Kainode



DEVELOPER GUIDE

Reseller API

Order, deliver and manage Kainode VPS and VDS servers from your own billing platform, under your own brand.

Version	Updated	Base URL
v1.10	November 2026	portal.kainode.com/api/reseller/v1

01

KAINODE RESELLER API

Overview

Sell Kainode VPS and VDS servers under your own brand. Your billing system talks to ours; your customers never see Kainode.

Order

Get a price with your discount, order and pay from credit in one call.

Deliver

Fetch each server's IP and root password, matched to the right order.

Control

Start, stop, reboot, reinstall, hostname, root password.

Monitor

Live status, CPU / memory / bandwidth gauges and one-hour charts.

Console

A one-time browser console for your customer. No logins shared.

Billing & support

Invoices, automatic renewals from credit, backups, tickets.

BASE URL	FORMAT
`https://portal.kainode.com/api/reseller/v1`	REST · JSON over HTTPS
ACCESS	**PAYMENT**
API key + your registered server IPs	Prepaid account credit

One rule

The API does exactly what you can do in the Kainode client area, nothing more. Your reseller account is a normal customer account: every order, invoice and server appears there too, under the same billing rules.

02 SETUP Getting started

Approval to first call in five steps.

1. **Get approved.** Kainode enables API access on your account and registers your server's public IP(s).
2. **Verify your email** and **accept the Terms** once on **Account → API**.
3. **Create a key** on the same page. It is shown once; copy it into your server's secret store.
4. **Add credit** in **Account → Credits**. Orders and renewals are paid from it.
5. **Call** `GET /me` from your registered server:

REQUEST

```
curl -s -H 'Authorization: Bearer $KEY' https://portal.kainode.com/api/reseller/v1/me
```

RESPONSE 200

```
{
  "data": {
    "id": 1042,
    "email": "you@example.com",
    "reseller_status": "approved",
    "default_currency": "USD",
    "credits": [{"currency": "USD", "amount": "245.50"}],
    "discount": {
      "code": "RESELLER10",
      "type": "percentage",
      "value": 10,
      "products": [33, 34, 35],
      "renewals": "first_invoice_only"
    },
    "key": {"name": "Billing server"}
  }
}
```

Security

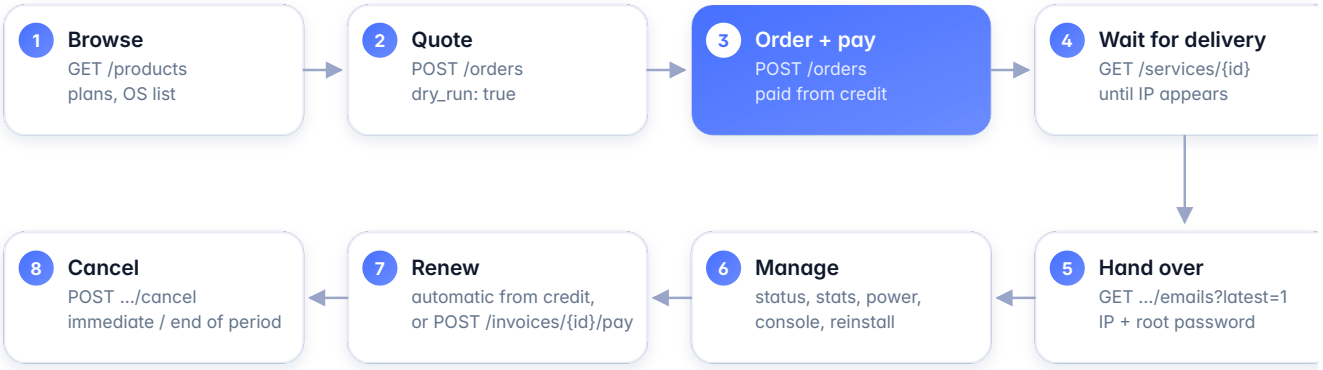
TOPIC	RULE
Header	`Authorization: Bearer <key>` on every request. Keys in URLs are refused.
Allowed IPs	Only your registered IPs are accepted (`403 IP_NOT_ALLOWED` shows the IP we saw). New IP or higher limit: open a ticket.
Your data only	Services, invoices, emails and tickets of other accounts always return `404`.
Key storage	Server-side only. Never in browser code, apps, emails or tickets. Kainode never asks for your key.
Rotation	Create a new key, switch over, delete the old one on Account → API. Up to 5 keys.
Abuse protection	Repeated failed requests temporarily block the sending IP (`429`).

03 WORKFLOW

Integration lifecycle

A few calls cover a server's whole life, from any billing platform.

One server, from first call to cancellation



YOUR PLATFORM DOES	CALL
Show products and prices	`GET /products`, `GET /products/{id}`
Quote with your discount	`POST /orders` + `"dry_run": true`
Create a server	`POST /orders` + `Idempotency-Key` header
Wait for delivery	`GET /services/{id}` every 10–15 s until `server.primary_ipv4` is set
Send login details	`GET /services/{id}/emails?latest=1`
Suspend / unsuspend your customer	`POST /services/{id}/power` `stop` / `start`
Terminate	`POST /services/{id}/cancel` `immediate`
Change password	`POST /services/{id}/reset-password` (returns the new password)
Customer panel buttons	`…/status`, `…/stats`, `…/network`, `…/power`, `…/console`, `…/reinstall`

Works with any platform

WHMCS, Blesta, HostBill, FOSSBilling or your own code: map your platform's create, suspend, unsuspend, terminate and password events to the calls above. Send your own ID as `reference` when ordering; `GET /services?reference=…` finds the service again at any time.

04 ORDERING

Ordering

Quote first, then order with an idempotency key. You can never be charged twice.

GET /PRODUCTS/33 · CHECKOUT FIELDS YOUR CUSTOMER FILLS


```
{
  "data": {
    "id": 33,
    "name": "VDS Pro",
    "category": "VDS-Singapore",
    "in_stock": true,
    "plans": [
      {
        "id": 87,
        "name": "Monthly",
        "billing_unit": "month",
        "price": "16.99",
        "currency": "USD"
      }
    ],
    "checkout_config": [
      {"name": "hostname", "type": "text", "required": true},
      {
        "name": "os",
        "type": "select",
        "required": true,
        "options": {"29": "Ubuntu 24.04", "30": "Debian 12"}
      }
    ]
  }
}
```


ORDER

```
curl -s -X POST https://portal.kainode.com/api/reseller/v1/orders -H 'Authorization: Bearer
$KEY' \
  -H 'Content-Type: application/json' -H 'Idempotency-Key: billing-5531' \
  -d '{"product_id":33,"plan_id":87,"reference":"billing-5531",
       "checkout_config":{"hostname":"vps1.example.com","os":"29"}}'
```

RESPONSE 201 · ORDERED AND PAID


```
{
  "data": {
    "order_id": 761,
    "paid": true,
    "coupon": "RESELLER10",
    "invoice": {"id": 4920, "number": "INV-4888", "status": "paid", "total": "15.29"},
    "services": [{"id": 1024, "status": "active"}]
  }
}
```


What to send

FIELD	NEEDED?	WHAT IT MEANS
`product_id`	Yes	Which server product to buy (from `GET /products`)
`plan_id`	Yes	Which billing period: daily, monthly or yearly (from the same product)
`checkout_config`	Yes for servers	Your customer's choices: `hostname` (e.g. `vps1.example.com`) and `os` (the number of the OS option, e.g. `29` = Ubuntu 24.04)
`config_options`	Only if the product has extras	Extra options listed on the product, as `{"option id": choice}`
`reference`	Optional, recommended	Your own order number, so you can find this server later with `GET /services?reference=…`
`dry_run`	Optional	Send `true` to only see the final price. Nothing is ordered or charged

Idempotency-Key header: protection against double orders

If your connection drops during an order, you can't know whether it went through. Send a unique value in the `Idempotency-Key` header (your own order number is ideal) and send the **same value** if you retry. Kainode then creates the order only once and returns the first result again, so your customer is never charged twice.

- New order = new value. Retry of the same order = same value.
- Using a value again for a **different** order is refused (`422 IDEMPOTENCY_KEY_REUSED`).

Price and credit

- Your discount coupon, if any, is applied automatically to the products it covers; others are charged normally.

- Not enough credit: `402 INSUFFICIENT_CREDIT` and nothing is created.
- `201` = ordered and paid. `202` = ordered, payment pending: pay with `POST /invoices/{id}/pay`.

05 OPERATIONS

Delivery & server management

Login details for each server, then every button of the client area.

How the server email reaches your customer



GET /SERVICES/1024/EMAILS?LATEST=1

```
{
  "data": {
    "id": 16134,
    "service_id": 1024,
    "subject": "Service activated",
    "status": "sent",
    "body_html": "<!DOCTYPE html> … branded email …",
    "body_text": "Service activated … IP Address: 203.0.113.25 … Root password: ••••••••"
  }
}
```

- `body_text` is the easiest to read: take the root password from it, and the IP from `GET /services/{id}`, then send your customer your own branded email.
- `POST /services/{id}/reset-password` returns the new password directly.
- Email history is kept for a limited time: store delivery details as soon as a server is ready.

Re-deliveries (reinstall by Kainode, failed setup, custom OS)

When Kainode support reinstalls or re-delivers a server, the new login email is sent to your account and linked to the same service. To pass it on automatically, poll `GET /emails?since=<last check>` every few minutes: any email with a `service_id` belongs to that server. Fetch it with `GET /emails/{id}` and send your customer the new details.

GET /SERVICES/1024/STATUS · LIVE GAUGES


```
{
  "data": {
    "status": "running",
    "cpu_percent": 4.1,
    "memory_percent": 13.5,
    "bandwidth": {"used_bytes": 16106127360, "limit_bytes": 3221225472000, "percent": 0.5},
    "mem": 872833024,
    "maxmem": 6442450944,
    "maxdisk": 85899345920,
    "uptime": 86400
  }
}
```


Giving your customer a console without sharing any login



ACTION	CALL	NOTES
Power	`POST …/power` `start | stop | reboot | shutdown`	`shutdown` is graceful, `stop` cuts power
Console	`POST …/console`	Returns a one-time URL, valid 5 minutes. Create it only when your customer clicks
Charts	`GET …/stats`	Last hour, one point per minute; poll once a minute at most
Network	`GET …/network`	Address, gateway, netmask
Hostname	`POST …/hostname` `{"hostname"}`	
Reinstall	`GET …/os`, `POST …/reinstall` `{"os_id", "confirm": true}`	Erases all data
Backups	`GET / POST / DELETE …/backups`, `POST …/backups/restore`	If the plan includes backups

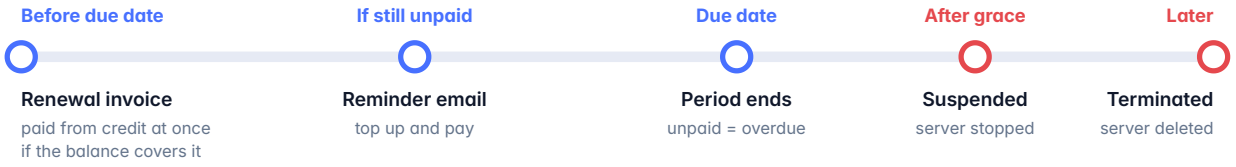
06

BILLING

Billing & renewals

Prepaid credit. Renewals follow the same billing rules as every Kainode customer.

Renewal timeline: the same billing rules as every Kainode customer



- Renewal invoices are created before the due date and paid from your credit automatically when the balance covers them.
- Invoice, reminder, suspension and termination timing follow Kainode's billing settings for all customers; each invoice and service shows its own dates (`due_at` , `expires_at`).
- Check `GET /invoices?status=pending` daily and pay anything open with `POST /invoices/{id}/pay` .
- Your customers' own non-payment is handled in your platform: stop or cancel the server as you decide.

Tip

Keep enough credit for upcoming renewals, and run your daily invoice check at a time other than midnight server time.

07

REFERENCE

Endpoint reference

All 31 calls. Lists accept `?page=` and `?per_page=` (max 100).

METHOD	PATH	PURPOSE / BODY
GET	`/me`	Account, credit, discount
GET	`/products`	Products for sale, plans, prices
GET	`/products/{id}`	One product, options, checkout fields
POST	`/orders`	Quote (dry_run) or order and pay
GET	`/services`	Your services · ?status= ?reference=
GET	`/services/{id}`	Service with server details
GET	`/services/{id}/status`	Live state and gauges
GET	`/services/{id}/stats`	Charts, last hour
GET	`/services/{id}/network`	IP, gateway, netmask
POST	`/services/{id}/power`	action: start, stop, reboot, shutdown
POST	`/services/{id}/console`	One-time console URL
POST	`/services/{id}/hostname`	hostname
POST	`/services/{id}/reset-password`	New root password
GET	`/services/{id}/os`	OS list for reinstall
POST	`/services/{id}/reinstall`	os_id, confirm: true
POST	`/services/{id}/cancel`	type: immediate | end_of_period, reason?
GET	`/services/{id}/backups`	Backups and plan limit
POST	`/services/{id}/backups`	name, type?, compression?
POST	`/services/{id}/backups/restore`	backup_id, confirm: true
DELETE	`/services/{id}/backups`	backup_id
GET	`/services/{id}/emails`	Emails of this service · ?latest=1

METHOD	PATH	PURPOSE / BODY
GET	`/emails`	All account emails · ?since= ?subject=
GET	`/emails/{id}`	One email with body
GET	`/invoices`	Invoices · ?status=pending
GET	`/invoices/{id}`	Invoice with items
POST	`/invoices/{id}/pay`	Pay in full from credit
GET	`/tickets/departments`	Departments and priorities
GET	`/tickets`	Your tickets
POST	`/tickets`	subject, message, priority, department, service_id?
GET	`/tickets/{id}`	Ticket with messages
POST	`/tickets/{id}/reply`	message

Conventions

Responses: `{"data": …}` , lists add `meta` . Money is a string with two decimals. Times are ISO 8601 with offset. Bytes are raw bytes; network and disk rates are bytes per second. Tickets are closed by Kainode support.

08 RELIABILITY

Errors & limits

Every error has a stable code. Act on the code, retry only where it is safe.

ERROR SHAPE

```
{
  "error": {
    "code": "VALIDATION_ERROR",
    "message": "The given data was invalid.",
    "details": {
      "checkout_config.hostname": ["The checkout config.hostname field format is invalid."]
    }
  }
}
```

HTTP	CODE	MEANING
400	IDEMPOTENCY_KEY_REQUIRED	Order sent without the header
401	UNAUTHENTICATED	Key missing, wrong or deleted, or account not active
402	INSUFFICIENT_CREDIT	Top up; nothing was created
403	IP_NOT_ALLOWED · EMAIL_NOT_VERIFIED · TOS_NOT_ACCEPTED	Fix access as named
404	NOT_FOUND · NOT_AVAILABLE	Not yours / feature not in this plan
409	SERVICE_NOT_ACTIVE · NO_SERVER · NOT_CANCELLABLE · NOT_PAYABLE · BACKUP_LIMIT · IN_PROGRESS	State does not allow it yet
422	VALIDATION_ERROR · ORDER_REJECTED · IDEMPOTENCY_KEY_REUSED · INVALID_OS	Fix the request; `details` names fields
429	RATE_LIMITED · TOO_MANY_* · BUSY · BACKUP_IN_PROGRESS	Wait `Retry-After` seconds, then retry
5xx	SERVER_ERROR · UPSTREAM_ERROR · ACTION_FAILED	Retry later; nothing was charged

LIMIT	VALUE
Requests	Per minute per account, shown on Account → API (headers `X-RateLimit-*`)
Orders	One at a time per account
Per server	Power 10/min · console 10/min · password reset 3/10 min · reinstall 5/hour · new backup 1/15 min
Tickets	1 new ticket per 30 s · 10 replies per minute

09

HELP

Support

Real people, same team that runs the servers.

CHANNEL	DETAILS
Tickets	Client area at portal.kainode.com, or `POST /tickets`
Email	support@kainode.com
Phone	+1 (307) 855-1647
Live chat & help	kainode.com/contact

Fastest help

Include the endpoint, time, HTTP status and error `code`. Never include your API key: Kainode will never ask for it.