

PLATFORM EXTENSION · RELEASE 10.2

REST API

Every account, room, notification and report the portal can touch — reachable by your own systems, on keys scoped to exactly what they need.

The XOP Networks **REST API** extension opens the Universal Service Node to the systems around it. A CRM, a provisioning system, an alarm panel or a custom application drives the platform over JSON on HTTPS: accounts and services provisioned, meeting rooms and stations created, notifications started, contacts and schedules maintained, results and recordings pulled back — each integration holding its own key, limited to the endpoints and the addresses it may use.

2

API generations — v2 for new work, v1 kept for existing integrations

23

resource areas in the v2 reference, from accounts to webcast

0

credentials accepted in a URL — keys and tokens travel in headers only

1

key per integration, each with its own endpoint list and IP allowlist

What it does

Version 2 is the API the portal itself is built on: JSON over HTTPS, one success and one error envelope, pagination and ordering on every list, and a reference page per endpoint in the documentation portal. A request carries an API key in a header; user-level operations add a session token obtained from a login call, so an integration acts as a named account with that account's privileges.

Keys are created on the REST API page: a name, an optional allowlist of IPv4 addresses or CIDR blocks, and a permission set chosen endpoint by endpoint or controller by controller — or All Endpoints, deliberately. The key is shown once, can be disabled without deletion, records its last use, and is bound to the application it serves.

Why it matters

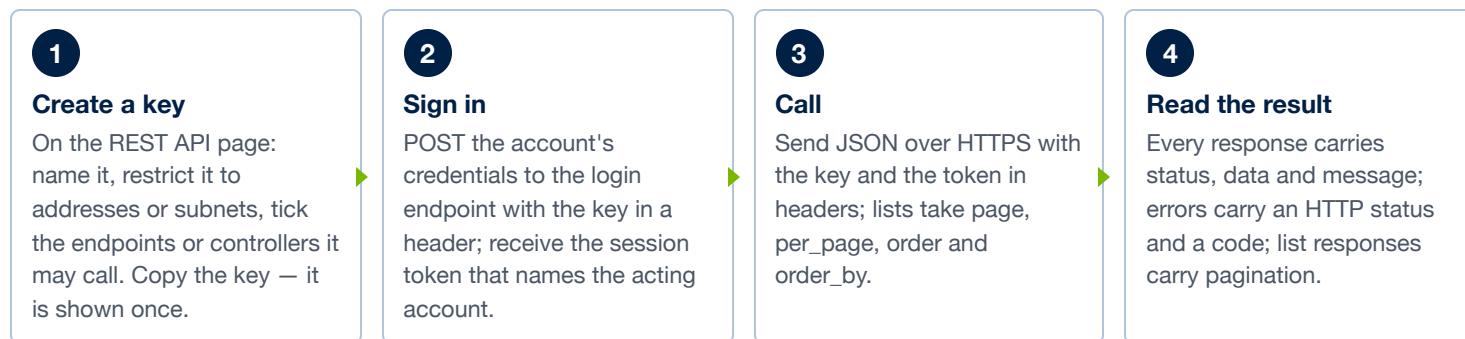
- ✓ Provision accounts, services and contacts from the system that already owns them.
- ✓ Start a notification or a blast from an alarm, a sensor or a ticket.
- ✓ Read results, call records and recordings into your own reporting.
- ✓ Least privilege by construction: an empty permission set cannot be saved.
- ✓ Keys never travel in URLs — headers only, so they stay out of logs and referrers.
- ✓ Everything the portal does is an API call you can make too.

How it works

An integration is a key, a login and a set of calls; the reference in the documentation portal lists every one.



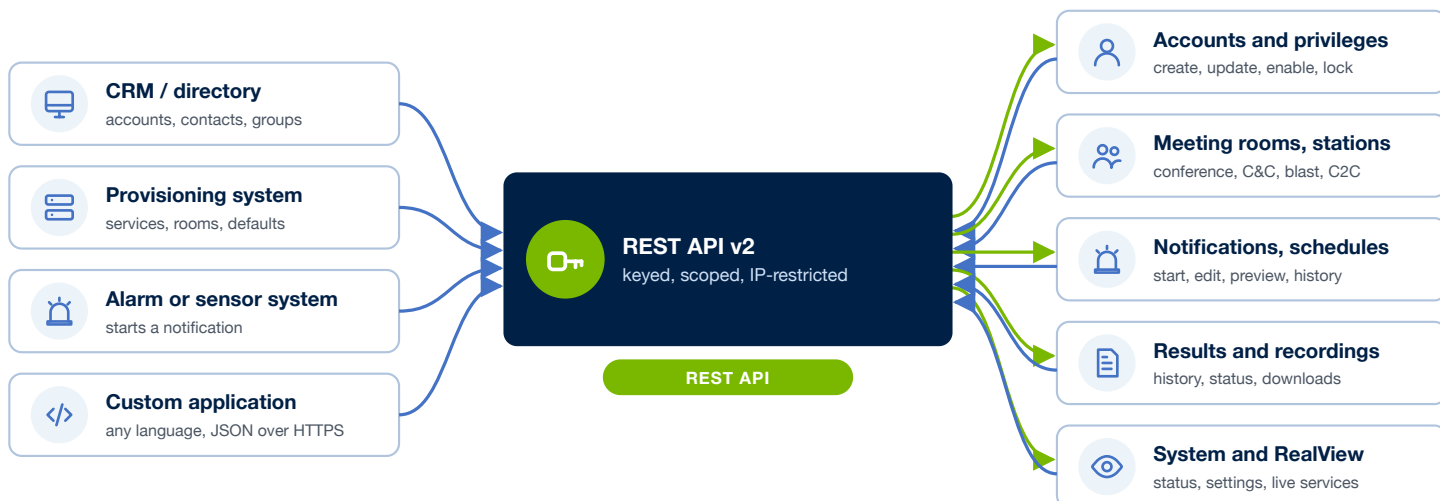
Each system holds its own key; the gateway admits only the endpoints and addresses that key was given, and hands back JSON.



YOUR SYSTEMS

UNIVERSAL SERVICE NODE

WHAT THEY REACH



External systems call in with a scoped key; every response comes back as JSON in the same envelope the portal reads.

Features and benefits

FEATURE	HOW IT WORKS	BENEFIT
Two API generations	v2 (/api/v2/) is current and receives every new feature. v1 (/api/v1/) stays for existing integrations — profile, account, conference, service selection, recordings, click-to-call, mass notification, HA role and state, ports and gateway status — behind its own account name and password.	New work on v2; nothing already built breaks.
Per-application keys	Each key has a name, the application it serves (portal, meeting client, RealView, Click to Call, documentation, Command & Control, calendar add-in), an active flag, created and last-used timestamps, and can be copied again by an administrator from the portal.	Revoke one integration without touching the others.
Endpoint-level permissions	Permissions are chosen per endpoint and HTTP method, per controller, or as All Endpoints; the list is read from the live route table, so it is always complete. A key with nothing selected cannot be saved, and unrestricted access has to be granted explicitly.	An integration that only starts notifications cannot read accounts.
IP allowlists	Single IPv4 addresses and CIDR blocks, comma-separated per key; the client address is taken from the trusted proxy, so a forged forwarding header cannot bypass the restriction.	A leaked key is useless from anywhere else.
Header-only credentials	The API key and the session token are read from request headers only; a key in a query string or a form field is refused with 401. State-changing requests are additionally covered by the platform's CSRF rules.	Credentials stay out of access logs, browser history and referrers.
Full platform coverage	Authentication, accounts, contacts and groups, meeting rooms and meetings, schedules, webcast, recordings, audio announcements, file manager, mass notifications, user events, click-to-call, call replication, command & control, remote control, RealView, tenants and billing — plus the system area: settings, network, licensing, HA, PBX, SNMP, SMSC, maintenance and reports.	If the portal can do it, so can your code.
Consistent envelopes	Success responses carry status, data and message; errors carry an HTTP status — 400, 401, 403, 404, 422, 429, 500 — and a machine-readable code with details; lists page with page, per_page, order and order_by.	One client library covers every endpoint.
Documentation in the portal	The documentation portal carries a REST API section with Getting Started, the v2 overview and error codes, the v1 reference, and a page per endpoint with request, response and curl example; the REST API admin page links to its own guide.	The reference ships with the server it describes.

Technical specifications

Protocol

Transport

HTTPS; JSON request and response bodies; Content-Type: application/json

Authentication

v2 key

Per-application key sent in a request header; never read from the URL or a form field

Base URLs	<code>https://<host>/api/v2/</code> and <code>https://<host>/api/v1/</code>	v2 session	Login with account credentials returns a session token sent in the <code>Authorization</code> header
Methods	GET, POST, PUT, DELETE	v1	Account name and password set on the REST API page; login creates the session
Pagination	<code>page</code> , <code>per_page</code> (default 20, maximum 100), <code>order</code> , <code>order_by</code>	Privileges	Calls act as the logged-in account with its privileges; administration endpoints need System Administrator
Errors	status / error envelope with code, message and details; HTTP status per class of failure		

Key management

Permissions	Per endpoint and method, per controller, or All Endpoints; an empty set is rejected
IP restriction	IPv4 addresses and CIDR blocks; optional
Lifecycle	Create, edit, enable or disable, copy, delete; shown once at creation; last use recorded
Applications	Portal, meeting client, RealView, Click to Call, documentation, Command & Control, calendar add-in

Coverage and licensing

v2 areas	23 resource areas, including a system area with 21 administration modules
v1 endpoints	login, profile, account, conference, service selection, recordings, click-to-call, mass notification, HA role and state, licensed and active ports, SIP stack and gateway state
Licence	X2NMCAPI — <code>web_services</code> feature; v1 refuses every request without it and the REST API page is disabled
Documentation	Documentation portal › REST API; Platform Extensions › REST API for keys

Sample use cases

Service provider provisioning

The order system creates the account, its meeting room and its defaults through v2 the moment a contract is signed, on a key that may call nothing else.

Alarm-driven notification

A building management system starts a prepared mass notification when a sensor trips, then polls the history endpoint for who acknowledged.

Reporting warehouse

A nightly job pulls conference history, notification results and recording lists into the corporate data warehouse on a read-only key limited to the reporting subnet.

One platform, every extension

The REST API is one of the platform extensions of the Universal Service Node. Extensions connect the platform to the systems around it — the directory, the phone system, your own applications; each is licensed on its own.

Call 972-590-0200
www.xopnetworks.com