STEELDOME

StratiSYSTEM API SDK Guide

Developer reference for the current StratiSYSTEM REST API

Revision	August 2026
API base path	/api/v1
Current route inventory	203 unique HTTP method/path combinations

This guide is for developers building a client, CLI, UI, automation workflow, or service integration that consumes the current StratiSYSTEM API. **It is implementation-oriented and reflects the supplied August 2026 API source.**

What changed in this revision

- Corrected authentication, privilege-claim, refresh-token, timeout, and RBAC behavior to match the current service.
- Added asynchronous module job submission, polling, cancellation, retention, and retry guidance.
- Added the complete local account, Active Directory, Cockpit AD, network, direct VM backup, direct replication, and standalone VMware import APIs.
- Added current StratiSERV coverage for VM network interfaces, managed VM jobs, managed shared-disk jobs, host HA-affinity groups, and managed VMware import jobs.
- Expanded the complete route catalog to 203 HTTP method/path combinations and the built-in RBAC catalog to 66 permissions.
- Added online shared-filesystem capacity eligibility and expansion coverage, and normalized endpoint formatting, canonical request field names, examples, status-code guidance, and destructive-operation cautions.

Intentional exclusion: Cockpit 2FA enrollment, provider configuration, one-time-code handling, QR/TOTP secret generation, and PAM activation are not exposed through REST. These remain controlled local administrative workflows.

STEELDOME

Table of Contents

Section	Topic	Page
1	Scope and API architecture	3
2	Base URL, TLS, and health	4
3	Authentication and token lifecycle	5
4	Request, response, and retry conventions	7
5	Asynchronous module jobs	8
6	RBAC, IAM, and audit	9
7	Local accounts, Active Directory, and Cockpit AD	11
8	Multi-node network API	15
9	Direct VM backup API	18
10	Direct VM replication API	19
11	Standalone VMware import API	20
12	Newly covered StratiSERV operations	21
13	SMB API	22
A	Complete endpoint catalog	25
B	Built-in permission catalog	32
C	SDK reference behavior	37
D	Corrections and consistency notes	39

Appendix A is the authoritative method/path inventory for the supplied API source. Detailed schemas for standalone module wrappers and newly covered StratiSERV operations appear in Sections 7-12.

1. Scope and API architecture

The StratiSYSTEM API is a TLS-protected, bearer-token REST service that wraps the StratiSTOR and StratiSERV command handlers and the standalone account, network, VM backup, replication, and VMware import modules. The API performs authentication and fine-grained authorization before invoking a fixed backend command mapping; it does not provide a generic shell-execution endpoint.

1.1 Backend locations

System role	Backend
StratiSTOR	/usr/share/cockpit/stratistor/handler/stratistorhandler
StratiSERV	/usr/share/cockpit/stratiserv/handler/vmchandler
Either role	/<stratistor|stratiserv>/modules/accounts/stratisystem-accounts
Either role	/<stratistor|stratiserv>/modules/network/stratisystem-network
Either role	/<stratistor|stratiserv>/modules/backup/vmbackup
Either role	/<stratistor|stratiserv>/modules/replicatevm/replicatevm
Either role	/<stratistor|stratiserv>/modules/vmware/vmwareimport

System-role availability: The router registers the complete API surface. A request still fails when the required backend module is not installed or is not applicable to the node role. SDKs should treat this as a capability/deployment error rather than assuming every operation is meaningful on every host.

1.2 API schema ownership

The stratisystem-api service owns and initializes its API users, groups, permissions, memberships, session fields, and API audit schema when the service starts. Upgrade tooling should not duplicate that API-schema responsibility.

2. Base URL, TLS, and health

The default listener is HTTPS on TCP port 3201. The runtime port is configurable, so SDKs should accept the base URL as configuration rather than hard-coding it.

```
API="https://192.168.59.101:3201"
curl -sS -k "$API/healthz" | jq
```

GET /healthz

This public endpoint returns HTTP 200 with {"status":"ok"} when the web service is responding. It does not validate database, backend-module, or cluster health.

2.1 TLS guidance

- The service enforces TLS 1.2 or newer.
- A self-signed certificate may be generated by the service. Use -k only in controlled test environments.
- Production clients should trust the API certificate or issuing CA and validate the expected hostname.
- Do not log authorization headers, refresh tokens, passwords, AD credentials, or import credentials.

3. Authentication and token lifecycle

The current API issues a JWT access token and a rotated opaque refresh token. Both login paths are supported and return the same response shape.

POST /login

POST /api/v1/auth/login

```
curl -sS -k -X POST "$API/api/v1/auth/login" \
  -H "Content-Type: application/json" \
  -d '{"username":"sdk-user","password":"<secret>"}' | jq
```

3.1 Login request and response

Field	Type	Required / behavior
username	`string`	Required. Leading/trailing whitespace is removed.
password	`string`	Required. Never persist or log in plaintext.

```
{
  "token": "<access JWT>",
  "refresh_token": "<rotated refresh token>",
  "privilege": "user|admin",
  "token_type": "Bearer",
  "expires_in": 3600,
  "refresh_expires_in": 604800,
  "idle_timeout_seconds": 900
}
```

Correction from the prior guide: The current implementation always returns refresh_token on successful login. The defaults are 1 hour for access tokens, 7 days for refresh tokens, and 15 minutes for idle-session expiration; all are configurable.

Privilege claim: The current implementation returns user for members of stratisystem-apiuser and admin for members of either stratisystem-apiadmin or stratisystem-apisuperadmin. Superadministrator status is represented by group membership and permissions, not by a distinct superadmin token claim.

3.2 Authenticated requests

```
TOKEN="..."
curl -sS -k "$API/api/v1/getstratistorinfo" \
  -H "Authorization: Bearer $TOKEN" | jq
```

Send the access token as Authorization: Bearer <token>. Refresh tokens are sent only in the refresh request body and must never be used as bearer tokens.

3.3 Refresh

POST /api/v1/auth/refresh

```
curl -sS -k -X POST "$API/api/v1/auth/refresh" \
  -H "Content-Type: application/json" \
  -d '{"refresh_token":"<refresh token>"}' | jq
```

- Replace both the access token and refresh token after every successful refresh.
- Retry the original request once after refresh succeeds.
- When refresh fails, clear both tokens and require a new login.
- Serialize refresh attempts so multiple concurrent 401 responses do not race and invalidate one another.

3.4 Logout

POST /api/v1/auth/logout

Permission: `api.auth.logout`

```
curl -sS -k -X POST "$API/api/v1/auth/logout" \
  -H "Authorization: Bearer $TOKEN" | jq
```

Logout clears the active access-token binding and refresh-token state for the user session, invalidates the in-memory token cache, and returns {"message":"Logged out"}. Clear local credentials even if the network response is lost after the server processes the request.

3.5 401 and 403 behavior

Status	Meaning	SDK action
401 Unauthorized	`Missing, expired, invalid, revoked, idle-expired, or authorization-version-mismatched token.`	Refresh once; if refresh fails, require login.
403 Forbidden	`The access token is valid, but the effective permission is missing.`	Do not retry. Surface the permission field.

```
{
  "error": "Insufficient permission",
  "permission": "network.write"
}
```

4. Request, response, and retry conventions

- **Content type:** Send application/json for JSON request bodies.
- **Canonical fields:** Use the field names documented in this guide. Compatibility aliases exist on several legacy endpoints but should not be emitted by new SDKs.
- **Strict module wrappers:** The account, network, direct VM backup, direct replication, and standalone VMware import endpoints reject unknown JSON fields. Their request body limit is 2 MiB.
- **Backend JSON:** Most handlers return the backend JSON document directly. A backend success:false or non-empty error normally becomes HTTP 400.
- **Secrets:** Redact credentials and tokens from logs, traces, crash reports, and job metadata.

4.1 Status codes

Status	Typical meaning	Automatic retry?
200 OK	`Synchronous operation completed; parse JSON.`	No retry needed.
202 Accepted	`Asynchronous operation queued; poll the returned job ID.`	Poll, do not resubmit.
400 Bad Request	`Invalid body, unsupported action, backend validation failure, or backend command failure.`	No. Correct the request or operator state.
401 Unauthorized	`Token/session failure.`	Refresh once, then login.
403 Forbidden	`Missing effective permission.`	No.
404 Not Found	`Route/entity/job not found or hidden by ownership rules.`	No, unless discovery confirms a transient entity.
408 Request Timeout	`Synchronous backend command was canceled.`	Only when the operation is known idempotent.
409 Conflict	`Job cannot be canceled because it is no longer running.`	No. Read the job.
500 Internal Server Error	`Service/database/backend integration error.`	Only safe, idempotent reads.
504 Gateway Timeout	`Synchronous backend command exceeded its timeout.`	Read state before deciding whether to retry.

4.2 Idempotency

GET requests and precheck/list/read operations are generally safe to repeat. Create, update, delete, import, restore, replication, backup, cluster restart, network apply, and service-control operations must not be retried automatically unless the SDK has verified idempotency for that specific workflow. A lost 202 response is especially important: do not blindly resubmit a long-running operation.

5. Asynchronous module jobs

Long-running standalone-module operations default to asynchronous execution. The initial response is HTTP 202 and includes an in-memory job record.

```
{
  "success": true,
  "job": {
    "job_id": "0c4600f7c6a143c8a6d428eb535a6430",
    "owner": "sdk-user",
    "kind": "replicatevm.run",
    "state": "queued",
    "created_at": "2026-07-13T20:10:00Z",
    "exit_code": -1
  }
}
```

GET /api/v1/jobs/:id

DELETE /api/v1/jobs/:id

- States are queued, running, succeeded, failed, or canceled.
- The job owner and users whose token privilege is admin can read or cancel the job.
- A completed job contains result, error, exit_code, started_at, and finished_at as applicable.
- Finished jobs are retained in memory for 24 hours; the store is capped at 1,000 records.
- Jobs and job history do not survive an API service restart.
- Canceling or timing out a job terminates the module process group and child processes.

```
JOB_ID="0c4600f7c6a143c8a6d428eb535a6430"
curl -sS -k "$API/api/v1/jobs/$JOB_ID" \
  -H "Authorization: Bearer $TOKEN" | jq

curl -sS -k -X DELETE "$API/api/v1/jobs/$JOB_ID" \
  -H "Authorization: Bearer $TOKEN" | jq
```

5.1 Default asynchronous behavior

Operation	Default
Local-account list	Synchronous
Local-account mutations and ensure	Asynchronous
AD join	Asynchronous
Cockpit AD list-groups / check-user	Synchronous
Other Cockpit AD actions	Asynchronous
Network precheck	Synchronous
Network apply	Asynchronous
VM backup plan / list / snapshot-sanity	Synchronous
VM backup backup / cleanup / restore / verify / prune / repair	Asynchronous
Direct replication run	Asynchronous

Operation	Default
Standalone VMware import	Asynchronous

Where documented, set async explicitly to override the default. Prefer asynchronous mode for operations that can exceed the HTTP write timeout.

6. RBAC, IAM, and audit

Authorization is based on effective permissions inherited through API group membership. The privilege claim is only a broad authentication tier; it is not sufficient by itself to authorize a route.

6.1 Built-in groups

Group	Purpose
stratisystem-apiuser	Default read/info and approved user operations.
stratisystem-apiadmin	Administrative configuration and management operations.
stratisystem-apisuperadmin	IAM governance and all active permissions.

Session invalidation: User, password, membership, group-permission, and permission changes can increment authorization state or clear session data. Clients performing IAM writes should expect affected sessions to receive 401 and reauthenticate.

6.2 IAM endpoints

All IAM routes require api.iam.manage. The middleware does not require a separate admin/superadmin token string; the effective permission is the gate.

Method	Endpoint	Purpose
GET	`/api/v1/iam/users`	List users and memberships.
POST	`/api/v1/iam/users`	Create a user.
POST	`/api/v1/iam/users/password`	Set a user password.
POST	`/api/v1/iam/users/delete`	Delete a user.
POST	`/api/v1/iam/users/groups`	Replace a user's memberships.
GET	`/api/v1/iam/groups`	List groups.
POST	`/api/v1/iam/groups`	Create a group.
POST	`/api/v1/iam/groups/delete`	Delete a group.
POST	`/api/v1/iam/groups/assign-user`	Add a user to a group.
POST	`/api/v1/iam/groups/remove-user`	Remove a user from a group.
GET	`/api/v1/iam/permissions`	List permissions.
GET	`/api/v1/iam/permission-catalog`	Return the grouped UI catalog.
POST	`/api/v1/iam/permissions`	Create a permission.
POST	`/api/v1/iam/permissions/delete`	Delete a permission.

Method	Endpoint	Purpose
POST	`/api/v1/iam/permissions/assign-group`	Grant a permission to a group.
POST	`/api/v1/iam/permissions/remove-group`	Revoke a permission from a group.
GET	`/api/v1/iam/users/effective-permissions?username=<user>`	Resolve a user's current effective permissions.

6.3 Create a user

```
curl -sS -k -X POST "$API/api/v1/iam/users" \
  -H "Authorization: Bearer $TOKEN" \
  -H "Content-Type: application/json" \
  -d '{
    "username": "automation-user",
    "password": "<strong secret>",
    "groups": ["stratisystem-apiuser"],
    "disabled": false
  }' | jq
```

6.4 Effective-permission diagnostics

```
curl -sS -k   "$API/api/v1/iam/users/effective-permissions?username=automation-user" \
  -H "Authorization: Bearer $TOKEN" | jq
```

6.5 API audit

GET /api/v1/audit/api?limit=100

Permission: `api.audit.read`

Returns API IAM/audit events. Treat limit as an upper-bound query parameter and paginate or archive externally if a client requires long-term audit retention.

7. Local accounts, Active Directory, and Cockpit AD

These endpoints wrap stratisystem-accounts. Local-account operations use the module's structured JSON interface. AD and Cockpit AD operations use explicit command mappings and return structured JSON.

7.1 Local account inventory

GET /api/v1/accounts/local

Permission: `accounts.local.read`

Query parameter	Default	Meaning
nodes	`Auto-discover`	Comma-separated target nodes.
include_non_login	`false`	Include service/non-login identities.
min_uid	`Module default 1000`	Minimum UID for managed inventory.
min_gid	`Module default`	Minimum GID for managed group inventory.
parallel	`Module default 8`	Maximum target-node fan-out.
verbose	`false`	Enable verbose backend execution while retaining JSON.

```
curl -sS -k   "$API/api/v1/accounts/local?nodes=node1,node2,node3&min_uid=1000" \
  -H "Authorization: Bearer $TOKEN" | jq
```

Inventory filtering: By default, non-login service identities are excluded even when a package allocates a UID above 1000. This keeps accounts such as clamav out of normal interactive-account inventory. Set include_non_login=true only for an explicit administrative use case.

7.2 Native local-account actions

POST /api/v1/accounts/local/action

Permission: `accounts.local.write`

Use the canonical action names below. The API also normalizes several shorter aliases, but SDKs should emit canonical values. dry_run is not supported on this endpoint; use /accounts/local/ensure for CLI-style dry-run behavior.

Action	Required fields	Important behavior	Default
list-local-accounts	`action`	Inventory; nodes may be omitted for discovery.	sync
create-local-account	`action, username, password`	Exact source account state is synchronized; shell defaults /bin/bash; create_home defaults true.	async

Action	Required fields	Important behavior	Default
set-local-account-password	`action, username, password`	Changes source then synchronizes the resulting shadow value.	async
set-local-account-groups	`action, username, primary_group and/or supplemental_groups`	Supplemental list is exact; [] clears supplemental groups.	async
sync-local-accounts	`action, usernames`	Use source_node or source_nodes to select authority.	async
lock-local-account	`action, username or usernames`	Idempotent; preserves underlying hash.	async
unlock-local-account	`action, username or usernames`	Fails when no usable hash can be restored.	async
remove-local-account	`action, username or usernames`	Missing accounts are successful no-ops; purge_home and force are optional.	async
create-local-group	`action, group`	gid optional; source allocates authoritative GID.	async
rename-local-group	`action, group, new_group`	Preserves authoritative GID.	async
remove-local-group	`action, group`	Refuses protected/primary groups.	async

Common request fields

Field	Type	Required / behavior
nodes	`string[]`	Optional target nodes; module may discover cluster hosts.
username / usernames	`string / string[]`	Single or multiple managed account names, depending action. root is not manageable.
group / groups	`string / string[]`	Single or multiple local group names, depending action.
new_group	`string`	New name for rename-local-group.
gid	`integer`	Optional requested group GID for create-local-group.
password	`string`	Required for create and password actions; newline/NUL rejected.
shell	`string`	Absolute shell path.
create_home	`boolean`	Create home directory when creating an account.
primary_group	`string`	Desired primary group.
supplemental_groups	`string[]`	Exact supplemental membership for native group action.
source_node	`string`	Authoritative node for source-driven operations.
source_nodes	`object`	Per-username source mapping for sync.
purge_home / force	`boolean`	Removal controls.
min_uid / min_gid	`integer`	Inventory/management lower bounds.
parallel / verbose	`integer / boolean`	Execution controls.

Field	Type	Required / behavior
async	`boolean`	Override default execution mode.

```
{
  "action": "create-local-account",
  "nodes": ["node1", "node2", "node3"],
  "username": "operator1",
  "password": "<secret>",
  "shell": "/bin/bash",
  "create_home": true,
  "primary_group": "operators",
  "supplemental_groups": ["wheel"],
  "source_node": "node1"
}
```

7.3 Ensure/update compatibility operation

POST /api/v1/accounts/local/ensure

Permission: `accounts.local.write`

This endpoint preserves the earlier ensure/update CLI behavior. It can append supplemental groups or replace them exactly and supports dry_run.

```
{
  "nodes": ["node1", "node2", "node3"],
  "username": "operator1",
  "password": "<secret>",
  "shell": "/bin/bash",
  "create_home": true,
  "primary_group": "operators",
  "groups": ["wheel"],
  "replace_groups": false,
  "dry_run": false,
  "async": true
}
```

7.4 Active Directory join

POST /api/v1/accounts/ad/join

Permission: `accounts.ad.join`

Field	Type	Required / behavior
nodes	`string[]`	Required.
domain_netbios	`string`	Required. Short domain name.
domain_fqdn	`string`	Required. Fully qualified domain name.
dns_server	`string`	Required. DNS server used for the join.
admin_user	`string`	Optional administrative account name.
admin_password	`string`	Backend join credential. Do not log.

Field	Type	Required / behavior
test_user	`string`	Optional post-join validation user.
default_cockpit_group	`string`	Optional default Cockpit allow group.
update_resolver	`boolean`	Optional explicit resolver-update setting.
configure_cockpit	`boolean`	Configure Cockpit AD access after join.
cockpit_groups	`string[]`	Initial allowed groups.
parallel, dry_run, verbose, log_file, async	`mixed`	Common execution controls; async defaults true.

```
{
  "nodes": ["node1", "node2", "node3"],
  "domain_netbios": "CORP",
  "domain_fqdn": "corp.example.com",
  "dns_server": "10.20.0.10",
  "admin_user": "join-operator",
  "admin_password": "<secret>",
  "update_resolver": true,
  "configure_cockpit": true,
  "cockpit_groups": ["CORP\Infrastructure Admins"]
}
```

Credential exposure limitation: The current AD join binary accepts the password as a command-line flag, so the API passes it in the child process argument vector. Restrict local process inspection and never include the request body in logs.

7.5 Cockpit AD access

POST /api/v1/accounts/cockpit-ad/action

Permission: `accounts.ad.write`

Action	Required action fields	Default
configure	`nodes; groups optional`	async
disable	`nodes`	async
add-group	`nodes, group`	async
remove-group	`nodes, group`	async
list-groups	`nodes`	sync
check-user	`nodes, user`	sync

Optional configuration fields are domain_netbios, test_user, default_group, sudo_group, allow_file, check_script, pam_file, sudoers_file, allow_unresolved_group, allow_local_accounts, and the common execution controls.

8. Multi-node network API

The network API wraps stratisystem-network and accepts the module's native JSON configuration. Always run precheck before apply in interactive workflows. The backend uses staged NetworkManager changes for disruptive bridge attachment and reconnect verification.

POST /api/v1/network/precheck

Permission: `network.read`

POST /api/v1/network/apply

Permission: `network.write`

> **Execution mode:** /network/precheck forces precheck_only=true and defaults to synchronous execution. /network/apply forces precheck_only=false and defaults to asynchronous execution. The client-supplied precheck_only value cannot change the selected endpoint behavior.

8.1 Supported actions

Canonical action	Minimum related fields
add-vlan	vlan_id and parent, globally or per host
remove-vlan	vlan_id and parent
add-bridge	bridge and bridge_port, or an inferred VLAN/VXLAN port
remove-bridge	bridge
add-vxlan	vxlan_id, vxlan_dev, and vxlan_remote or vxlan_group
remove-vxlan	vxlan_if or vxlan_id

Specify actions as an array. The request model also includes boolean add_vlan, remove_vlan, add_bridge, remove_bridge, add_vxlan, and remove_vxlan fields for module compatibility; new SDKs should prefer actions.

8.2 Request schema

Field	Type	Required / behavior
nodes	`string[]`	Required target hosts.
actions	`string[]`	Preferred operation list. action supports a single operation.
dry_run	`boolean`	Validate/render without committing changes where supported.
verbose	`boolean`	Verbose module execution.
parallel	`integer`	Maximum host fan-out.
log_file	`string`	Backend log path.
require_stratisystem	`boolean`	Require target platform validation.
host_overrides	`object`	Per-node network values.
vlan_id, parent	`integer / string`	Global VLAN ID and physical/bond parent.

Field	Type	Required / behavior
bridge, bridge_port	`string / string`	Global bridge and member interface.
vxlan_id, vxlan_if, vxlan_dev	`mixed`	VNI, interface name, and underlay interface.
vxlan_remote / vxlan_group	`string`	Unicast remote or multicast group.
vxlan_local	`string`	Optional local source address.
vxlan_dstport	`integer`	Optional UDP destination port.
neigh_suppress	`string`	Backend on/off-style value.
bridge_snooping / bridge_querier	`string`	Bridge multicast settings.
ssh	`object`	SSH connection configuration.
async	`boolean`	Override default execution mode.

8.3 Per-host overrides

Field	Type	Required / behavior
skip	`boolean`	Skip this host.
parent	`string`	Host-specific VLAN parent.
bridge	`string`	Host-specific bridge.
bridge_port	`string`	Host-specific bridge member.
vlan_id	`integer`	Host-specific VLAN ID.
vxlan_id	`integer`	Host-specific VNI.
vxlan_if	`string`	Host-specific VXLAN interface name.
vxlan_dev	`string`	Host-specific underlay interface.
vxlan_remote / vxlan_group / vxlan_local	`string`	Host-specific VXLAN addressing.

8.4 SSH object

Field	Type	Required / behavior
user	`string`	SSH user.
port	`string`	SSH port.
key_path	`string`	Private-key path on the API host.
connect_timeout_seconds	`integer`	SSH connection timeout.
command_timeout_seconds	`integer`	Remote command timeout.
options	`string[]`	Additional safe SSH options.

Rejected SSH options: ProxyCommand, LocalCommand, RemoteCommand, PermitLocalCommand, KnownHostsCommand, PKCS11Provider, and SecurityKeyProvider are rejected because they can execute or load local commands/providers.

8.5 VLAN-backed bridge example

```
{
  "nodes": ["node1", "node2", "node3"],
  "actions": ["add-vlan", "add-bridge"],
  "vlan_id": 30,
  "bridge": "br30",
  "host_overrides": {
    "node1": {"parent": "eno1"},
    "node2": {"parent": "enp3s0"},
    "node3": {"parent": "bond0"}
  }
}
```

8.6 Physical VM bridge example

```
{
  "nodes": ["node1", "node2", "node3"],
  "actions": ["add-bridge"],
  "bridge": "br0",
  "host_overrides": {
    "node1": {"bridge_port": "eno1"},
    "node2": {"bridge_port": "enp4s0"},
    "node3": {"bridge_port": "bond0"}
  }
}
```

A successful precheck returns resolved per-host configuration, hard errors, warnings, and planned operations. If any precheck fails, the backend does not apply changes to any host.

9. Direct VM backup API

These endpoints invoke the standalone vmbackup module directly. They are separate from the scheduled backup-job API exposed by vmchandler.

Endpoint	Permission	Command-specific fields	Default
`/api/v1/vmbackup/plan`	`backup.read`	domain; full optional	sync
`/api/v1/vmbackup/backup`	`backup.execute`	domain; full optional	async
`/api/v1/vmbackup/list`	`backup.read`	domain; include_failed optional	sync
`/api/v1/vmbackup/snapshot-sanity`	`backup.read`	domain	sync
`/api/v1/vmbackup/cleanup-failed`	`backup.write`	domain	async
`/api/v1/vmbackup/restore`	`backup.restore`	domain, point, mode; target fields optional	async
`/api/v1/vmbackup/verify`	`backup.read`	domain, point	async
`/api/v1/vmbackup/prune`	`backup.write`	domain	async
`/api/v1/vmbackup/repair-checkpoints`	`backup.write`	domain	async

Common request schema

Field	Type	Required / behavior
config	`string`	Optional absolute config-file path on the API host. Relative paths are rejected.
domain	`string`	Required for every command. Newline, carriage return, and NUL are rejected.
full	`boolean`	Plan or run a full backup.
include_failed	`boolean`	Include failed history in list.
point	`string`	Required for restore and verify.
mode	`string`	Restore: in_place, recreate_missing, or alternate. Aliases are normalized.
target_domain	`string`	Alternate restore target domain.
target_file_dir	`string`	Alternate file-backed disk directory.
target_rbd_pool	`string`	Alternate block target pool.
start_after	`boolean`	Start restored VM after successful restore.
async	`boolean`	Override command default.

9.1 Backup example

```
curl -sS -k -X POST "$API/api/v1/vmbackup/backup" \
  -H "Authorization: Bearer $TOKEN" \
  -H "Content-Type: application/json" \
  -d '{"domain":"application-vm-01","full":false}' | jq
```

9.2 Alternate restore example

```
{
  "domain": "application-vm-01",
  "point": "2026-07-13T18:30:00Z",
  "mode": "alternate",
  "target_domain": "application-vm-01-recovered",
  "target_file_dir": "/stratiserv/clustermounts/recovery",
  "start_after": false,
  "async": true
}
```

Restore caution: Restore is destructive or state-changing. Never retry automatically after a network timeout or lost response. Poll the returned job and inspect target state before resubmitting.

10. Direct VM replication API

GET /api/v1/replicatevm/version

Permission: `replication.read`

POST /api/v1/replicatevm/run

Permission: `replication.execute`

The run endpoint exposes the current standalone replicatevm flag surface and defaults to asynchronous execution.

Field	Type	Required / behavior
vm	`string`	Required source VM.
remote_host	`string`	Required remote/DR host.
remote_path	`string`	Required remote replication path.
checkpoint	`string`	Required checkpoint/state path.
libvirt_uri	`string`	Optional source libvirt URI.
ssh_user / ssh_port	`string`	Remote SSH connection.
wait_secs / sleep_secs / stale_overlay_secs	`integer`	Timing controls.
state_dir / debug_log_path	`string`	Local state and debug paths.
net_retries / net_retry_sleep_secs / net_retry_max_sleep	`integer`	Network retry policy.
sync_timeout_secs / ssh_connect_timeout	`integer`	Transfer/connect timeouts.
allow_crash_consistent	`boolean`	Permit crash-consistent fallback.
sync_sparse_existing	`boolean`	Sparse-existing synchronization setting.
checksum / checksum_algo	`string`	Checksum enable/mode and algorithm.
compress / compress_algo / compress_level	`string`	Compression controls.

Field	Type	Required / behavior
remote_space_headroom_pct	`integer`	Percentage headroom.
remote_space_headroom_gib / remote_space_min_free_gib	`integer`	Absolute free-space controls.
dr_only / dr_import / dr_import_sync / dr_start_vm	`boolean`	DR workflow controls.
dr_libvirt_uri / dr_suffix / dr_lock_name	`string`	DR import naming/locking.
dr_preserve_source_uuid / dr_preserve_source_name	`boolean`	Source identity preservation.
keep_workdir_on_fail	`boolean`	Retain work directory on failure.
install_remote_qemu_modules	`boolean`	Install required remote QEMU modules.
async	`boolean`	Defaults true.

```
{
  "vm": "application-vm-01",
  "remote_host": "dr-node-01",
  "remote_path": "/stratiserv/clustermounts/replicas",
  "checkpoint": "/stratiserv/clustermounts/replication-state",
  "checksum": "on",
  "checksum_algo": "xxh3",
  "compress": "on",
  "compress_algo": "zstd",
  "dr_import": true,
  "dr_start_vm": false,
  "async": true
}
```

11. Standalone VMware import API

POST /api/v1/vmwareimport/import

Permission: `vm.import.manage`

Field	Type	Required / behavior
esx_host_address	`string`	Required ESXi host address.
username	`string`	Required source credential.
password	`string`	Required source credential.
vm_name	`string`	Required source VM name.
bridge	`string`	Required destination bridge, commonly br0.
destination	`string`	Required destination storage path.
verbose	`boolean`	Enable verbose backend output.
async	`boolean`	Defaults true.

```
{
  "esx_host_address": "192.0.2.50",
  "username": "migration-user",
  "password": "<secret>",
  "vm_name": "legacy-app-01",
  "bridge": "br0",
  "destination": "/stratiserv/clustermounts/machines",
  "async": true
}
```

The standalone import binary does not emit native JSON. The API converts completion state, stdout, stderr, exit code, VM name, source host, bridge, and destination into a stable JSON response/job result.

> **Credential exposure limitation:** The current standalone VMware import binary accepts the password as a command-line flag. Restrict local process inspection and never log the request body.

12. Newly covered StratiSERV operations

GET /api/v1/getclustervmsummary

Permission: `cluster.vm.read`

Returns the lightweight cluster VM summary from vmchandler under the vms response key.

GET /api/v1/getcpumodels?arch=x86_64

Permission: `cluster.vm.read`

Returns libvirt CPU models. arch is optional. The API normalizes an inconsistent backend case where vmchandler may emit success:true together with an error.

POST /api/v1/unmapblockdevicefromvm

Permission: `block.device.write`

```
{
  "devicename": "database-volume-01",
  "primarypoolname": "production-block",
  "domainidtounmapfrom": "<domain UUID>"
}
```

Canonical fields are devicename, primarypoolname, and domainidtounmapfrom. Compatibility aliases are accepted, but SDKs should use the canonical form.

12.1 Shared filesystem capacity expansion

POST /api/v1/getshareddiskfilesystemexpandinfo

Permission: cluster.shareddisk.read

Performs a lightweight, non-mutating eligibility check for a managed shared GFS2 filesystem. Canonical request fields are hostnames, devicepath, deviceserial, and fsname.

```
{
  "hostnames": ["stratistor1-1", "stratistor1-2", "stratistor1-3"],
  "devicepath": "/dev/disk/by-id/dm-uuid-mpath-<WWID>",
  "deviceserial": "<WWID>",
  "fsname": "ds01"
}
```

A successful HTTP 200 response includes expandable/reason plus available LVM/GFS2 geometry such as expandable_bytes, needs_pv_resize, needs_lv_extend, needs_gfs2_grow, participant_hosts, and mounted_hosts.

POST /api/v1/expandshareddiskfilesystem

Permission: cluster.shareddisk.write

Uses the same request body and submits the online capacity expansion as a backend-managed shared-disk job. Successful submission returns HTTP 202 with job_id/status. Poll GET /api/v1/managedshareddiskjobs/:id; GET /api/v1/managedshareddiskjobs lists jobs. The current backend does not permit interrupting an active clustered-storage job after submission, so DELETE /api/v1/managedshareddiskjobs/:id may report the job as non-cancelable.

13. SMB API

SMB endpoints use the legacy StratiSTOR handler with JSON request wrappers. New SDKs should use canonical fields and treat all share-relative paths as opaque strings. Raw JSON strings may contain spaces; percent-encoded values are also accepted and decoded by the API.

13.1 Read endpoints

Endpoint	Permission	Purpose
GET /api/v1/getsmbshareinfo	`smb.share.read`	List shares.
GET /api/v1/getsmbuserinfo	`smb.user.read`	List SMB users.
GET /api/v1/getsmbgroupinfo	`smb.group.read`	List SMB groups.
POST /api/v1/listsmbsharedirectorycontents	`smb.share.read`	List a share-relative directory.
POST /api/v1/getsmbshareusage	`smb.share.read`	Return usedcapacitybytes for a share.

```
{
  "smbsharename": "EngineeringDocs",
  "directorypath": "/projects/2026"
}
```

```
{
  "smbsharename": "EngineeringDocs"
}
```

directorypath defaults to /. Directory listing metadata may include filename, filetype, filesize, lastmodified, lastaccessed, and created. created can be empty when the underlying filesystem does not expose birth time.

13.2 Canonical payload fields

Operation	Canonical fields
SMB group	smbgroupname; users for full membership replacement
SMB user	smbusername, smbuserpassword, smbusergroups
SMB share	smbsharename, smbsharefilesystem, smbsharefilesystemtype, smbsharesubdir, assignedusers
Directory list/create/delete	smbsharename, directorypath
Path rename/copy/move	smbsharename, sourcepath, destinationpath
File delete	smbsharename, filepath; confirmation for wildcard delete
Share usage	smbsharename

13.3 User and group examples

```
{
  "smbgroupname": "engineering"
}
```

```
{
  "smbgroupname": "engineering",
  "users": ["alice", "bob", "charlie"]
}
```

editsmbgroup replaces membership with the provided list. An empty list clears membership.

```
{
  "smbusername": "alice",
  "smbuserpassword": "<secret>",
  "smbusergroups": ["engineering", "staff"]
}
```

13.4 Share example

```
{
  "smbsharename": "EngDocs",
  "smbsharefilesystem": "shared-data-01",
  "smbsharefilesystemtype": "gfs2",
  "smbsharesubdir": "/engineering/docs",
  "assignedusers": ["engineering", "staff"]
}
```

Use the filesystem type value supported by the deployed StratiSTOR backend. Share assignment lists are replacement lists on edit.

13.5 Share path safety and destructive confirmations

- All paths are resolved relative to the selected SMB share root.
- Traversal above the share root is rejected.
- Listing may use / for the share root.
- Create/delete/rename/copy/move cannot target the share root itself as an object.
- Recursive directory deletion requires confirmation: CONFIRM_RECURSIVE_DELETE.
- Wildcard file deletion requires confirmation: CONFIRM_WILDCARD_DELETE.
- SDKs must never auto-add destructive confirmation unless the caller explicitly requested that destructive operation.

```
{
  "smbsharename": "EngineeringDocs",
  "directorypath": "projects/archive-2025",
  "confirmation": "CONFIRM_RECURSIVE_DELETE"
}
```

```
{
  "smbsharename": "EngineeringDocs",
  "filepath": "projects/2026/*.tmp",
  "confirmation": "CONFIRM_WILDCARD_DELETE"
}
```

Wildcard matching is non-recursive. If a match is a directory, the request is rejected and the directory endpoint must be used.

Appendix A. Complete endpoint catalog

This catalog reflects the current router source: 203 unique HTTP method/path combinations. It includes compatibility aliases that map to the same backend action. Detailed request schemas are provided earlier for authentication, IAM, SMB, accounts, network, direct VM backup, direct replication, standalone VMware import, and shared-filesystem expansion.

Public, authentication, diagnostics, and jobs

Method	Endpoint	Permission / access rule
GET	/healthz	Public / authentication only
POST	/login	Public / authentication only
POST	/api/v1/auth/login	Public / authentication only
POST	/api/v1/auth/refresh	Public / authentication only
POST	/api/v1/auth/logout	api.auth.logout
GET	/api/v1/testfunction	api.test
GET	/api/v1/jobs/:id	Authenticated owner or API admin
DELETE	/api/v1/jobs/:id	Authenticated owner or API admin

System state and licensing

Method	Endpoint	Permission / access rule
GET	/api/v1/getstratistorinfo	system.info.read
GET	/api/v1/geteulainfo	eula.read
POST	/api/v1/accepteula	eula.accept
GET	/api/v1/getlicenseinfo	license.read
POST	/api/v1/updatelicense	license.write

StratiSERV platform and cluster

Method	Endpoint	Permission / access rule
GET	/api/v1/getstratiservinfo	stratiserv.info.read
GET	/api/v1/getservicesinfo	stratiserv.services.read
GET	/api/v1/getresourceschedulersettings	stratiserv.scheduler.read
POST	/api/v1/modifyresourceschedulersettings	stratiserv.scheduler.write
GET	/api/v1/getclusternodeinfo	cluster.node.read
POST	/api/v1/getshareddiskinfo	cluster.shareddisk.read
GET	/api/v1/getclustervminfo	cluster.vm.read
GET	/api/v1/getclustervmsummary	cluster.vm.read
GET	/api/v1/getcpumodels	cluster.vm.read
POST	/api/v1/fetchspecificvmdata	cluster.vm.read
GET	/api/v1/getclusteraffinitygroupinfo	cluster.affinity.read

Method	Endpoint	Permission / access rule
GET	/api/v1/getclusterantiaffinitygroupinfo	cluster.affinity.read
GET	/api/v1/getclusterhaaffinitygroupinfo	cluster.affinity.read
POST	/api/v1/addclusteraffinitygroup	cluster.affinity.write
POST	/api/v1/addclusterantiaffinitygroup	cluster.affinity.write
POST	/api/v1/addclusterhaaffinitygroup	cluster.affinity.write
POST	/api/v1/updateclusteraffinitygroup	cluster.affinity.write
POST	/api/v1/updateclusterantiaffinitygroup	cluster.affinity.write
POST	/api/v1/updateclusterhaaffinitygroup	cluster.affinity.write
POST	/api/v1/removeclusteraffinitygroup	cluster.affinity.write
POST	/api/v1/removeclusterantiaffinitygroup	cluster.affinity.write
POST	/api/v1/removeclusterhaaffinitygroup	cluster.affinity.write
POST	/api/v1/clusterrestart	cluster.operations.write
POST	/api/v1/rescanallstoragetargets	cluster.operations.write
POST	/api/v1/createclusterfencedevice	cluster.shareddisk.write
POST	/api/v1/removeclusterfencedevice	cluster.shareddisk.write
POST	/api/v1/createshareddiskfilesystem	cluster.shareddisk.write
POST	/api/v1/extendaccessshareddiskfilesystem	cluster.shareddisk.write
POST	/api/v1/removeshareddiskfilesystem	cluster.shareddisk.write
POST	/api/v1/getshareddiskfilesystemexpandinfo	cluster.shareddisk.read
POST	/api/v1/expandshareddiskfilesystem	cluster.shareddisk.write
GET	/api/v1/managedshareddiskjobs	cluster.shareddisk.read
GET	/api/v1/managedshareddiskjobs/:id	cluster.shareddisk.read
DELETE	/api/v1/managedshareddiskjobs/:id	cluster.shareddisk.write
POST	/api/v1/enterhostmaintenancemode	cluster.maintenance.write
POST	/api/v1/exithostmaintenancemode	cluster.maintenance.write

Virtual machines and external imports

Method	Endpoint	Permission / access rule
POST	/api/v1/enableclustervmhamode	vm.ha.write
POST	/api/v1/disableclustervmhamode	vm.ha.write
POST	/api/v1/enabletemplatemodeforvm	vm.template.write
POST	/api/v1/disabletemplatemodeforvm	vm.template.write
POST	/api/v1/deletevm	vm.manage
POST	/api/v1/updatevmsettings	vm.manage
POST	/api/v1/refreshvmconfigindb	vm.manage
POST	/api/v1/unregistervm	vm.manage

Method	Endpoint	Permission / access rule
POST	/api/v1/resizevmdisk	vm.disk.write
POST	/api/v1/getvmnetworkinterfaces	vm.network.read
POST	/api/v1/addvmnetworkinterface	vm.network.write
POST	/api/v1/updatevmnetworkinterface	vm.network.write
POST	/api/v1/removevmnetworkinterface	vm.network.write
POST	/api/v1/managedvmjobs/clone	vm.manage
POST	/api/v1/managedvmjobs/deploy-template	vm.template.write
POST	/api/v1/managedvmjobs/convert-disks	vm.disk.write
POST	/api/v1/managedvmjobs/migrate-vm	vm.manage
POST	/api/v1/managedvmjobs/migrate-disks	vm.disk.write
GET	/api/v1/managedvmjobs	cluster.vm.read
GET	/api/v1/managedvmjobs/:id	cluster.vm.read
DELETE	/api/v1/managedvmjobs/:id	vm.manage
POST	/api/v1/vmwareimportjobs	vm.import.manage
GET	/api/v1/vmwareimportjobs	vm.import.read
GET	/api/v1/vmwareimportjobs/:id	vm.import.read
DELETE	/api/v1/vmwareimportjobs/:id	vm.import.manage
POST	/api/v1/performvmwareapiauthentication	vm.import.manage
POST	/api/v1/getvmwarevmmanifest	vm.import.read
POST	/api/v1/enablesshvmwarehost	vm.import.manage
POST	/api/v1/disablesshvmwarehost	vm.import.manage
POST	/api/v1/deleteallsnapshotsvmwarevm	vm.import.manage
POST	/api/v1/poweroffvmwarevm	vm.import.manage
POST	/api/v1/poweronvmwarevm	vm.import.manage
POST	/api/v1/importvmwarevm	vm.import.manage
POST	/api/v1/validatevmwarehostcredentials	vm.import.manage
POST	/api/v1/performmicrosoftapiauthentication	vm.import.manage
POST	/api/v1/getmicrosoftvmmanifest	vm.import.read
POST	/api/v1/poweroffmicrosoftvm	vm.import.manage
POST	/api/v1/poweronmicrosoftvm	vm.import.manage
POST	/api/v1/importmicrosoftvm	vm.import.manage
POST	/api/v1/deleteallsnapshotsmicrosoftvm	vm.import.manage

Backup, locks, performance, and replication

Method	Endpoint	Permission / access rule
GET	/api/v1/getbackupjobinfo	backup.read
POST	/api/v1/addbackuprepo	backup.write

Method	Endpoint	Permission / access rule
POST	/api/v1/removebackuprepo	backup.write
POST	/api/v1/addbackupjob	backup.write
POST	/api/v1/editbackupjob	backup.write
POST	/api/v1/removebackupjob	backup.write
POST	/api/v1/runbackupjob	backup.execute
POST	/api/v1/changestatebackupjob	backup.write
POST	/api/v1/restorevm	backup.restore
POST	/api/v1/verifybackup	backup.read
POST	/api/v1/getbackupsinfo	backup.read
POST	/api/v1/getvmmigrationlockstate	locks.read
POST	/api/v1/setvmmigrationlockstate	locks.write
POST	/api/v1/gethostlockstate	locks.read
GET	/api/v1/gethostslockstate	locks.read
POST	/api/v1/sethostlockstate	locks.write
POST	/api/v1/sethostslockstate	locks.write
GET	/api/v1/getperfmetrics	performance.read
GET	/api/v1/getperfresources	performance.read
GET	/api/v1/getreplicationdestinationinfo	replication.read
POST	/api/v1/addreplicationdestination	replication.write
POST	/api/v1/editreplicationdestination	replication.write
POST	/api/v1/removereplicationdestination	replication.write
GET	/api/v1/getreplicationjobinfo	replication.read
POST	/api/v1/addreplicationjob	replication.write
POST	/api/v1/editreplicationjob	replication.write
POST	/api/v1/removereplicationjob	replication.write
POST	/api/v1/setreplicationjobstate	replication.write
POST	/api/v1/changestatereplicationjob	replication.write
POST	/api/v1/runreplicationjobnow	replication.execute
POST	/api/v1/runreplicationjob	replication.execute

Block storage

Method	Endpoint	Permission / access rule
GET	/api/v1/getblockdevicepoolinfo	block.pool.read
POST	/api/v1/addblockdevicepool	block.pool.write
POST	/api/v1/renameblockdevicepool	block.pool.write
POST	/api/v1/deleteblockdevicepool	block.pool.write
GET	/api/v1/getblockdeviceinfo	block.device.read
POST	/api/v1/addblockdevice	block.device.write
POST	/api/v1/renameblockdevice	block.device.write

Method	Endpoint	Permission / access rule
POST	/api/v1/deleteblockdevice	block.device.write
POST	/api/v1/mapblockdevicetovm	block.device.write
POST	/api/v1/mapblockdevicetohost	block.device.write
POST	/api/v1/unmapblockdevicefromhost	block.device.write
POST	/api/v1/unmapblockdevicefromvm	block.device.write

Filesystems

Method	Endpoint	Permission / access rule
GET	/api/v1/getfilesysteminfo	filesystem.read
POST	/api/v1/addfilesystem	filesystem.write
POST	/api/v1/renamefilesystem	filesystem.write
POST	/api/v1/deletefilesystem	filesystem.write

SMB

Method	Endpoint	Permission / access rule
GET	/api/v1/getsmbshareinfo	smb.share.read
GET	/api/v1/getsmbuserinfo	smb.user.read
GET	/api/v1/getsmbgroupinfo	smb.group.read
POST	/api/v1/listsmbsharedirectorycontents	smb.share.read
POST	/api/v1/getsmbshareusage	smb.share.read
POST	/api/v1/addsmbgroup	smb.group.write
POST	/api/v1/editsmbgroup	smb.group.write
POST	/api/v1/deletesmbgroup	smb.group.write
POST	/api/v1/addsmbshare	smb.share.write
POST	/api/v1/editsmbshare	smb.share.write
POST	/api/v1/deletesmbshare	smb.share.write
POST	/api/v1/createsmbsharedirectory	smb.share.write
POST	/api/v1/deletesmbsharedirectory	smb.share.write
POST	/api/v1/renamesmbsharepath	smb.share.write
POST	/api/v1/copysmbsharepath	smb.share.write
POST	/api/v1/movesmbsharepath	smb.share.write
POST	/api/v1/deletesmbsharefile	smb.share.write
POST	/api/v1/addsmbuser	smb.user.write
POST	/api/v1/deletesmbuser	smb.user.write
POST	/api/v1/changepasswordsmbuser	smb.user.write

Web engine

Method	Endpoint	Permission / access rule
GET	/api/v1/getwebengineinfo	webengine.read
GET	/api/v1/getwebengineservicestate	webengine.read
POST	/api/v1/editwebengineconfig	webengine.write
POST	/api/v1/webengineservicecontrol	webengine.service

Remote support

Method	Endpoint	Permission / access rule
GET	/api/v1/getremotesupportinfo	remotesupport.read
GET	/api/v1/getremotesupportservicestate	remotesupport.read
POST	/api/v1/editremotesupportconfig	remotesupport.write
POST	/api/v1/remotesupportservicecontrol	remotesupport.service

Updates

Method	Endpoint	Permission / access rule
GET	/api/v1/checkforupdates	updates.read
POST	/api/v1/applyupdates	updates.apply

Standalone module wrappers

Method	Endpoint	Permission / access rule
GET	/api/v1/accounts/local	accounts.local.read
POST	/api/v1/accounts/local/action	accounts.local.write
POST	/api/v1/accounts/local/ensure	accounts.local.write
POST	/api/v1/accounts/ad/join	accounts.ad.join
POST	/api/v1/accounts/cockpit-ad/action	accounts.ad.write
POST	/api/v1/network/precheck	network.read
POST	/api/v1/network/apply	network.write
POST	/api/v1/vmbackup/plan	backup.read
POST	/api/v1/vmbackup/backup	backup.execute
POST	/api/v1/vmbackup/list	backup.read
POST	/api/v1/vmbackup/snapshot-sanity	backup.read
POST	/api/v1/vmbackup/cleanup-failed	backup.write
POST	/api/v1/vmbackup/restore	backup.restore
POST	/api/v1/vmbackup/verify	backup.read
POST	/api/v1/vmbackup/prune	backup.write
POST	/api/v1/vmbackup/repair-checkpoints	backup.write
GET	/api/v1/replicatevm/version	replication.read

Method	Endpoint	Permission / access rule
POST	/api/v1/replicatevm/run	replication.execute
POST	/api/v1/vmwareimport/import	vm.import.manage

IAM and audit

Method	Endpoint	Permission / access rule
GET	/api/v1/iam/users	api.iam.manage
POST	/api/v1/iam/users	api.iam.manage
POST	/api/v1/iam/users/password	api.iam.manage
POST	/api/v1/iam/users/delete	api.iam.manage
POST	/api/v1/iam/users/groups	api.iam.manage
GET	/api/v1/iam/groups	api.iam.manage
POST	/api/v1/iam/groups	api.iam.manage
POST	/api/v1/iam/groups/delete	api.iam.manage
POST	/api/v1/iam/groups/assign-user	api.iam.manage
POST	/api/v1/iam/groups/remove-user	api.iam.manage
GET	/api/v1/iam/permissions	api.iam.manage
GET	/api/v1/iam/permission-catalog	api.iam.manage
POST	/api/v1/iam/permissions	api.iam.manage
POST	/api/v1/iam/permissions/delete	api.iam.manage
POST	/api/v1/iam/permissions/assign-group	api.iam.manage
POST	/api/v1/iam/permissions/remove-group	api.iam.manage
GET	/api/v1/iam/users/effective-permissions	api.iam.manage
GET	/api/v1/audit/api	api.audit.read

Appendix B. Built-in permission catalog

The current service seeds 66 built-in permissions and synchronizes grants when it initializes the API schema. The superadministrator group receives every active permission. Custom groups and permissions remain available through IAM.

Authentication

Permission	Risk	Display name	Description
`api.auth.logout`	`low`	Logout Session	Allow an authenticated API user to log out and revoke their current session.
`api.auth.refresh`	`low`	Refresh Session Token	Allow refresh-token exchange for a new access token.

Diagnostics

Permission	Risk	Display name	Description
`api.test`	`low`	Run API Test Function	Allow access to diagnostic/test API functions.

Identity & Access

Permission	Risk	Display name	Description
`api.iam.manage`	`critical`	Manage API IAM	Manage API users, groups, permissions, and group permission assignments.
`api.audit.read`	`high`	Read API Audit Log	View API audit log entries.
`accounts.local.read`	`medium`	Read Local Accounts	Read local Linux account and group inventory across StratiSYSTEM nodes.
`accounts.local.write`	`critical`	Manage Local Accounts	Create, synchronize, lock, unlock, update, and remove local Linux accounts and groups across nodes.
`accounts.ad.join`	`critical`	Join Active Directory	Join StratiSYSTEM nodes to Active Directory and optionally configure Cockpit access.
`accounts.ad.write`	`critical`	Manage Cockpit AD Access	Configure, disable, inspect, and update Cockpit Active Directory access controls.

System

Permission	Risk	Display name	Description
`system.info.read`	`low`	Read System Status	Read aggregated StratiSTOR system status and platform health.

Licensing

Permission	Risk	Display name	Description
`eula.read`	`low`	Read EULA	Read the current EULA state and text.

Permission	Risk	Display name	Description
`eula.accept`	`medium`	Accept EULA	Accept the current EULA on behalf of the authenticated caller.
`license.read`	`low`	Read License	Read the currently installed license information.
`license.write`	`high`	Update License	Install or replace the active product license.

Storage

Permission	Risk	Display name	Description
`block.pool.read`	`low`	Read Block Device Pools	Read block device pool inventory and details.
`block.pool.write`	`high`	Manage Block Device Pools	Create, rename, and delete block device pools.
`block.device.read`	`low`	Read Block Devices	Read block device inventory and mappings.
`block.device.write`	`high`	Manage Block Devices	Create, rename, delete, and map block devices.
`filesystem.read`	`low`	Read Filesystems	Read filesystem inventory and status.
`filesystem.write`	`high`	Manage Filesystems	Create, rename, and delete filesystems.

SMB

Permission	Risk	Display name	Description
`smb.share.read`	`low`	Read SMB Shares	Read SMB share information, usage, and directory listings.
`smb.share.write`	`high`	Manage SMB Shares	Create, edit, delete, and manage SMB shares and share paths.
`smb.user.read`	`low`	Read SMB Users	Read SMB user information.
`smb.user.write`	`high`	Manage SMB Users	Create, delete, and update SMB user credentials.
`smb.group.read`	`low`	Read SMB Groups	Read SMB group information.
`smb.group.write`	`high`	Manage SMB Groups	Create, edit, and delete SMB groups.

Web Engine

Permission	Risk	Display name	Description
`webengine.read`	`low`	Read Web Engine	Read web engine configuration and service state.
`webengine.write`	`high`	Manage Web Engine Configuration	Update web engine configuration.
`webengine.service`	`high`	Control Web Engine Service	Start, stop, restart, or otherwise control the web engine service.

Remote Support

Permission	Risk	Display name	Description
`remotesupport.read`	`low`	Read Remote Support	Read remote support activation state and service state.
`remotesupport.write`	`high`	Manage Remote Support Configuration	Configure remote support activation settings.
`remotesupport.service`	`high`	Control Remote Support Service	Start, stop, restart, or otherwise control the remote support service.

Updates

Permission	Risk	Display name	Description
`updates.read`	`medium`	Check for Updates	Check the current repository version available for upgrade.
`updates.apply`	`critical`	Apply Updates	Apply system updates through the backend handler.

StratiSERV

Permission	Risk	Display name	Description
`stratiserv.info.read`	`low`	Read StratiSERV Platform Status	Read StratiSERV platform status and management metadata.
`stratiserv.services.read`	`low`	Read StratiSERV Services	Read StratiSERV service inventory and compute service state.
`stratiserv.scheduler.read`	`low`	Read Resource Scheduler Settings	Read resource scheduler settings and balancing configuration.
`stratiserv.scheduler.write`	`high`	Manage Resource Scheduler Settings	Modify resource scheduler thresholds and balancing behavior.

Cluster

Permission	Risk	Display name	Description
`cluster.node.read`	`low`	Read Cluster Nodes	Read cluster node inventory and cluster host metadata.
`cluster.vm.read`	`low`	Read Cluster VMs	Read cluster VM inventory and host placement metadata.
`cluster.shareddisk.read`	`low`	Read Shared Disk State	Read shared disk inventory, shared filesystem expansion eligibility, managed shared-disk jobs, and shared storage state.
`cluster.shareddisk.write`	`high`	Manage Shared Disk Resources	Create and remove fence devices; create, extend, remove, and expand shared disk filesystems; and submit managed shared-disk cancellation requests when the backend considers them safe.
`cluster.affinity.read`	`low`	Read Cluster Affinity Groups	Read VM affinity, VM anti-affinity, and host HA-affinity groups.
`cluster.affinity.write`	`high`	Manage Cluster Affinity Groups	Create, update, and remove VM affinity, VM anti-affinity, and host HA-affinity groups.
`cluster.operations.write`	`critical`	Run Cluster Operations	Run cluster-wide operational actions such as restarts and storage rescans.

Permission	Risk	Display name	Description
`cluster.maintenance.write`	`high`	Manage Host Maintenance Mode	Enter and exit cluster host maintenance mode.
`locks.read`	`medium`	Read Cluster Locks	Read migration and host lock state across the cluster.
`locks.write`	`high`	Manage Cluster Locks	Set or clear migration and host locks across the cluster.

Virtual Machines

Permission	Risk	Display name	Description
`vm.ha.write`	`high`	Manage VM HA Mode	Enable or disable high-availability protection for cluster VMs.
`vm.template.write`	`high`	Manage VM Template Mode	Enable or disable template mode for VMs.
`vm.manage`	`high`	Manage Cluster VMs	Delete, unregister, refresh, and update cluster VM configuration.
`vm.disk.write`	`high`	Resize VM Disks	Resize a VM disk by domain reference, disk target, and requested size.
`vm.network.read`	`low`	Read VM Network Interfaces	Read persistent and live network-interface configuration for cluster VMs.
`vm.network.write`	`high`	Manage VM Network Interfaces	Hot-add, update, and remove VM network interfaces while keeping persistent configuration synchronized.
`vm.import.read`	`medium`	Read External VM Manifests	Retrieve external VM manifests and VMware import-job status.
`vm.import.manage`	`critical`	Manage External VM Imports	Authenticate to external platforms and start, monitor, cancel, or run VM imports.

Backup

Permission	Risk	Display name	Description
`backup.read`	`medium`	Read Backup Metadata	Read backup jobs, repositories, restore points, and verification data.
`backup.write`	`high`	Manage Backup Jobs	Create, edit, delete, and toggle backup repositories and jobs.
`backup.execute`	`high`	Run Backup Jobs	Trigger backup jobs to execute immediately.
`backup.restore`	`critical`	Restore VMs from Backup	Restore VM data from backup repositories and restore points.

Performance

Permission	Risk	Display name	Description
`performance.read`	`low`	Read Performance Metrics	Read live performance metrics and system resource manifests.

Replication

Permission	Risk	Display name	Description
`replication.read`	`medium`	Read Replication State	Read replication destinations, jobs, and their configuration.
`replication.write`	`high`	Manage Replication	Create, edit, remove, and enable or disable replication jobs and destinations.
`replication.execute`	`high`	Run Replication Jobs	Trigger replication jobs to execute immediately.

Network

Permission	Risk	Display name	Description
`network.read`	`high`	Precheck Network Changes	Validate multi-node VLAN, bridge, and VXLAN changes without applying them.
`network.write`	`critical`	Manage Cluster Networking	Apply multi-node VLAN, bridge, VXLAN, and related NetworkManager changes.

Appendix C. SDK reference behavior

C.1 Required client operations

- Login(username, password) and securely retain access/refresh tokens.
- Do(method, path, body) with centralized TLS, authorization, JSON, timeout, and error handling.
- Refresh() with refresh-token rotation and single-flight concurrency.
- Retry the original request once after a successful refresh; never loop on 401.
- Do not retry 403.
- Submit long-running operations once, persist the job ID in client state, and poll.
- Logout() and clear both tokens.

C.2 Bash reference helper

```
API="https://192.168.59.101:3201"
TOKEN=""
REFRESH_TOKEN=""

login() {
  local user="$1" pass="$2" resp
  resp=$(curl -sS -k -X POST "$API/api/v1/auth/login" \
    -H "Content-Type: application/json" \
    -d "$(jq -nc --arg u "$user" --arg p "$pass"       '{username:$u,password:$p}')") || return
1
  TOKEN=$(jq -r '.token // empty' <<<"$resp")
  REFRESH_TOKEN=$(jq -r '.refresh_token // empty' <<<"$resp")
  [[ -n "$TOKEN" && -n "$REFRESH_TOKEN" ]]
}

refresh() {
  local resp
  [[ -n "$REFRESH_TOKEN" ]] || return 1
  resp=$(curl -sS -k -X POST "$API/api/v1/auth/refresh" \
    -H "Content-Type: application/json" \
    -d "$(jq -nc --arg r "$REFRESH_TOKEN" '{refresh_token:$r}')")     || return 1
  TOKEN=$(jq -r '.token // empty' <<<"$resp")
  REFRESH_TOKEN=$(jq -r '.refresh_token // empty' <<<"$resp")
  [[ -n "$TOKEN" && -n "$REFRESH_TOKEN" ]]
}

api_request() {
  local method="$1" path="$2" body="${3:-}" tmp code
  tmp=$(mktemp)
  local args=(-sS -k -X "$method" "$API$path"     -H "Authorization: Bearer $TOKEN" -o "$tmp"
-w '%{http_code}')
  [[ -n "$body" ]] && args+=(-H "Content-Type: application/json" -d "$body")
  code=$(curl "${args[@]}") || { rm -f "$tmp"; return 1; }
  if [[ "$code" == 401 ]] && refresh; then
    args=(-sS -k -X "$method" "$API$path"       -H "Authorization: Bearer $TOKEN" -o "$tmp" -w
'%{http_code}')
    [[ -n "$body" ]] && args+=(-H "Content-Type: application/json" -d "$body")
    code=$(curl "${args[@]}") || { rm -f "$tmp"; return 1; }
  fi
  cat "$tmp"
  rm -f "$tmp"
  [[ "$code" =~ ^2[0-9][0-9]$ ]]
}
```

Production TLS: Replace -k with certificate validation, such as --cacert /path/to/ca.pem, in production.

C.3 End-to-end validation sequence

1. Call GET /healthz and verify HTTP 200.
2. Login and verify both token and refresh_token are present.
3. Call GET /api/v1/testfunction with a user that has api.test.
4. Read the current user's effective permissions.
5. Run a safe read endpoint appropriate to the node role.
6. Submit a network precheck with no apply.
7. Submit a long-running operation in a non-production test environment and poll its job to completion.
8. Test refresh-token rotation and verify the prior refresh token is no longer used.
9. Test a deliberately unauthorized route and verify 403 includes permission.
10. Logout and verify the old access token returns 401.

Appendix D. Corrections and consistency notes

- Login and refresh behavior now reflects the current implementation rather than optional/observed compatibility language.
- The privilege claim is documented as user or admin; superadministrator is a built-in group and permission set.
- IAM and audit access are documented as permission-gated, not role-tier-gated.
- The endpoint count is corrected to 203 unique method/path combinations for the current router.
- The guide distinguishes scheduled vmchandler backup/replication endpoints from direct standalone vmbackup/replicatevm endpoints.
- The async job lifecycle, ownership rules, 24-hour retention, 1,000-record cap, and restart behavior are documented.
- All examples use canonical request fields; aliases are described only as compatibility behavior.
- SMB share usage and paths containing spaces/percent encoding are covered.
- The built-in RBAC catalog now documents all 66 seeded permissions, including VM network-interface read/write controls and expanded shared-disk/affinity descriptions.
- The endpoint catalog now includes VM network-interface management, managed VM/shared-disk/VMware import jobs, host HA-affinity group operations, and shared-filesystem capacity expansion.
- 2FA REST operations are explicitly excluded.