anoBytes | TECHNOLOGY SOLUTIONS

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

Replace the default self-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 2026-08-16.

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 complains

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). Browser validation runs two separate checks, and the default certificate can pass only the first:

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 fixes the issuer-trust half; ACME (the Automatic Certificate Management Environment, the protocol Let's Encrypt speaks) is how Proxmox obtains and renews it without manual work.

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 at all, and it can issue wildcard certificates. The node needs outbound HTTPS to Cloudflare and Let's Encrypt, and no inbound access at all.

No public DNS record is required for the node

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.

API credentials end up on the host

DNS-01's trade-off is that API credentials sit on the server. 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.

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 has appropriate access to both Cloudflare and Proxmox.
- The node has outbound HTTPS access to Cloudflare and Let's Encrypt, and its date and time are correct.
- The chosen FQDN resolves to the node's reachable address from authorized client devices.
- 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:

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

If internal DNS is used, create or update the internal A or AAAA record before the final verification step.

Collect the Account ID. Sign in to the Cloudflare dashboard, open **Account Home**, press **Ctrl+K** (or **Cmd+K** on a Mac), search for **Copy account ID**, and select the result. Store the value for the plugin form in step 04.

Collect the Zone ID. Open **Domains**, select the base domain, and copy the **Zone ID** from the **API** section of the domain's **Overview** page. Use the base zone's ID: for `pve01.lab.example.com`, that is the Zone ID of `example.com`.

Also have ready an administrative email address for the ACME account contact in step 03.

02 Create a restricted Cloudflare API token

Use a scoped API token, never the legacy Cloudflare Global API Key: the token lets Proxmox create and remove only the DNS records ACME validation needs, in one zone, while a leaked Global API Key compromises the whole account. In

Cloudflare, open **Manage Account > API Tokens**, select **Create Token**, name it (for example `proxmox-acme`), and build a custom permission policy:

SETTING	VALUE
Resource scope	**Specified Domains**, with the client's base domain, such as `example.com`
DNS & Zones permissions	**DNS: Edit** and **Zone: Read**, and nothing else

Leave every other permission disabled; in particular do not grant **Zone: Edit**. Review the summary, 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. If it is ever exposed, revoke it in Cloudflare and create a replacement. Record the token's owner, purpose, scope, 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	`default`, 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 significantly higher rate limits; Let's Encrypt recommends it for testing and troubleshooting so production limits are not consumed. This guide registers against the production directory.

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 `cf-example`
Validation Delay	`30` 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:

FIELD	VALUE
`CF_Account_ID`	The Cloudflare Account ID
`CF_Token`	The restricted API token secret
`CF_Zone_ID`	The Zone ID for the base domain
`CF_Email`	Leave blank
`CF_Key`	Leave blank

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**. Leave the task window open and do not select **Stop**. 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'
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.

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 a name Let's Encrypt did not issue, and it can still produce a certificate warning. 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 its conditions true

Proxmox's ACME client requests no certificate profile, so Let's Encrypt issues its current default: a 90-day certificate. Renewal therefore has to run unattended, and Proxmox handles it: a daily task (`pve-daily-update`) 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. 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.
- If the token was created with an expiration date, rotate it before it lapses and update `CF_Token` in the challenge plugin.
- Certificate expiration is watched by normal infrastructure monitoring. Let's Encrypt ended its expiration notice emails in 2025, so monitoring is the only warning before a lapsed certificate.

- **The stored token stays sensitive after setup**
  The Cloudflare API token lives in the Proxmox configuration from now on. Limit administrative access to the host, and protect configuration backups the way credentials are protected.

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 the attack surface only shrinks by restricting 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 immediately if disclosure is suspected.
- 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.

HELP Troubleshooting

- **The browser still says "Not Secure"**
  Confirm the address bar carries the complete FQDN from the certificate, in a new tab rather than one opened against 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 it is still being served the old certificate. Confirm the ACME task ended with `TASK OK` and that the browser is reaching the correct node by the issued FQDN.

- **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` is the base domain's Zone ID, 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.

- **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**. Production limits allow five authorization failures per identifier per hour, so correct the cause rather than retrying in a loop.

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

  Issuance and reachability are separate: DNS-01 proves control of the DNS zone and never creates the client-facing A or AAAA record or the network path. Confirm internal DNS, routing, VPN access, firewall policy, and connectivity to TCP port 8006.

DONE 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 the management interface stayed private.
- The Cloudflare token can edit DNS in one zone and read it, and can do nothing else; the Global API Key was never used.
- Renewal is automatic through the daily Proxmox task, with the token, plugin, and monitoring in place to keep it that way.

REF 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; API token permissions; Find account and zone IDs · `developers.cloudflare.com/fundamentals`

Let's Encrypt documentation: Challenge Types · `letsencrypt.org/docs/challenge-types`

acme.sh DNS API reference (the Cloudflare provider Proxmox uses) · `github.com/acmesh-official/acme.sh/wiki/dnsapi`