ProjectX

OBJECT STORAGE · DATASHEET

S3-compatible buckets behind one endpoint.

Buckets with versioning and lifecycle rules, reached through a unified S3 endpoint that names no provider. Access is a key scoped to one of your principals, and a presigned URL is exactly as powerful as the key that signed it.

Why teams choose it

01

Your tools already speak it

One S3-compatible endpoint means your existing SDKs, CLIs and backup tools work without a provider-specific integration.

02

A key can never exceed its owner

Access keys are scoped to your own principals, so a key does exactly what its owner may do — and nothing more, ever.

03

Sharing without handing out keys

Presigned URLs give time-limited upload or download and inherit exactly the grant of the signing key — a read-only key cannot presign a write.

04

Your bucket names cannot be probed

Another tenant's bucket answers the same 404 as one that does not exist, so nobody can enumerate what you store.

In numbers

1
S3-compatible endpoint per bucket, whoever holds the bytes

404
for another tenant's bucket — the same answer as one that does not exist

0
keys that can do more than the principal that owns them

What it does

Bucket lifecycle

Create, configure and delete buckets as synchronous CRUD, with another tenant's bucket answering the same 404 as one that does not exist, so a name cannot be probed.

Versioning and lifecycle rules are set on the bucket itself, so old versions expire by policy rather than by someone's cleanup script. Buckets are per tenant — from the outside, yours are indistinguishable from ones that do not exist.

Scoped access keys

Keys scoped to your platform principals rather than a provider account, addressing the bucket through its own endpoint and naming no upstream.

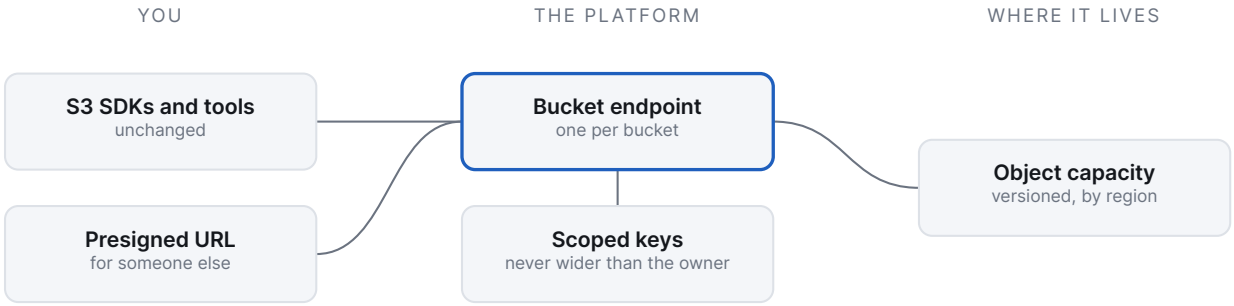
Issue a key to one of your principals and it can do exactly what that principal may do — no shared provider account behind it, no privilege it did not inherit. Rotate or revoke the key and the access ends with it.

Presigned URLs

Time-limited URLs for direct upload and download, inheriting exactly the grant of the key that signed them — a read-only key cannot presign a write.

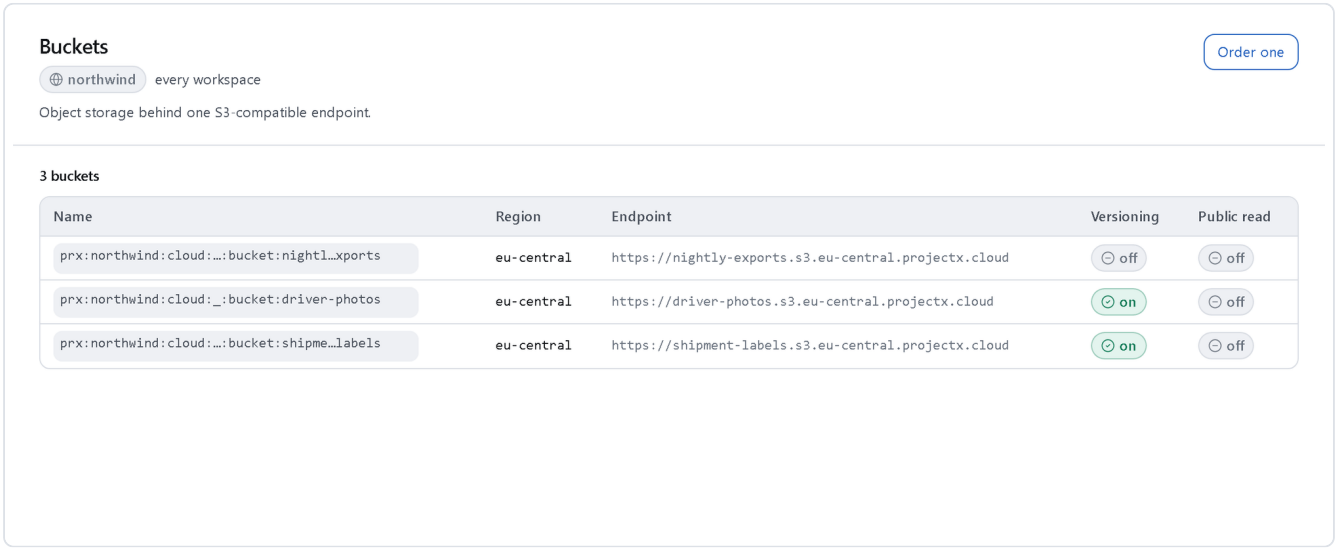
Hand a customer or a partner a URL for one upload or one download, valid for exactly as long as you set. No account to create for them, no credential to claw back afterwards — the link expires and the access ends.

How it fits together

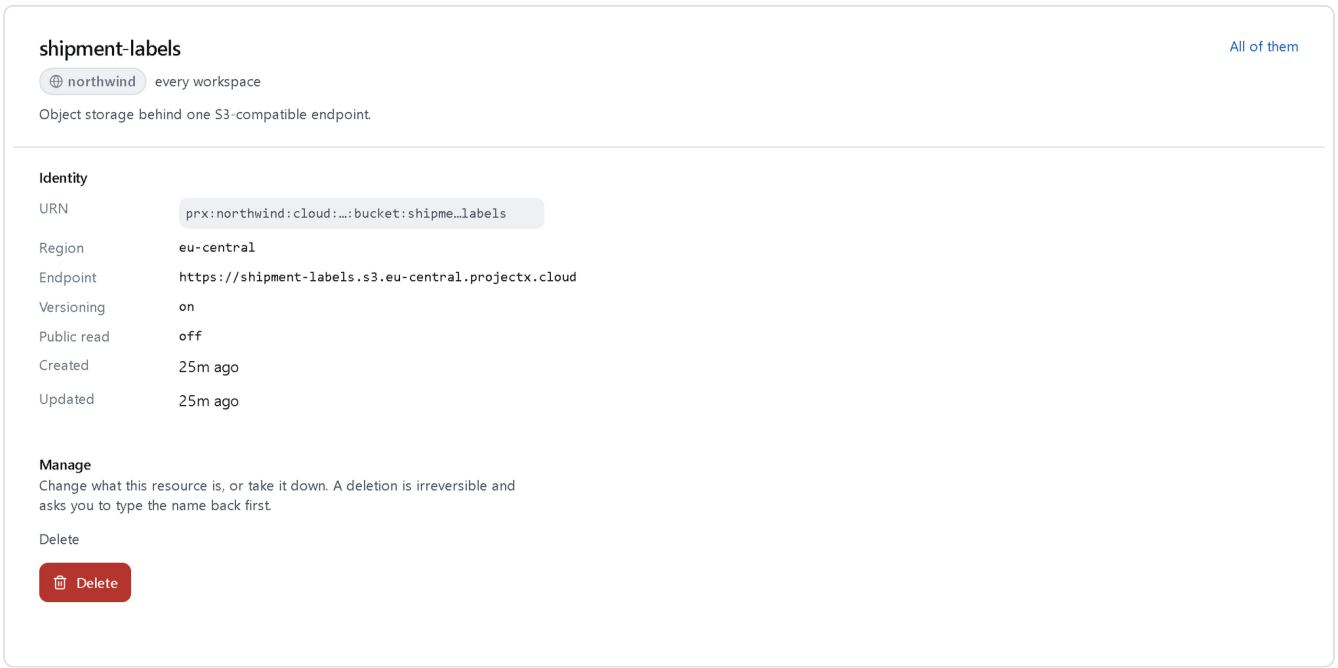


Your S3 tools address one endpoint per bucket. A key scoped to one of your principals signs the request, and the platform checks it against your policy before the object is read or written on the capacity that holds the bucket.

In the console



Every bucket, its region and its versioning, behind one endpoint. Screens shown with demo data.



One bucket: its S3 endpoint, versioning and public access. Screens shown with demo data.

Key ideas

S3-compatible endpoint
The address your SDKs, CLIs and backup tools already know how to talk to. It names your region, not the provider underneath.

Scoped access key
A key issued to one of your own principals, so it can do exactly what that principal may do. Revoke the principal's access and the key loses it too.

Presigned URL
A link for one upload or download that expires when you say. It carries the grant of the key that signed it and no more — a read-only key cannot presign a write.

Versioning and lifecycle rules
Keep every version of an object, and let rules expire old versions or move them to a colder tier on a schedule rather than by hand.

In depth

Security model
Keys belong to your principals, so access is governed by the same policy engine as everything else. Public read is off by default, CORS and bucket policy are set on the bucket, and a bucket name in another tenant cannot be discovered.

Durability and cost
Versioning keeps old copies; lifecycle rules decide how long. Replication to a second provider and an archive tier with retrieval times are there when one copy in one place is not enough.

Sharing and sites
Hand out presigned links instead of accounts, and serve a static site straight from a bucket on its own website address.

How it works

01 **Create a bucket** · Console
Buckets with versioning and lifecycle rules, created as one synchronous call.

02 **Scope the keys** · Console
Access keys belong to your own principals, so a key does exactly what its owner may — nothing more.

03 **Point your tools at one endpoint** · S3 API
Your SDKs, CLIs and backup tools already speak S3 — the unified endpoint names no provider.

04 **Share safely** · API
Presigned URLs give time-limited upload and download, with exactly the power of the key that signed them.

Who it is for

Application teams

You need durable object storage for uploads, assets and exports.

Versioned buckets with lifecycle rules behind an endpoint your stack already speaks.

Teams sharing large files

Customers or partners must upload or download without accounts.

Presigned URLs that expire and carry exactly the access you granted.

Data platforms

A landing zone for data that must not leak between tenants.

Per-tenant buckets whose names cannot even be probed from outside.

Works with

S3 API · S3-compatible tools · TypeScript SDK · Python SDK · Console

Questions buyers ask

Is it really S3-compatible?
Yes — buckets sit behind a unified endpoint your S3 SDKs and tools already speak, with versioning and lifecycle rules on the bucket.

Who can access a bucket?
Only keys scoped to your own principals. A presigned URL inherits exactly the grant of the key that signed it, for exactly as long as you set.

Can another customer see my buckets?
No. A bucket that is not yours answers the same 404 as a bucket that does not exist, so names cannot be enumerated.

How is it billed?
By the gigabyte stored at the published rate, on the one monthly invoice. The rate card is on the pricing page.

Pricing

Current published rates, in your currency: https://adaydev.com/pricing/cloud