

Configure a trusted Let's Encrypt certificate for Proxmox VE with Cloudflare DNS

Replace the default cluster-signed certificate on the Proxmox web interface with a publicly trusted Let's Encrypt certificate, validated through the Cloudflare API with the ACME DNS-01 challenge, while the management interface stays private.

PROXMOX VE 9.2 · LET'S ENCRYPT · ACME DNS-01 · CLOUDFLARE API TOKEN · HOW-TO GUIDE

General educational guidance, not affiliated with or endorsed by Proxmox Server Solutions GmbH, Cloudflare, or Let's Encrypt. Treat the Cloudflare API token like a password, and keep the management interface off the public internet. Validated on Proxmox VE 9.2.10. Published 2026-09-08.

0 ports opened to the internet

DNS-01 validates through the Cloudflare API; the node stays private

2 token permissions

DNS Edit and Zone Read, on one zone

Proxmox VE serves its web interface with a certificate signed by its own cluster certificate authority, so every login starts with a browser warning. This guide replaces that certificate with a publicly trusted one from Let's Encrypt, using the ACME DNS-01 challenge through the Cloudflare API: Proxmox publishes a temporary DNS record, Let's Encrypt checks it, and the node never opens TCP port 80, 443, or 8006 to the public internet.

CONTEXT Why the browser shows a warning

Proxmox installs a certificate for its web service, `pveproxy`, at `/etc/pve/nodes/<node>/pve-ssl.pem`, signed by the cluster's own certificate authority (CA: the issuer a browser decides whether to trust). Administrators usually call this a self-signed certificate. It is signed by the cluster CA rather than by itself, and the browser treats the two alike, because nothing in its trust store vouches for either issuer. Browser validation runs two separate checks, and the default certificate fails the second:

VALIDATION CHECK	DEFAULT PROXMOX CERTIFICATE
Does the certificate contain the hostname being used?	May pass, when the name is in its Subject Alternative Name (SAN) field
Does the browser trust the certificate issuer?	Fails, unless the Proxmox CA is manually installed as trusted

Firefox reports the second failure as `SEC_ERROR_UNKNOWN_ISSUER`, even when the fully qualified domain name (FQDN) appears in the certificate's SAN field. A Let's Encrypt certificate is issued by a certificate authority browsers already trust, which is the check the default certificate fails. ACME (the Automatic Certificate Management Environment, the protocol Let's Encrypt speaks) is how Proxmox obtains and renews it without manual work.

Ordering an ACME certificate leaves `pve-ssl.pem` alone: Proxmox writes the new certificate and key to `/etc/pve/local/pveproxy-ssl.pem` and `/etc/pve/local/pveproxy-ssl.key`, and `pveproxy` prefers those for the web interface. The cluster-signed pair keeps its cluster role, and the Proxmox Administration Guide warns against editing or replacing it.

Why DNS-01 and not HTTP-01

The HTTP-01 challenge requires Let's Encrypt to reach the server over the public internet. DNS-01 instead proves control of the domain by publishing a TXT record at `_acme-challenge.<node FQDN>`, so it works for hosts with no public exposure. It can also issue wildcard certificates, which Proxmox cannot order: its ACME domain field rejects any name containing `*`, and the form answers "This is not a valid hostname". The node needs outbound HTTPS to Cloudflare and Let's Encrypt, and no inbound access.

TRADE-OFFS What this route requires and costs

The node needs no public DNS record

The node FQDN may resolve through internal DNS, split DNS, or another private name-resolution service. Do not configure the management hostname as a Cloudflare-proxied record; this guide assumes direct private access to Proxmox on TCP port 8006.

DNS-01 leaves API credentials on the host

That is the trade-off for keeping the node off the public internet. The token in this guide is restricted to two permissions on one zone to limit what a leaked token could do; step 08 covers protecting it after setup.

The node FQDN becomes public

Let's Encrypt submits every certificate it issues to public Certificate Transparency logs, so `pve01.lab.example.com` reaches a public log the moment the certificate is issued, becomes searchable at sites such as crt.sh soon after, and stays there. No port opens and no route to the node is created; the name is what becomes public, along with whatever the naming convention reveals. Choose the FQDN with that in mind before ordering, because a name in the logs cannot be withdrawn.

MAP Example values and assumptions

The guide uses these example values throughout. Replace them with the client environment's own:

ITEM	EXAMPLE VALUE
Cloudflare DNS zone	example.com
Proxmox node name	pve01
Proxmox node FQDN	pve01.lab.example.com
Proxmox web address	https://pve01.lab.example.com:8006
ACME account name	default
Cloudflare challenge plugin ID	cf-example

- Cloudflare is authoritative for the base domain's public DNS zone, and the administrator can change DNS settings in Cloudflare and datacenter settings in Proxmox.

- Let's Encrypt checks CAA at `pve01.lab.example.com` first, then `lab.example.com`, then `example.com`, and stops at the first name carrying any CAA record. That record set allows `letsencrypt.org`, and where it carries a `validationmethods` parameter, `dns-01` is in it. Check each name with `dig CAA <name>`, or read the zone's records in Cloudflare.
- The node can resolve DNS and reach Cloudflare and Let's Encrypt over HTTPS, and its date and time are correct.
- Authorized client devices can resolve the chosen FQDN to an address the node answers on.
- Every configuration step runs in the web interface. The checks in the troubleshooting section need a root shell on the node.
- The guide configures one node. In a cluster, the ACME account and challenge plugin are datacenter-wide, while each node carries its own certificate, so repeat steps 05 to 07 on every node with its own FQDN.

01 Confirm name resolution and collect the Cloudflare IDs

Confirm the FQDN resolves. From an authorized workstation, confirm the intended name points at an address the node is reachable on. On Linux, `getent` answers through the workstation's own resolver, so it reflects the hosts file as well as DNS:

```
getent hosts pve01.lab.example.com
```

On Windows or macOS, use `nslookup pve01.lab.example.com`, which asks the DNS server directly and will not show a hosts-file entry. If internal DNS is used, create or update the internal A or AAAA record before the final verification step.

Collect the Zone ID and the Account ID. Sign in to the Cloudflare dashboard, select the domain, and copy both from the **API** section at the foot of its **Overview** page. Store both for the plugin form in step 04.

Use the Zone ID of the zone that is authoritative for the node FQDN, usually the base domain: for `pve01.lab.example.com`, the Zone ID of `example.com`. Where the account holds `lab.example.com` as its own zone, use that one and scope the token in step 02 to it, because `CF_Zone_ID` pins the lookup and Proxmox writes the record nowhere else.

Have an administrative email address ready too; step 03 uses it as the ACME account contact.

02 Create a restricted Cloudflare API token

Use a scoped API token, never the legacy Cloudflare Global API Key. A scoped token limits Proxmox, and anyone who steals the token, to DNS in one zone; a leaked Global API Key gives an attacker the whole account. Inside that zone the token can edit any record, so treat it as write access to the zone's DNS rather than to the ACME challenge records alone. In Cloudflare, open **Manage Account** and go to the account API tokens page, select **Create Token**, name it (for example `proxmox-acme`), and build a custom permission policy.

Create it there rather than under **My Profile > API Tokens**: an account token keeps renewal working after the person who set it up loses access, where a personal token stops with them. Creating or updating one needs the Super Administrator role.

SETTING	VALUE
Permissions	Two rows: Zone / DNS / Edit , and Zone / Zone / Read . Nothing else
Zone Resources	Include , then the one zone from step 01, such as <code>example.com</code> , rather than All zones

Leave every other permission disabled, in particular **Zone / Zone / Edit**. The form also offers a **TTL**, an expiry date for the token: setting one bounds what a leaked token is worth and puts a known date on the rotation, and leaving it unset keeps renewal running until someone revokes the token. Record the choice either way, because step 08 depends on it. The

summary should read `DNS:Edit, Zone:Read` against the one zone. Create the token and copy the secret immediately: Cloudflare shows it only once.

- **Treat the token as a password**

Keep it out of screenshots, email, tickets, chat messages, and documentation. Record the token's owner, purpose, scope, expiry, and rotation requirements in the client's credential inventory.

03 Register the ACME account in Proxmox

Sign in to the Proxmox web interface with an account that can administer the Datacenter ACME configuration, select **Datacenter** in the tree, and open **ACME**. Under **Accounts**, select **Add**:

FIELD	VALUE
Account Name	<code>default</code> , or another descriptive name
E-Mail	The administrative address from step 01
ACME Directory	Let's Encrypt V2
Accept TOS	Enabled, after reviewing the terms of service

Select **Register** and confirm the account appears under **Datacenter > ACME > Accounts**.

The directory list also offers **Let's Encrypt V2 Staging**, which issues untrusted test certificates and carries much higher rate limits. Proxmox recommends staging for a first ACME setup, and Let's Encrypt recommends it for troubleshooting, so failed attempts do not use up the production limits.

Skipping staging means working inside two production limits: five authorization failures per hostname per account each hour, refilling one every 12 minutes, and five certificates for the same set of names every seven days, refilling one every 34 hours. The second has no override, so a few rounds of ordering, deleting and re-ordering can block the FQDN for days.

An account's directory cannot be changed afterwards: to move to production, register a second account against the production directory and select it on the node, or deactivate the staging account and register the same name again; the second of those, deactivating and re-registering, is the route Proxmox documents.

This guide registers against production directly, which is the shorter path once the values from steps 01 and 02 are confirmed.

04 Add the Cloudflare challenge plugin

Still in **Datacenter > ACME**, under **Challenge Plugins**, select **Add**. The first three fields configure the plugin itself:

FIELD	VALUE
Plugin ID	A descriptive ID, such as <code>cf-example</code>
Validation Delay	<code>30</code> seconds, the time Proxmox waits between writing the DNS record and asking for validation
DNS API	Cloudflare Managed DNS

The remaining fields carry the Cloudflare credentials, in the order the form lists them:

FIELD	VALUE
CF_Account_ID	The Cloudflare Account ID
CF_Email	Leave blank
CF_Key	Leave blank
CF-Token	The restricted API token secret
CF_Zone_ID	The Zone ID from step 01

Select **Add** and confirm the plugin appears under **Challenge Plugins** with **cf** in the API column.

CF_Email and **CF_Key** belong to Cloudflare's legacy Global API Key authentication. They must stay blank when **CF-Token** is used.

05 Assign the domain to the node

Select the node (such as **pve01**) in the tree and open **System > Certificates**. In the **ACME** section, confirm the intended account is selected (shown as **Using Account: default**), then select **Add**:

FIELD	VALUE
Challenge Type	DNS
Plugin	The Cloudflare plugin ID, such as cf-example
Domain	The complete node FQDN, such as pve01.lab.example.com

Select **Create** and confirm the FQDN appears in the ACME domain list with the type **dns**.

06 Order and install the certificate

In **System > Certificates > ACME**, select the configured domain and then **Order Certificates Now**. Keep the task window open so you can read the log, and do not select **Stop**. Closing the window does not stop the order; **Stop** does. Proxmox places the ACME order, creates a temporary TXT record at **_acme-challenge.pve01.lab.example.com** through Cloudflare, waits out the validation delay, asks Let's Encrypt to validate, removes the TXT record, installs the issued certificate, and restarts **pveproxy**. The output should end with **TASK OK**; along the way it typically prints:

```
Status is 'valid', domain 'pve01.lab.example.com' OK!
All domains validated!
Setting pveproxy certificate and key
Restarting pveproxy
```

The web interface disconnects briefly while **pveproxy** restarts. That is expected, and the session resumes on reload.

To undo the change, select the **pveproxy-ssl.pem** row in **System > Certificates** and use **Delete Custom Certificate**: it removes the certificate and key and restarts **pveproxy**, which goes back to the cluster-signed certificate and the browser warning. **pvenode cert delete 1** does the same from a root shell. The ACME account, the challenge plugin, and the domain entry stay configured, and the daily task attempts no renewal until you order again.

07 Verify in the browser

Open a new browser tab and use the exact FQDN the certificate was issued for: `https://pve01.lab.example.com:8006`. A short hostname such as `https://pve01:8006` is not one of the names on the certificate, so the browser warns again. Confirm that:

- The browser reports a secure HTTPS connection, and the TLS certificate's issuer is Let's Encrypt (through its current chain), no longer the Proxmox cluster CA.
- The certificate contains the node FQDN and its validity period is current.
- The Proxmox login page loads.

08 Renewal is automatic; keep the token and the plugin working

Proxmox's ACME client requests no certificate profile, so Let's Encrypt issues its classic default, a 90-day certificate. Renewal has to happen without anyone watching, and a daily task (`pve-daily-update`) does it: it reorders any certificate that has expired or expires within the next 30 days, using the same Cloudflare plugin to publish and remove a fresh TXT record.

Each renewal also asks Let's Encrypt to revoke the certificate it replaced, so a certificate restored from an older backup of `/etc/pve` will not be trusted.

Let's Encrypt has published a schedule that takes the classic profile to 64 days on 10 February 2027 and to 45 days on 16 February 2028. Nothing here changes for that: renewals become more frequent, and the 30 days of retry runway before expiry stay the same. Automatic renewal keeps working as long as these conditions hold:

- The Cloudflare token stays active, with its **DNS Edit** and **Zone Read** permissions intact.
- The ACME account, the node domain entry, and the challenge plugin stay configured.
- The node keeps outbound internet access and correct system time.

Nothing announces a failed renewal, so put the certificate's expiry date in your normal infrastructure monitoring. Let's Encrypt ended its expiration notice emails in 2025, so monitoring is the only warning before a certificate lapses. Where the token was given a TTL in step 02, replace it before that date and update `CF_Token` in the challenge plugin.

• The stored token stays sensitive after setup

Proxmox writes the token to `/etc/pve/priv/acme/plugins.cfg`. That directory is readable by root only, and in a cluster pmxcfs copies it to every node, so one token sits on all of them. Limit administrative access to the hosts, and treat any backup that captures `/etc/pve` as a backup that holds a credential.

HARDEN Keep the interface private and the accounts strong

- Do not expose the Proxmox web interface to the public internet. A trusted certificate changes what the browser shows and does not change who can reach the login page. Restrict access to an administrative network, management VLAN, or approved VPN.
- Use multi-factor authentication on both the Cloudflare and Proxmox administrative accounts.
- Keep the API token dedicated to this service, scoped to the one required zone and the two required permissions, and revoke and replace it as soon as you suspect it has been exposed. An account token's secret begins with `cfat_`. Credential scanners match on that prefix, and it is the string to search for when checking whether a token has reached a repository or a ticket.
- Where the node's outbound address is static, Cloudflare's Client IP Address Filtering can restrict the token to that address; skip it where the address changes.

Troubleshooting

- **The browser still says "Not Secure"**

Check that the address bar shows the complete FQDN from the certificate. Open a new tab rather than reusing one you opened with the short hostname. Inspect the presented certificate: its issuer should no longer be the Proxmox local CA. Confirm the FQDN resolves to the intended node.

- **SEC_ERROR_UNKNOWN_ISSUER remains**

The browser does not trust the certificate issuer, which usually means the browser is still receiving the old certificate. Confirm the ACME task ended with `TASK OK` and that the browser is reaching the correct node by the issued FQDN. On the node, confirm the file the order should have written and read its issuer with `openssl x509 -in /etc/pve/local/pveproxy-ssl.pem -noout -issuer -dates`. A missing file means the order installed nothing.

- **Cloudflare authentication or authorization fails**

Confirm `CF_Token` holds the token secret (never the token's name or ID), `CF_Account_ID` holds the alphanumeric Account ID (never an email address), `CF_Zone_ID` holds the Zone ID from step 01, of the zone that is authoritative for the node FQDN, and `CF_Email` and `CF_Key` are blank. Then confirm the token covers the correct domain, carries **DNS Edit** and **Zone Read**, and has not expired or been revoked.

- **The order is refused before validation runs**

Check the CAA records, starting at the node FQDN and working up. Let's Encrypt reads CAA at issuance and stops at the first name carrying any record, so a set that names other authorities and not `letsencrypt.org` stops the order whatever the DNS challenge does. A set that allows `letsencrypt.org` but restricts `validationmethods` to `http-01` or `tls-alpn-01` stops it too. Cloudflare manages CAA records under **DNS > Records** on the domain.

- **DNS validation times out**

Confirm Cloudflare is authoritative for the domain and the token can create and delete records in the zone. Raise the plugin's **Validation Delay** when the TXT record is not visible before validation begins; 30 seconds worked in the validated configuration, and some environments need longer. Confirm outbound DNS and HTTPS work from the node, then retry **Order Certificates Now**. Let's Encrypt allows one account five authorization failures per hostname each hour, so correct the cause rather than retrying in a loop.

- **The certificate is valid but the FQDN does not load**

Getting a certificate and reaching the node are two different jobs. DNS-01 proves you control the DNS zone; it does not create the A or AAAA record clients look up, and it does not open a route to the node. Confirm internal DNS, routing, VPN access, firewall policy, and connectivity to TCP port 8006.

What you achieved

- The Proxmox web interface presents a publicly trusted Let's Encrypt certificate for its FQDN, and browsers stop warning.
- Domain control was proven with a temporary TXT record through the Cloudflare API, so no port was opened and no route to the node was created. The node FQDN is now in the public Certificate Transparency logs.
- The Cloudflare token can edit DNS records in one zone and read that zone, and nothing beyond it. The Global API Key was never used.
- Renewal runs automatically through the daily Proxmox task, and the token, the plugin, and the monitoring are all in place to keep it running.

References

Proxmox VE wiki: Certificate Management, and the Administration Guide · pve.proxmox.com/wiki/Certificate_Management · pve.proxmox.com/pve-docs

Cloudflare Fundamentals: Create an API token; Account API tokens; API token permissions; Find account and zone IDs · developers.cloudflare.com/fundamentals

Let's Encrypt documentation: Challenge Types; Rate Limits; CAA · letsencrypt.org/docs/challenge-types · letsencrypt.org/docs/rate-limits · letsencrypt.org/docs/caa

Let's Encrypt: Decreasing Certificate Lifetimes to 45 Days, 2 December 2025 · letsencrypt.org/2025/12/02/from-90-to-45
acme.sh DNS API reference (the Cloudflare provider Proxmox uses) · github.com/acmesh-official/acme.sh/wiki/dnsapi