

Deployment & Segmentation

This chapter explains the heart of the module: how a service is deployed, why you split your infrastructure across servers, and the three deployment models (Split, Unified, Vanity) you can sell. Read it before building products — the choices here decide how your hosting scales and how your costs and isolation trade off.

- [Deployment Models \(Split · Unified · Vanity\)](#)
- [Server Segmentation \(Web / Mail / DNS tiers\)](#)
- [How Deployment Works](#)

Deployment Models (Split · Unified · Vanity)

PUQ Web Hosting module **WHMCS**

[Order now](#) | [Download](#) | [Community](#)

Every product you sell has a **Deployment mode** set on the product's *Module Settings* → *General* tab. It decides **how many HestiaCP accounts** back each service and **where** the web, mail and DNS roles live. This is the single most important product decision.

The screenshot shows the WHMCS interface for the PUQ Web Hosting module. The 'Deployment mode' section is expanded, showing three options: Split, Unified, and Vanity. The Vanity option is selected. Below the options, there is detailed text explaining how each mode works and how role gating is implemented.

Deployment mode

Split (one Hestia user per role; web / mail / dns can live on different servers) Unified (single Hestia user holds web + mail; DNS joins the same account when the server is in the DNS cluster)

Vanity (sell name-<your-domain> sites + name@<your-domain> mailboxes on a domain YOU own)

Split is the classic mode and supports any group composition. Unified requires the chosen server group to contain at least one server with both *Web* and *Mail* capabilities ticked (Addon → Servers → Edit → Capabilities). Customer-facing change: a single *Account backups* quota replaces the separate Web / Mail backup counts. The mode is locked at `_CreateAccount` time — switching the product of an existing service between split and unified is refused.

Vanity sells slots on a domain you own (configured under Addon → Server Groups → Edit → *Vanity domains*). The buyer picks the parent domain via a Configurable Option named `vanity_domain` and enters their name in a Custom Field named `vanity_name`; the service then gets a normal per-service web account for `name.<domain>` and ONE mailbox `name@<domain>` on your shared mail user. The module never creates or deletes the parent domain / zone / your provider user. Set the per-order site and mailbox sizes on the *Vanity limits* tab.

How role gating works: A role is *enabled* for a service when BOTH the package-level checkbox is ticked AND the corresponding `disk_quota` resolves to a non-zero value. Setting `web_disk_quota=0` (or `mail_disk_quota=0`) via the Web/Mail tab or via a WHMCS Configurable Option behaves exactly like un-ticking the box here — the role is skipped entirely:

- Provisioner skips `createpackage` / `createuser` / `addomain` for the disabled role.
- Client area hides the matching tabs (Mailboxes / Databases / FTP / Cron / Web settings / Custom SSL / Backups) and dashboard cards.
- Admin inline panel skips role-specific pills.
- DNS is never gated — DNS records are managed locally regardless of which Hestia roles are active. If the server group has no DNS servers attached, DNS provisioning is silently skipped.

At least one of *Web* / *Mail* must be ticked — provisioning errors otherwise.

The three modes

Mode	Hestia accounts per service	Best for
------	-----------------------------	----------

Split	A separate Hestia user per role — one web user, one mail user, one DNS user (each can live on a different server).	Classic shared/business hosting where you want maximum isolation and the freedom to put web, mail and DNS on different, independently-scaled servers.
Unified	One Hestia user holds web and mail (and local DNS) for the service.	Simpler, denser, cheaper hosting where web and mail naturally live together on one node.
Vanity	A per-service web user for one subdomain + one mailbox on your shared provider mail account.	Selling <code> name.yourdomain.com </code> websites and <code> name@yourdomain.com </code> mailboxes on a domain <i>you</i> own — see the dedicated Vanity Mode chapter.

Split deployment

In Split mode the module provisions up to three independent HestiaCP users for the service — for example `|customer-com-web|`, `|customer-com-mail|` and `|customer-com-dns|`. Each is placed on a server that has the matching capability, so the **website**, the **mailboxes** and the **DNS zone** can sit on completely different machines.

The admin service panel shows this clearly — a **Web & DNS** card and a separate **Mail** card, each with its own Hestia user, server and certificate:

The screenshot displays the HestiaCP admin interface for the domain `test.puq.info`. It features two main service cards: **WEB & DNS** and **MAIL**.

- WEB & DNS Card:** Shows separate Hestia users for `test-puq-info-web` (ISP: `isp-web02-test.uuq.pl`) and `test-puq-info-dns` (ISPs: `isp-ns1-test.uuq.pl`, `isp-ns2-test.uuq.pl`). It also displays SSL status (456 fails), hosts required, and resource usage for web disk, bandwidth, DNS records, databases, FTP, cron jobs, and web backups.
- MAIL Card:** Shows a separate Hestia user for `test-puq-info-mail` (ISP: `isp-mx01-test.uuq.pl`). It displays SSL status (467 fails), hosts required, and resource usage for mail disk, mail accounts, and mail backups.

Below the service cards is a navigation menu with options like Overview, Web settings, SSL, FTP, Databases, DNS, Mailboxes, Cron, Backups, Deploy, Tasks, and Logs. The **Databases** section is active, showing a table with columns: ID, Name, User, Engine, Charset, Status, Created, and Error. The table is currently empty, displaying "No data available in table".

At the bottom, there is an **Addons** section with a table header: Reg Date, Name, Pricing, Status, Next Due Date. It currently shows "No Records Found".

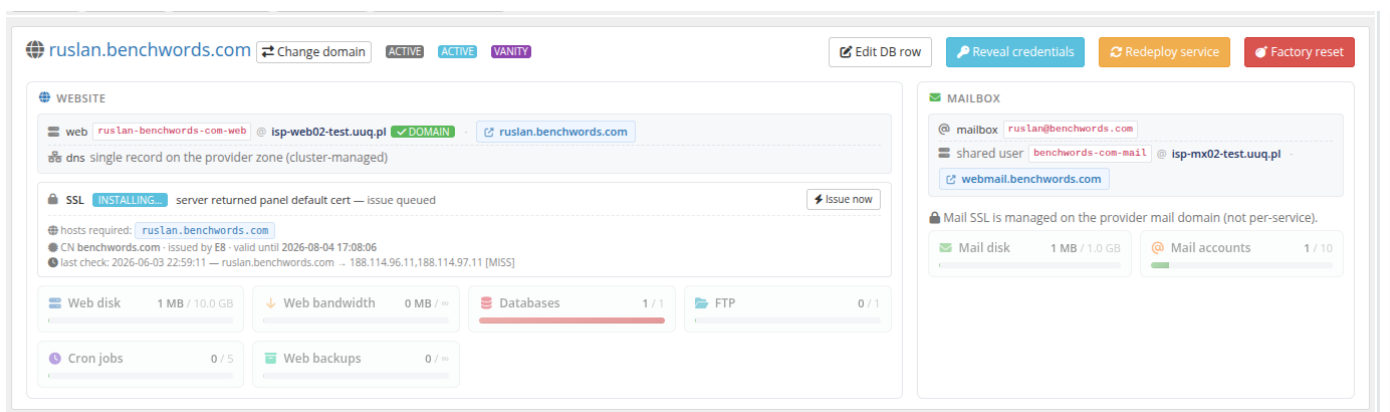
Split is the mode to choose when you want to **segment your fleet by role** (the next page) and keep mailboxes off your web servers.

Unified deployment

In Unified mode a single Hestia user owns the website and the mailboxes, so web and mail are always co-located. There is no separate mail user to manage. Unified products use a single combined account package and a single set of backups (`role = all`). Choose Unified when you run general-purpose nodes that do both jobs and you want fewer accounts to manage.

Vanity deployment

Vanity is a different business model rather than a different server layout: you own a domain (e.g. `benchwords.com`) and sell **slots** on it. Each order becomes `name.benchwords.com` (a normal per-service web account) plus `name@benchwords.com` (one mailbox on your **shared** provider mail user). The parent domain, its DNS zone and the provider mail account are **never** modified per order — the model is destructive-safe by design.



Vanity has its own chapter because the setup (sellable parent domains, reserved names, the order flow and a drop-anywhere shop widget) is substantial — see **Vanity Mode**.

Which roles does a product include?

Independently of the mode, the product's *General* tab lets you tick which **roles** the package includes — **Web, Mail, DNS**. Unticking a role (or setting its disk quota to `0` in the limits tab) simply omits it. For example, a "mail-only" product ticks Mail and leaves Web off; a "website-only" product ticks Web and DNS.

The limits for each ticked role are configured on the matching *Web limits / Mail limits / DNS limits* tab (or the single *Vanity limits* tab in Vanity mode) — see **Installation & Configuration** → **Create a product**.

“ **Rule of thumb:** Split = isolation + segmentation; Unified = density + simplicity; Vanity = a productised "personal site + email" offer on your own domain. You can sell all three side-by-side — they coexist on the same servers.

Server Segmentation (Web / Mail / DNS tiers)

PUQ Web Hosting module **WHMCS**

[Order now](#) | [Download](#) | [Community](#)

One server or many?

You can run the whole module on a **single** HestiaCP node that does everything. That is perfect for getting started, for small offers, and for testing.

But the module is designed so you can **segment** your fleet as you grow: put **websites** on one set of servers, **mailboxes** on dedicated **mail (MX)** servers, and **DNS** on separate **nameservers**. Each role then scales on its own hardware, and a busy mail queue or a heavy website can't starve the others.

The mechanism that makes this possible is **server capabilities** + **server groups**.

Capabilities — what a node is allowed to host

Each server you add (Infrastructure → *Web / Mail / DNS Servers*) has three capability flags: **Web**, **Mail**, **DNS**. You tick them when you add or edit the node. A node can carry one role, two, or all three.

The OPcache PHP extension is enabled. This extension can cause problems with cached data use and PHP script execution in WHMCS. We recommend disabling OPcache. [Learn More](#)

PUQ Web Hosting

Web Servers

ID	Server
2	isp-web01-test.uuq.pl
3	isp-web02-test.uuq.pl

Edit Web Server

Capabilities *
 Web Mail DNS
 Tick every role this physical server is provisioned for. Unified-deployment products need a server with both Web and Mail ticked. PowerDNS supports DNS only.

Driver * hestiacp **Status** active

Name * isp-web01-test.uuq.pl **Group** pl

Hostname * isp-web01-test.uuq.pl **IP *** 77.87.125.156 **SSH port** 22

SSH username * admin **SSH auth method *** Password Private key
 Either **root** (recommended), or any user with full passwordless sudo (**NOPASSWD: ALL**).

SSH password
 leave blank to keep existing

Requirements on the HestiaCP server:

- SSH access enabled on port **22** (or custom port above).
- SSH user can log in via password or private key.
- FULL passwordless sudo is required.** The module runs both Hestia **v-*** commands and system probes (**df**, **/proc/***, custom scripts) — partial NOPASSWD scoped only to **/usr/local/hestia/bin/*** is not enough.

Two ways to satisfy this:

- (a) SSH as **root** — easiest, no sudoers config needed. The module auto-detects **uid=0** and skips **sudo** entirely.
- (b) Use a **sudoer user** — create **/etc/sudoers.d/puq-webhosting** on the Hestia server:

```
sudo tee /etc/sudoers.d/puq-webhosting <<'EOF'
# Full passwordless sudo for PUQ Web Hosting integration.
# Replace "admin" with the SSH username configured above.
admin ALL=(ALL) NOPASSWD: ALL
EOF
sudo chmod 440 /etc/sudoers.d/puq-webhosting
sudo visudo -c # syntax check - must say "parsed OK"
```

Verify it works: **sudo -n -l** on the server must list a **NOPASSWD: ALL** rule.

4. For **private-key** auth — public key in **~/.ssh/authorized_keys** of the SSH user, perms **600**.

This is how you decide your topology. Typical patterns:

Pattern	Web nodes	Mail nodes	DNS nodes
All-in-one	one node ticks Web + Mail + DNS	(same node)	(same node)
Web / mail split	nodes tick Web only	dedicated Mail nodes (your MX)	nodes tick DNS only
Full three-tier	Web pool	Mail pool	Nameserver pool (e.g. <code>ns1</code> , <code>ns2</code>)

In the Infrastructure pages the same physical servers appear under **Web Servers**, **Mail Servers** and **DNS Servers** filtered by their capability — so a mail-only node only shows up under Mail:

PUQ Web Hosting

Home Services Statistics **Infrastructure** Logs

Settings Help v1.0

Web Servers

All groups + Add Web Server

ID	Server	Group	Status	Capacity	Actions
2	<p>isp-web01-test.uuq.pl isp-web01-test.uuq.pl:22 HESTIACP OK</p> <p>2026-06-03 21:11</p> <p>CPU 1.2% RAM 51.7% Disk 47% Load 6%</p> <p>v1.9.4 Ubuntu 3 4 26/26 1</p>	PL	ACTIVE	3 / 0	
3	<p>isp-web02-test.uuq.pl isp-web02-test.uuq.pl:22 HESTIACP OK</p> <p>2026-06-03 21:12</p> <p>CPU 3.4% RAM 41.7% Disk 11% Load 4%</p> <p>v1.9.6 Debian 6 7 25/25 1</p>	PL	ACTIVE	6 / 0	

PUQ Web Hosting

Home Services Statistics **Infrastructure** Logs

Settings Help v1.0

Mail Servers

All groups + Add Mail Server

ID	Server	Group	Status	Capacity	Actions
8	<p>isp-mx01-test.uuq.pl isp-mx01-test.uuq.pl:22 HESTIACP OK</p> <p>2026-06-03 21:12</p> <p>CPU 3.4% RAM 29.2% Disk 22% Load 2%</p> <p>v1.9.6 Debian 4 5 16/16 1</p>	PL	ACTIVE	4 / 0	
9	<p>isp-mx02-test.uuq.pl isp-mx02-test.uuq.pl:22 HESTIACP OK</p> <p>2026-06-03 21:12</p> <p>CPU 2.3% RAM 49.8% Disk 10% Load 5%</p> <p>v1.9.6 Debian 2 2 16/16 1</p>	PL	ACTIVE	2 / 0	

PUQ Web Hosting

Home Services Statistics **Infrastructure** Logs

Settings Help v1.0

DNS Servers

All drivers + Add DNS Server

DNS servers are independent — attach them to groups from Group → DNS servers tab. One DNS server can serve many groups.

ID	Server	Status	Actions
10	<p>isp-ns1-test.uuq.pl isp-ns1-test.uuq.pl:22 OK</p> <p>2026-06-03 21:12</p> <p>CPU 6.7% RAM 37.3% Disk 10% Load 8%</p> <p>v1.9.6</p>	ACTIVE	
11	<p>isp-ns2-test.uuq.pl isp-ns2-test.uuq.pl:22 OK</p> <p>2026-06-03 21:12</p> <p>CPU 6.8% RAM 36.7% Disk 10% Load 12%</p> <p>v1.9.6</p>	ACTIVE	

When a **Split** service is provisioned, the module places each role on a **least-loaded active node that has the matching capability**: the web user lands on a Web node, the mailbox on a Mail node, the DNS zone on the group's DNS nodes. That is the load balancing — it is **by role and by capacity**, automatically.

Server groups — the unit you sell

from

A **server group** ties a set of nodes together and is what a product points at (the product's *Server Group* dropdown). Open a group to manage it.

PUQ Web Hosting

Server	Type	Status	Last collected	
isp-mx01-test.uuq.pl isp-mx01-test.uuq.pl	MAIL	OK	2026-06-03 21:12:13	Open
isp-mx02-test.uuq.pl isp-mx02-test.uuq.pl	MAIL	OK	2026-06-03 21:12:23	Open
isp-web01-test.uuq.pl isp-web01-test.uuq.pl	WEB	OK	2026-06-03 21:11:46	Open
isp-web02-test.uuq.pl isp-web02-test.uuq.pl	WEB	OK	2026-06-03 21:12:00	Open

The group's **Actions** tab shows the members with their **Type** (WEB / MAIL) and lets you push configuration to **all servers**, **web servers only**, **mail servers only** or **DNS servers only**. That last point is the key to safe segmentation: a web-specific setting is **never** applied to a mail node and vice-versa.

Role-targeted configuration

The group editor has separate config tabs whose values are pushed **only** to nodes of that role during *Apply Hestia config*:

- **All-servers config** — keys applied to every node (language, theme, security/API, user policies...).
- **Web config** — web-only keys (Apache/Nginx ports, web backend, phpMyAdmin alias...).

Settings applied only to web (Hestia) servers in this group. Pushed via `v-change-sys-config-value KEY VALUE` during the same "Apply Hestia config" action — the worker filters keys by category target.

Web Server

Apache port `WEB_PORT`
(not managed)

Default 8080. Behind nginx proxy.

Apache SSL port `WEB_SSL_PORT`
(not managed)

Default 8443.

Nginx (proxy) port `PROXY_PORT`
(not managed)

Default 80 — public HTTP.

Web backend `WEB_BACKEND`
php-fpm

Sync error pages on rebuild `POLICY_SYNC_ERROR_DOCUMENTS`
yes

Sync skeleton on rebuild `POLICY_SYNC_SKELETON`
yes

Databases

phpMyAdmin alias `DB_PMA_ALIAS`
(not managed)

Default "phpmyadmin" — /phpmyadmin URL.

phpPgAdmin alias `DB_PGSQL_ALIAS`
(not managed)

- **Mail config** — mail-only keys (SMTP/IMAP system, antispam/antivirus, webmail, notification from-address...).

Settings applied only to mail (Hestia) servers in this group. Pushed via `v-change-sys-config-value KEY VALUE` during the same "Apply Hestia config" action — the worker filters keys by category target, so mail-only keys never land on web servers and vice-versa.

Mail

Mail (SMTP) system `MAIL_SYSTEM`
(not managed by group)

Outbound SMTP / inbound MTA, Hestia ships exim4.

IMAP/POP3 system `IMAP_SYSTEM`
(not managed by group)

Mailbox access daemon, Hestia ships Dovecot.

Sieve filters (ManageSieve) `SIEVE_SYSTEM`
(not managed by group)

Server-side mail filtering rules (RFC 5228).

Antispam `ANTISPAM_SYSTEM`
(not managed by group)

Spam filter applied to inbound mail.

Antivirus `ANTIVIRUS_SYSTEM`
(not managed by group)

Virus scanner applied to inbound mail attachments.

Webmail alias `WEBMAIL_ALIAS`
(not managed)

Subdomain prefix, e.g. "webmail" — webmail.domain.com

Webmail system `WEBMAIL_SYSTEM`
(not managed by group)

Use server SMTP for system mail `USE_SERVER_SMTP`
(not managed by group)

Route Hestia's own notifications through local exim.

Notifications from-email `FROM_EXIM`
(not managed)

Notifications from-name `FROM_NAME`
(not managed)

- **DNS config** — DNS-only keys (DNSSEC, DNS cluster). PowerDNS nodes have no `hestia.conf` and are skipped automatically.

The worker filters every managed key by its category target, so "mail-only keys never land on web servers and vice-versa". This means you can, for instance, enable ClamAV/SpamAssassin only on your mail tier without touching the web tier.

DNS is active-active across the tier

DNS servers are **independent** — you attach them to a group on the group's **DNS servers** tab, and **one DNS server can serve many groups**. When a zone is created it is **replicated to every attached DNS node**, so you get an active-active nameserver cluster out of the box. The **DNS Zones** page shows the per-server deploy state for each logical zone:

ID	Zone	Service	Cluster	Records	Actions
36	test.appuw.com	test.appuw.com	ISP-NS1-TEST.UUQ.PL: ACTIVE ISP-NS2-TEST.UUQ.PL: ACTIVE	25	Redeploy
34	test.puq.info	test.puq.info	ISP-NS1-TEST.UUQ.PL: ACTIVE ISP-NS2-TEST.UUQ.PL: ACTIVE	25	Redeploy
35	test2.puq.info	test2.puq.info	ISP-NS1-TEST.UUQ.PL: ACTIVE ISP-NS2-TEST.UUQ.PL: ACTIVE	25	Redeploy

Watching the balance

The addon **Statistics** page reports usage **per domain, split into Web and Mail columns**, so you can see at a glance how load is distributed and when a tier needs another node:

Domain	Status	Web							Mail			Updated
		Disk	Bandwidth	DNS rec	DBs	FTP	Cron	Backups	Disk	Accounts	Backups	
ddd.benchwords.com	ACTIVE	1 MB / 1.0 GB 0%	0 / ∞	0 / 50 0%	0 / 1 0%	0 / 1 0%	0 / 5 0%	0 / ∞	1 MB / 10.0 GB 0%	1 / 10 10%	0 / ∞	2026-06-03 21:12
ddfsq.benchwords.com	ACTIVE	1 MB / 1.0 GB 0%	0 / ∞	0 / 50 0%	0 / 1 0%	0 / 1 0%	0 / 5 0%	0 / ∞	1 MB / 512 MB 0%	1 / 10 10%	0 / ∞	2026-06-03 21:12
ruslan.benchwords.com	ACTIVE	1 MB / 10.0 GB 0%	0 / ∞	0 / 50 0%	1 / 1 100%	0 / 1 0%	0 / 5 0%	0 / ∞	1 MB / 1.0 GB 0%	1 / 10 10%	1 / ∞	2026-06-03 21:12
test.appuw.com	ACTIVE	1 MB / 1.0 GB 0%	0 / ∞	25 / 500 5%	2 / 5 40%	0 / 10 0%	0 / 10 0%	9 / 5 180%	1 MB / 10.0 GB 0%	0 / 10 0%	5 / 2 250%	2026-06-03 21:12
test2.puq.info	ACTIVE	1 MB / 1.0 GB 0%	0 / ∞	25 / 500 5%	0 / 5 0%	0 / 10 0%	0 / 10 0%	12 / 5 240%	1 MB / 10.0 GB 0%	0 / 10 0%	8 / 2 400%	2026-06-03 21:12
test.puq.info	ACTIVE	1 MB / 1.0 GB 0%	0 / ∞	25 / 500 5%	0 / 5 0%	1 / 10 10%	0 / 10 0%	12 / 5 240%	1 MB / 10.0 GB 0%	0 / 10 0%	8 / 2 400%	2026-06-03 21:12

The Home dashboard's **Server capacity** table lists each node by type with its capacity and last sync time — a quick health view of every tier.

“ **Why segment?** Mail and web have very different load profiles. Keeping them on separate, capability-tagged tiers lets you scale each independently, isolate failures, and apply role-specific tuning — while the module's per-capability

placement spreads new accounts across the least-loaded node automatically.

How Deployment Works

PUQ Web Hosting module **WHMCS**


[Order now](#) | [Download](#) | [Community](#)

Orders return instantly — work happens in the background

When a service is created (a WHMCS order goes active, an admin clicks **Create**, or the API is called) the module does **not** try to build everything inside the HTTP request. It writes a service record, returns `|success|` to WHMCS immediately, and lets the **cron-driven task queue** do the real work.

This is what the customer sees right after ordering — a live deployment splash that polls for progress and refreshes itself:

- ★ Overview ^
- i Information
- 🔧 Actions ^
- 🔄 Request Cancellation



Vanity
HOSTING

ACTIVE

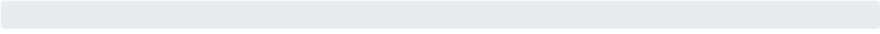
Request Cancellation

Registration Date
Wednesday, June 3rd, 2026
Recurring Amount
\$6.33
Billing Cycle
Annually
Next Due Date
Thursday, June 3rd, 2027
Payment Method
PayPal

- 🌐 Manage
- ⚙️ Configurable Options
- i Additional Information

🚧 Service is being deployed
Domain: ruslan2.benchwords.com

What's happening now: Configuring web plan



Progress: 0 of 4 steps completed · 4 pending

🕒 This page refreshes automatically. Full deployment typically takes 5-15 minutes.

▶ Recent activity

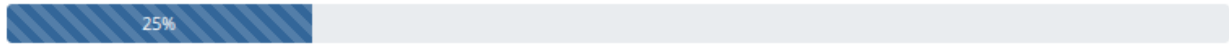
Powered by WHMCompleteSolution

Each step is a queued **task** processed by the cron runner. As steps complete, the bar advances and the recent-activity feed updates:

Service is being deployed

Domain: ruslan2.benchwords.com

What's happening now: Setting up web account



Progress: 1 of 4 steps completed · 1 in progress · 2 pending

This page refreshes automatically. Full deployment typically takes 5-15 minutes.

Recent activity

Stage	State	Updated
Configuring web plan	done	2026-06-03 23:41:32
Setting up web account	in progress	2026-06-03 23:41:32
Configuring web hosting	queued	2026-06-03 23:41:10
Creating mailbox	queued	2026-06-03 23:41:10

When the last step finishes the page flips to the live control panel:

Manage Configurable Options Additional Information

Service is being deployed
Domain: ruslan2.benchwords.com

What's happening now: Deployment finished — refreshing...

100%

Progress: 5 of 5 steps completed

This page refreshes automatically. Full deployment typically takes 5-15 minutes.

Recent activity

Stage	State	Updated
Installing SSL certificate	done	2026-06-03 23:42:20
Creating mailbox	done	2026-06-03 23:41:51
Configuring web hosting	done	2026-06-03 23:41:46
Setting up web account	done	2026-06-03 23:41:37
Configuring web plan	done	2026-06-03 23:41:32

This design makes provisioning resilient to slow panels, SSH timeouts, apt-get locks and long operations — none of them block the order, and any failed step is simply retried.

The pipeline (Split example)

A Split service is built by a chain of idempotent tasks, roughly in this order:

1. **Create the web package + web user** on a Web node.
2. **Add the web domain** (and the web account's quota/PHP settings).
3. **Create the mail user + mail domain** on a Mail node (with DKIM).
4. **Create the DNS user + zone** and **deploy the zone** to every attached DNS node.
5. **Seed Auto-SSL** so Let's Encrypt issues as soon as DNS resolves.

Unified runs the same idea with one combined account; Vanity runs a trimmed chain (one web account, one mailbox on the shared mail user, one DNS record). Every task is **idempotent** — re-running it is safe — so the chain can resume from wherever it stopped.

Watching and driving it from the

admin panel

The admin service panel's **Deploy** tab shows the live deploy status and a full **timeline** of every task with its duration and result, plus a **Redeploy (hard reset)** button:

PUQ Web Hosting — admin panel

The screenshot shows the 'Deploy' tab in the admin panel. At the top, there's a navigation bar with various service icons: Overview, Web settings, SSL, FTP (1), Databases (1), DNS (25), Mailboxes (2), Cron (1), Backups (7), Deploy (0), and Tasks (11). Below this, the 'Logs' section shows the 'Deploy status' as 'ACTIVE' with a 100% progress bar. There are 'Refresh' and 'Redeploy (hard reset)' buttons. A 'success: 11' indicator is present. The main area is a 'Timeline' showing a list of tasks with their timestamps, durations, and results. The tasks include server recomputations, service resyncs, and mail backups. At the bottom, there's a table header with columns: Reg Date, Name, Pricing, Status, and Next Due Date.

The **Tasks** tab lists this service's tasks (action, server, attempts, timestamps); each row opens a detail modal with the streamed SSH log:

PUQ Web Hosting — admin panel

The screenshot shows the 'Tasks' tab in the admin panel. It features a 'Reload' button and '2 task(s)' count. A 'Show 25 entries' dropdown and a search box are also visible. The main content is a table with columns: ID, Action, Target, Server, Status, Attempts, Created, Started, Finished, Error, and Actions. Two tasks are listed, both with a 'SUCCESS' status. The first task is 'web.addCron' on server #3, and the second is 'web.addFTP' on server #3. At the bottom, there's a 'Showing 1 to 2 of 2 entries' indicator and 'Previous' and 'Next' navigation buttons.

ID	Action	Target	Server	Status	Attempts	Created	Started	Finished	Error	Actions
#2568	web.addCron	cron #16	#3	SUCCESS	1/3	2026-06-03 23:06:13	2026-06-03 23:07:24	2026-06-03 23:07:26		
#2567	web.addFTP	ftp #20	#3	SUCCESS	1/3	2026-06-03 23:05:35	2026-06-03 23:06:03	2026-06-03 23:06:06		

Retries, failures and tickets

A failed task is retried with exponential back-off. Once all attempts are exhausted it is **terminally failed**, the service shows a deploy error, and — if you enabled it — the module can open a **support ticket** for your staff. The retry policy and the failure-ticket behaviour are configured in addon **Settings** (see *Cron & Automation* → *Task queue, retries & tickets*).

“ **Key takeaway:** deployment is an asynchronous, idempotent, resumable pipeline. The order is accepted instantly, the queue does the work, failures self-heal on retry, and both you and the customer get live progress.