was well-formed JSON but rejected by request-specific validation, for example an out-of-range field or a value that conflicts with another field. `detail` is a human-readable explanation.
`401`	Unauthorized.	The `Authorization` header is missing, the token is invalid or expired, or (for `POST /auth/login`) the credentials were wrong. Login failures and unknown users answer with the same body, to avoid user and tenant enumeration.
`403`	Forbidden.	The caller is authenticated but lacks the permission the endpoint requires, or a request-specific rule blocks the call for a caller in this state (for example, a delegated tenant-switch session cannot issue an API token).
`404`	Not found.	No resource exists at the given identifier, or (for endpoints that avoid leaking existence, such as another user's API token) it belongs to someone else and answers identically to not existing at all.
`409`	Conflict.	The request is valid but conflicts with the current state of the system, for example creating a volume whose pathname already exists or deleting a resource something else still depends on.
`422`	Validation error.	The request body or parameters failed schema validation before any business logic ran. `detail` is a list of `ValidationError` objects, each naming the offending field (`loc`), the problem (`msg`), and the failure type (`type`). See `HTTPValidationError` in Appendix B.
`500`	Internal server error.	An unexpected failure on the server side. The request may or may not have taken effect; check the resource's current state before retrying a non-idempotent call.

Status	Meaning	Details
`503`	Service unavailable.	A backend dependency (typically the storage backend) was unreachable, or something the request needs is not in place, such as a running S3 gateway instance for a bucket call. The request changed nothing. A 503 for an unreachable dependency is worth retrying once it recovers; a 503 for a missing prerequisite will not clear on retry until the prerequisite is met.
`504`	Gateway timeout.	The operation took too long to answer. It may still have completed, since a volume or bucket can appear afterward, so list the resources before retrying a create.

Appendix B. Schema Index

Every schema referenced anywhere in this guide, alphabetically, with a one-line description and the fields of any object schema. Enum schemas (such as `VolumeLayout` or `EfsaMode`) show their allowed values instead of a field table. Field tables include an **Example** column drawn from the specification's own sample values where the field carries one, and a schema carrying a whole-object example (such as `AboutResponse`) shows it as a JSON block after its field table. Request and response bodies in Chapters 4 through 15 already expand the schema they use directly, examples included; this appendix is for looking up a schema named only in passing, such as a nested field's type.

Schema entries below have been deliberately left out of the table of contents. Use the index table here or the in-page search to jump to an entry.

Schema	Schema	Schema
`AboutResponse`	`Action`	`AgeUnit`
`AlertEvent`	`ApiTokenCreateRequest`	`ApiTokenCreatedResponse`
`ApiTokenListResponse`	`ApiTokenResponse`	`ApiTokenState`
`ApiTokenStateReason`	`ArchivedTenantSchema`	`Attachment`
`AttachmentCreateRequest`	`AttachmentListResponse`	`AttachmentTunnel`
`AuditDiagnostics`	`AuditRecord`	`AuditRecordDetail`
`Body_refresh_api_v1_auth_refresh_post`	`ChangePasswordRequest`	`CollectedHostResponse`
`DelegatedTokenResponse`	`DirectorEntry`	`EfsaMode`
`FsUser`	`FsUserCreateRequest`	`FsUserListResponse`
`FsUserPatchRequest`	`FsUserResponse`	`FsUserS3View`
`FsUserSecretResponse`	`GetS3ConfigResponse`	`GetTenantResponse`
`GetTenantStorageSetResponse`	`GetUserResponse`	`HTTPValidationError`
`Hardware`	`HardwareBlinkDriveLedActionRequest`	`HardwareBlinkNodeLedActionRequest`
`HardwareDriveLedActionParams`	`HardwareGetResponse`	`HardwareListResponse`

Schema	Schema	Schema
HardwareStopDriveLedActionRequest	HardwareStopNodeLedActionRequest	HostResult
ListArchivedTenantResponse	ListS3BucketsResponse	ListTenantResponse
ListTenantStorageSetResponse	LocalAuthSettings	LogAgeRequest
LogAgeResponse	LogCollectionCreateRequest	LogCollectionListResponse
LogCollectionOptionsResponse	LogCollectionResponse	LogCollectionScope
LogCollectionStatus	LoginForm	MeResponse
MountInfoResponse	Node	NodeDrainState
NodeDrainStatusResponse	NodeFlag	NodeGetResponse
NodeHaltActionRequest	NodeListResponse	NodeRebootActionRequest
NodeRole	NodeStartDrainRequest	NodeStatus
NodeStopDrainRequest	ObservabilityLabels	ObservabilityTarget
PamAuthSettings	PanfsEvent	Principal
QuotasPolicy	RaidType	Realm
RealmClientMode	RealmLockActionRequest	RealmLogSizing
RealmMoveServiceActionParams	RealmMoveServiceActionRequest	RealmOfflineActionParams
RealmOfflineActionRequest	RealmOfflineMode	RealmOnlineActionRequest
RealmPresidentFlag	RealmRebootActionRequest	RealmRefResponse
RealmRehomeRootActionParams	RealmRehomeRootActionRequest	RealmResponse
RealmSegmentTypeRequest	RealmSegmentTypeResponse	RealmState
RealmStateFlag	RealmUnlockActionRequest	Resource
RolesResponse	S3AggregatorConfigSchema	S3BucketCreateRequest
S3BucketSchema	S3ConfigCreateRequest	S3ConfigPatchRequest
S3ConfigSchema	S3ConfigUpdateResponse	S3DnsConfigSchema

Schema	Schema	Schema
S3DnsMode	S3EnabledRequest	S3EnabledResponse
S3InstanceCreateRequest	S3InstanceCreateResponse	S3InstanceFanoutResponse
S3InstanceHealthSchema	S3InstanceListResponse	S3InstancePerNodeResult
S3InstanceSchema	S3InstanceViewSchema	S3PrefixRequest
S3PrefixResponse	S3Profile	S3TlsConfigResponseSchema
S3TlsConfigSchema	S3TlsMode	ServiceAccountCreateRequest
ServiceAccountListResponse	ServiceAccountResponse	ServiceAccountUpdateRequest
ServiceInstance	SetRolesRequest	StorageSet
StorageSetAddStorageActionRequest	StorageSetAddStorageMode	StorageSetAddStorageParams
StorageSetAlertStatus	StorageSetAssignment	StorageSetCondition
StorageSetCreateRequest	StorageSetCreateResponse	StorageSetFlag
StorageSetGetResponse	StorageSetListResponse	StorageSetPatchRequest
StorageSetSpaceCapacity	StorageSetSpaceResponse	StorageSetState
StorageSetStatus	SubjectKind	Subnet
SubnetCreateRequest	SubnetCreateResponse	SubnetGetResponse
SubnetListResponse	SubnetPatchRequest	SubnetSegmentType
Tenant	TenantCreateRequest	TenantCreateResponse
TenantPatchRequest	TenantRole	TenantRolesResponse
TenantStatus	TenantStorageSetQuotaRequest	TenantSummarySchema
TenantSwitchRequest	TokenResponse	User
UserCreateRequest	UserCreateResponse	UserListResponse
UserPatchRequest	ValidationError	VictoriaLogsLogsqlEndpoint
Volume	VolumeAccessMode	VolumeCompressionMode

Schema	Schema	Schema
`VolumeCreateRequest`	`VolumeCreateResponse`	`VolumeGetResponse`
`VolumeLayout`	`VolumeListResponse`	`VolumePatchRequest`
`VolumePermission`	`VolumeRecoverActionParams`	`VolumeRecoverActionRequest`
`VolumeRecoverMode`	`VolumeRenameActionParams`	`VolumeRenameActionRequest`
`VolumeSpaceResponse`	`VolumeStateOption`	`VolumeStatus`

`AboutResponse`

About response exposing service identity and version.

Field	Type	Required	Default	Example	Description
`name`	string	yes	n/a	`PanAPI`	Service name
`panfs_version`	string, nullable	no	`null`	`12.0.0`	PanFS realm software version read from the local filesystem; null when unavailable
`version`	string	yes	n/a	`1.2.3`	Service version, derived from the package metadata at build time

`Action`

RBAC actions that can be performed on resources. The `finalize-delete` action applies to calls made by the platform's own S3 services on each node. Customers, Vantage, and panctl do not use it.

Allowed values: `list`, `view`, `create`, `update`, `delete`, `action`, `read-s3-secret`, `rotate-s3-secret`, `switch`, `finalize-delete`, `download`

`AgeUnit`

Units of the `age` option.

Allowed values: `minutes`, `hours`, `days`, `weeks`, `months`

`AlertEvent`

One alert vmalert is currently firing, projected from an `ALERTS` series.

The server-shaped payload for both the alerts snapshot (`GET /observability/alerts`) and the `added` / `resolved` SSE deltas.

Which fields carry a value depends on the rule that fired: only `alert_rule` , `message` , `severity` , `alert_group` and `source` come from every rule. There is no timestamp - an alert states that a condition is true now.

Field	Type	Required	Default	Example	Description
`action_message`	string	no	(empty string)		Recommended action (“User Action”); empty when none.
`alert_code`	string	no	(empty string)		PanFS alert code, shown as `#CODE` in the UI. Empty on non-PanFS rules.
`alert_group`	string	no	(empty string)		Rule group, e.g. `panfs.alerts` , `panfs.processes` .
`alert_name`	string	no	(empty string)		PanFS event constant, e.g. `PAN_MGMT_EV_BRICK_NIC6_NOT_ACTIVE` .
`alert_rule`	string	yes	n/a		vmalert rule that fired, e.g. `PanFSAlertActive` , `ProcessDown` .
`category`	string	no	(empty string)		Condition category, e.g. `hardware` , `software` .
`id`	string	yes	n/a		Stable alert ID (`alert_rule|node_name|message`).
`ip_address`	string	no	(empty string)		That node’s management IP address.
`is_manually_clearable`	boolean	no	`false`		Whether the alert can be cleared by a user.

Field	Type	Required	Default	Example	Description
`kind`	string	no	(empty string)		Condition kind, e.g. `state`.
`message`	string	yes	n/a		Human-readable alert message.
`node_dataserial`	string	no	(empty string)		That node's data serial, for linking to its detail page.
`node_name`	string	no	(empty string)		Name of the node the alert is about; empty on realm-wide alerts.
`node_type`	string	no	(empty string)		Type of that node (e.g. vpod, vdp, director_l).
`process_name`	string	no	(empty string)		Process the alert is about (`ProcessDown` only).
`severity`	one of: `error`, `warning`, `info`	yes	n/a		Normalized severity level.
`source`	string	no	(empty string)		Signal source, e.g. `panfs`.
`tenant_id`	string	no	(empty string)		Tenant the alert is about; the platform tenant on a realm-wide condition. Attribution only - it does not scope who may read the alert.

`ApiTokenCreateRequest`

Request to issue a long-lived API token for the caller.

Field	Type	Required	Default	Example	Description
`description`	string, nullable	no	`null`		What this token is for. Optional; `name` remains the handle an operator identifies it by.
`expires_in_days`	integer	yes	n/a	`30`, `90`, `180`, `365`	Lifetime in days. Required, and must name one of the offered values (30, 90, 180, 365), so a long lifetime is always a deliberate choice.
`name`	string	yes	n/a		Name for this token, unique among the caller's usable tokens. An operator identifies a token by it. A revoked or expired token releases its name, so a rotated credential can keep the name it had.

Example

```
{
   "description" : "Nightly backup runner",
   "expires_in_days" : 90,
   "name" : "nightly-backup@ci-runner-04"
}
```

`ApiTokenCreatedResponse`

The issuance response - the only one that ever contains the secret.

Field	Type	Required	Default	Example	Description
`created_at`	string (date-time)	yes	n/a		When the secret was created. Never changes.
`created_by`	`Principal` object	yes	n/a		Who issued the token.
`description`	string, nullable	no	`null`		What this token is for, if a description was given.
`expires_at`	string (date-time)	yes	n/a		Absolute expiry.
`id`	string	yes	n/a		Public token identifier. Appears in audit events; not a secret.
`name`	string	yes	n/a		The handle given at issuance, unique among the owner's usable tokens.
`revoked_at`	string (date-time), nullable	no	`null`		When the token was revoked, if it was. The only lifecycle fact the record stores.
`revoked_by`	`Principal` object, nullable	no	`null`		Who revoked the token, if it was.

Field	Type	Required	Default	Example	Description
`roles`	array of string, nullable	no	`null`		The role ceiling held in the record: the owner's own roles as at issuance. Not chosen - a token carries what its owner held, and never more than they hold now. Null for a service account key, which carries no ceiling: it resolves whatever the account holds at the time of each request.
`state`	`ApiTokenState` (`active` , `disabled`)	yes	n/a		Whether the token still authenticates. Computed per response, never stored.
`state_reason`	`ApiTokenStateReason` (`expired` , `revoked`), nullable	no	`null`		Why the token is disabled. Null while it is active. Revocation outranks expiry.
`token`	string	yes	n/a	`vdt1_6XuU1pyDS0abcDEF12345w_9fK3mZq7Xw2Lb1cD_hT8n...7bQ2xM`	The token. Returned once and never retrievable again; a lost token must be replaced.
`warning`	string	no	`Store this token now. It cannot be retrieved again.`		Reminder that the secret is shown exactly once.

`ApiTokenListResponse`

The caller's own tokens.

Field	Type	Required	Default	Example	Description
`tokens`	array of `ApiTokenResponse` object	yes	n/a		Token metadata, newest first.
`total`	integer	yes	n/a		Number of tokens returned.

`ApiTokenResponse`

A token's metadata. Carries neither the secret nor its digest.

Field	Type	Required	Default	Example	Description
`created_at`	string (date-time)	yes	n/a		When the secret was created. Never changes.
`created_by`	`Principal` object	yes	n/a		Who issued the token.
`description`	string, nullable	no	`null`		What this token is for, if a description was given.
`expires_at`	string (date-time)	yes	n/a		Absolute expiry.
`id`	string	yes	n/a		Public token identifier. Appears in audit events; not a secret.
`name`	string	yes	n/a		The handle given at issuance, unique among the owner's usable tokens.
`revoked_at`	string (date-time), nullable	no	`null`		When the token was revoked, if it was. The only lifecycle fact the record stores.
`revoked_by`	`Principal` object, nullable	no	`null`		Who revoked the token, if it was.

Field	Type	Required	Default	Example	Description
`roles`	array of string, nullable	no	`null`		The role ceiling held in the record: the owner's own roles as at issuance. Not chosen - a token carries what its owner held, and never more than they hold now. Null for a service account key, which carries no ceiling: it resolves whatever the account holds at the time of each request.
`state`	`ApiTokenState` (`active` , `disabled`)	yes	n/a		Whether the token still authenticates. Computed per response, never stored.
`state_reason`	`ApiTokenStateReason` (`expired` , `revoked`), nullable	no	`null`		Why the token is disabled. Null while it is active. Revocation outranks expiry.

`ApiTokenState`

Whether a token still authenticates. Derived per response from the record's own dates.

Allowed values: `active` , `disabled`

`ApiTokenStateReason`

Why a disabled token stopped authenticating.

Allowed values: `expired` , `revoked`

`ArchivedTenantSchema`

A deleted tenant, as the platform archive kept it.

Its own entity, not a tenant with fields missing: nothing here can be acted on, and the tenant it describes exists in neither the storage fabric nor the configuration store. `display_name` and `email` are the values it was deleted with - the archive is the only remaining copy - while `created_at` and `updated_at` are the RM timestamps as they stood at that moment.

Field	Type	Required	Default	Example	Description
`created_at`	string (date-time)	yes	n/a	`2026-01-10T08:00:00Z`	Tenant creation timestamp.
`deleted_at`	string (date-time)	yes	n/a	`2026-04-02T11:20:00Z`	When the tenant was deleted.
`description`	string	yes	n/a	`Decommissioned engineering tenant`	Free-form description the tenant held when it was deleted.
`display_name`	string, nullable	no	`null`	`Engineering (retired)`	Human-readable name the tenant held when it was deleted. Null when its configuration detail was already gone at that point, so the name was never captured.
`email`	string, nullable	no	`null`	`eng-owner@vdura.example.com`	Contact email the tenant held when it was deleted. Null when its configuration detail was already gone at that point.

Field	Type	Required	Default	Example	Description
`id`	string (uuid)	yes	n/a	`3f2b1c0d-5e4a-4b8c-9d7e-6a5b4c3d2e1f`	Identifier (UUID) of the deleted tenant.
`name`	string	yes	n/a	`oldeng`	Name the tenant held when it was deleted. Free for reuse by a new tenant.
`status`	`TenantStatus` (`pending` , `configuring` , `active` , `archived`)	no	`archived`	`archived`	Always `archived` , so a client merging this list with the live one can tell them apart.
`updated_at`	string (date-time)	yes	n/a	`2026-04-02T11:15:00Z`	Tenant last-modification timestamp, as it stood when the tenant was deleted.

Example


```
{
   "created_at" : "2026-01-10T08:00:00Z",
   "deleted_at" : "2026-04-02T11:20:00Z",
   "description" : "Decommissioned engineering tenant",
   "display_name" : "Engineering (retired)",
   "email" : "eng-owner@vdura.example.com",
   "id" : "3f2b1c0d-5e4a-4b8c-9d7e-6a5b4c3d2e1f",
   "name" : "oldeng",
   "status" : "archived",
   "updated_at" : "2026-04-02T11:15:00Z"
}
```

`Attachment`

One host's binding to one subnet, as reads and creates return it.

Field	Type	Required	Default	Example	Description
`hostname`	string	no	(empty string)		Host name as supplied at create. Informational.
`id`	string (uuid)	yes	n/a		Unique identifier of the attachment.
`joined_at`	string (date-time)	yes	n/a		First attach time (UTC).
`mtu`	integer	yes	n/a		The MTU the host sets on its tenant interface, which is the MTU the realm nodes run the subnet at: the subnet's declared MTU, or the realm's primary network MTU less the encapsulation overhead when the subnet declares 0. Never 0.
`overlay_ip`	string	yes	n/a		The host's address on the tenant network.
`overlay_prefix`	integer	yes	n/a		CIDR prefix length of the subnet, which the host applies to its own address.
`segment_id`	integer	yes	n/a		The subnet VLAN ID under one segment type and its VNI under the other. 0 means unassigned.
`segment_type`	`SubnetSegmentType` (`none` , `vlan` , `vxlan`)	yes	n/a		Kind of network segment the subnet is placed on, as the realm has it configured.

Field	Type	Required	Default	Example	Description
`subnet_id`	string (uuid)	yes	n/a		UUID of the subnet the host is attached to.
`tenant_id`	string (uuid)	yes	n/a		UUID of the tenant owning the subnet.
`tunnel`	`AttachmentTunnel` object, nullable	no	`null`		Tunnel recipe, present only when the segment is tunnelled and null otherwise.
`underlay_ip`	string	yes	n/a	`10.128.9.0`	The host's address on the realm underlay. Identifies the attachment inside its subnet, and on a tunnelled segment it is the endpoint the realm sends the host traffic to.

Example


```
{
   "hostname" : "client-0",
   "id" : "8f14e45f-ceea-467a-9575-5c4bcbdcd2f1",
   "joined_at" : "2026-08-28T12:00:00Z",
   "mtu" : 1450,
   "overlay_ip" : "192.168.50.100",
   "overlay_prefix" : 24,
   "segment_id" : 4257216,
   "segment_type" : "vxlan",
   "subnet_id" : "36a8995f-026b-47e1-88bd-7fb9e7239995",
   "tenant_id" : "00000000-0000-0000-0000-000000000001",
   "tunnel" : {
      "mac" : "92:b2:99:59:f3:fa",
      "udp_port" : 4789,
      "vteps" : [
         "10.128.1.0",
         "10.128.1.1",
         "10.128.1.2"
      ]
   },
   "underlay_ip" : "10.128.9.0"
}
```

`AttachmentCreateRequest`

Request body for attaching one host to a subnet.

Field	Type	Required	Default	Example	Description
`hostname`	string	no	(empty string)	`client-0`	Informational host name, carried for the operator and never interpreted.
`overlay_ip`	string, nullable	no	`null`	`192.168.50.100`	Requested address on the tenant network. Must sit inside the subnet, outside the realm node pool and be unallocated. Omit to let the backend allocate one. Applied on first attach only, so a repeated create never moves a host.
`underlay_ip`	string	yes	n/a	`10.128.9.0`	The host's address on the realm underlay. Identifies the attachment inside its subnet, and on a tunnelled segment it is the endpoint the realm sends the host traffic to.

Example

```
{
   "hostname" : "client-0",
   "overlay_ip" : "192.168.50.100",
   "underlay_ip" : "10.128.9.0"
}
```

`AttachmentListResponse`

A collection of attachments with a total count.

Field	Type	Required	Default	Example	Description
`attachments`	array of `Attachment` object	no	n/a		Array of attachment objects. Empty array if none are found.
`total`	integer	yes	n/a	`0`, `1`, `12`	Total count of attachments returned.

Example


```
{
   "attachments" : [
      {
         "hostname" : "client-0",
         "id" : "8f14e45f-ceea-467a-9575-5c4bcbdcd2f1",
         "joined_at" : "2026-08-28T12:00:00Z",
         "mtu" : 1450,
         "overlay_ip" : "192.168.50.100",
         "overlay_prefix" : 24,
         "segment_id" : 4257216,
         "segment_type" : "vxlan",
         "subnet_id" : "36a8995f-026b-47e1-88bd-7fb9e7239995",
         "tenant_id" : "00000000-0000-0000-0000-000000000001",
         "tunnel" : {
            "mac" : "92:b2:99:59:f3:fa",
            "udp_port" : 4789,
            "vteps" : [
               "10.128.1.0",
               "10.128.1.1",
               "10.128.1.2"
            ]
         },
         "underlay_ip" : "10.128.9.0"
      }
   ],
   "total" : 1
}
```

`AttachmentTunnel`

The tunnel recipe, present only on a tunnelled segment.

A VLAN fabric forwards, learns, and floods on its own, so a subnet on it produces no tunnel: no link-layer address is assigned, no encapsulation port applies and no flood list exists.

Field	Type	Required	Default	Example	Description
`mac`	string	yes	n/a	`92:b2:99:59:f3:fa`	Link-layer address assigned by the backend, derived from the subnet and the host so that every realm node computes the same value without the host publishing it. The host must set it on its interface, because the realm programs a static entry for it.
`udp_port`	integer	yes	n/a	`4789`	UDP port carrying the encapsulation.
`vteps`	array of string	no	n/a		The realm nodes' underlay addresses, which are the host's flood list. Realm nodes only: another attached host is deliberately absent.

`AuditDiagnostics`

Failure diagnostics for one audit record, parsed from the correlated pan-api ERROR line.

Attached to the single-record get (`GET /observability/audit/{request_id}`) response, never to the list. Every structured field is empty when its source is absent - parsed, never guessed. `log_lines` is returned to platform callers only; tenant callers always receive it empty.

Field	Type	Required	Default	Example	Description
`code`	string	no	(empty string)		Error code token parsed from the error line, e.g. `11411` .
`log_lines`	array of string	no	n/a		Correlated pan-api log lines, ascending time. Always empty for a non-platform caller.
`message`	string	no	(empty string)		Human-readable message parsed from the error line.
`subsystem`	string	no	(empty string)		Subsystem token parsed from the error line, e.g. `rmm` .
`summary`	string	no	(empty string)		Error classification, e.g. `RpcUnexpected subsystem=rmm code=11411` .
`symbol`	string	no	(empty string)		Error symbol token parsed from the error line, e.g. `PANFS_11411` .

`AuditRecord`

One audit record, projected from the API's own audit-trail log line.

The payload for the audit snapshot (`GET /observability/audit`).

Field	Type	Required	Default	Example	Description
`action`	string	no	(empty string)		Action performed on that resource, e.g. `create` .
`auth_token_id`	string	no	(empty string)		Id of the API token that authenticated the request; empty for a session.
`client_ip`	string	no	(empty string)		Client IP address of the request.
`duration_ms`	number	no	`0`		Request duration in milliseconds.
`id`	string	yes	n/a		The record's own identifier, which is its `request_id` .
`method`	string	yes	n/a		HTTP method of the audited request.
`principal_id`	string	no	(empty string)		Who acted, as `<tenant id>/<principal id>` . Under a tenant switch this is the person who switched, in their own tenant. Empty when the request was not authenticated.
`principal_name`	string	no	(empty string)		`<tenant name>/<name>` , or the bare name when the tenant name was not known.
`principal_type`	string	no	(empty string)		`user` | `service-account` .
`query`	string	no	(empty string)		Query string of the audited request.

Field	Type	Required	Default	Example	Description
`reason`	any, nullable	no	`null`		Reason for a denied or failed result, if recorded.
`request_body`	any, nullable	no	`null`		Request body, secret-redacted at write time.
`request_id`	string	yes	n/a		Audit request ID; the identity used for deduplication.
`resource`	string	no	(empty string)		Resource the request acted on, e.g. `volumes`.
`result`	string	yes	n/a		Outcome of the request: `success` | `denied` | `failed`.
`severity`	string	yes	n/a		Audit-significance severity: `INFO` | `WARNING`.
`status_code`	integer, nullable	no	`null`		HTTP response status code.
`target_id`	string	no	(empty string)		Id of the specific instance acted on, if any.
`tenant_id`	string	no	(empty string)		The tenant the credential acts in: the caller's own, or under a tenant switch the tenant switched into. It differs from the tenant half of `principal_id` only under a switch.
`time`	string	yes	n/a		When the request was audited (RFC 3339, from VictoriaLogs `_time`).
`url`	string	yes	n/a		URL path of the audited request.
`user_agent`	string	no	(empty string)		Client `User-Agent` header of the request.

`AuditRecordDetail`

The single-record get's response (`GET /observability/audit/{request_id}`).

Shared fields agree field-for-field with the list's `AuditRecord` , which stays unchanged.

Field	Type	Required	Default	Example	Description
`action`	string	no	(empty string)		Action performed on that resource, e.g. `create` .
`auth_token_id`	string	no	(empty string)		Id of the API token that authenticated the request; empty for a session.
`client_ip`	string	no	(empty string)		Client IP address of the request.
`diagnostics`	`AuditDiagnostics` object	no	n/a		Correlated pan-api log detail for this request.
`duration_ms`	number	no	`0`		Request duration in milliseconds.
`id`	string	yes	n/a		The record's own identifier, which is its `request_id` .
`method`	string	yes	n/a		HTTP method of the audited request.
`principal_id`	string	no	(empty string)		Who acted, as `<tenant id>/<principal id>` . Under a tenant switch this is the person who switched, in their own tenant. Empty when the request was not authenticated.
`principal_name`	string	no	(empty string)		`<tenant name>/<name>` , or the bare name when the tenant name was not known.
`principal_type`	string	no	(empty string)		`user` | `service-account` .

Field	Type	Required	Default	Example	Description
`query`	string	no	(empty string)		Query string of the audited request.
`reason`	any, nullable	no	`null`		Reason for a denied or failed result, if recorded.
`request_body`	any, nullable	no	`null`		Request body, secret-redacted at write time.
`request_id`	string	yes	n/a		Audit request ID; the identity used for deduplication.
`resource`	string	no	(empty string)		Resource the request acted on, e.g. `volumes`.
`result`	string	yes	n/a		Outcome of the request: `success` | `denied` | `failed`.
`severity`	string	yes	n/a		Audit-significance severity: `INFO` | `WARNING`.
`status_code`	integer, nullable	no	`null`		HTTP response status code.
`target_id`	string	no	(empty string)		Id of the specific instance acted on, if any.
`tenant_id`	string	no	(empty string)		The tenant the credential acts in: the caller's own, or under a tenant switch the tenant switched into. It differs from the tenant half of `principal_id` only under a switch.
`time`	string	yes	n/a		When the request was audited (RFC 3339, from VictoriaLogs `_time`).
`url`	string	yes	n/a		URL path of the audited request.

Field	Type	Required	Default	Example	Description
`user_agent`	string	no	(empty string)		Client `User-Agent` header of the request.

`Body_refresh_api_v1_auth_refresh_post`

Field	Type	Required	Default	Example	Description
`refresh_token`	string	yes	n/a		

`ChangePasswordRequest`

Self-service change-password payload. Carries no user/tenant ID - the caller comes from the token.

Field	Type	Required	Default	Example	Description
`current_password`	string (password)	yes	n/a		The caller's current password, verified server-side.
`new_password`	string (password)	yes	n/a		New password. Must contain at least 8 characters with an upper-case letter, a lower-case letter, a digit, and a special character, and must not exceed 72 bytes.

Example

```
{
   "current_password" : "Curr3nt!pw",
   "new_password" : "N3w!password"
}
```

`CollectedHostResponse`

One node a run collected from.

Field	Type	Required	Default	Example	Description
`address`	string	yes	n/a	`10.97.26.240`	The IPv4 address collect_logs used.
`data_serial`	string, nullable	no	`null`	`f2c172cea18e5e`	Node data serial in the RM.
`name`	string, nullable	no	`null`	`VCH-7,4`	Node name in the RM.
`result`	`HostResult` (`collected` , `failed` , `not_collected`)	yes	n/a		`collected` : the archive holds a non-empty node file from it. `failed` : collect_logs reported a failure, or the archive has no logs from it. `not_collected` : the run failed, so there is no archive.
`role`	string, nullable	no	`null`	`qnode` , `vosd` , `vhost`	Node role in the RM.

`DelegatedTokenResponse`

Assumed-tenant token issued by the tenant-switch endpoint. Deliberately carries no refresh token.

Field	Type	Required	Default	Example	Description
`access_token`	string	yes	n/a		Short-lived JWT scoped to the target tenant.
`expires_in`	integer	yes	n/a	`3600`	Granted token lifetime in seconds.
`tenant_id`	string (uuid)	yes	n/a		Target tenant the token acts in (`tid` claim).
`token_type`	literal `bearer`	no	`bearer`		

`DirectorEntry`

One realm Director's address on the subnet.

Field	Type	Required	Default	Example	Description
`dataserial`	string	yes	n/a	`SN-D01`	The Director's serial, which is how the realm's own tooling names the node.
`ip`	string	yes	n/a	`10.20.0.11`	The Director's address on this tenant subnet, which a client mounts to.
`repset`	boolean	yes	n/a		True when the node carries a replica of the realm database. These are the realm managers a client bootstraps against, so rmlist is built from them.

Field	Type	Required	Default	Example	Description
`status`	`NodeStatus` (`unknown` , `new` , `secure_erase` , `fsck` , `booting` , `shutdown` , `online` , `offline` , `mia_timer` , `dead` , `mia_notimer` , `lowbattery` , `clocksync` , `not_ready` , `network_err` , `not_reconciled` , `config_err`)	yes	n/a		Operational state of the node, as the realm manager reports it, carried for the operator to read. An address for a node that is down is inert and starts working the moment that node returns, so it stays in the list.

Example

```
{
   "dataserial" : "SN-D01",
   "ip" : "10.20.0.11",
   "repset" : true,
   "status" : "online"
}
```

`EfsaMode`

Extended File System Availability (EFSA) mode.

Mirrors CLI EFSA Mode column and volume list options.

Modes:

- `retry` : clients block and retry I/O until storage recovers (default)
- `file-unavailable` : I/O requests fail with errors when FT is exceeded
- `default` : apply system default EFSA setting

Allowed values: `retry` , `file-unavailable` , `default`

`FsUser`

A PanFS user as published.

Field	Type	Required	Default	Example	Description
`gid`	integer	yes	n/a	`65534`	POSIX group ID. Always concrete.
`id`	string (uuid)	yes	n/a	`b7c1f0e2-3a4d-4c5e-8f90-1a2b3c4d5e6f`	PanFS user identifier (UUID).
`name`	string	yes	n/a	`alice`	Unique handle within the tenant. A POSIX short name by convention.
`s3`	`FsUserS3View` object, nullable	no	`null`		S3 access, or null if the user has none.
`uid`	integer	yes	n/a	`65534`	POSIX user ID. Always concrete.

Example


```
{
   "gid" : 65534,
   "id" : "b7c1f0e2-3a4d-4c5e-8f90-1a2b3c4d5e6f",
   "name" : "alice",
   "s3" : {
      "access_key" : "alice",
      "profile" : "user"
   },
   "uid" : 65534
}
```

`FsUserCreateRequest`

Request payload for creating a PanFS user.

Field	Type	Required	Default	Example	Description
`access_key`	string, nullable	no	`null`		S3 access key. Defaults to name, which is why both carry the same charset rule.
`gid`	integer, nullable	no	`null`		POSIX group ID. Same defaulting and bounds as uid.
`name`	string	yes	n/a		Unique handle within the tenant.
`profile`	`S3Profile` (`disabled`, `user`, `privileged`)	no	`user`		S3 access profile.
`uid`	integer, nullable	no	`null`		POSIX user ID. Omit to take the tenant's configured default, or nobody (65534) when it has none or when that default is not a usable ID, 0 included. The resolved value is written onto the record, so a later change to the tenant default does not move this identity. 0 is not accepted: it would name root. Nor is anything above 4294967294, the last ID a 32-bit uid_t can hold.

Example

```
{
   "gid" : 1000,
   "name" : "alice",
   "uid" : 1000
}
```

`FsUserListResponse`

Envelope for a PanFS user listing.

Field	Type	Required	Default	Example	Description
`fs_users`	array of `FsUser` object	yes	n/a		
`total`	integer	yes	n/a		How many users the listing carries.

`FsUserPatchRequest`

Request payload for updating a PanFS user. The secret is not among the fields.

Field	Type	Required	Default	Example	Description
`access_key`	string, nullable	no	`null`		Updated S3 access key.
`gid`	integer, nullable	no	`null`		Updated POSIX group ID.
`profile`	`S3Profile` (`disabled` , `user` , `privileged`), nullable	no	`null`		Updated S3 access profile.
`uid`	integer, nullable	no	`null`		Updated POSIX user ID.

`FsUserResponse`

Envelope for a single PanFS user.

Field	Type	Required	Default	Example	Description
`fs_user`	`FsUser` object	yes	n/a		A PanFS user as published.

`FsUserS3View`

The S3 half of a PanFS user, as published. No secret.

Field	Type	Required	Default	Example	Description
`access_key`	string	yes	n/a		S3 access key. The handle the gateway resolves a request by.
`profile`	`S3Profile` (`disabled`, `user`, `privileged`)	yes	n/a		S3 access profile.
`secret_rotated_at`	string (date-time), nullable	no	`null`		When the S3 secret was last rotated, or null if it never has been.

`FsUserSecretResponse`

The one place an S3 secret reaches a caller.

Field	Type	Required	Default	Example	Description
`access_key`	string	yes	n/a		The access key the secret belongs to.
`secret_key`	string	yes	n/a		The S3 secret key.

Example

```
{
   "access_key" : "alice",
   "secret_key" : "..."
}
```

`GetS3ConfigResponse`

Response wrapper for S3 config retrieval.

Returned on GET /api/v1/tenants/{tenant_id}/s3.

Field	Type	Required	Default	Example	Description
`s3_config`	`S3ConfigSchema` object	yes	n/a		Tenant S3 gateway configuration.

Example

```
{
   "s3_config" : {
      "default_dir_mode" : "755",
      "default_group" : "gid:65534",
      "default_obj_mode" : "644",
      "default_owner" : "uid:65534",
      "default_prefix" : "/vol1/t1",
      "enabled" : true,
      "tls" : {}
   }
}
```

`GetTenantResponse`

Detailed view of a single tenant.

Returned on GET /api/v1/tenants/{tenant_id}.

Field	Type	Required	Default	Example	Description
`tenant`	`Tenant` object	yes	n/a		Tenant entity with all available details.

`GetTenantStorageSetResponse`

Detailed view of a single tenant storageset assignment.

Field	Type	Required	Default	Example	Description
`storageset`	`StorageSetAssignment` object	yes	n/a		Storageset assignment (ID + quota + name).

Example

```
{
   "storageset" : {
      "quota_bytes" : 2199023255552,
      "storageset_id" : 1,
      "storageset_name" : "Set 1"
   }
}
```

`GetUserResponse`

Detailed view of a single Tenant User.

Returned on GET /tenants/{tenant_id}/users/{user_id}.

Field	Type	Required	Default	Example	Description
`user`	`User` object	yes	n/a		Tenant User entity with all available details.

Example


```
{
   "user" : {
      "auth_settings" : {
         "auth_mode" : "local"
      },
      "created_at" : "2026-01-01T00:00:00Z",
      "display_name" : "Alice Anderson",
      "email" : "alice@fintech.example.com",
      "id" : "a1b2c3d4-e5f6-7890-abcd-ef1234567890",
      "updated_at" : "2026-01-01T00:00:00Z",
      "username" : "alice"
   }
}
```


`HTTPValidationError`

Field	Type	Required	Default	Example	Description
`detail`	array of `ValidationError` object	no	n/a		

`Hardware`

A single hardware model from the catalog.

Declared attributes are typed for documentation; any further catalog fields are preserved as-is. Only `id` is guaranteed to be present.

Field	Type	Required	Default	Example	Description
`allow_lacp_link_agg`	integer, nullable	no	`null`		Whether LACP link aggregation is allowed.
`allow_static_link_agg`	integer, nullable	no	`null`		Whether static link aggregation is allowed.
`allow_upgrade`	integer, nullable	no	`null`		Whether upgrade is allowed (0 or 1).
`arch`	integer, nullable	no	`null`	`64`	CPU architecture width.
`capacity_gb`	integer, nullable	no	`null`		Storage capacity in GB.
`codename`	string, nullable	no	`null`		Internal codename(s).
`compatibility_type`	string, nullable	no	`null`	`bravo`	Compatibility line.
`component_type`	string, nullable	no	`null`	`node`, `enclosure`	Component type.
`cpu`	string, nullable	no	`null`		CPU description.
`data_sector_size`	integer, nullable	no	`null`		Drive data sector size in bytes.
`datastores`	array of any, nullable	no	`null`		Datastore definitions.
`deny_downgrade_to`	string, nullable	no	`null`		Version this model must not be downgraded to.

Field	Type	Required	Default	Example	Description
`drives`	object, nullable	no	`null`		Drive definitions keyed by drive number.
`generation`	string, nullable	no	`null`	`100`	Hardware generation.
`has_bladeset_affinity`	boolean, nullable	no	`null`		Whether the model has bladeset affinity.
`hdinfo`	string, nullable	no	`null`	`(1) 512GB SSD`	Human-readable drive summary.
`id`	string	yes	n/a	`301002001`	Catalog identifier of the hardware model.
`intent_log`	object, nullable	no	`null`		Intent log configuration.
`introduced`	string, nullable	no	`null`	`7.0.0`	First version to support this model.
`loops`	object, nullable	no	`null`		Loop device definitions.
`manufacturer`	string, nullable	no	`null`		Hardware manufacturer.
`memory`	string, nullable	no	`null`	`96GB`	Amount of memory.
`metadatastores`	array of any, nullable	no	`null`		Metadata store definitions.
`name`	string, nullable	no	`null`	`ASD-1-100-96-80`	Model name (SKU).
`network`	string, nullable	no	`null`	`2x200GbE`	Network interface summary.
`node_osd_filesystem_version`	integer, nullable	no	`null`		Node OSD filesystem version.

Field	Type	Required	Default	Example	Description
`node_role_ids`	array of integer, nullable	no	`null`		Node role identifiers.
`nvdimm_namespaces`	object, nullable	no	`null`		NVDIMM namespace definitions.
`osd_filesystem_version`	integer, nullable	no	`null`		OSD filesystem version.
`platform`	string, nullable	no	`null`	`freebsd_142_amd64`	OS/platform identifier.
`powerdraw`	integer, nullable	no	`null`	`400`	Power draw in watts.
`product_name`	string, nullable	no	`null`		Product name (product type plus generation).
`product_type`	string, nullable	no	`null`		Product type (name without generation).
`role`	string, nullable	no	`null`	`director`	Software role the device serves.
`storage_controller`	string, nullable	no	`null`		Storage controller description.
`tlsize`	integer, nullable	no	`null`		Transaction log size in GB.
`vpod_skus`	object, nullable	no	`null`		Associated VPOD SKUs and counts.
`watts`	integer, nullable	no	`null`		Power supply rating in watts.

Example


```
{
   "arch" : 64,
   "compatibility_type" : "bravo",
   "component_type" : "node",
   "cpu" : "Intel Xeon E5-1630 v4 3.7GHz",
   "generation" : "100",
   "hdinfo" : "(1) 512GB SSD",
   "id" : "301002001",
   "introduced" : "7.0.0",
   "memory" : "96GB",
   "name" : "ASD-1-100-96-80",
   "network" : "2x200GbE",
   "platform" : "freebsd_142_amd64",
   "powerdraw" : 400,
   "product_name" : "Bravo Director Node 100",
   "product_type" : "Bravo Director Node",
   "role" : "director"
}
```


`HardwareBlinkDriveLedActionRequest`

Request to turn on a drive's identify ('locate') LED so the drive can be found physically.

Field	Type	Required	Default	Example	Description
`action`	literal `blink_drive_led`	yes	n/a		The action to perform on the component.
`params`	`HardwareDriveLedActionParams` object	yes	n/a		Identifies the drive on the hosting node.

`HardwareBlinkNodeLedActionRequest`

Request to turn on a node's identify ('locate') LED so the component can be found physically.

Field	Type	Required	Default	Example	Description
`action`	literal `blink_node_led`	yes	n/a		The action to perform on the component.
`params`	null (not used)	no	n/a		

`HardwareDriveLedActionParams`

Identifies the drive whose LED to drive, by its number on the hosting node.

Field	Type	Required	Default	Example	Description
`drive_number`	integer	yes	n/a	`1` , `3`	The drive's number on the hosting node (the panfs_drive `drive_number`). Numbering starts at 1, so a node with four drives accepts 1 to 4.

`HardwareGetResponse`

Detailed representation of a single hardware model.

Field	Type	Required	Default	Example	Description
`hardware`	`Hardware` object	yes	n/a		Hardware model with all available details.

Example


```
{
   "hardware" : {
      "arch" : 64,
      "compatibility_type" : "bravo",
      "component_type" : "node",
      "cpu" : "Intel Xeon E5-1630 v4 3.7GHz",
      "generation" : "100",
      "hdinfo" : "(1) 512GB SSD",
      "id" : "301002001",
      "introduced" : "7.0.0",
      "memory" : "96GB",
      "name" : "ASD-1-100-96-80",
      "network" : "2x200GbE",
      "platform" : "freebsd_142_amd64",
      "powerdraw" : 400,
      "product_name" : "Bravo Director Node 100",
      "product_type" : "Bravo Director Node",
      "role" : "director"
   }
}
```

`HardwareListResponse`

A collection of hardware models with the total number available.

Field	Type	Required	Default	Example	Description
`hardware`	array of `Hardware` object	no	n/a		Array of hardware models present in the current realm. Empty array if none are found.
`total`	integer	yes	n/a	`0`, `1`, `3`	Total count of hardware models present in the current realm.

Example


```
{
   "hardware" : [
      {
         "arch" : 64,
         "compatibility_type" : "bravo",
         "component_type" : "node",
         "cpu" : "Intel Xeon E5-1630 v4 3.7GHz",
         "generation" : "100",
         "hdinfo" : "(1) 512GB SSD",
         "id" : "301002001",
         "introduced" : "7.0.0",
         "memory" : "96GB",
         "name" : "ASD-1-100-96-80",
         "network" : "2x200GbE",
         "platform" : "freebsd_142_amd64",
         "powerdraw" : 400,
         "product_name" : "Bravo Director Node 100",
         "product_type" : "Bravo Director Node",
         "role" : "director"
      }
   ],
   "total" : 1
}
```

`HardwareStopDriveLedActionRequest`

Request to turn off a drive's identify ('locate') LED.

Field	Type	Required	Default	Example	Description
`action`	literal `stop_drive_led`	yes	n/a		The action to perform on the component.
`params`	`HardwareDriveLedActionParams` object	yes	n/a		Identifies the drive on the hosting node.

`HardwareStopNodeLedActionRequest`

Request to turn off a node's identify ('locate') LED.

Field	Type	Required	Default	Example	Description
`action`	literal `stop_node_led`	yes	n/a		The action to perform on the component.
`params`	null (not used)	no	n/a		

`HostResult`

Whether `collect_logs` got a node's logs into the archive.

Allowed values: `collected`, `failed`, `not_collected`

`ListArchivedTenantResponse`

Collection response for the archive listing.

Returned on GET /api/v1/tenants/archived (Platform Admin only), newest deletion first.

Field	Type	Required	Default	Example	Description
`tenants`	array of `ArchivedTenantSchema` object	no	n/a		Array of archived tenants, newest deletion first. Empty array if none were ever deleted.
`total`	integer	yes	n/a	`0`, `5`, `50`	Total count of archived tenants in this response.

Example


```
{
   "tenants" : [
      {
         "created_at" : "2026-01-10T08:00:00Z",
         "deleted_at" : "2026-04-02T11:20:00Z",
         "description" : "Decommissioned engineering tenant",
         "display_name" : "Engineering (retired)",
         "email" : "eng-owner@vdura.example.com",
         "id" : "3f2b1c0d-5e4a-4b8c-9d7e-6a5b4c3d2e1f",
         "name" : "oldeng",
         "status" : "archived",
         "updated_at" : "2026-04-02T11:15:00Z"
      }
   ],
   "total" : 1
}
```

`ListS3BucketsResponse`

Response wrapper for listing S3 buckets.

Field	Type	Required	Default	Example	Description
`s3_buckets`	array of `S3BucketSchema` object	yes	n/a		List of S3 buckets.

`ListTenantResponse`

Collection response for tenant list operations.

Returned on GET /api/v1/tenants, to any platform role holding `tenants:list`. Returns summary info for each tenant, not full details.

Field	Type	Required	Default	Example	Description
`tenants`	array of `TenantSummarySchema` object	no	n/a		Array of tenant summary objects. Empty array if no tenants found
`total`	integer	yes	n/a	`0`, `5`, `50`	Total count of tenants in this response.

Example


```
{
   "tenants" : [
      {
         "created_at" : "2026-03-02T12:00:00Z",
         "description" : "FinTech tenant",
         "id" : "e97b94d6-9cc1-4325-a267-1064ae538ccd",
         "name" : "fintech",
         "status" : "active",
         "updated_at" : "2026-03-02T12:00:00Z"
      }
   ],
   "total" : 1
}
```

`ListTenantStorageSetResponse`

Collection response listing a tenant's storageset assignments.

Field	Type	Required	Default	Example	Description
`storagesets`	array of `StorageSetAssignment` object	no	n/a		Array of storageset assignments (ID + quota + name). Empty array if none are assigned.
`total`	integer	yes	n/a	`0`, `2`, `5`	Total count of storageset assignments in this response.

Example


```
{
   "storagesets" : [
      {
         "quota_bytes" : 2199023255552,
         "storageset_id" : 1,
         "storageset_name" : "Set 1"
      },
      {
         "quota_bytes" : 5497558138880,
         "storageset_id" : 3,
         "storageset_name" : "Set 3"
      }
   ],
   "total" : 2
}
```

`LocalAuthSettings`

Local bcrypt-based authentication.

Field	Type	Required	Default	Example	Description
`auth_mode`	literal `local`	no	`local`		

`LogAgeRequest`

Skip rotated logs and corefiles older than this.

Field	Type	Required	Default	Example	Description
`count`	integer	yes	n/a		How many units old a file may be. 0 means no limit.
`unit`	`AgeUnit` (`minutes` , `hours` , `days` , `weeks` , `months`)	yes	n/a		The unit of `count` .

`LogAgeResponse`

The age limit a run applied.

Field	Type	Required	Default	Example	Description
`count`	integer	yes	n/a	`1`	How many units old a file may be.
`unit`	`AgeUnit` (`minutes` , `hours` , `days` , `weeks` , `months`)	yes	n/a	`hours`	The unit of `count` .

`LogCollectionCreateRequest`

What to collect, and from which nodes. An empty body collects everything from every node.

Field	Type	Required	Default	Example	Description
`age`	`LogAgeRequest` object, nullable	no	`null`		Skip rotated `/var/log` files and corefiles older than this. Default: no age limit.
`hosts`	array of string	no	n/a		Nodes to collect from, by IPv4 address, serial, or name (e.g. `VCH-1,1`). Adds to `scope`.
`include_cores`	boolean	no	`true`		Collect corefiles and executables. Stack files always are.
`include_databases`	boolean	no	`true`		Collect the system databases (sanitised copies).
`include_sed`	boolean	no	`true`		Collect the self-encrypting drive (SED) event log, `/pan/var/lib/pan_sed/event.log` (ASU nodes only).
`node_timeout_seconds`	integer, nullable	no	`null`		Seconds each node may take to collect, and to copy back (collect_logs `timeout`). Default 900; 0 means no limit. The server also limits the whole run.
`run_diagnostics`	boolean	no	`true`		Run the diagnostic tools on each node before collecting.

Field	Type	Required	Default	Example	Description
`scope`	`LogCollectionScope` (`all` , `repset` , `directors` , `storage`), nullable	no	`null`		A whole class of nodes: `all` , `repset` (the Directors in the replication set), `directors` or `storage` . Defaults to `all` when neither `hosts` nor `storagesets` is given. `all` cannot be combined with them, because it already covers every node.
`storagesets`	array of string	no	n/a		StorageSets whose nodes to collect from, by name (e.g. `Set 1`). Adds to `scope` .
`wait_for_savecore`	boolean	no	`false`		Wait for a running savecore to finish first.

`LogCollectionListResponse`

Every run the server knows of, newest first.

Field	Type	Required	Default	Example	Description
`collections`	array of `LogCollectionResponse` object	yes	n/a		Running and finished collections, newest first.

`LogCollectionOptionsResponse`

The options a run was started with, as the request gave them (an omitted scope reads `all`).

Field	Type	Required	Default	Example	Description
`age`	`LogAgeResponse` object, nullable	yes	n/a		The age limit on rotated logs and corefiles, if any.
`hosts`	array of string	yes	n/a	`["10.97.26.200"]`	Nodes named in the request.
`include_cores`	boolean	yes	n/a		Corefiles and executables were collected.
`include_databases`	boolean	yes	n/a		System databases were collected.
`include_sed`	boolean	yes	n/a		The self-encrypting drive (SED) event log was collected.
`node_timeout_seconds`	integer, nullable	yes	n/a		Per-node timeout, if set; `null` means 900 s.
`run_diagnostics`	boolean	yes	n/a		Diagnostic tools ran on each node first.
`scope`	`LogCollectionScope` (`all`, `repset`, `directors`, `storage`), nullable	yes	n/a	`all`	`null` when the request named only `hosts` or `storagesets`.
`storagesets`	array of string	yes	n/a	`["Set1"]`	StorageSets named in the request.
`wait_for_savecore`	boolean	yes	n/a		The run waited for a running savecore.

`LogCollectionResponse`

One log collection run.

Field	Type	Required	Default	Example	Description
`archive_sha256`	string, nullable	no	`null`	`0f26932a2db4c86847bbf2b28e8f38ef8e9fd1b802e9e228471c13cd82db208c`	SHA-256 of the archive, in hex. The download sends it as its `ETag`.
`archive_size_bytes`	integer, nullable	no	`null`		Size of the archive, once it is ready.
`archive_url`	string, nullable	yes	n/a		Where to download the archive, once it is ready.
`duration_seconds`	number, nullable	yes	n/a		Seconds from start to end, once the run has ended.
`error`	string, nullable	no	`null`		Why the run failed.
`failed_hosts`	array of string	no	n/a		Addresses of the `hosts` that failed. A run with some failed hosts still succeeds.
`finished_at`	string (date-time), nullable	no	`null`		When the run ended (UTC); the archive name carries this time.
`host_count`	integer, nullable	no	`null`		Nodes the run collects from, once it has listed them.
`hosts`	array of `CollectedHostResponse` object	no	n/a		Every node the run collected from, once it has ended.

Field	Type	Required	Default	Example	Description
`id`	string (uuid)	yes	n/a		Identifier of the run.
`options`	`LogCollectionOptionsResponse` object	yes	n/a		What the run collects, as requested.
`realm`	`RealmRefResponse` object, nullable	no	`null`		The realm, when the RM could be read.
`requested_by`	string, nullable	no	`null`	`platform/admin`	Who started the run: `<tenant>/<principal>`.
`started_at`	string (date-time)	yes	n/a		When the run started (UTC).
`status`	`LogCollectionStatus` (`running`, `succeeded`, `failed`)	yes	n/a		`running` until the run ends, then `succeeded` (an archive is ready) or `failed`.

`LogCollectionScope`

A class of nodes to collect from, named as `collect_logs` names it.

Allowed values: `all`, `repset`, `directors`, `storage`

`LogCollectionStatus`

The state of a run. A run is deleted, not canceled, so there is no canceled state.

Allowed values: `running`, `succeeded`, `failed`

`LoginForm`

Credentials for `POST /auth/login`, where the username carries the tenant.

Any other form field an OAuth2 client sends is accepted and ignored.

Field	Type	Required	Default	Example	Description
`password`	string (password)	yes	n/a		The user password.
`username`	string	yes	n/a		Username, tenant-qualified as `<tenant-name>/<user>` (for example `fintech/alice`). An unprefixed name belongs to the platform tenant.

Example

```
{
   "password" : "Secret1!",
   "username" : "admin"
}
```

`MeResponse`

Current user identity, and what it may do in the effective tenant.

`roles` and `permissions` are the same policy read from either end: the roles are the bindings the caller holds, `permissions` the effective set they resolve to. A client that only gates on capability reads `permissions`; one that also names the caller's role reads `roles`. Both are scoped to `tenant_id`, so a tenant switch reports the target tenant.

Field	Type	Required	Default	Example	Description
`actor_tenant_id`	string (uuid), nullable	no	`null`	`00000000-0000-0000-0000-000000000000`	Home tenant of the acting user; set only while a tenant switch is active.
`actor_user_id`	string (uuid), nullable	no	`null`	`00000000-0000-0000-0000-000000000001`	The user that asked for the switch; set only while a tenant switch is active.
`is_delegated`	boolean	no	`false`		True while the caller acts in another tenant through a tenant switch. `user_id` is then the platform-support principal, and the delegation fields name the real caller.
`is_platform_admin`	boolean	yes	n/a		True if the authenticated user is the built-in Platform Admin account acting in its own tenant. Informational only - it grants nothing that the platform policy does not already grant. Always false while a tenant switch is active.

Field	Type	Required	Default	Example	Description
`permissions`	array of string	no	n/a	`["volumes:list","volumes:create","users:view"]`	Every permission the caller's roles grant, as `resource:action`, deduplicated and in policy order. This is the effective set the server enforces against, so a client can gate a control on membership alone without knowing which role supplied it.
`roles`	array of string	no	n/a	`["admin"]`	Roles the caller holds in the effective tenant. Empty when the caller is bound to no role there - a role defined by another tenant grants nothing here. The full catalog, with the permissions of every role the tenant defines, is at `GET /tenants/roles`.
`tenant_id`	string (uuid)	yes	n/a	`123e4567-e89b-12d3-a456-426614174000`	Effective tenant UUID. Resolved from the `tid` claim in the JWT.
`user_id`	string (uuid)	yes	n/a	`123e4567-e89b-12d3-a456-426614174000`	The authenticated user identifier (JWT `sub` claim).

Example


```
{
   "is_delegated" : false,
   "is_platform_admin" : false,
   "permissions" : [
      "tenants:view",
      "volumes:list",
      "volumes:view",
      "users:list",
      "users:view",
      "users:update",
      "users:delete",
      "observability:list",
      "observability:view",
      "roles:view",
      "roles:update"
   ],
   "roles" : [
      "admin"
   ],
   "tenant_id" : "123e4567-e89b-12d3-a456-426614174000",
   "user_id" : "123e4567-e89b-12d3-a456-426614174000"
}
```

`MountInfoResponse`

The mount targets a client on one tenant subnet can use.

Field	Type	Required	Default	Example	Description
`directors`	array of `DirectorEntry` object	no	n/a		The realm Directors holding an address on this subnet, nodes carrying a realm-database replica first and then ordered by address. Every such Director appears here, in either address family. An empty array means the subnet exists and carries no address records yet.
`rmlist`	string	yes	n/a	`10.20.0.11;10.20.0.12`	The addresses for mount_panfs, joined with ";" in the order above and wrapped by the caller as rmlist=(). It narrows the Directors array to replica nodes, which is what the option means and what keeps it under the five addresses mount_panfs accepts, and to IPv4, which is all that grammar can express. Empty when no replica node holds an IPv4 address here, even when Directors is not empty.
`subnet_id`	string (uuid)	yes	n/a		UUID of the subnet the addresses belong to.
`tenant_id`	string (uuid)	yes	n/a		UUID of the tenant owning the subnet.

Example


```
{
   "directors" : [
      {
         "dataserial" : "SN-D01",
         "ip" : "10.20.0.11",
         "repset" : true,
         "status" : "online"
      },
      {
         "dataserial" : "SN-D02",
         "ip" : "10.20.0.12",
         "repset" : true,
         "status" : "online"
      },
      {
         "dataserial" : "SN-D03",
         "ip" : "10.20.0.13",
         "repset" : false,
         "status" : "online"
      }
   ],
   "rmlist" : "10.20.0.11;10.20.0.12",
   "subnet_id" : "36a8995f-026b-47e1-88bd-7fb9e7239995",
   "tenant_id" : "e97b94d6-9cc1-4325-a267-1064ae538ccd"
}
```

`Node`

Node representation.

Field	Type	Required	Default	Example	Description
`capacity_bytes`	integer	no	`0`	`5565002547 2` , `0`	Storage capacity of the node in bytes.
`dataserial`	string	yes	n/a	`e9ff4075d18233`	The data serial identifier used for data replication and recovery processes
`flags`	array of `NodeFlag` (`impaired` , `jettisoned` , `upgrade` , `reinit` , `offline` , `reinit_pending` , `recon_pending` , `need_to_offline` , `draining` , `alias_uncommitted` , `disallow_aliases` , `repset` , `drained` , `activated` , `10ge_secondary_only` , `admin_nic_disabled`)	yes	n/a	`repset` , `activated`	Operational flags indicating special conditions or states of the node

Field	Type	Required	Default	Example	Description
`hw_serial`	string	yes	n/a	`3e7e0000310fff001`	The hardware serial identifier used for inventory tracking and support
`ip_address`	string	yes	n/a	`10.97.18.200`	The management interface IP address used to reach the node for control and monitoring
`name`	string	yes	n/a	`ASD-1,1`, `VCH-2,2`	Human-readable node name formed as `<node group>,<slot>` (e.g. `ASD-1,1`, `VCH-2,2`).
`role`	`NodeRole` (`none`, `any`, `any_director`, `any_storage`, `qnode`, `snode`, `ibr_node`, `vhost`, `vosd`)	yes	n/a	`any_director`	The functional role of the node indicating how the node is used in the system
`shelf_id`	string	yes	n/a	`3e7e000140200100 1`	Unique identifier of a storage shelf

Field	Type	Required	Default	Example	Description
`status`	`NodeStatus` (`unknown` , `new` , `secure_erase` , `fsck` , `booting` , `shutdown` , `online` , `offline` , `mia_timer` , `dead` , `mia_notimer` , `lowbattery` , `clocksync` , `not_ready` , `network_err` , `not_reconciled` , `config_err`)	yes	n/a	`online`	The status of the node
`storageset_id`	integer	no	`0`	`1` , `2` , `5`	StorageSet ID where this node resides
`vhost_id`	string	yes	n/a	`bf806fa0c17feb`	Identifier of the virtual host (vhost) the node belongs to; empty if not associated with a vhost

Example


```
{
   "capacity_bytes" : 55650025472,
   "dataserial" : "e9ff4075d18233",
   "flags" : [
      "repset",
      "activated"
   ],
   "hw_serial" : "3e7e0000310fff001",
   "ip_address" : "10.97.18.200",
   "name" : "ASD-1,1",
   "role" : "any_director",
   "shelf_id" : "3e7e0001402001001",
   "status" : "online",
   "storageset_id" : 1,
   "vhost_id" : "bf806fa0c17feb"
}
```


`NodeDrainState`

High-level state of an OSD drain operation.

Allowed values: `draining` , `not_draining` , `suspended` , `unknown`

`NodeDrainStatusResponse`

Progress of an OSD drain, including derived completion percentage and ETA.

Field	Type	Required	Default	Example	Description
`amount_error`	integer	yes	n/a		Bytes that could not be migrated.
`amount_moved`	integer	yes	n/a		Bytes already migrated off the node.
`amount_remaining`	integer	yes	n/a		Bytes still to be migrated off the node.
`eta_seconds`	integer, nullable	yes	n/a	`51`, `0`, `null`	Estimated seconds to completion at the current rate, or `null` when the rate is zero.
`objects_error`	integer	yes	n/a		Objects that could not be migrated.
`objects_moved`	integer	yes	n/a		Objects already migrated off the node.
`objects_remaining`	integer	yes	n/a		Objects still to be migrated off the node.
`percent_complete`	number, nullable	yes	n/a	`50`, `100`, `null`	Percentage of bytes migrated, or `null` when there is nothing to drain.
`pm_id`	integer	yes	n/a		Identifier of the performance manager (PM) reporting this drain.
`rate_bytes_per_sec`	integer	yes	n/a		Current data-movement rate for the drain, in bytes per second.

Field	Type	Required	Default	Example	Description
`state`	`NodeDrainState` (`draining` , `not_draining` , `suspended` , `unknown`)	yes	n/a		High-level state of the drain operation.

Example


```
{
   "amount_error" : 0,
   "amount_moved" : 5368709120,
   "amount_remaining" : 5368709120,
   "eta_seconds" : 51,
   "objects_error" : 0,
   "objects_moved" : 1024,
   "objects_remaining" : 1024,
   "percent_complete" : 50,
   "pm_id" : 4,
   "rate_bytes_per_sec" : 104857600,
   "state" : "draining"
}
```


`NodeFlag`

Node flags.

Allowed values: `impaired` , `jettisoned` , `upgrade` , `reinit` , `offline` , `reinit_pending` , `recon_pending` , `need_to_offline` , `draining` , `alias_uncommitted` , `disallow_aliases` , `repset` , `drained` , `activated` , `10ge_secondary_only` , `admin_nic_disabled`

`NodeGetResponse`

Detailed representation of a single node.

Field	Type	Required	Default	Example	Description
`node`	`Node` object	yes	n/a		Node entity with all available details.

Example


```
{
   "node" : {
      "capacity_bytes" : 55650025472,
      "dataserial" : "e9ff4075d18233",
      "flags" : [
         "repset",
         "activated"
      ],
      "hw_serial" : "3e7e0000310fff001",
      "ip_address" : "10.97.18.200",
      "name" : "ASD-1,1",
      "role" : "any_director",
      "shelf_id" : "3e7e0001402001001",
      "status" : "online",
      "storageset_id" : 1,
      "vhost_id" : "bf806fa0c17feb"
   }
}
```

`NodeHaltActionRequest`

Request to perform a halt (power-down without auto-restart) action on a Node.

Field	Type	Required	Default	Example	Description
`action`	literal `halt`	yes	n/a		The action to perform on the node.
`params`	null (not used)	no	n/a		

`NodeListResponse`

Represents a collection of node objects along with the total number of available objects.

Field	Type	Required	Default	Example	Description
`nodes`	array of `Node` object	no	n/a		Array of node objects matching the query. Empty array if no nodes found
`total`	integer	yes	n/a	`0`, `2`, `150`	Total count of nodes available (may exceed returned array size if paginated).

Example


```
{
   "nodes" : [
      {
         "capacity_bytes" : 55650025472,
         "dataserial" : "e9ff4075d18233",
         "flags" : [
            "repset",
            "activated"
         ],
         "hw_serial" : "3e7e0000310fff001",
         "ip_address" : "10.97.18.200",
         "name" : "ASD-1,1",
         "role" : "any_director",
         "shelf_id" : "3e7e0001402001001",
         "status" : "online",
         "storageset_id" : 1,
         "vhost_id" : "bf806fa0c17feb"
      }
   ],
   "total" : 1
}
```

`NodeRebootActionRequest`

Request to perform a reboot action on a Node.

Field	Type	Required	Default	Example	Description
`action`	literal `reboot`	yes	n/a		The action to perform on the node.
`params`	null (not used)	no	n/a		

`NodeRole`

Node roles.

Allowed values: `none` , `any` , `any_director` , `any_storage` , `qnode` , `snode` , `ibr_node` , `vhost` , `vosd`

`NodeStartDrainRequest`

Request to start draining a node (migrate its data onto other nodes in the StorageSet).

Field	Type	Required	Default	Example	Description
`action`	literal `start_drain`	yes	n/a		The drain action to perform.
`params`	null (not used)	no	n/a		

`NodeStatus`

Node operational statuses.

`not_reconciled` : The node's network configuration or network ports have not yet been reported and recorded. This is normally temporary while a node joins; if it persists, check the node's network setup.

Allowed values: `unknown` , `new` , `secure_erase` , `fsck` , `booting` , `shutdown` , `online` , `offline` , `mia_timer` , `dead` , `mia_notimer` , `lowbattery` , `clocksync` , `not_ready` , `network_err` , `not_reconciled` , `config_err`

`NodeStopDrainRequest`

Request to stop an in-progress node drain.

Field	Type	Required	Default	Example	Description
`action`	literal `stop_drain`	yes	n/a		The drain action to perform.
`params`	null (not used)	no	n/a		

`ObservabilityLabels`

Labels for observability targets.

Field	Type	Required	Default	Example	Description
`__meta_node_state`	string	yes	n/a	`online`, `offline`	The PanFS node state (mgmt node status) of the nodes in this group. It describes the node, not whether the scrape endpoint answers.
`__meta_node_type`	string	yes	n/a	`any_director`, `vhost`, `vosd`	The type/role of the node

Example

```
{
   "__meta_node_state" : "online",
   "__meta_node_type" : "any_director"
}
```

`ObservabilityTarget`

Target for observability metrics collection.

Field	Type	Required	Default	Example	Description
`labels`	`ObservabilityLabels` object	yes	n/a		Labels associated with these targets
`targets`	array of string	yes	n/a	`["10.97.15.200:9273"]`	Scrape endpoints as host:port. The port is the observability agent (telegraf) Prometheus exporter on each node, which serves /metrics; 9273 unless the deployment overrides it.

Example

```
{
   "labels" : {
      "__meta_node_state" : "online",
      "__meta_node_type" : "any_director"
   },
   "targets" : [
      "10.97.15.200:9273"
   ]
}
```

`PamAuthSettings`

PAM-based authentication.

The `service` field maps to `/etc/pam.d/<service>`.

Field	Type	Required	Default	Example	Description
`auth_mode`	literal `pam`	no	`pam`		
`service`	string	no	`login`		

`PanfsEvent`

One PanFS management event, projected from a `job="EVENTS"` log record.

The payload for both the events snapshot (`GET /observability/events`) and the `append` SSE deltas. Every field describes the event's **subject** node; the record's stream labels name the Director that shipped the log line and are not carried here.

Field	Type	Required	Default	Example	Description
`blade_serial`	string	no	(empty string)		Blade serial of the subject node.
`category`	string	no	(empty string)		Event category, e.g. `hardware` , `volume` .
`component_type`	string	no	(empty string)		Subject component type, e.g. `Director` , `VPOD Container` .
`event_code`	string	no	(empty string)		PanFS event code, shown as `#CODE` in the UI.
`event_name`	string	no	(empty string)		PanFS event constant, e.g. `PAN_MGMT_EV_BRICK_ONLINE` .
`hw_serial`	string	no	(empty string)		Subject node's hardware serial.
`id`	string	yes	n/a		PanFS `event_id` - a hash of the event, unique per occurrence.
`ip_address`	string	no	(empty string)		Subject node's management IP address.
`message`	string	yes	n/a		Human-readable event message.
`node_dataserial`	string	no	(empty string)		Subject node's data serial, for linking to its detail page.

Field	Type	Required	Default	Example	Description
`severity`	one of: `error`, `warning`, `info`	yes	n/a		Normalized severity level.
`shelf_id`	string	no	(empty string)		Serial of the shelf holding the subject node.
`slot_id`	string	no	(empty string)		Slot the subject node occupies in that shelf.
`tenant_id`	string	no	(empty string)		Tenant the event belongs to; empty for a realm-wide event.
`time`	string	yes	n/a		When the event occurred (ISO 8601).
`volume_service_id`	string	no	(empty string)		Volume-service handle for volume events, e.g. `0x0400000000000005`.

`Principal`

A principal as a token record names it: IDs and names, recorded at the time and never re-read.

Field	Type	Required	Default	Example	Description
`id`	string (uuid)	yes	n/a		Principal ID.
`kind`	`SubjectKind` (`user`, `service-account`)	yes	n/a		`user` | `service-account`.
`name`	string	yes	n/a		Username or service account name.
`tenant_id`	string (uuid)	yes	n/a		The principal's own tenant.
`tenant_name`	string	yes	n/a		The principal's tenant name.

`QuotasPolicy`

User and Group Quotas Policy status.

Determines whether user and group quota enforcement is enabled on the volume.

States:

- `DISABLE` : quotas enforcement is disabled
- `ENABLE` : quotas enforcement is enabled and active
- `REPORT` : quota reporting mode without enforcement
- `INHERIT` : quota policy inherited from system defaults
- `INVALID` : invalid quota state, indicates an error condition

Allowed values: `disable` , `enable` , `report` , `inherit` , `invalid`

`RaidType`

RAID types supported by volume layout.

Allowed values: `raid6+` , `raid5+` , `raid10+` , `raid5` , `raid10` , `default`

`Realm`

Realm for the API response.

Field	Type	Required	Default	Example	Description
`boot_epoch_seconds`	number	yes	n/a		Realm boot epoch timestamp in seconds
`clientmode_current`	`RealmClientMode` (`invalid` , `balanced` , `favor_df`)	yes	n/a		Current client mode
`clientmode_next`	`RealmClientMode` (`invalid` , `balanced` , `favor_df`)	yes	n/a		Next client mode (after reboot)
`data_generation`	integer	yes	n/a		Data generation counter
`heartbeat_failure_grace_seconds`	number	yes	n/a		Heartbeat failure grace period in seconds
`heartbeat_period_seconds`	number	yes	n/a		Heartbeat period in seconds
`id`	string	yes	n/a		Realm unique identifier as a hex string
`locked`	boolean	yes	n/a		Whether the realm is locked to new node joins (via PanDHCP). true prevents new nodes from joining; false allows joins. Independent of state - a realm can be online and locked at the same time.

Field	Type	Required	Default	Example	Description
`log_sizing`	`RealmLogSizing` (`invalid` , `4g` , `12g` , `dm1` , `custom_1` , `custom_2` , `custom_3` , `custom_4` , `custom_5` , `current`)	yes	n/a		Log sizing configuration
`mem_sizing_current`	integer	yes	n/a		Current memory sizing level
`mem_sizing_next`	integer	yes	n/a		Next memory sizing level (after reboot)
`mia_timer`	integer	yes	n/a		MIA timer in seconds
`name`	string	yes	n/a		Realm display name
`panfs_mounted`	boolean	yes	n/a		Whether the realm root filesystem is mounted. Creating a tenant, and creating or renaming a volume, need it: while it is false, those calls answer 503. Each API host checks its own mount on every request, so two hosts behind a load balancer may disagree.

Field	Type	Required	Default	Example	Description
`president_flags`	array of `RealmPresidentFlag` (`settled_storage` , `settled_mgr` , `vsm_backup_rebal`)	yes	n/a		President flags
`state`	`RealmState` (`invalid` , `unknown` , `online` , `going_offline` , `offline` , `powered_down` , `rebooting` , `snapping` , `powering_down` , `reinitializing` , `pausing` , `paused` , `upgrade_quiescing` , `upgrade`)	yes	n/a		Current operational state of the realm

Field	Type	Required	Default	Example	Description
`state_flags`	array of `RealmStateFlag` (`offline` , `paused` , `upgrade` , `ure_default` , `log_sizing_bad`)	yes	n/a		Active realm state flags
`volume_layout_type_default`	`RaidType` (`raid6+` , `raid5+` , `raid10+` , `raid5` , `raid10` , `default`)	yes	n/a		Default volume RAID layout type
`volume_max_vols_realm`	integer	yes	n/a		Maximum number of volumes in the realm
`volume_recovery_priority_default`	integer	yes	n/a		Default volume recovery priority

Example


```
{
   "boot_epoch_seconds" : 0,
   "clientmode_current" : "balanced",
   "clientmode_next" : "balanced",
   "data_generation" : 6,
   "heartbeat_failure_grace_seconds" : 30,
   "heartbeat_period_seconds" : 30,
   "id" : "a538deadbeef0001889abcdef1234567",
   "locked" : false,
   "log_sizing" : "12g",
   "mem_sizing_current" : 1,
   "mem_sizing_next" : 1,
   "mia_timer" : 360,
   "name" : "exp-aic-97-5",
   "panfs_mounted" : true,
   "president_flags" : [
      "settled_storage",
      "settled_mgr"
   ],
   "state" : "online",
   "state_flags" : [],
   "volume_layout_type_default" : "raid6+",
   "volume_max_vols_realm" : 1200,
   "volume_recovery_priority_default" : 50
}
```


`RealmClientMode`

Realm client modes.

Allowed values: `invalid` , `balanced` , `favor_df`

`RealmLockActionRequest`

Request to lock the realm (prevent new nodes from joining via PanDHCP).

Field	Type	Required	Default	Example	Description
`action`	literal `lock`	yes	n/a		The action to perform on the realm.
`params`	null (not used)	no	n/a		

`RealmLogSizing`

Realm log sizing options.

Allowed values: `invalid`, `4g`, `12g`, `dm1`, `custom_1`, `custom_2`, `custom_3`, `custom_4`, `custom_5`, `current`

`RealmMoveServiceActionParams`

Parameters for moving a service to a different Director Node.

Field	Type	Required	Default	Example	Description
`node_dataserial`	string	yes	n/a	`50f6dd3e7103a9`, `253d193e710424`	Dataserial of the destination Director Node that will host the service.
`service_id`	string	yes	n/a	`0x0400000000000004`, `0x0400000000000005`	Hexadecimal manager ID of the service to move, e.g. "0x0400000000000004". A volume service's `volume_service_id` (from the volumes API) is one such ID.
`service_type`	`ServiceInstance` (`primary`, `backup`)	yes	n/a	`primary`, `backup`	Which service instance to move: "primary" (live failover) or "backup" (standby).

`RealmMoveServiceActionRequest`

Request to move a service to a different Director Node.

Field	Type	Required	Default	Example	Description
`action`	literal `move_service`	yes	n/a		The action to perform on the realm.
`params`	`RealmMoveServiceActionParams` object	yes	n/a		Parameters identifying the service to move and its destination node.

`RealmOfflineActionParams`

Parameters for the `offline` realm action.

Field	Type	Required	Default	Example	Description
`mode`	`RealmOfflineMode` (`paused` , `unavailable`)	no	`paused`	`paused` , `unavailable`	How to take the realm offline. `paused` performs a graceful drain; `unavailable` makes volumes inaccessible immediately. Defaults to `paused` .

`RealmOfflineActionRequest`

Request to take the realm offline.

Field	Type	Required	Default	Example	Description
`action`	literal `offline`	yes	n/a		The action to perform on the realm.
`params`	`RealmOfflineActionParams` object	no	n/a		Parameters for taking the realm offline.

`RealmOfflineMode`

Mode for the `offline` realm action.

Allowed values: `paused`, `unavailable`

`RealmOnlineActionRequest`

Request to bring the realm online.

Field	Type	Required	Default	Example	Description
`action`	literal `online`	yes	n/a		The action to perform on the realm.
`params`	null (not used)	no	n/a		

`RealmPresidentFlag`

Realm president flags.

Allowed values: `settled_storage`, `settled_mgr`, `vsm_backup_rebal`

`RealmRebootActionRequest`

Request to reboot the realm.

Field	Type	Required	Default	Example	Description
`action`	literal `reboot`	yes	n/a		The action to perform on the realm.
`params`	null (not used)	no	n/a		

`RealmRefResponse`

The realm a run collected from.

Field	Type	Required	Default	Example	Description
`id`	string	yes	n/a	`010bb45f01eb42c188e04703e06b0368`	Realm ID in the RM.
`name`	string	yes	n/a	`exp-aic-97-26`	Realm name, also used in the archive name.

`RealmRehomeRootActionParams`

Parameters required to reassign the realm root volume to a different StorageSet.

Field	Type	Required	Default	Example	Description
`raid_type`	`VolumeLayout` (`raid6+` , `raid5+` , `raid10+` , `raid5` , `raid10`)	no	`raid6+`	`raid6+` , `raid5+` , `raid10+`	RAID layout for the new realm root volume. Defaults to raid6+ when omitted.
`storageset_id`	integer	yes	n/a	`1` , `3` , `5`	Identifier of the StorageSet that will host the new realm root volume.

`RealmRehomeRootActionRequest`

Request to reassign the realm root volume to a different StorageSet.

Field	Type	Required	Default	Example	Description
`action`	literal `rehome_root`	yes	n/a		The action to perform on the realm.
`params`	`RealmRehomeRootActionParams` object	yes	n/a		Parameters required to reassign the realm root volume to a different StorageSet.

`RealmResponse`

Response wrapper for realm, plus the API host's timezone.

Field	Type	Required	Default	Example	Description
`realm`	`Realm` object	yes	n/a		Realm details
`server_tzone`	string	yes	n/a	`America/Los_Angeles`, `UTC`	IANA timezone name of the API server's host (e.g. 'America/Los_Angeles').

Example


```
{
   "realm" : {
      "boot_epoch_seconds" : 0,
      "clientmode_current" : "balanced",
      "clientmode_next" : "balanced",
      "data_generation" : 6,
      "heartbeat_failure_grace_seconds" : 30,
      "heartbeat_period_seconds" : 30,
      "id" : "a538deadbeef0001889abcdef1234567",
      "locked" : false,
      "log_sizing" : "12g",
      "mem_sizing_current" : 1,
      "mem_sizing_next" : 1,
      "mia_timer" : 360,
      "name" : "exp-aic-97-5",
      "panfs_mounted" : true,
      "president_flags" : [
         "settled_storage",
         "settled_mgr"
      ],
      "state" : "online",
      "state_flags" : [],
      "volume_layout_type_default" : "raid6+",
      "volume_max_vols_realm" : 1200,
      "volume_recovery_priority_default" : 50
   },
   "server_tzone" : "America/Los_Angeles"
}
```

`RealmSegmentTypeRequest`

Set the network segment type new subnets are placed on.

Field	Type	Required	Default	Example	Description
`segment_type`	one of: `vlan`, `vxlan`	yes	n/a	`vlan`, `vxlan`	The segment type new subnets are placed on.

Example

```
{
   "segment_type" : "vlan"
}
```

`RealmSegmentTypeResponse`

The network segment type every new subnet of this realm is placed on.

Realm-wide rather than a property of any subnet, so it answers before the first one exists, which is what a client building a create request needs.

Field	Type	Required	Default	Example	Description
`segment_type`	`SubnetSegmentType` (`none` , `vlan` , `vxlan`)	yes	n/a	`vxlan` , `vlan` , `none`	The segment type a new subnet is placed on. `vlan` carries a VLAN ID in `segment_id` and `vxlan` a VNI. `none` is the realm naming no segment at all, which is a state to report rather than an error: a create on such a realm is refused, with a reason.

Example

```
{
   "segment_type" : "vxlan"
}
```

`RealmState`

Realm operational states.

Allowed values: `invalid` , `unknown` , `online` , `going_offline` , `offline` , `powered_down` , `rebooting` , `snapping` , `powering_down` , `reinitializing` , `pausing` , `paused` , `upgrade_quiescing` , `upgrade`

`RealmStateFlag`

Realm state flags.

Allowed values: `offline` , `paused` , `upgrade` , `ure_default` , `log_sizing_bad`

`RealmUnlockActionRequest`

Request to unlock the realm (re-enable PanDHCP joins).

Field	Type	Required	Default	Example	Description
`action`	literal `unlock`	yes	n/a		The action to perform on the realm.
`params`	null (not used)	no	n/a		

`Resource`

RBAC-protected resources. The `s3-reconcile-state` resource applies to calls made by the platform's own S3 services on each node. Customers, Vantage, and panctl do not use it.

Allowed values: `tenants`, `storagesets`, `nodes`, `hardware`, `volumes`, `subnets`, `attachments`, `users`, `service-accounts`, `s3-config`, `s3-instances`, `fs-users`, `s3-buckets`, `s3-admin`, `s3-reconcile-state`, `realm`, `realm-logs`, `observability`, `observability-realm`, `roles`, `tokens`

`RolesResponse`

The current role set of a principal, returned by every GET and PATCH on a roles route.

Field	Type	Required	Default	Example	Description
`roles`	array of string	yes	n/a	`["admin"]`	Roles currently bound to the principal.

`S3AggregatorConfigSchema`

Per-tenant tenant-level aggregator policy.

Opt-in, and defaulting to off. The per-instance endpoints are the tenant's data plane: the same fixed port on every node, inside the tenant's own network namespace, on a subnet its clients are already attached to. `enabled=true` adds an `agg_<uuid>` frontend and backend on each Director, scoped to the tenant namespace.

There is no port field. The aggregator binds a fixed realm-wide port inside that namespace, so there is nothing to allocate or report.

Field	Type	Required	Default	Example	Description
`enabled`	boolean	no	`false`		Whether the tenant-level S3 aggregator is generated for this tenant.

Example

```
{
   "enabled" : false
}
```

`S3BucketCreateRequest`

Request payload for creating an S3 bucket.

Field	Type	Required	Default	Example	Description
`name`	string	yes	n/a		Bucket name.
`owner`	string	yes	n/a		S3 access_key of the PanFS user the bucket belongs to. Required, and must name a PanFS user in this tenant: the gateway resolves the owner through its IAM backend, so a value that is not a PanFS user cannot be resolved there at all.

Example

```
{
   "name" : "my-bucket",
   "owner" : "alice"
}
```

`S3BucketSchema`

S3 bucket data model for API responses.

`owner` names a *PanFS* user, by access key. It is not an authorization input: deletion is governed by `s3-buckets:delete` and ownership is not consulted, matching S3's own `s3:DeleteBucket` semantics. It is the owner in the bucket's ACL on the gateway, which the create sets, so the owning user holds the bucket creator's rights.

The caller is a control-plane principal and has no S3 identity of its own, which is why the owner is named in the request rather than derived from whoever made it.

Field	Type	Required	Default	Example	Description
`name`	string	yes	n/a		Bucket name.
`owner`	string	no	(empty string)		S3 access_key of the PanFS user the bucket belongs to, as the bucket ACL on the gateway records it. Empty string for a bucket whose ACL names no owner.

`S3ConfigCreateRequest`

Request payload for creating a tenant's S3 configuration.

Every field carries a default, so `PUT /s3 {}` is a complete request: it creates the disabled config with the empty prefix and the POSIX defaults a tenant starts with. Those defaults live here rather than in a factory beside the service, because this is the only place that can publish them - an operator reading the API document sees what an omitted field will be, and there is no second copy to drift from it.

`access_key` , `secret_key` and `control_plane` have no field here at all;
:mod: `app.use_cases.create_s3_config` mints all three.

Field	Type	Required	Default	Example	Description
`access_log`	boolean	no	`true`		Whether the S3 access log is enabled for this tenant (the S3 gateway's –access-log audit log). Defaults to true.
`aggregator`	`S3AggregatorConfigSchema` object	no	n/a		Tenant-level S3 aggregator policy (opt-in via `enabled=true`).
`debug_log`	boolean	no	`false`		Whether the S3 debug request log is enabled for this tenant. Defaults to false.
`default_dir_mode`	string	no	`755`	`755` , `0755`	Default directory permission mode in octal (e.g. 755).
`default_group`	string	no	`gid:65534`	`gid:65534` , `gid:1000`	Default POSIX group for objects (e.g. gid:65534). An fs-user created without a gid takes it. gid:65534 (nogroup) is the unset value; an empty string is not accepted.
`default_obj_mode`	string	no	`644`	`644` , `0644`	Default object/file permission mode in octal (e.g. 644).

Field	Type	Required	Default	Example	Description
`default_owner`	string	no	`uid:65534`	`uid:65534`, `uid:1000`	Default POSIX owner for objects (e.g. uid:65534). An fs-user created without a uid takes it. uid:65534 (nobody) is the unset value; an empty string is not accepted.
`default_prefix`	string	no	(empty string)	`/vol1`, `/vol1/data`	Tenant-relative path served by S3 (e.g. /vol1 or /vol1/data). Must name an existing volume of the tenant (or a directory inside one); stored resolved into the tenant volume namespace. Optional: omit it to create the config before the tenant has a volume to serve. The empty value skips volume validation and cannot be enabled until a real prefix is set by PATCH.
`dns`	`S3DnsConfigSchema` object	no	n/a		DNS/named-access policy (default mode=off).
`enabled`	boolean	no	`false`		Whether S3 access is enabled for this tenant.
`metrics_enabled`	boolean	no	`true`		Whether S3 metrics are collected for this tenant (the S3 gateway's StatsD counters to the node-local Telegraf listener). Defaults to true.
`tls`	`S3TlsConfigSchema` object	no	n/a		TLS configuration for the S3 gateway.

Example


```
{
   "aggregator" : {
      "enabled" : false
   },
   "default_dir_mode" : "755",
   "default_group" : "gid:65534",
   "default_obj_mode" : "644",
   "default_owner" : "uid:65534",
   "default_prefix" : "/vol1/t1",
   "enabled" : true,
   "tls" : {}
}
```

`S3ConfigPatchRequest`

Request payload for partially updating a tenant's S3 configuration.

At least one field must be provided. Null values are not allowed - omit fields that should not change.

Field	Type	Required	Default	Example	Description
`access_log`	boolean, nullable	no	`null`		Whether the S3 access log is enabled for this tenant.
`aggregator`	`S3AggregatorConfigSchema` object, nullable	no	`null`		Tenant-level S3 aggregator policy (opt-in via `enabled=true`).
`debug_log`	boolean, nullable	no	`null`		Whether the S3 debug request log is enabled for this tenant.
`default_dir_mode`	string, nullable	no	`null`		Default directory permission mode in octal.
`default_group`	string, nullable	no	`null`		Default POSIX group for objects. To unset it, send gid:65534 (nogroup), the value a new config starts with; an empty string is not accepted.
`default_obj_mode`	string, nullable	no	`null`		Default object/file permission mode in octal.
`default_owner`	string, nullable	no	`null`		Default POSIX owner for objects. To unset it, send uid:65534 (nobody), the value a new config starts with; an empty string is not accepted.

Field	Type	Required	Default	Example	Description
`default_prefix`	string, nullable	no	`null`		Tenant-relative path served by S3. Must name an existing volume of the tenant (or a directory inside one); immutable while the tenant has S3 enabled.
`dns`	`S3DnsConfigSchema` object, nullable	no	`null`		DNS/named-access policy (mode off/panfs/external).
`enabled`	boolean, nullable	no	`null`		Whether S3 access is enabled for this tenant.
`metrics_enabled`	boolean, nullable	no	`null`		Whether S3 metrics are collected for this tenant.
`tls`	`S3TlsConfigSchema` object, nullable	no	`null`		TLS configuration for the S3 gateway.

Example

```
{
   "default_prefix" : "/vol1/t1",
   "enabled" : false
}
```

`S3ConfigSchema`

S3 configuration data model for API responses.

The tenant-root `access_key` / `secret_key` are not returned by this API at all, neither in full nor redacted: they are the gateway's own administrator account, internal to PanFS. `tls.private_key` is the one value still redacted rather than dropped - it has a field because a tenant supplies it, so the response has to say whether one is set.

That is what makes `s3-config:view` an ordinary read, held by the read-only platform roles like any other: it describes a configuration and hands over no credential at all.

Field	Type	Required	Default	Example	Description
`access_log`	boolean	no	`true`		Whether the S3 access log is enabled for this tenant.
`aggregator`	`S3AggregatorConfigSchema` object	yes	n/a		Tenant-level S3 aggregator policy.
`debug_log`	boolean	no	`false`		Whether the S3 debug request log is enabled for this tenant.
`default_dir_mode`	string	yes	n/a		Default directory permission mode in octal.
`default_group`	string	yes	n/a		Default POSIX group for objects.
`default_obj_mode`	string	yes	n/a		Default object/file permission mode in octal.
`default_owner`	string	yes	n/a		Default POSIX owner for objects.
`default_prefix`	string	yes	n/a		Filesystem path served by S3.
`enabled`	boolean	yes	n/a		Whether S3 access is enabled for this tenant.
`metrics_enabled`	boolean	no	`true`		Whether S3 metrics are collected for this tenant.

Field	Type	Required	Default	Example	Description
`tls`	`S3TlsConfigResponseSchema` object	yes	n/a		TLS configuration for the S3 gateway.

Example


```
{
   "aggregator" : {
      "enabled" : false
   },
   "default_dir_mode" : "755",
   "default_group" : "gid:65534",
   "default_obj_mode" : "644",
   "default_owner" : "uid:65534",
   "default_prefix" : "/vol1/t1",
   "enabled" : true,
   "tls" : {
      "ca_certificate" : "-----BEGIN CERTIFICATE-----\n...",
      "certificate" : "-----BEGIN CERTIFICATE-----\n...",
      "mode" : "realm-generated",
      "private_key" : "***"
   }
}
```

`S3ConfigUpdateResponse`

What a PATCH of the S3 config could not apply to the gateways already running.

It is a body so the answer travels in the OpenAPI document and through a JSON client, which is what a header or a log line could not do. An empty list is the ordinary case and is a positive statement - every change this request made is in effect now.

`restart_required` names the fields this request actually *changed* whose new value a running gateway will only pick up when it is restarted - never the tenant's other standing configuration, and never a field re-sent at the value it already held. A tenant with no started instance always answers `[]` : there is nothing holding the old values. Which instances those are is deliberately not repeated here - `GET /tenants/{tenant_id}/s3/instances` already answers that, and a second copy could disagree with it.

Field	Type	Required	Default	Example	Description
`restart_required`	array of string	yes	n/a		Names of S3 config fields this request changed whose new value reaches already-running S3 gateway instances only after those instances are restarted. Empty when every change is in effect now, when the request changed nothing, or when the tenant has no started instance. The set of names can change between releases.

Example

```
{
   "restart_required" : [
      "access_log",
      "debug_log"
   ]
}
```

`S3DnsConfigSchema`

Per-tenant DNS / named-access policy.

`mode` selects whether the realm publishes DNS records; it is never a precondition for reachability (every endpoint is reachable by `IP:port`) and drives no routing:

- `off` (default): IP:port only - no DNS records.
- `panfs`: `pan_dns_server` publishes the aggregator + per-instance records. Not supported for a tenant other than the platform tenant, and refused with 400: the realm DNS server cannot publish addresses on a tenant subnet.
- `external`: the customer runs their own DNS at the tenant's addresses; PanFS publishes nothing.

Field	Type	Required	Default	Example	Description
`mode`	`S3DnsMode` (`off`, `panfs`, `external`)	no	`off`		DNS/named-access mode: off (default), panfs, or external. panfs is supported for the platform tenant only; any other tenant is refused with 400.

Example

```
{
   "mode" : "off"
}
```

`S3DnsMode`

Per-tenant DNS / named-access mode.

DNS records are optional, never a precondition for reachability - every endpoint is reachable by `IP:port` in all modes, and no mode drives routing. The mode is tenant-scoped (the aggregator apex is a tenant name; per-instance names are derived subdomains).

Allowed values: `off`, `panfs`, `external`

`S3EnabledRequest`

Request body for PUT /s3/enabled.

Field	Type	Required	Default	Example	Description
`enabled`	boolean	yes	n/a		Realm-wide S3 enable flag.

Example

```
{
   "enabled" : true
}
```

`S3EnabledResponse`

Response body for GET /s3/enabled and PUT /s3/enabled.

When PUT disables S3 at the realm level while at least one tenant still has tenant-level S3 enabled, `warning` carries a human- readable message and `tenants_with_s3_enabled` lists the short names of the offending tenants. Both fields are absent on GET and on PUT-enable.

Field	Type	Required	Default	Example	Description
`enabled`	boolean	yes	n/a		Realm-wide S3 enable flag.
`tenants_with_s3_enabled`	array of string, nullable	no	`null`		Tenant short names that still have tenant-level S3 enabled after the realm-wide disable. Absent on get and on enable.
`warning`	string, nullable	no	`null`		Warning surfaced when realm-disable leaves tenants with S3 enabled.

Example


```
{
   "enabled" : false,
   "tenants_with_s3_enabled" : [
      "tn1"
   ],
   "warning" : "1 tenant(s) still have S3 enabled at the tenant level"
}
```

`S3InstanceCreateRequest`

Create a new S3 gateway instance for a tenant on a specific node.

Field	Type	Required	Default	Example	Description
`dataserial`	string	yes	n/a		Target node's dataserial (14 hex chars) as returned by `GET /nodes` and consumed by every `/nodes/{dataserial}` endpoint (e.g. `08823f71a1eecc`). The stable placement identity - unaffected by node renames.
`instance_id`	string	yes	n/a		Instance identifier: exactly 5 alphanumeric characters (D10).

Example

```
{
   "dataserial" : "08823f71a1eecc",
   "instance_id" : "abc12"
}
```

`S3InstanceCreateResponse`

Response body for a successful instance create.

Field	Type	Required	Default	Example	Description
`instance`	`S3InstanceSchema` object	yes	n/a		Desired-state view of an S3 gateway instance.

`S3InstanceFanoutResponse`

Aggregate body for fan-out verbs (start / stop / delete / show).

`aggregate_status` reflects the worst-case per-node outcome. Callers that want a single success/failure signal should inspect this field; detail-level consumers should walk `results`.

Field	Type	Required	Default	Example	Description
`aggregate_status`	integer	yes	n/a		Aggregate HTTP status: the rows' shared success code (202 for a recorded state change, 200 for a read) if every node succeeded, the shared error code if every node failed with the same code, otherwise 207. delete-all answers 200 once every instance is gone, and start-all/stop-all answer 200 with no results for a tenant with no instances.
`confirmed`	boolean	no	`true`		For delete fan-out, true once the caller has confirmed the destructive operation. When false, the request body enumerates matches without deleting anything.
`results`	array of `S3InstancePerNodeResult` object	yes	n/a		

`S3InstanceHealthSchema`

Observed/health view merged in by the reader path.

Field	Type	Required	Default	Example	Description
`consecutive_failures`	integer	yes	n/a		
`last_error`	string, nullable	no	`null`		
`last_seen_ts`	integer, nullable	yes	n/a		When a probe last reached the gateway (epoch seconds); null until one has.
`restart_count_60s`	integer	yes	n/a		

`S3InstanceListResponse`

Response body for list and fan-out operations.

Field	Type	Required	Default	Example	Description
`instances`	array of `S3InstanceViewSchema` object	yes	n/a		

`S3InstancePerNodeResult`

Per-node outcome inside a fan-out response body (D10).

`status` is the HTTP status the call would have returned if it had targeted this single node; `error` carries the detail message on non-2xx cases. `instance` is populated on read verbs; null for start/stop/delete outcomes, which record the new desired state without waiting for the instance to reach it (show the instance for its observed state). `data_serial` identifies the target node.

Field	Type	Required	Default	Example	Description
`data_serial`	string	yes	n/a		
`error`	string, nullable	no	`null`		
`instance`	`S3InstanceViewSchema` object, nullable	no	`null`		
`status`	integer	yes	n/a		Per-node HTTP status (200, 202, 404, 409, 503, ...).

`S3InstanceSchema`

Desired-state view of an S3 gateway instance.

Field	Type	Required	Default	Example	Description
`created_at`	integer	yes	n/a		
`data_serial`	string	yes	n/a		
`desired`	string	yes	n/a		created | started | stopped | deleting
`instance_id`	string	yes	n/a		
`last_transition_at`	integer	yes	n/a		
`module_instance_token`	string	yes	n/a		

`S3InstanceViewSchema`

Merged desired + observed/health row returned by list/show.

Field	Type	Required	Default	Example	Description
`desired`	`S3InstanceSchema` object	yes	n/a		Desired-state view of an S3 gateway instance.
`dns_name`	string, nullable	no	`null`		Optional DNS endpoint https://.s3-., populated only when the tenant has dns.mode=panfs.
`endpoint`	string, nullable	no	`null`		Primary S3 endpoint as https://:3000; null when the hosting node holds no address on the tenant subnet.
`health`	`S3InstanceHealthSchema` object, nullable	no	`null`		Null when the observed state could not be read: nothing was measured.
`last_health_transition_at`	integer, nullable	no	`null`		When observed last changed (epoch seconds); null when it could not be read or is not yet known.
`observed`	string	yes	n/a		unknown | starting | running | unresponsive | crashloop | stopped; unknown, with observed_reason, when the hosting node has no provider for the instance yet or cannot be reached.

Field	Type	Required	Default	Example	Description
`observed_reason`	string, nullable	no	`null`		Why PanAPI could not read the instance's observed state, when observed is unknown for that reason. Never the gateway's own error, which is health.last_error.
`tenant_id`	string	yes	n/a		

`S3PrefixRequest`

Request body for PUT /s3/prefix.

Field	Type	Required	Default	Example	Description
`prefix`	string	yes	n/a		Realm-wide POSIX-style path prefix. Empty string means "no realm-wide prefix"; otherwise the value MUST begin with "/" and MUST NOT contain trailing whitespace.

Example

```
{
   "prefix" : "/realm-root"
}
```

`S3PrefixResponse`

Response body for GET /s3/prefix and PUT /s3/prefix.

Field	Type	Required	Default	Example	Description
`prefix`	string	yes	n/a		Current realm-wide path prefix.

Example

```
{
   "prefix" : "/realm-root"
}
```

`S3Profile`

S3 access profile of a PanFS user.

- `DISABLED` - credentials are stored but the reveal endpoint returns 404 (which the S3 gateway maps to `auth.ErrNoSuchUser`).
- `USER` - S3 gateway `role=user`.
- `PRIVILEGED` - S3 gateway `role=userplus` (bucket-creation rights). The operator-facing profile name is `privileged`; the gateway-internal role string it maps to stays `userplus`.

This docstring is published as the enum's description in the OpenAPI document, so it says what a caller may choose and nothing else.

Allowed values: `disabled`, `user`, `privileged`

`S3TlsConfigResponseSchema`

TLS settings as returned by GET /tenants/{tid}/s3.

Same fields as the TLS settings accepted on write, but with `private_key` redacted, so the response body never carries the tenant TLS private key in plaintext.

Field	Type	Required	Default	Example	Description
`ca_certificate`	string, nullable	no	`null`		PEM-encoded CA certificate.
`certificate`	string, nullable	no	`null`		PEM-encoded server certificate.
`mode`	`S3TlsMode` (`realm-generated`, `tenant-provided`)	no	`realm-generated`		Source of the tenant-wide S3 edge cert.
`private_key`	string, nullable	no	`null`		PEM-encoded private key. Redacted to `***` on response when present; null when no key is configured.

Example

```
{
   "ca_certificate" : "-----BEGIN CERTIFICATE-----\n...",
   "certificate" : "-----BEGIN CERTIFICATE-----\n...",
   "mode" : "realm-generated",
   "private_key" : "***"
}
```

`S3TlsConfigSchema`

Tenant-wide TLS material for the S3 gateway.

The S3 data plane is always TLS-terminated. A single cert+key is served by the whole tenant S3 surface (the aggregator frontend and every per-instance endpoint), whatever SNI the client sends.

`mode` selects the cert source explicitly - never inferred from the presence of cert material:

- `realm-generated` (default): the realm mints the cert and stores it, with the SANs `*.s3-<short>.<zone>`, `s3-<short>.<zone>` when the aggregator is on, and the addresses on the tenant's subnets of the nodes that serve it. The request body MUST NOT carry cert material in this mode.
- `tenant-provided`: the tenant supplies `certificate` and `private_key` (and optionally `ca_certificate`); the realm validates them at config-set. The cert carries the names and addresses its clients use; only with `dns.mode=panfs` must it also carry the names the realm publishes (`*.s3-<short>.<zone>`, plus `s3-<short>.<zone>` when the aggregator is on).

Field	Type	Required	Default	Example	Description
`ca_certificate`	string, nullable	no	`null`		PEM-encoded CA certificate.
`certificate`	string, nullable	no	`null`		PEM-encoded server certificate.
`mode`	`S3TlsMode` (`realm-generated`, `tenant-provided`)	no	`realm-generated`		Explicit source of the tenant-wide S3 edge cert: `realm-generated` (the realm mints it) or `tenant-provided` (tenant supplies the cert+key).
`private_key`	string, nullable	no	`null`		PEM-encoded private key.

Example


```
{
   "ca_certificate" : "-----BEGIN CERTIFICATE-----\n...",
   "certificate" : "-----BEGIN CERTIFICATE-----\n...",
   "mode" : "tenant-provided",
   "private_key" : "-----BEGIN PRIVATE KEY-----\n..."
}
```


`S3TlsMode`

Explicit source of the tenant-wide S3 edge cert.

The cert source is an explicit, tenant-wide choice - never inferred from the presence of cert material. A single cert+key in the blob is served by the whole tenant S3 surface in both modes; the mode decides who produced the bytes, and with it which names the cert must carry ().

Allowed values: `realm-generated` , `tenant-provided`

`ServiceAccountCreateRequest`

Request to create a service account.

Field	Type	Required	Default	Example	Description
`description`	string, nullable	no	`null`		What this account is for. Newlines and tabs are allowed; other control characters are not.
`display_name`	string, nullable	no	`null`		Human-readable label. Control and invisible characters are not allowed.
`name`	string	yes	n/a		Unique name within the tenant, 4-30 characters of lowercase letters, digits, and hyphens, starting with a letter. Identifies the account wherever it appears; it cannot be changed afterwards.

Example

```
{
   "display_name" : "Nightly CI",
   "name" : "ci-nightly"
}
```

`ServiceAccountListResponse`

Every service account in the caller's tenant, newest first.

Field	Type	Required	Default	Example	Description
`service_accounts`	array of `ServiceAccountResponse` object	yes	n/a		The accounts.
`total`	integer	yes	n/a		How many accounts are listed.

`ServiceAccountResponse`

One service account.

Field	Type	Required	Default	Example	Description
`created_at`	string (date-time)	yes	n/a		When the account was created.
`created_by`	string (uuid)	yes	n/a		The user that created it.
`description`	string, nullable	no	`null`		What this account is for.
`display_name`	string, nullable	no	`null`		Human-readable label.
`id`	string (uuid)	yes	n/a		Identifier of the service account.
`name`	string	yes	n/a		Unique name within the tenant, 4-30 characters of lowercase letters, digits, and hyphens, starting with a letter. Identifies the account wherever it appears; it cannot be changed afterwards.
`updated_at`	string (date-time)	yes	n/a		When the account was last changed.

`ServiceAccountUpdateRequest`

Request to update a service account. Only the fields present are applied.

Field	Type	Required	Default	Example	Description
`description`	string, nullable	no	`null`		What this account is for. Newlines and tabs are allowed; other control characters are not.
`display_name`	string, nullable	no	`null`		Human-readable label. Control and invisible characters are not allowed.

Example

```
{
   "description" : "Nightly backup runner"
}
```

`ServiceInstance`

Which instance of a service the `move_service` action targets.

Allowed values: `primary`, `backup`

`SetRolesRequest`

Request payload for replacing the full role set of a Tenant User.

Field	Type	Required	Default	Example	Description
`roles`	array of string	yes	n/a	`["admin"]`	Roles to assign to the user; replaces the current set. A role the target tenant does not define, or a reserved one, is rejected with HTTP 400.

`StorageSet`

StorageSet for list/detail responses.

Only fields relevant for the API response are included; more can be added in future versions.

Field	Type	Required	Default	Example	Description
`capacity_balancing`	boolean	yes	n/a		Whether automatic capacity balancing across nodes is enabled
`encrypted`	boolean	yes	n/a		Whether the StorageSet encrypts data at rest
`fips_mode`	boolean	yes	n/a		Whether the StorageSet operates in FIPS mode
`id`	integer	yes	n/a		StorageSet ID
`mlec`	boolean	yes	n/a		Whether Multi-Level Erasure Coding is enabled. Fixed when the StorageSet is created
`name`	string	yes	n/a		StorageSet name
`nodes_not_responding`	integer	yes	n/a		Number of offline nodes/OSDs that are unreachable with a recovery timer still running
`nodes_offline`	integer	yes	n/a		Number of nodes/OSDs not online for any reason. Includes nodes_not_responding
`nodes_online`	integer	yes	n/a		Number of nodes/OSDs currently online in this StorageSet
`nodes_reconstructing`	integer	yes	n/a		Number of nodes/OSDs with a pending data rebuild, whether or not they are online

Field	Type	Required	Default	Example	Description
`nodes_total`	integer	yes	n/a		Total number of nodes/OSDs assigned to this StorageSet
`spares_available`	integer	yes	n/a		Current number of spare units available for reconstruction
`spares_requested`	integer	yes	n/a		Configured number of requested spare units
`state`	`StorageSetState` object	yes	n/a		Conditions the StorageSet currently holds, and their severity.
`topology_aware_placement`	boolean	yes	n/a		Whether objects are placed with the enclosure topology in mind
`ure_default`	boolean	yes	n/a		Whether the StorageSet uses the default unrecoverable-read-error handling
`vhosts_total`	integer	yes	n/a		Total number of virtual storage hosts assigned to this StorageSet

Example


```
{
   "capacity_balancing" : true,
   "encrypted" : false,
   "fips_mode" : false,
   "id" : 1,
   "mlec" : false,
   "name" : "Set 1",
   "nodes_not_responding" : 0,
   "nodes_offline" : 0,
   "nodes_online" : 6,
   "nodes_reconstructing" : 0,
   "nodes_total" : 6,
   "spares_available" : 2,
   "spares_requested" : 2,
   "state" : {
      "conditions" : [],
      "status" : "ok"
   },
   "topology_aware_placement" : false,
   "ure_default" : false,
   "vhosts_total" : 6
}
```


`StorageSetAddStorageActionRequest`

Request to perform an add_storage action on a StorageSet.

Field	Type	Required	Default	Example	Description
`action`	literal `add_storage`	yes	n/a		The action to perform on the storageset.
`params`	`StorageSetAddStorageParams` object	yes	n/a		Parameters required to perform the add_storage action.

`StorageSetAddStorageMode`

Operation mode for add_storage action.

Allowed values: `add_spare` , `migration`

`StorageSetAddStorageParams`

Parameters for the add_storage action on a StorageSet.

Supports two modes:

- add_spare: Expand capacity and adjust spare Storage Node count
- migration: Add storage to help drain another enclosure

Dataserials are Storage Nodes (OSDs/VPODs) by name or serial. If not specified, all unassigned nodes are added.

Field	Type	Required	Default	Example	Description
`dataserials`	array of string	no	n/a		List of dataserial names or serial numbers (OSD and VPOD nodes) to add. List the nodes to add. Omitting this field, or sending an empty list, adds nothing and still returns 204.
`mode`	`StorageSetAddStorageMode` (`add_spare` , `migration`)	yes	n/a		Operation mode: “add_spare” to expand capacity while adjusting spare Storage Nodes, “migration” to add storage for draining another enclosure
`spares_requested`	integer, nullable	no	`null`		For add_spare mode: count of spare Storage Nodes to add (0 or greater). A value of 0 means add zero spare nodes, keeping the current spare count unchanged. For migration mode: must not be provided (omit from request). Providing this field in migration mode will fail validation.

`StorageSetAlertStatus`

Active-alert presence for one StorageSet.

Feeds the storage-set status badge, which the frontend computes by combining this with the set's node/spare counts. Only sets carrying at least one active PanFS alert appear in the response; absence means "no alerts".

Field	Type	Required	Default	Example	Description
`has_critical`	boolean	yes	n/a		At least one active critical/error alert.
`has_warning`	boolean	yes	n/a		At least one active warning alert.
`storageset_name`	string	yes	n/a		StorageSet name as it appears in the alert message.

`StorageSetAssignment`

StorageSet assignment with per-tenant quota.

Represents the allocation of a specific StorageSet to a tenant, including the maximum storage quota the tenant can consume within that StorageSet.

Field	Type	Required	Default	Example	Description
`quota_bytes`	integer	yes	n/a	`2199023255552`, `54975581388880`	Maximum storage quota in bytes that the tenant can consume within this StorageSet.
`storageset_id`	integer	yes	n/a	`1`, `2`, `5`	Identifier of the assigned StorageSet.
`storageset_name`	string	no	(empty string)	`Set 1`, `Production Set`	Name of the assigned StorageSet, resolved server-side so tenant callers need no access to the platform storageset endpoints. Empty only if the StorageSet is no longer present on the cluster.

Example

```
{
   "quota_bytes" : 2199023255552,
   "storageset_id" : 1,
   "storageset_name" : "Set 1"
}
```

`StorageSetCondition`

A condition a storageset can be in. Several hold at once; each carries a status.

Allowed values: `no_osds_assigned`, `activating`, `partly_offline`, `all_nodes_offline`, `reconstructing`, `spare_shortfall`, `recon_reserve_not_enough_osds`, `recon_reserve_not_enough_space`

`StorageSetCreateRequest`

Request payload for creating a StorageSet.

Requires a unique StorageSet name. Optional parameters use system defaults when omitted.

Field	Type	Required	Default	Example	Description
`flags`	array of `StorageSetFlag` (`ure_default` , `capacity_balancing_disabled` , `encrypted` , `fips_mode` , `topology_aware_placement` , `mlec`)	no	n/a	`["mlec"]` , `[]`	Flags to set during StorageSet creation.
`name`	string	yes	n/a	`Set 2` , `Production Set`	Unique display name of the StorageSet. Must be unique within the realm.
`spares_requested`	integer	no	`2`	`2` , `4` , `6`	Number of VPODs to be reserved for reconstruction within the new StorageSet. Default: 2.

Example


```
{
   "flags" : [
      "mlec"
   ],
   "name" : "Set 1",
   "spares_requested" : 4
}
```


`StorageSetCreateResponse`

Minimal response for successful StorageSet creation.

Returned on POST /storagesets to signal successful creation. Clients should fetch full details via GET /storagesets if needed.

Field	Type	Required	Default	Example	Description
`id`	integer	yes	n/a	`1`, `2`, `3`	Unique StorageSet identifier assigned by the system

Example


```
{
   "id" : 3
}
```


`StorageSetFlag`

Flags a client may set when creating a storageset. The realm's own flags are read back as fields.

Allowed values: `ure_default`, `capacity_balancing_disabled`, `encrypted`, `fips_mode`, `topology_aware_placement`, `mlec`

`StorageSetGetResponse`

Detailed view of a single StorageSet.

Field	Type	Required	Default	Example	Description
`storageset`	`StorageSet` object	yes	n/a		StorageSet entity with all available details.

Example


```
{
   "storageset" : {
      "capacity_balancing" : true,
      "encrypted" : false,
      "fips_mode" : false,
      "id" : 1,
      "mlec" : false,
      "name" : "Set 1",
      "nodes_not_responding" : 0,
      "nodes_offline" : 0,
      "nodes_online" : 6,
      "nodes_reconstructing" : 0,
      "nodes_total" : 6,
      "spares_available" : 2,
      "spares_requested" : 2,
      "state" : {
         "conditions" : [],
         "status" : "ok"
      },
      "topology_aware_placement" : false,
      "ure_default" : false,
      "vhosts_total" : 6
   }
}
```

`StorageSetListResponse`

Collection response for StorageSet list operations.

Field	Type	Required	Default	Example	Description
`storagesets`	array of `StorageSet` object	no	n/a		Array of StorageSet objects
`total`	integer	yes	n/a	`1`, `5`, `10`	Total count of StorageSets in this response.

Example


```
{
   "storagesets" : [
      {
         "capacity_balancing" : true,
         "encrypted" : false,
         "fips_mode" : false,
         "id" : 1,
         "mlec" : false,
         "name" : "Set 1",
         "nodes_not_responding" : 0,
         "nodes_offline" : 0,
         "nodes_online" : 6,
         "nodes_reconstructing" : 0,
         "nodes_total" : 6,
         "spares_available" : 2,
         "spares_requested" : 2,
         "state" : {
            "conditions" : [],
            "status" : "ok"
         },
         "topology_aware_placement" : false,
         "ure_default" : false,
         "vhosts_total" : 6
      }
   ],
   "total" : 1
}
```

`StorageSetPatchRequest`

Request payload for updating an existing StorageSet via PATCH.

Field	Type	Required	Default	Example	Description
`capacity_balancing_disabled`	boolean, nullable	no	`null`	`true`, `false`	True to disable capacity balancing (set flag), False to enable it (clear flag).
`name`	string, nullable	no	`null`	`Set 3`, `Production Set`	Updated StorageSet name. Must be unique within the realm.
`spares_requested`	integer, nullable	no	`null`	`1`, `2`, `4`	Optional updated number of VPOD spares reserved for reconstruction. If provided, must be at least 1.

Example

```
{
   "capacity_balancing_disabled" : false,
   "name" : "Set 3",
   "spares_requested" : 3
}
```

`StorageSetSpaceCapacity`

Capacity bucket with total, free, and used bytes.

Field	Type	Required	Default	Example	Description
`free`	integer	yes	n/a	`278234398720`, `55650041856`	Free (available) capacity in this bucket in bytes.
`total`	integer	yes	n/a	`3339001528 32`, `278250110976`	Total capacity of this bucket in bytes.
`used`	integer	yes	n/a	`15712256`, `0`	Used capacity in this bucket in bytes (total - free).

`StorageSetSpaceResponse`

Space usage response for a StorageSet.

Three capacity buckets: raw - overall physical capacity across all buckets. usable - capacity available for data storage. reserved - reconstruction reserve held for spare rebuilds.

Field	Type	Required	Default	Example	Description
`capacity_balancing_active`	boolean	yes	n/a	`true`, `false`	Whether automatic capacity balancing is currently active for this StorageSet.
`node_count`	integer	yes	n/a	`12`, `6`	Number of OSD nodes (storage blades) in this StorageSet.
`raw`	`StorageSetSpaceCapacity` object	yes	n/a		Overall physical capacity across all buckets in this StorageSet.
`reserved`	`StorageSetSpaceCapacity` object	yes	n/a		Reconstruction reserve capacity. Held for spare rebuilds; not available for regular data storage.
`usable`	`StorageSetSpaceCapacity` object	yes	n/a		Capacity available for data storage (usable bucket).
`volume_hard_quota`	integer	yes	n/a	`524288000`, `0`	Sum of hard quotas for all volumes in this StorageSet, in bytes. A create is reflected immediately; after a delete the RM releases the quota asynchronously, so this figure can briefly overstate the true sum.

Example


```
{
   "capacity_balancing_active" : true,
   "node_count" : 6,
   "raw" : {
      "free" : 333884440576,
      "total" : 333900152832,
      "used" : 15712256
   },
   "reserved" : {
      "free" : 55650041856,
      "total" : 55650041856,
      "used" : 0
   },
   "usable" : {
      "free" : 278234398720,
      "total" : 278250110976,
      "used" : 15712256
   },
   "volume_hard_quota" : 524288000
}
```

`StorageSetState`

Conditions a StorageSet currently holds, and the status they reduce to.

Field	Type	Required	Default	Example	Description
`conditions`	array of `StorageSetCondition` (`no_osds_assigned` , `activating` , `partly_offline` , `all_nodes_offline` , `reconstructing` , `spare_shortfall` , `recon_reserve_not_enough_osds` , `recon_reserve_not_enough_space`)	no	n/a	`["partly_offline","reconstructing"]` , `[]`	Every condition currently active. More than one can hold at a time; an empty list means "ok".
`status`	`StorageSetStatus` (`ok` , `informational` , `warning` , `critical`)	yes	n/a	`ok` , `warning`	Severity of the most severe active condition, ordered "ok" < "informational" < "warning" < "critical". "ok" means no condition is active.

`StorageSetStatus`

Severity of a storageset's worst active condition. Declared in ascending severity.

Allowed values: `ok`, `informational`, `warning`, `critical`

`SubjectKind`

What a token authenticates as.

Allowed values: `user`, `service-account`

`Subnet`

Subnet representation.

Field	Type	Required	Default	Example	Description
`addr_count`	integer	no	`0`		Number of addresses bound to this subnet. Computed by the server.
`addr_external`	boolean	no	`false`		True when an outside allocator hands out the addresses. Fixed at create.
`created_at`	string (date-time)	yes	n/a		Creation timestamp (UTC).
`gateway`	string, nullable	no	`null`		Default gateway address, or null when none is configured.
`id`	string (uuid)	yes	n/a		Unique identifier of the subnet.
`ip_range_base`	string	yes	n/a		Base IP address of the subnet range (IPv4 or IPv6).
`ip_range_prefix`	integer	yes	n/a		CIDR prefix length.
`mtu`	integer	yes	n/a		Maximum transmission unit (0 means the platform default).
`name`	string	yes	n/a		Subnet name, unique per tenant.
`range_end`	string, nullable	no	`null`		Upper bound of the usable-address window, or null when the subnet has no window.
`range_start`	string, nullable	no	`null`		Lower bound of the usable-address window, or null when the subnet has no window.

Field	Type	Required	Default	Example	Description
`segment_id`	integer	no	`0`		Network segment identifier: a VLAN ID when segment_type is vlan and a VNI when it is vxlan. 0 means the identifier is unassigned. Bounded by 0-16777215, the wider of the two identifier spaces, so the document states the same ceiling a create request is held to. Fixed at create.
`segment_type`	`SubnetSegmentType` (`none` , `vlan` , `vxlan`)	no	`none`		Kind of network segment the subnet is placed on. The realm owns the choice, so this is returned and never accepted. Fixed at create.
`tenant_id`	string (uuid)	yes	n/a		UUID of the owning tenant, which is always a named tenant.
`tunnel_udp_port`	integer	no	`0`		UDP port carrying a tunnelled segment. 0 means unset, and the realm manager applies its own default. Bounded by 0-65535, the width of a UDP port. Fixed at create.
`updated_at`	string (date-time)	yes	n/a		Last-modification timestamp (UTC).

Example


```
{
   "addr_count" : 12,
   "addr_external" : false,
   "created_at" : "2026-06-29T12:00:00Z",
   "gateway" : "192.168.50.1",
   "id" : "36a8995f-026b-47e1-88bd-7fb9e7239995",
   "ip_range_base" : "192.168.50.0",
   "ip_range_prefix" : 24,
   "mtu" : 1450,
   "name" : "storage-net",
   "range_end" : "192.168.50.200",
   "range_start" : "192.168.50.10",
   "segment_id" : 10,
   "segment_type" : "vlan",
   "tenant_id" : "00000000-0000-0000-0000-000000000001",
   "tunnel_udp_port" : 0,
   "updated_at" : "2026-06-29T12:30:00Z"
}
```

`SubnetCreateRequest`

Request payload for creating a subnet.

Field	Type	Required	Default	Example	Description
`addr_external`	boolean	no	`false`	`false`, `true`	Selects the address-allocation model. False leaves PanFS managing the addresses, true hands that to an outside allocator. Fixed at create and cannot be changed afterwards.
`gateway`	string, nullable	no	`null`	`192.168.50.1`, `null`	Optional default gateway address, in the same family as ip_range_base.
`ip_range_base`	string	yes	n/a	`192.168.50.0`, `fd00:dead:beef::`	Base IP address of the subnet range (IPv4 or IPv6).
`ip_range_prefix`	integer	yes	n/a	`24`, `64`	CIDR prefix length. 1-32 for IPv4, 1-128 for IPv6.
`mtu`	integer	no	`0`	`0`, `1450`, `8950`	The MTU of the subnet's tenant interface on every realm node and every attached host. 0 selects the default, which is the realm's primary network MTU less the encapsulation overhead (50 bytes on vxlan, 4 bytes on vlan): 1450 on vxlan over a 1500-byte primary network. Any other value, from 68 to 9216, has to leave that overhead within the primary network MTU. A larger one is refused with 400 naming the largest value that fits.

Field	Type	Required	Default	Example	Description
`name`	string	yes	n/a	`storage-net`, `tenant_a-vlan10`	Subnet name, unique per tenant. 3-31 chars from [A-Za-z0-9_-].
`range_end`	string, nullable	no	`null`	`192.168.50.200`, `null`	Upper bound of the usable-address window, which is the range the realm allocates its own node addresses from. Supply it together with range_start, or omit both and let the realm derive the window, as range_start describes.
`range_start`	string, nullable	no	`null`	`192.168.50.10`, `null`	Lower bound of the usable-address window, which is the range the realm allocates its own node addresses from. The window must hold at least as many addresses as the realm primary pool. A host attached from outside the realm is placed outside this window. Supply it together with range_end, or omit both and the realm derives the window from the first usable host, as deep as its primary pool, so the subnet as stored reports a window this request did not carry. The realm refuses with 400 when it cannot read that pool, and then the window has to be named. Nothing is derived for a subnet whose addresses an outside allocator owns.

Field	Type	Required	Default	Example	Description
`segment_id`	integer	no	`0`	`0`, `10`, `100100`	Network segment identifier: a VLAN ID when the realm runs 802.1Q and a VNI when it runs VXLAN. If you leave it out, the realm assigns an unused identifier; 0 leaves it unassigned. Otherwise the bound here is the wider of the two spaces, 1-16777215. The realm manager owns the narrower range that follows the realm segment type, and the uniqueness of the identifier across the realm. Fixed at create and cannot be changed afterwards.
`tunnel_udp_port`	integer	no	`0`	`0`, `4789`	UDP port carrying a tunnelled segment, which is meaningful when the realm runs VXLAN. 0 leaves it unset and the realm manager applies its own default. Otherwise the bound is 1-65535. Fixed at create and cannot be changed afterwards.

Example


```
{
   "addr_external" : false,
   "gateway" : "192.168.50.1",
   "ip_range_base" : "192.168.50.0",
   "ip_range_prefix" : 24,
   "mtu" : 1450,
   "name" : "storage-net",
   "range_end" : "192.168.50.200",
   "range_start" : "192.168.50.10",
   "segment_id" : 10,
   "tunnel_udp_port" : 0
}
```


`SubnetCreateResponse`

Minimal response for successful subnet creation.

Field	Type	Required	Default	Example	Description
`id`	string (uuid)	yes	n/a		Server-assigned unique identifier of the created subnet.

Example


```
{
   "id" : "36a8995f-026b-47e1-88bd-7fb9e7239995"
}
```

`SubnetGetResponse`

Detailed representation of a single subnet.

Field	Type	Required	Default	Example	Description
`subnet`	`Subnet` object	yes	n/a		Subnet entity with all available details.

Example


```
{
   "subnet" : {
      "addr_count" : 12,
      "addr_external" : false,
      "created_at" : "2026-06-29T12:00:00Z",
      "gateway" : "192.168.50.1",
      "id" : "36a8995f-026b-47e1-88bd-7fb9e7239995",
      "ip_range_base" : "192.168.50.0",
      "ip_range_prefix" : 24,
      "mtu" : 1450,
      "name" : "storage-net",
      "range_end" : "192.168.50.200",
      "range_start" : "192.168.50.10",
      "segment_id" : 10,
      "segment_type" : "vlan",
      "tenant_id" : "00000000-0000-0000-0000-000000000001",
      "tunnel_udp_port" : 0,
      "updated_at" : "2026-06-29T12:30:00Z"
   }
}
```

`SubnetListResponse`

A collection of subnets with a total count.

Field	Type	Required	Default	Example	Description
`subnets`	array of `Subnet` object	no	n/a		Array of subnet objects. Empty array if none are found.
`total`	integer	yes	n/a	`0`, `1`, `12`	Total count of subnets returned.

Example


```
{
   "subnets" : [
      {
         "addr_count" : 12,
         "addr_external" : false,
         "created_at" : "2026-06-29T12:00:00Z",
         "gateway" : "192.168.50.1",
         "id" : "36a8995f-026b-47e1-88bd-7fb9e7239995",
         "ip_range_base" : "192.168.50.0",
         "ip_range_prefix" : 24,
         "mtu" : 1450,
         "name" : "storage-net",
         "range_end" : "192.168.50.200",
         "range_start" : "192.168.50.10",
         "segment_id" : 10,
         "segment_type" : "vlan",
         "tenant_id" : "00000000-0000-0000-0000-000000000001",
         "tunnel_udp_port" : 0,
         "updated_at" : "2026-06-29T12:30:00Z"
      }
   ],
   "total" : 1
}
```

`SubnetPatchRequest`

Request payload for updating a subnet via PATCH (partial update).

Only the supplied fields are modified, and a patch carrying no field at all is refused here, so it answers 422 with the same envelope every other patch schema uses. A field supplied as null is cleared, which is how a gateway is removed and how a usable window is withdrawn by clearing both of its endpoints. `name` and `mtu` carry no meaning when cleared, so an explicit null is refused for them.

`addr_external` is absent by design: the flag is fixed at create, and `extra='forbid'` refuses a request that carries it. The four network segment values are absent for the same reason, and a patch naming any of them is refused with that field in the error location.

Field	Type	Required	Default	Example	Description
`gateway`	string, nullable	no	`null`	`192.168.50.254`, `null`	New default gateway address (IPv4 or IPv6). Null clears the gateway.
`mtu`	integer, nullable	no	`null`	`1450`, `8950`	New MTU. The MTU of the subnet's tenant interface on every realm node and every attached host. 0 selects the default, which is the realm's primary network MTU less the encapsulation overhead (50 bytes on vxlan, 4 bytes on vlan): 1450 on vxlan over a 1500-byte primary network. Any other value, from 68 to 9216, has to leave that overhead within the primary network MTU. A larger one is refused with 400 naming the largest value that fits.
`name`	string, nullable	no	`null`		

Field	Type	Required	Default	Example	Description
`range_end`	string, nullable	no	`null`	`192.168.50.220`, `null`	New upper bound of the usable-address window. Patchable on its own. Null clears it, and clearing both bounds together has the realm derive the window anew, as range_start describes.
`range_start`	string, nullable	no	`null`	`192.168.50.20`, `null`	New lower bound of the usable-address window. Patchable on its own, and the realm manager validates the resulting pair against the subnet CIDR. Null clears it. Clearing both bounds together has the realm derive the window anew, on the same terms a create naming none gets one, and is refused with 409 while a subnet the realm addresses itself has an attached host.

Example

```
{
   "gateway" : "192.168.50.254",
   "mtu" : 1500,
   "name" : "storage-net-2",
   "range_end" : "192.168.50.220",
   "range_start" : "192.168.50.20"
}
```

`SubnetSegmentType`

Kind of network segment a subnet is placed on.

The realm owns the choice, so a create request never names it. `none` means the realm named no type when the subnet was created.

- `none` : the subnet's segment is not described
- `vlan` : 802.1Q, and the segment identifier is a VLAN ID
- `vxlan` : VXLAN, and the segment identifier is a VNI

Allowed values: `none` , `vlan` , `vxlan`

`Tenant`

Core tenant data model.

Represents a tenant with all its attributes. Used as the inner model in response wrappers (GetTenantResponse, ListTenantResponse).

The two names are published the other way round from the domain: `name` is the handle the domain calls `short_name`, and `display_name` is the label the domain calls `name`. The aliases hold that mapping, so a domain object validates straight into this view.

Field	Type	Required	Default	Example	Description
`auth_settings`	`LocalAuthSettings` object or `PamAuthSettings` object	yes	n/a	`{"auth_mode":"local"}`, `{"auth_mode":"pam","service":"login"}`	Tenant authentication settings.
`created_at`	string (date-time)	yes	n/a	`2026-03-02T12:00:00Z`	Tenant creation timestamp.
`description`	string, nullable	no	`null`	`FinTech department tenant`, `Engineering team workspace`	Tenant description.
`display_name`	string	yes	n/a	`FinTech`, `Engineering`	Human-readable tenant name.
`email`	string	yes	n/a	`admin@fintech.example.com`	Tenant contact email address.

Field	Type	Required	Default	Example	Description
`id`	string (uuid)	yes	n/a	`e97b94d6-9cc1-4325-a267-1064ae538ccd`	Tenant identifier (UUID).
`name`	string	yes	n/a	`fintech`, `eng`	Tenant name used for volume mounting (3-10 alphanumeric characters).
`status`	`TenantStatus` (`pending`, `configuring`, `active`, `archived`)	yes	n/a	`pending`, `configuring`, `active`	How far the tenant's network has been provisioned, as the realm manager reports it on every read: `pending` owns no subnet, `configuring` owns subnets that hold no node address yet, and `active` carries at least one node address.
`storageset_assignments`	array of `StorageSetAssignment` object	no	n/a		List of StorageSets assigned to the tenant, with per-set quotas and names.
`updated_at`	string (date-time)	yes	n/a	`2026-03-15T09:30:00Z`	Last update timestamp.

`TenantCreateRequest`

Request payload for creating a tenant.

Field	Type	Required	Default	Example	Description
`auth_settings`	`LocalAuthSettings` object or `PamAuthSettings` object	yes	n/a	`{"auth_mode":"local"}`, `{"auth_mode":"pam","service":"login"}`	Tenant authentication settings. Discriminated on `auth_mode`.
`description`	string, nullable	no	`null`	`FinTech department tenant`, `Engineering team workspace`	Tenant description.
`display_name`	string	yes	n/a	`FinTech`, `Engineering`, `Data Science`	Human-readable tenant name. Must be 1-255 printable characters.
`email`	string	yes	n/a	`admin@fintech.example.com`, `ops@vdura.example.com`	Tenant contact email address.

Field	Type	Required	Default	Example	Description
`name`	string	yes	n/a	`fintech`, `eng`, `datasci`	Tenant name, used for volume mounting and as the prefix of a tenant-qualified username. Must be 3-10 alphanumeric characters and unique within the system. Stored in lower case. `platform` is reserved. Immutable after creation.

`TenantCreateResponse`

Minimal response for successful tenant creation.

Returned on POST /tenants to signal successful creation. Clients should fetch full details via GET /tenants/{tenant_id} if needed.

Field	Type	Required	Default	Example	Description
`id`	string (uuid)	yes	n/a	`e97b94d6-9cc1-4325-a267-1064ae538ccd`	Unique tenant identifier (UUID) assigned by the system

Example

```
{
   "id" : "e97b94d6-9cc1-4325-a267-1064ae538ccd"
}
```

`TenantPatchRequest`

Request payload for updating a tenant via PATCH.

Allows partial updates. At least one field must be provided. Patchable fields: `display_name`, `email`, `description`, `auth_settings`. Only a Platform Admin can update a tenant; the tenant's own Admin role can view it but not change it. `name` is the immutable handle and is not patchable.

Field	Type	Required	Default	Example	Description
`auth_settings`	`LocalAuthSettings` object or `PamAuthSettings` object, nullable	no	`null`	`{"auth_mode":"local"}`, `{"auth_mode":"pam","service":"login"}`	Updated tenant authentication settings.
`description`	string, nullable	no	`null`	`Updated tenant description`, `New department focus`	Updated tenant description.
`display_name`	string, nullable	no	`null`	`FinTech Renamed`, `Engineering 2.0`	Updated human-readable tenant name. Must be 1-255 printable characters.
`email`	string, nullable	no	`null`	`admin@fintech.example.com`	Updated tenant contact email address.

`TenantRole`

One role of the tenant catalog: its ID, and the actions it allows on each resource.

Field	Type	Required	Default	Example	Description
`id`	string	yes	n/a	`admin`	Role identifier, as sent and received on the roles endpoints.
`permissions`	object	no	n/a		Resource to the actions this role may perform on it.
`reserved`	boolean	no	`false`	`false`	Whether the role is reserved: a reserved role cannot be assigned to or revoked from any principal through the API.

Example


```
{
   "id" : "admin",
   "permissions" : {
      "roles" : [
         "view",
         "update"
      ],
      "tenants" : [
         "view"
      ],
      "users" : [
         "list",
         "view",
         "create",
         "update",
         "delete"
      ],
      "volumes" : [
         "list",
         "view",
         "create",
         "update",
         "delete",
         "action"
      ]
   },
   "reserved" : false
}
```

`TenantRolesResponse`

Catalog of roles available on the current tenant.

Returned on GET /tenants/roles.

Field	Type	Required	Default	Example	Description
`roles`	array of `TenantRole` object	no	n/a		Array of roles the tenant policy defines. Empty array if the policy defines none.
`total`	integer	yes	n/a	`2`, `5`	Total count of roles in this response.

Example


```
{
   "roles" : [
      {
         "id" : "admin",
         "permissions" : {
            "roles" : [
               "view",
               "update"
            ],
            "tenants" : [
               "view"
            ],
            "users" : [
               "list",
               "view",
               "create",
               "update",
               "delete"
            ],
            "volumes" : [
               "list",
               "view",
               "create",
               "update",
               "delete",
               "action"
            ]
         },
         "reserved" : false
      },
      {
         "id" : "s3-gateway",
         "permissions" : {
            "fs-users" : [
               "read-s3-secret",
               "list"
            ]
```

```
        },
        "reserved" : true
      }
   ],
   "total" : 2
}
```

`TenantStatus`

Tenant lifecycle statuses.

States:

- `PENDING` : tenant owns no subnet
- `CONFIGURING` : the tenant owns subnets and no node has taken an address on any of them
- `ACTIVE` : at least one node holds an address on one of the tenant's subnets
- `ARCHIVED` : tenant deleted and its data removed, and only the archive entry remains

Allowed values: `pending` , `configuring` , `active` , `archived`

`TenantStorageSetQuotaRequest`

Request body for assigning or resizing a tenant storageset quota.

Field	Type	Required	Default	Example	Description
`quota_bytes`	integer	yes	n/a	`2199023255552` , `54975581388880`	Maximum storage quota in bytes that the tenant can consume within this storageset.

Example

```
{
   "quota_bytes" : 2199023255552
}
```

`TenantSummarySchema`

Short tenant info visible to platform admin.

The `display_name` and `email` are returned only by GET /tenants/{tenant_id}, not in this list view.

Deleted tenants are not here at all: they have their own entity and their own endpoint, `GET /tenants/archived`.

Field	Type	Required	Default	Example	Description
`created_at`	string (date-time)	yes	n/a	`2026-03-02T12:00:00Z`	Tenant creation timestamp.
`description`	string	yes	n/a	`FinTech tenant`	Free-form tenant description.
`id`	string (uuid)	yes	n/a	`e97b94d6-9cc1-4325-a267-1064ae538ccd`	Tenant identifier (UUID).
`name`	string	yes	n/a	`fintech`, `eng`	Tenant name used for volume mounting.
`status`	`TenantStatus` (`pending`, `configuring`, `active`, `archived`)	yes	n/a	`pending`, `configuring`, `active`	How far the tenant's network has been provisioned, as the realm manager reports it on every read: `pending` owns no subnet, `configuring` owns subnets that hold no node address yet, and `active` carries at least one node address.
`updated_at`	string (date-time)	yes	n/a	`2026-03-02T12:00:00Z`	Tenant last-modification timestamp.

Example


```
{
   "created_at" : "2026-03-02T12:00:00Z",
   "description" : "FinTech tenant",
   "id" : "e97b94d6-9cc1-4325-a267-1064ae538ccd",
   "name" : "fintech",
   "status" : "active",
   "updated_at" : "2026-03-02T12:00:00Z"
}
```


`TenantSwitchRequest`

Platform Admin request to exchange the current access token for a target-tenant one.

Field	Type	Required	Default	Example	Description
`tenant_id`	string (uuid)	yes	n/a	`e97b94d6-9cc1-4325-a267-1064ae538ccd`	Target tenant to act in.
`ttl_minutes`	integer, nullable	no	`null`	`60` , `480`	Requested lifetime in minutes. Clamped server-side to the supported range (30 minutes - 12 hours); the granted lifetime is returned in `expires_in` .

Example


```
{
   "tenant_id" : "e97b94d6-9cc1-4325-a267-1064ae538ccd",
   "ttl_minutes" : 60
}
```

`TokenResponse`

Issued token pair returned by login and refresh endpoints.

Field	Type	Required	Default	Example	Description
`access_token`	string	yes	n/a		Short-lived JWT access token.
`expires_in`	integer	yes	n/a	`3600`	Access token lifetime in seconds (60 minutes by default).
`refresh_token`	string	yes	n/a		Long-lived JWT refresh token (7 days by default).
`token_type`	literal `bearer`	no	`bearer`		

`User`

Core Tenant User data model.

Represents a user account scoped to a single tenant. Used as the inner model in response wrappers (GetUserResponse, UserListResponse).

Field	Type	Required	Default	Example	Description
`auth_settings`	`LocalAuthSettings` object or `PamAuthSettings` object	yes	n/a	`{"auth_mode":"local"}`, `{"auth_mode":"pam","service":"login"}`	Authentication backend for this user, discriminated on `auth_mode`.
`created_at`	string (date-time)	yes	n/a		When the user was created.
`created_by`	string (uuid), nullable	no	`null`		The user that created it. Null for the built-in admin, which the bootstrap seeds.
`display_name`	string, nullable	no	`null`	`Alice Anderson`	Human-readable full name. Optional; `username` remains the unique handle.
`email`	string, nullable	no	`null`	`alice@fintech.example.com`	Contact email for the Tenant User, or null if none is on file.
`id`	string (uuid)	yes	n/a	`a1b2c3d4-e5f6-7890-abcd-ef1234567890`	Tenant User identifier (UUID).
`updated_at`	string (date-time)	yes	n/a		When the user was last changed.

Field	Type	Required	Default	Example	Description
`username`	string	yes	n/a	`alice`	Tenant User username.

Example


```
{
   "auth_settings" : {
      "auth_mode" : "local"
   },
   "created_at" : "2026-01-01T00:00:00Z",
   "display_name" : "Alice Anderson",
   "email" : "alice@fintech.example.com",
   "id" : "a1b2c3d4-e5f6-7890-abcd-ef1234567890",
   "updated_at" : "2026-01-01T00:00:00Z",
   "username" : "alice"
}
```

`UserCreateRequest`

Request payload for creating a Tenant User.

The password is hashed server-side before storage; the plaintext value is never persisted. The name must be unique within the tenant.

Field	Type	Required	Default	Example	Description
`auth_settings`	`LocalAuthSettings` object or `PamAuthSettings` object	yes	n/a	`{"auth_mode":"local"}`, `{"auth_mode":"pam","service":"login"}`	Authentication backend for this user, discriminated on `auth_mode`.
`display_name`	string, nullable	no	`null`	`Alice Anderson`	Human-readable full name. Optional; `username` remains the unique handle.
`email`	string, nullable	no	`null`	`alice@fintech.example.com`	Optional contact email for the Tenant User.
`password`	string (password)	yes	n/a		Tenant User password (min 8 characters). Hashed server-side before storage.
`username`	string	yes	n/a	`alice`	Tenant User username. Must be unique within the tenant. Whitespace, control, and invisible characters are not allowed.

Example


```
{
   "auth_settings" : {
      "auth_mode" : "local"
   },
   "display_name" : "Alice Anderson",
   "email" : "alice@fintech.example.com",
   "password" : "secure-password",
   "username" : "alice"
}
```


`UserCreateResponse`

Minimal response for successful Tenant User creation.

Returned on POST /tenants/{tenant_id}/users. Clients should fetch full details via GET /tenants/{tenant_id}/users/{user_id} if needed.

Field	Type	Required	Default	Example	Description
`id`	string (uuid)	yes	n/a	`a1b2c3d4-e5f6-7890-abcd-ef1234567890`	Unique Tenant User identifier (UUID) assigned by the system.

Example


```
{
   "id" : "a1b2c3d4-e5f6-7890-abcd-ef1234567890"
}
```

`UserListResponse`

Collection response for Tenant User list operations.

Returned on GET /tenants/{tenant_id}/users.

Field	Type	Required	Default	Example	Description
`total`	integer	yes	n/a	`0`, `5`, `50`	Total count of Tenant Users in this response.
`users`	array of `User` object	no	n/a		Array of Tenant User objects. Empty array if no users found.

Example


```
{
   "total" : 1,
   "users" : [
      {
         "auth_settings" : {
            "auth_mode" : "local"
         },
         "created_at" : "2026-01-01T00:00:00Z",
         "display_name" : "Alice Anderson",
         "email" : "alice@fintech.example.com",
         "id" : "a1b2c3d4-e5f6-7890-abcd-ef1234567890",
         "updated_at" : "2026-01-01T00:00:00Z",
         "username" : "alice"
      }
   ]
}
```

`UserPatchRequest`

Request payload for updating a Tenant User.

Allows partial updates of `new_password`, `auth_settings`, `email` and/or `display_name`; at least one must be supplied. Administrative operation - the caller does not need to supply the current password. Intended for Tenant Admin or Platform Admin use only.

Field	Type	Required	Default	Example	Description
`auth_settings`	`LocalAuthSettings` object or `PamAuthSettings` object, nullable	no	`null`	`{"auth_mode":"local"}`, `{"auth_mode":"pam","service":"login"}`	Updated authentication backend for this user, discriminated on `auth_mode`.
`display_name`	string, nullable	no	`null`	`Alice Anderson`	Updated human-readable full name for the Tenant User.
`email`	string, nullable	no	`null`	`alice@fintech.example.com`	Updated contact email for the Tenant User.
`new_password`	string (password), nullable	no	`null`		New password for the Tenant User (min 8 characters). Hashed server-side before storage.

`ValidationError`

Field	Type	Required	Default	Example	Description
`ctx`	object	no	n/a		
`input`	any	no	n/a		
`loc`	array of string or integer	yes	n/a		
`msg`	string	yes	n/a		
`type`	string	yes	n/a		

`VictoriaLogsLogsqlEndpoint`

Allowed path segments under VictoriaLogs `/select/logsql/` (HTTP API).

Allowed values: `query` , `tail` , `hits` , `facets` , `stats_query` , `stats_query_range` , `stream_ids` , `streams` , `stream_field_names` , `stream_field_values` , `field_names` , `field_values`

`Volume`

Storage volume representation.

A volume is a logical storage container within a StorageSet that provides isolated namespace, quota management, and configurable data protection. Volumes can be exported via NFS and support features like snapshots, quotas, and recovery operations.

Field	Type	Required	Default	Example	Description
`efsa_mode`	`EfsaMode` (`retry` , `file-unavailable` , `default`)	yes	n/a	`retry` , `file-unavailable` , `default`	Current Extended File System Availability (EFSA) mode. Defines volume behavior during partial storage unavailability.
`hard_quota_bytes`	integer, nullable	no	`null`	`549755813888` , `1099511627776`	Configured hard quota limit in bytes. null indicates that no hard limit is enforced.
`id`	integer	yes	n/a	`1` , `2` , `42`	Unique volume identifier within the system
`max_width`	integer, nullable	no	`null`	`12` , `20`	Configured maximum number of VPODs used for striping data in the volume.
`object_id`	string	yes	n/a	`I-xD0200000000000005-xG82c418af-xU00004a75726a656e`	Root object ID as compound identifier string
`pathname`	string	yes	n/a	`/` , `/home` , `/data`	Unique volume pathname within the system.

Field	Type	Required	Default	Example	Description
`quotas_policy`	`QuotasPolicy` (`disable`, `enable`, `report`, `inherit`, `invalid`)	yes	n/a	`disable`, `report`, `enable`	User and Group Quotas Policy status indicating whether quota enforcement is active
`raid_type`	string	yes	n/a	`raid6+`, `raid5+`, `raid10+`	Configured RAID layout of the volume.
`recovery_priority`	integer	no	`50`	`0`, `1`, `50`, `100`	Recovery priority of the volume (1-100). Lower values indicate higher recovery priority.
`rg_depth`	integer, nullable	no	`null`	`4`, `16`	Configured RAID group depth for the volume.
`rg_width`	integer, nullable	no	`null`	`8`, `12`	Configured RAID group width.
`soft_quota_bytes`	integer, nullable	no	`null`	`524288000000`, `989560465024`	Configured soft quota limit in bytes. null indicates that no soft limit is enforced.

Field	Type	Required	Default	Example	Description
`status`	`VolumeStatus` (`none`, `creating`, `deleting`, `online`, `fsrc`, `offline`, `down`, `degraded`, `recon_available`, `snapping`, `recon_unavailable`, `paused`)	no	`none`		Current operational status: ONLINE, DEGRADED, OFFLINE, etc.
`storageset_id`	integer	yes	n/a	`1`, `2`, `5`	Identifier of the StorageSet where this volume resides.
`stripe_unit_bytes`	integer, nullable	no	`null`	`262144`, `1048576`	Configured stripe unit size. Specifies the number of bytes written to a VPOD before advancing to the next VPOD in the stripe.
`tenant_id`	string (uuid)	yes	n/a	`550e8400-e29b-41d4-a716-446655440000`, `00000000-0000-0000-0000-000000000000`	UUID of the owning tenant. System/backend-created volumes with no tenant prefix (e.g. "/", "/home") are owned by the default tenant (all-zeros / PAN_UUID_ZERO).

Field	Type	Required	Default	Example	Description
`volume_service_id`	string	yes	n/a	`0x0400000000000005`, `0x0400000000000006`	Hexadecimal identifier of the volume service currently managing this volume.

Example


```
{
   "efsa_mode" : "retry",
   "hard_quota_bytes" : 549755813888,
   "id" : 1,
   "max_width" : 12,
   "object_id" : "I-xD0200000000000005-xG82c418af-xU00004a75726a656e",
   "pathname" : "/",
   "quotas_policy" : "enable",
   "raid_type" : "raid6+",
   "recovery_priority" : 1,
   "rg_depth" : 4,
   "rg_width" : 8,
   "soft_quota_bytes" : 524288000000,
   "status" : "online",
   "storageset_id" : 1,
   "stripe_unit_bytes" : 262144,
   "tenant_id" : "00000000-0000-0000-0000-000000000001",
   "volume_service_id" : "0x0400000000000005"
}
```

`VolumeAccessMode`

Volume access permission modes.

Used by the 'volume set' CLI command to control read/write access to the volume.

Modes:

- `read-only` : volume is read-only for administrative purposes
- `read-write` : volume is readable and writable (default)

Allowed values: `read-only` , `read-write`

`VolumeCompressionMode`

Volume compression mode.

Controls whether data compression is enabled for the volume.

Modes:

- `enable` : compression is enabled for new data
- `disable` : compression is disabled

Allowed values: `enable` , `disable`

`VolumeCreateRequest`

Request payload for creating a volume.

Mirrors the CLI “volume create” command. Requires a volume name and a StorageSet ID. Optional parameters use system defaults when omitted.

Field	Type	Required	Default	Example	Description
`efsa_mode`	`EfsaMode` (`retry` , `file-unavailable` , `default`)	no	`default`	`retry` , `file-unavailable` , `default`	EFSA mode for the volume. Defines behavior during partial storage unavailability. Supported: retry, file-unavailable, default. If not specified, system default (retry) is applied. Only for RAID 5+, 6+, 10+ volumes.
`group`	string, nullable	no	`null`	`0` , `100`	Owner GID of the volume, as a numeric string (e.g. “0”, “100”). Defaults to root group (gid=0) if not specified. String name resolution and Windows SID are not supported.
`group_permissions`	`VolumePermission` (`none` , `read-only` , `write-only` , `execute-only` , `read-write` , `read-execute` , `write-execute` , `all`), nullable	no	`null`	`read-only` , `read-execute`	Filesystem permissions for the owner group on the root of the volume. Allowed values: none, read-only, write-only, execute-only, read-write, read-execute, write-execute, all.

Field	Type	Required	Default	Example	Description
`hard_quota_bytes`	integer, nullable	no	`null`	`54975581388` , `null`	Hard quota in bytes. Prevents additional writes when limit is reached. A volume in a tenant requires a value greater than 0; a request without one returns 400.
`max_width`	integer, nullable	no	`null`	`12` , `20`	Maximum number of VPODs to stripe data across. Minimum depends on RAID layout: 2 for RAID 5+, 3 for RAID 6+, 2 for RAID 10+. If not specified, the system allows striping across the StorageSet.
`other_permissions`	`VolumePermission` (`none` , `read-only` , `write-only` , `execute-only` , `read-write` , `read-execute` , `write-execute` , `all`), nullable	no	`null`	`none` , `read-only`	Filesystem permissions for others on the root of the volume. Allowed values: none, read-only, write-only, execute-only, read-write, read-execute, write-execute, all.
`pathname`	string	yes	n/a	`/vol2` , `/data`	Required unique mount path of the volume. Must start with / and be unique within the system.

Field	Type	Required	Default	Example	Description
`raid_type`	`VolumeLayout` (`raid6+`, `raid5+`, `raid10+`, `raid5`, `raid10`)	no	`raid6+`	`raid6+`, `raid5+`, `raid10+`, `raid5`, `raid10`	RAID layout for the volume. Default: raid6+. Cannot be changed after creation.
`recovery_priority`	integer	no	`50`	`0`, `1`, `50`, `100`	Recovery priority (1-100) of the volume. Lower values indicate higher recovery priority, where 1 represents the highest priority. If not specified the system default (50) is applied.
`rg_depth`	integer, nullable	no	`null`	`4`, `16`	RAID group depth (number of stripes per RAID parity group). Only for RAID 5+, 6+. If not specified, a system default value is applied.
`rg_width`	integer, nullable	no	`null`	`8`, `12`	RAID group width (VPODs to stripe over per single RAID group, 3-20). Only for RAID 5+, 6+. If not specified, a system default value is applied.
`soft_quota_bytes`	integer, nullable	no	`null`	`549755813888`, `null`	Soft quota limit in bytes. Triggers quota notifications when the threshold is exceeded. Use 0 or null for unlimited.
`storageset_id`	integer	yes	n/a	`1`, `2`, `5`	Identifier of the StorageSet where the volume will be created. Required if multiple StorageSets exist.

Field	Type	Required	Default	Example	Description
`stripe_unit_bytes`	integer, nullable	no	`null`	`262144`, `1048576`	Specifies the number of bytes written to a VPOD before advancing to the next VPOD in the stripe. Must be a multiple of 16 KB (min: 16 KB, max: 4 MB). If not specified, a system default value is applied.
`user`	string, nullable	no	`null`	`0`, `1001`	Owner UID of the volume, as a numeric string (e.g. “0”, “1001”). Defaults to root (uid=0) if not specified. String name resolution and Windows SID are not supported.
`user_permissions`	`VolumePermission` (`none`, `read-only`, `write-only`, `execute-only`, `read-write`, `read-execute`, `write-execute`, `all`), nullable	no	`null`	`read-write`, `all`	Filesystem permissions for the owner user on the root of the volume. Allowed values: none, read-only, write-only, execute-only, read-write, read-execute, write-execute, all.

Field	Type	Required	Default	Example	Description
`volume_service_id`	string, nullable	no	`null`	`0x0400000000000005`	Identifier of the volume service that will manage this volume, in hexadecimal format (e.g., "0x0400000000000005"). This allows control over how volume management load is distributed across Directors. If not specified, the system automatically assigns the volume to an existing service.

Example

```
{
   "efsa_mode" : "retry",
   "pathname" : "/vol2",
   "raid_type" : "raid6+",
   "recovery_priority" : 50,
   "storageset_id" : 1
}
```

`VolumeCreateResponse`

Minimal response for successful volume creation.

Returned on POST /volumes to signal successful creation without the full volume detail payload. Clients should fetch full details via GET /volumes/{volume_id} if needed.

Field	Type	Required	Default	Example	Description
`id`	integer	yes	n/a	`1`, `2`, `42`	Unique volume identifier assigned by RM

Example

```
{
   "id" : 1
}
```

`VolumeGetResponse`

Detailed representation of a single volume.

Field	Type	Required	Default	Example	Description
`volume`	`Volume` object	yes	n/a		Volume entity with all available details.

Example


```
{
   "volume" : {
      "efsa_mode" : "retry",
      "hard_quota_bytes" : 549755813888,
      "id" : 1,
      "max_width" : 12,
      "object_id" : "I-xD0200000000000005-xG82c418af-xU00004a75726a656e",
      "pathname" : "/",
      "quotas_policy" : "disable",
      "raid_type" : "raid6+",
      "recovery_priority" : 1,
      "rg_depth" : 4,
      "rg_width" : 8,
      "soft_quota_bytes" : 524288000000,
      "status" : "online",
      "storageset_id" : 1,
      "stripe_unit_bytes" : 262144,
      "tenant_id" : "00000000-0000-0000-0000-000000000001",
      "volume_service_id" : "0x0400000000000005"
   }
}
```

`VolumeLayout`

Volume layout options derived from the CLI “volume create” command.

Layouts:

- `raid6+` : RAID 6+ with dual parity and enhanced metadata redundancy (default)
- `raid5+` : RAID 5+ with single parity and enhanced metadata redundancy
- `raid10+` : RAID 10+ with mirroring and enhanced metadata redundancy
- `raid5` : Legacy RAID 5 layout (service mode only)
- `raid10` : Legacy RAID 10 layout (service mode only)

Allowed values: `raid6+` , `raid5+` , `raid10+` , `raid5` , `raid10`

`VolumeListResponse`

Represents a collection of volume objects along with the total number of available objects.

Field	Type	Required	Default	Example	Description
`total`	integer	yes	n/a	`0`, `2`, `150`	Total count of volumes available (may exceed returned array size if paginated).
`volumes`	array of `Volume` object	no	n/a		Array of volume objects matching the query. Empty array if no volumes found

Example


```
{
   "total" : 2,
   "volumes" : [
      {
         "efsa_mode" : "retry",
         "hard_quota_bytes" : 549755813888,
         "id" : 1,
         "object_id" : "I-xD0200000000000005-xG82c418af-xU00004a75726a656e",
         "pathname" : "/",
         "quotas_policy" : "enable",
         "raid_type" : "raid6+",
         "recovery_priority" : 1,
         "soft_quota_bytes" : 524288000000,
         "status" : "online",
         "storageset_id" : 1,
         "tenant_id" : "00000000-0000-0000-0000-000000000001",
         "volume_service_id" : "0x0400000000000005"
      },
      {
         "efsa_mode" : "retry",
         "id" : 2,
         "object_id" : "I-xD0200000000000006-xGc4f7a724-xU00004a75726a656e",
         "pathname" : "/home",
         "quotas_policy" : "disable",
         "raid_type" : "raid6+",
         "recovery_priority" : 50,
         "status" : "online",
         "storageset_id" : 1,
         "tenant_id" : "e97b94d6-9cc1-4325-a267-1064ae538ccd",
         "volume_service_id" : "0x0400000000000006"
      }
   ]
}
```

`VolumePatchRequest`

Request payload for updating an existing volume via PATCH.

Field	Type	Required	Default	Example	Description
`access_permission`	`VolumeAccessMode` (`read-only`, `read-write`), nullable	no	`null`	`read-only`, `read-write`	Volume access permission mode. Set to read-only for administrative purposes
`compression`	`VolumeCompressionMode` (`enable`, `disable`), nullable	no	`null`	`enable`, `disable`	Enable or disable compression for the volume. Affects new data only
`efsa_mode`	`EfsaMode` (`retry`, `file-unavailable`, `default`), nullable	no	`null`	`retry`, `file-unavailable`, `default`	EFSA mode for the volume. Defines behavior during partial storage unavailability. Supported: retry, file-unavailable, default. If not specified, system default (retry) is applied. Only for RAID 5+, 6+, 10+ volumes.
`hard_quota_bytes`	integer, nullable	no	`null`	`1099511627776`, `0`	Hard quota in bytes. Prevents additional writes when limit is reached. A volume in a tenant requires a value greater than 0; a request without one returns 400.

Field	Type	Required	Default	Example	Description
`recovery_priority`	integer, nullable	no	`null`	`0`, `1`, `50`, `100`	Recovery priority (1-100) of the volume. Lower values indicate higher recovery priority, where 1 represents the highest priority. If not specified the system default (50) is applied.
`soft_quota_bytes`	integer, nullable	no	`null`	`549755813888`, `0`	Soft quota limit in bytes. Triggers quota notifications when the threshold is exceeded. Use 0 for unlimited.
`state`	`VolumeStateOption` (`online`, `offline`, `paused`), nullable	no	`null`	`online`, `offline`, `paused`	Target operational state for the volume. Use to bring volume online/offline

Example


```
{
   "access_permission" : "read-write",
   "efsa_mode" : "file-unavailable",
   "hard_quota_bytes" : 1099511627776,
   "recovery_priority" : 75,
   "soft_quota_bytes" : 549755813888,
   "state" : "online"
}
```

`VolumePermission`

File permission presets for volume ownership configuration.

Permissions:

- `none` : no permissions (000)
- `read-only` : read-only access (r–)
- `write-only` : write-only access (-w-)
- `execute-only` : execute-only access (–x)
- `read-write` : read and write access (rw-)
- `read-execute` : read and execute access (r-x)
- `write-execute` : write and execute access (-wx)
- `all` : full permissions (rwx)

Allowed values: `none` , `read-only` , `write-only` , `execute-only` , `read-write` , `read-execute` , `write-execute` , `all`

`VolumeRecoverActionParams`

Parameters required to start File System Recovery on a volume.

Field	Type	Required	Default	Example	Description
`ignore_eligibility`	boolean	no	`false`		When true, skip the volume eligibility precondition (PAN_VOLUME_F_FSRC_RECOMMENDED or PAN_VOLUME_F_FSRC_FAILED).
`mode`	`VolumeRecoverMode` (`recover` , `parity_scrub` , `no_scrub` , `check_only`)	no	`recover`	`recover` , `parity_scrub` , `no_scrub` , `check_only`	File System Recovery mode controlling the underlying RPC flag bitmask.

`VolumeRecoverActionRequest`

Request to start File System Recovery on a volume.

Field	Type	Required	Default	Example	Description
`action`	literal `recover`	yes	n/a		The action to perform on the volume.
`params`	`VolumeRecoverActionParams` object	yes	n/a		Parameters required to perform the recover action.

`VolumeRecoverMode`

Modes available when starting File System Recovery on a volume.

Allowed values: `recover`, `parity_scrub`, `no_scrub`, `check_only`

`VolumeRenameActionParams`

Parameters required to perform a rename action on a volume.

Field	Type	Required	Default	Example	Description
`new_pathname`	string	yes	n/a	`/new_volume_name`, `/data_archive`	The new unique mount path for the volume. Must start with / and be unique within the system.

`VolumeRenameActionRequest`

Request to perform a rename action on a Volume.

Field	Type	Required	Default	Example	Description
`action`	literal `rename`	yes	n/a		The action to perform on the volume.
`params`	`VolumeRenameActionParams` object	yes	n/a		Parameters required to perform the rename action.

`VolumeSpaceResponse`

Space usage response for a volume.

A volume's capacity is bounded by its hard quota. Free space is the remaining capacity within that quota: `hard_quota_bytes - used_bytes`.

Field	Type	Required	Default	Example	Description
`free_bytes`	integer	yes	n/a	`107374182400`, `0`	Remaining capacity within the hard quota (hard_quota_bytes - used_bytes).
`hard_quota_bytes`	integer, nullable	yes	n/a	`214748364800`, `549755813888`	Hard quota limit in bytes. This is the total capacity allocated to the volume.
`soft_quota_bytes`	integer, nullable	no	`null`	`193273528320`, `null`	Soft quota limit in bytes. Advisory threshold that triggers warnings. Null if not configured.
`used_bytes`	integer	yes	n/a	`107374182400`, `0`	Bytes consumed by this volume on disk.

Example

```
{
   "free_bytes" : 107374182400,
   "hard_quota_bytes" : 214748364800,
   "soft_quota_bytes" : 193273528320,
   "used_bytes" : 107374182400
}
```

`VolumeStateOption`

Volume state options for administrative state transitions.

Used by the 'volume state' CLI command to move volumes between operational states.

States:

- `online` : bring volume online and serve I/O
- `offline` : take volume offline (clients block and retry)
- `paused` : take volume offline with I/O errors (service mode only)

Allowed values: `online` , `offline` , `paused`

`VolumeStatus`

Human-readable volume status.

Mirrors `pan_volume_state` from the proto definition and exposes descriptive string values in the REST API instead of raw integers.

States:

- `NONE` : state is unknown or not yet initialized
- `CREATING` : volume is being created and not fully available
- `DELETING` : volume is being deleted and may be partially unavailable
- `ONLINE` : volume is fully online and serving I/O
- `FSRC` : volume is undergoing FSRC (file system recovery/check) operations
- `OFFLINE` : volume is intentionally offline and not serving I/O
- `DOWN` : volume is down due to failures and not accessible
- `DEGRADED` : volume is online but running with reduced redundancy
- `RECON_AVAILABLE` : reconstruction is in progress but volume is still available
- `SNAPPING` : volume snapshot operation is in progress
- `RECON_UNAVAILABLE` : reconstruction is in progress and volume is unavailable
- `PAUSED` : volume activity is administratively paused

Allowed values: `none` , `creating` , `deleting` , `online` , `fsrc` , `offline` , `down` , `degraded` , `recon_available` , `snapping` , `recon_unavailable` , `paused`