

Traefik

Purpose

Traefik is the central HTTPS reverse proxy for the Bytek homelab.

Traefik receives HTTPS requests, selects the correct application according to the requested hostname, and forwards the request to the appropriate private backend.

Traefik provides:

- HTTPS termination.
- Hostname-based routing.
- Let's Encrypt certificate issuance and renewal.
- ACME DNS-01 validation through the WHC cPanel API.
- Routing to private application backends.
- Authentik ForwardAuth for selected administrative services.
- A dashboard for routers, services, middlewares, and certificates.
- Consistent internal and external application URLs.
- Protection of private backend addresses from direct public exposure.

Service Information

Setting	Value
Service name	Traefik
Private IP address	192.168.2.182
HTTPS entry point	TCP 443
Public dashboard hostname	proxy.bytek.ca
Deployment directory	/opt/traefik
Container DNS	192.168.2.65
Certificate authority	Let's Encrypt

ACME challenge type	DNS-01
DNS automation	WHC cPanel API
Normal dashboard authentication	Authentik ForwardAuth
Secondary dashboard authentication	Basic Auth while retained
Backup coverage	Proxmox scheduled backup
Proxmox VM ID	Confirm current VM ID

Dashboard:

[Open the Traefik Dashboard](#)

Architecture Role

Traefik is the common application-routing layer for both external and internal clients.

External Request Path

1. WHC public DNS resolves the application hostname to the VPS at `38.29.213.101`.
2. The VPS receives HTTPS traffic.
3. The VPS forwards the traffic through WireGuard.
4. The home WireGuard gateway sends the request to Traefik at `192.168.2.182`.
5. Traefik examines the requested hostname.
6. Traefik forwards the request to the matching private backend.

Internal Request Path

1. A LAN client queries Pi-hole.
2. Pi-hole resolves the application hostname to `192.168.2.182`.
3. The client connects directly to Traefik.
4. Traefik forwards the request to the matching private backend.

The same application hostname is used internally and externally.

Application Routing Reference

Hostname	Backend Destination	Authentication
<code>portal.bytek.ca</code>	<code>192.168.2.162:9000</code>	Authentik native authentication
<code>cloud.bytek.ca</code>	<code>192.168.2.100:11000</code>	Nextcloud native OIDC
<code>projects.bytek.ca</code>	<code>192.168.2.121:3456</code>	Vikunja native OIDC
<code>docs.bytek.ca</code>	BookStack private IP on TCP 6875	BookStack native OIDC
<code>status.bytek.ca</code>	Uptime Kuma or Authentik embedded outpost	Authentik Proxy Provider
<code>proxy.bytek.ca</code>	Traefik internal dashboard service	Authentik ForwardAuth

The BookStack private address must be confirmed and added to this page.

Configuration Locations

Purpose	Location
Docker Compose file	<code>/opt/traefik/compose.yml</code>
Static Traefik configuration	<code>/opt/traefik/traefik.yml</code>
Environment variables	<code>/opt/traefik/.env</code>
Dynamic configuration	<code>/opt/traefik/dynamic/</code>
Certificate state	<code>/opt/traefik/data/acme.json</code>
Traefik log	<code>/opt/traefik/logs/traefik.log</code>
Access log	Confirm current path
Additional persistent data	<code>/opt/traefik/data/</code>

The exact certificate and log paths should be confirmed against the current Compose file before relying on them during recovery.

Dynamic Configuration

Each application normally has its own dynamic configuration file.

Known files include:

File	Purpose
<code>authentik.yml</code>	Authentik router and service
<code>authentik-forwardauth.yml</code>	Shared Authentik middleware and outpost service
<code>bookstack.yml</code>	BookStack router and service
<code>dashboard.yml</code>	Traefik dashboard router
<code>nextcloud.yml</code>	Nextcloud router and service
<code>uptime-kuma.yml</code>	Uptime Kuma router and service
<code>vikunja.yml</code>	Vikunja router and service

Separating services into individual files makes configuration changes easier to review and reverse.

Before editing a working file:

1. Create a dated or descriptive backup copy.
2. Change only the required section.
3. Validate the YAML indentation.
4. Check the Traefik log.
5. Test the route locally.
6. Test the route from the LAN.
7. Test the route externally if the service is public.
8. Remove obsolete backup files after the change is documented.

Static Configuration

The static Traefik configuration defines platform-level settings such as:

- Entry points.
- File-provider location.
- Dashboard availability.
- Logging.
- Access logging.
- Certificate resolvers.
- ACME settings.

- Trusted forwarded headers.
- Provider behavior.

Static configuration changes normally require a Traefik container restart or recreation.

Dynamic routing changes are usually detected automatically by the file provider when file watching is enabled.

Docker DNS

The Traefik container explicitly uses Pi-hole at `192.168.2.65`.

This configuration was added after Docker's embedded resolver failed to resolve the Let's Encrypt ACME API following a power interruption.

The observed error reported that the Docker resolver at `127.0.0.11` was misbehaving.

Explicit Pi-hole DNS allows Traefik to resolve:

- Let's Encrypt endpoints.
- WHC API endpoints.
- Internal Bytek hostnames.
- External Docker and operating-system dependencies.

Verify the configured Docker DNS using:

```
sudo docker inspect --format='DNS={{json .HostConfig.Dns}}' traefik
```

Expected result:

- `192.168.2.65` appears in the DNS list.

The container's internal resolver file may still show Docker's embedded resolver at `127.0.0.11`.

That is normal when Docker forwards requests to the explicitly configured upstream DNS server.

Container Management

Move to the deployment directory before running Compose commands:

```
cd /opt/traefik
```

Check the stack:

```
sudo docker compose ps
```

View recent logs:

```
sudo docker compose logs --tail 100 traefik
```

Follow logs live:

```
sudo docker compose logs -f traefik
```

Validate the Compose file:

```
sudo docker compose config --quiet
```

Apply normal Compose changes:

```
sudo docker compose up -d
```

Recreate Traefik after changing container-level settings such as DNS:

```
sudo docker compose up -d --force-recreate traefik
```

Restart Traefik after a static configuration change:

```
sudo docker compose restart traefik
```

A normal restart does not apply changes to Docker-level settings such as container DNS, network mode, volume mounts, or environment variables. Those changes require container recreation.

Naming and Routing Standard

Every Traefik application should have:

- A descriptive router name.
- A descriptive service name.
- A hostname rule.
- The `websecure` entry point.
- TLS enabled.

- The correct certificate resolver.
- A private backend destination.
- Host-header forwarding where required.
- Authentication middleware only when appropriate.

A route should not contain Authentik ForwardAuth if the application already uses native OIDC and requires direct API or client access.

Applications That Must Not Use ForwardAuth

Do not attach Authentik ForwardAuth to:

- Nextcloud.
- Vikunja.
- BookStack.
- Authentik itself.
- Proxmox VE.
- Nextcloud AIO management.
- Plex.
- Game-server traffic.

Nextcloud

Nextcloud mobile clients, desktop clients, WebDAV, CalDAV, CardDAV, and application authentication must reach Nextcloud directly.

Nextcloud uses native OIDC.

Vikunja

Vikunja uses native Authentik OIDC.

BookStack

BookStack uses native Authentik OIDC.

Authentik

Authentik must never depend on its own ForwardAuth middleware.

Proxmox VE

Proxmox uses native OIDC and remains outside Traefik.

Plex

Plex clients use Plex authentication and Plex-specific APIs.

Applications That Use Authentik Proxy Integration

Traefik Dashboard

The Traefik dashboard uses single-application ForwardAuth.

The dashboard request is validated by the Authentik embedded outpost before access is granted.

Uptime Kuma

Uptime Kuma uses an Authentik Proxy Provider because Uptime Kuma does not provide the same native OIDC integration as the other applications.

Selected status-page and monitoring endpoints may bypass authentication when public access is intentionally required.

Traefik Dashboard Authentication

The dashboard is available at:

[Open the Traefik Dashboard](#)

Access is restricted by:

- Authentik group `bytek-admin`.
- Authentik policy `Allow bytek.ca users`.
- Policy engine mode `ALL`.

During the initial ForwardAuth validation, Traefik Basic Auth remains enabled as a second authentication layer.

The expected initial flow is:

1. The administrator opens `proxy.bytek.ca`.
2. Traefik invokes Authentik ForwardAuth.
3. Authentik validates the user and MFA.
4. Authentik checks the application bindings.
5. Traefik receives the authenticated request.
6. Traefik presents the existing Basic Auth challenge.
7. The administrator enters the Traefik Basic Auth credentials.
8. The dashboard opens.

Basic Auth may be removed only after ForwardAuth and recovery procedures are fully tested.

Authentik ForwardAuth Middleware

The ForwardAuth middleware sends authentication requests to the Authentik embedded outpost.

The outpost backend is hosted by Authentik at `192.168.2.162` on TCP port 9000.

The middleware should trust the forwarded request headers and return Authentik identity headers to the protected application.

Typical identity headers include:

- Authentik username.
- Authentik group list.
- Authentik entitlements.
- Email address.
- Display name.
- User identifier.
- Provider metadata.
- Application metadata.
- Outpost metadata.

Do not log full authentication headers in a location accessible to untrusted users.

Authentik Outpost Callback Route

The path under `/outpost.goauthentik.io/` must remain reachable without the ForwardAuth middleware.

If the callback path is protected by the same middleware, an authentication loop can occur.

The callback router should:

- Match `proxy.bytek.ca`.
- Match the `/outpost.goauthentik.io/` path prefix.
- Use the Authentik outpost service.
- Have a higher routing priority than the dashboard router.
- Use HTTPS.
- Avoid the dashboard ForwardAuth middleware.

Test the outpost with:

```
curl -k -I https://proxy.bytek.ca/outpost.goauthentik.io/ping
```

Expected result:

- HTTP 204.

A normal browser link is not useful for this endpoint because an HTTP 204 response intentionally contains no page content.

Certificate Management

Traefik obtains HTTPS certificates from Let's Encrypt.

The current design uses:

- ACME.
- DNS-01 validation.
- WHC cPanel API.
- Certificate resolver named `letsencrypt`.

Each public router should reference the correct certificate resolver.

A router without the certificate resolver may use the Traefik default certificate instead of requesting the intended hostname certificate.

ACME DNS-01 Flow

Certificate issuance follows this sequence:

1. A Traefik router requests a certificate.
2. Traefik contacts the Let's Encrypt ACME API.
3. Traefik creates the required DNS challenge through the WHC API.
4. Let's Encrypt validates the challenge.
5. Traefik receives the certificate.
6. Traefik stores the certificate state in `acme.json`.
7. Traefik serves the certificate for the matching hostname.

Traefik must have working DNS before this process can succeed.

Certificate Storage

Certificate state is stored in:

```
/opt/traefik/data/acme.json
```

The file must:

- Remain persistent across container recreation.
- Have restrictive permissions.
- Be included in backups.
- Never be committed to a public repository.
- Never be pasted into BookStack.
- Never be deleted during routine troubleshooting.

Deleting `acme.json` can remove all stored certificates and account state.

That may trigger unnecessary reissuance and certificate-rate-limit risk.

Certificate Validation

Inspect a certificate served by Traefik using:

```
openssl s_client -connect 127.0.0.1:443 -servername projects.bytek.ca </dev/null 2>/dev/null |  
openssl x509 -noout -subject -issuer -dates -ext subjectAltName
```

Verify:

- The subject or Subject Alternative Name contains the requested hostname.
- The issuer is Let's Encrypt.
- The certificate is currently valid.
- The expiration date is reasonable.
- Traefik is not serving the default certificate.

Repeat with the relevant hostname when validating another service.

Certificate Troubleshooting

Certificate Missing From the Dashboard

Check:

1. The dynamic router exists.
2. The hostname rule is correct.
3. TLS is enabled.
4. The certificate resolver is named correctly.
5. Traefik DNS works.
6. The WHC API credentials work.
7. The ACME log contains the hostname.
8. The DNS challenge was validated.

ACME DNS Failure

A previously observed error reported:

- Failure to contact the Let's Encrypt API.
- DNS lookup through `127.0.0.11`.
- Docker DNS server misbehaving.

The correction was to configure the Traefik container to use Pi-hole explicitly and recreate the container.

Wrong Certificate Served

Possible causes include:

- Router not loaded.
 - Hostname typo.
 - Wrong certificate resolver.
 - Duplicate router rule.
 - Default certificate served.
 - Client or browser cache.
 - DNS points to the wrong server.
-

Application Onboarding

When adding a new public application:

1. Confirm the application backend works directly.
 2. Reserve the VM or LXC IP in Pi-hole.
 3. Add the public WHC A record pointing to `38.29.213.101`.
 4. Add the Pi-hole local DNS record pointing to `192.168.2.182`.
 5. Create a dynamic Traefik configuration file.
 6. Add the hostname router.
 7. Add the private backend service.
 8. Enable TLS with the correct resolver.
 9. Validate YAML formatting.
 10. Confirm Traefik loaded the router.
 11. Confirm certificate issuance.
 12. Test from the Traefik VM.
 13. Test from the home LAN.
 14. Test from an external network.
 15. Add Uptime Kuma monitors.
 16. Update BookStack documentation.
-

Backend Validation

Before troubleshooting an application through Traefik, test the direct backend from the Traefik VM.

Examples:

Nextcloud

```
nc -zv -w 5 192.168.2.100 11000
```

Vikunja

```
curl -i http://192.168.2.121:3456/health
```

Authentik

```
curl -I http://192.168.2.162:9000/-/health/ready/
```

BookStack

Use the confirmed BookStack IP on TCP port 6875.

A direct backend failure must be resolved before changing the Traefik router.

Local Router Validation

Traefik can be tested locally without relying on public DNS.

Example for Vikunja:

```
curl -k --resolve projects.bytek.ca:443:127.0.0.1 https://projects.bytek.ca/health
```

This forces the requested hostname to the local Traefik listener.

A successful response confirms:

- The router loaded.
 - TLS negotiation works.
 - Traefik selected the expected service.
 - The backend is reachable.
 - The application responded.
-

Logging

Review the main Traefik log using:

```
sudo tail -n 100 /opt/traefik/logs/traefik.log
```

Filter common issues using:

```
sudo grep -iE 'error|acme|certificate|middleware|service|router|dns' /opt/traefik/logs/traefik.log  
| tail -n 100
```

Look for:

- YAML parsing errors.
- Unknown fields.
- Missing services.
- Missing middlewares.
- Incorrect certificate resolver.
- ACME validation failures.
- DNS resolution failures.
- Backend connection failures.
- Router conflicts.

If access logging is enabled, use it to distinguish:

- Requests reaching Traefik.
- Router selection.
- Backend status.
- HTTP response code.
- Client source.
- Authentication redirects.

Do not copy session cookies, bearer tokens, or authorization headers into BookStack.

YAML Validation

YAML indentation must use spaces, not tabs.

Check dynamic configuration files using:

```
sudo grep -RnP '\t' /opt/traefik/dynamic/
```

Expected result:

- No output.

Review a specific file using:

```
sudo cat /opt/traefik/dynamic/<filename>.yaml
```

Common YAML problems include:

- Incorrect indentation.
 - Tabs.
 - Misspelled field names.
 - Router placed under the wrong section.
 - Service placed under the wrong section.
 - Incorrect middleware name.
 - Incorrect backend URL.
 - Old IP address.
 - Duplicate router name.
-

IP Address Changes

If an application IP changes:

1. Confirm the new reservation in Pi-hole.
2. Confirm the VM owns the new address.
3. Update the Traefik backend.
4. Search `/opt/traefik` for the old address.
5. Update Proxmox firewall rules.
6. Update Authentik internal hosts if applicable.
7. Update Uptime Kuma monitors.
8. Test the direct backend.
9. Test the local Traefik route.
10. Test the public route.
11. Update BookStack documentation.

A stale backend address causes gateway errors even when the application itself is healthy.

Firewall Requirements

The Traefik VM must accept:

- TCP 443 from the home LAN.
- TCP 443 from the WireGuard gateway.
- Administrative SSH from approved LAN sources.
- Monitoring traffic where required.

Application VM firewalls must permit Traefik to reach their backend ports.

Known requirements include:

Destination	Source	Port
Nextcloud	192.168.2.182	TCP 11000
Vikunja	192.168.2.182	TCP 3456
Authentik	192.168.2.182	TCP 9000
BookStack	192.168.2.182	TCP 6875
Uptime Kuma direct route, if retained	192.168.2.182	TCP 3001

Do not allow public clients to connect directly to these backend ports.

Monitoring

Recommended Uptime Kuma monitors include:

Monitor	Target
Traefik VM	Ping 192.168.2.182
Traefik HTTPS	TCP 443
Authentik route	portal.bytek.ca readiness endpoint
Nextcloud route	Nextcloud status endpoint
Vikunja route	Vikunja health endpoint
BookStack route	BookStack login endpoint
Certificate expiry	Each important public hostname
Public VPS to Traefik path	External application monitor

Monitoring should distinguish between:

- Traefik VM unavailable.
 - TCP 443 unavailable.
 - Router unavailable.
 - Certificate failure.
 - Backend unavailable.
 - Authentik middleware failure.
 - VPS or WireGuard path failure.
-

Power-Failure Recovery

After a power interruption:

1. Confirm Pi-hole is running.
2. Confirm the WireGuard gateway is running.
3. Confirm Authentik is ready.
4. Confirm the Traefik VM owns `192.168.2.182`.
5. Confirm Docker is active.
6. Confirm the Traefik container is running.
7. Confirm the container uses Pi-hole DNS.
8. Review Traefik logs for DNS errors.
9. Confirm routers loaded.
10. Confirm certificates are present.
11. Test Authentik locally.
12. Test an application locally.
13. Test a public application externally.
14. Restart or recreate Traefik only if required.

If Traefik started before Pi-hole and DNS remains broken, recreate the Traefik container after Pi-hole is healthy.

Backup

The Traefik VM is included in the Proxmox scheduled backup job.

Protect the following paths:

- `/opt/traefik/compose.yml`
- `/opt/traefik/traefik.yml`
- `/opt/traefik/.env`
- `/opt/traefik/dynamic/`
- `/opt/traefik/data/acme.json`
- Relevant logs.
- Any authentication files.
- Any custom certificate files.
- Any dashboard Basic Auth configuration.

Secrets must remain in a password manager or encrypted backup.

BookStack should document the secret location, not the secret itself.

Recovery Procedure

If Traefik configuration is broken:

1. SSH into the Traefik VM.
2. Confirm Docker is active.
3. Confirm the Traefik container state.
4. Review the most recent log entries.
5. Identify the most recently changed dynamic file.
6. Restore the known-good backup copy.
7. Confirm the file provider reloads the configuration.
8. Test the affected router locally.
9. Restart Traefik only if the static configuration changed or reload failed.
10. Confirm certificates remain present.
11. Test from the LAN.
12. Test externally.

If the VM must be restored:

1. Restore the most recent Proxmox backup under a temporary VM ID.
2. Disconnect the restored VM network adapter.
3. Verify `/opt/traefik`.

4. Verify `acme.json`.
 5. Verify environment and dynamic configuration.
 6. Shut down the failed production VM.
 7. Assign the expected network identity.
 8. Start the restored VM.
 9. Confirm `192.168.2.182`.
 10. Confirm Docker and Traefik.
 11. Test all routers.
 12. Reconnect public traffic only after validation.
-

Security Checklist

- Traefik TCP 443 is the only intended public application entry point.
 - Private backend ports are not publicly exposed.
 - The Traefik dashboard requires `bytek-admin`.
 - The dashboard callback path bypasses ForwardAuth.
 - Authentik is not protected by its own ForwardAuth.
 - Nextcloud uses native OIDC.
 - Vikunja uses native OIDC.
 - BookStack uses native OIDC.
 - Certificate API credentials are protected.
 - `acme.json` has restrictive permissions.
 - Container DNS points to Pi-hole.
 - The Docker socket is mounted read-only where supported.
 - The Docker API is not publicly exposed.
 - SSH uses keys.
 - Dynamic configuration is backed up.
 - Basic Auth recovery credentials are stored securely.
 - Proxmox backup includes the Traefik VM.
-

Validation Checklist

- Traefik VM owns `192.168.2.182`.
 - Docker is active.
 - Traefik container is running.
 - TCP 443 is listening.
 - Pi-hole DNS works inside the container.
 - Dynamic YAML contains no tabs.
 - Authentik router works.
 - Nextcloud router works.
 - Vikunja router works.
 - BookStack router works.
 - Uptime Kuma router works.
 - Dashboard authentication works.
 - Outpost ping returns HTTP 204.
 - Certificates match every public hostname.
 - Certificate renewal works.
 - Public ingress works through the VPS and WireGuard.
 - Internal split-DNS access works directly.
 - `acme.json` is included in backup.
 - A known-good dynamic configuration backup exists.
-

Document Control

- **Owner:** Bryan Gagne-Plante
- **Private address:** `192.168.2.182`
- **Public dashboard:** `proxy.bytek.ca`
- **Primary entry point:** TCP 443
- **Deployment directory:** `/opt/traefik`
- **Certificate authority:** Let's Encrypt
- **ACME challenge:** DNS-01 through WHC cPanel API
- **Container DNS:** `192.168.2.65`
- **Dashboard authentication:** Authentik ForwardAuth
- **Secondary authentication:** Basic Auth while retained
- **Proxmox VM ID:** Confirm current VM ID
- **Last verified:** YYYY-MM-DD
- **Last certificate-renewal test:** YYYY-MM-DD
- **Last ForwardAuth test:** YYYY-MM-DD

- **Last restore test:** YYYY-MM-DD
 - **Known limitation:** Public applications depend on one Traefik VM and one certificate store
-

Revision #2

Created 2026-08-17 14:50:39 UTC by Bryan Gagne-Plante

Updated 2026-08-18 00:23:15 UTC by Bryan Gagne-Plante