EN

Let's Encrypt and ACME on OPNsense

Free TLS Certificates for All Your Internal Services

BOTUM OPNsense Stack Series — Episode 14/15

Mars 2026

EPISODE 14

Table of Contents

Let's Encrypt and ACME on OPNsense: Free TLS Certificates for All Your Internal Services

In Episode 14 of the BOTUM OPNsense Stack Series, we set up **Let's Encrypt** via the ACME protocol directly on OPNsense. Goal: every internal service (Grafana, Wazuh, AdGuard, HAProxy) gets a valid TLS certificate, automatically renewed, without opening port 80.

1. Why TLS Even for Internal Services?

Most administrators skip TLS on the internal network: *"nobody can intercept LAN traffic anyway"*. This is a mistake.

Critical reasons:

• **HSTS (HTTP Strict Transport Security)**: Modern browsers refuse HTTP connections to domains that previously responded over HTTPS. Without internal TLS, your services become unreachable after a first HTTPS visit.

• **Browser trust**: 'Invalid certificate' security warnings train users to ignore alerts — making them vulnerable to real attacks.

• **Internal Zero-Trust**: An attacker on the IoT VLAN (Episode 2) can intercept unencrypted traffic between services. Internal TLS + MTLS = defense in depth.

• **APIs and webhooks**: Grafana, Wazuh, AdGuard Home expose APIs. Without TLS, authentication tokens travel in plaintext.

• **Let's Encrypt = free**: No economic reason not to use it. OPNsense's ACME plugin automates everything.

2. Installing the os-acme-client Plugin on OPNsense

The **os-acme-client** plugin is available directly in the OPNsense plugin manager. It integrates the acme.sh ACME client and natively manages Let's Encrypt.

```
# Via OPNsense Web UI:
# System > Firmware > Plugins > search "acme"
# Install: os-acme-client
# Restart OPNsense after installation

# Verification via OPNsense SSH:
pkg info | grep acme
# -> os-acme-client-x.x_x   ACME client plugin

# The menu appears under:
# Services > ACME Client > Settings / Accounts / Certificates / Automations
```

After installation, create an ACME account (email for expiry notifications):

```
# Services > ACME Client > Accounts > Add
Name        : botum-letsencrypt
E-Mail      : admin@botum.ca
ACME Server : Let's Encrypt (production)
              https://acme-v02.api.letsencrypt.org/directory
# Note: use "staging" for tests (limited to 5 certs/week in prod)
```

3. Domain Validation: DNS Challenge Without Port 80

The **DNS-01** validation is the only method that works without exposing port 80 publicly — essential for internal services.

3.1 Cloudflare (recommended)

```
# 1. Create a Cloudflare API token (minimal permissions):
# Cloudflare Dashboard > My Profile > API Tokens > Create Token
# Permissions: Zone > DNS > Edit
# Zone Resources: Include > Specific zone > botum.ca
# Copy the generated token

# 2. OPNsense > Services > ACME Client > Challenge Types > Add
Name          : cloudflare-dns
Challenge     : DNS-01
DNS Service   : Cloudflare
CF_Token      : [your-cloudflare-api-token]
CF_Account_ID : [your-account-id] (visible in Cloudflare Dashboard > right)

# 3. Alternative: CF_Key + CF_Email (legacy Global API Key method)
# Discouraged - prefer scoped API token
```

3.2 OVH / Gandi

```
# OVH - Services > ACME Client > Challenge Types > Add
Name         : ovh-dns
Challenge    : DNS-01
DNS Service  : OVH
OVH_AK       : [Application Key from api.ovh.com]
OVH_AS       : [Application Secret]
OVH_CK       : [Consumer Key]
OVH_ENDPOINT : ovh-ca   # or ovh-eu depending on region

# Gandi
Name         : gandi-dns
Challenge    : DNS-01
DNS Service  : Gandi LiveDNS
GANDI_LIVEDNS_KEY : [Gandi API key]

# Generate Gandi API key:
# account.gandi.net > Security > API Keys > Create
```

4. Generating a Wildcard Certificate *.botum.ca

A **wildcard** certificate covers all subdomains: *grafana.botum.ca*, *wazuh.botum.ca*, *adguard.botum.ca*, etc. One certificate, centralized renewal.

```
# Services > ACME Client > Certificates > Add
Name               : wildcard-botum-ca
Common Name        : *.botum.ca
Challenge Type     : cloudflare-dns  (created in section 3)
Key Length         : 4096 bit (or ec-384 for ECDSA)
OCSP Must-Staple   : disabled (optional)
Auto Renew         : enable
Renew before (d)   : 30  (renew 30 days before expiry)

# After creation, click "Issue / Renew"
# The ACME plugin will:
#  1. Contact Let's Encrypt
#  2. Create DNS TXT record _acme-challenge.botum.ca via Cloudflare API
#  3. Let's Encrypt validates the TXT record
#  4. Certificate + private key downloaded and stored in OPNsense
#  5. TXT record automatically deleted

# Verification:
# System > Trust > Certificates
# -> wildcard-botum-ca should appear with ~90 days expiry
```

5. Distributing the Certificate: OPNsense, HAProxy, Nginx, Docker

5.1 OPNsense Web UI (admin panel)

```
# System > Settings > Administration
SSL Certificate : wildcard-botum-ca
-> OPNsense admin UI switches to valid HTTPS
-> Accessible via https://fw.botum.ca/ without browser warning
```

5.2 HAProxy (reverse proxy)

```
# Services > HAProxy > Settings
# In each HTTPS Frontend:
SSL Certificates : wildcard-botum-ca
SSL offloading   : enable
# HAProxy handles TLS termination for all backend services

# Example Grafana frontend:
# Name : grafana-https
# Bind : 0.0.0.0:443 (Servers VLAN)
# SSL  : wildcard-botum-ca
# Backend : grafana-backend (192.168.20.x:3000)
```

5.3 Nginx (internal reverse proxy)

```
# On the Nginx server, via SFTP/SCP from OPNsense:
# OPNsense stores certificates in:
# /var/etc/acme-client/home/[cert-name]/

# Automation via ACME post-renewal action:
# Services > ACME Client > Automations > Add
Name    : deploy-to-nginx
Run Command : /usr/local/sbin/deploy_cert_nginx.sh

# /usr/local/sbin/deploy_cert_nginx.sh
#!/bin/sh
CERT_SRC="/var/etc/acme-client/home/wildcard-botum-ca"
NGINX_HOST="192.168.20.5"
scp $CERT_SRC/fullchain.cer admin@$NGINX_HOST:/etc/nginx/ssl/botum.ca.crt
scp $CERT_SRC/*.key         admin@$NGINX_HOST:/etc/nginx/ssl/botum.ca.key
ssh admin@$NGINX_HOST "nginx -t && nginx -s reload"
echo "Nginx cert deployed: $(date)" >> /var/log/acme_deploy.log
```

5.4 Docker Services

```
# Mount certificate into containers via Docker volume
# On Docker host (us-srv-dck-01):

# docker-compose.yml - Grafana with TLS example
services:
  grafana:
    image: grafana/grafana:latest
    volumes:
      - /opt/certs/botum.ca.crt:/etc/grafana/ssl/cert.pem:ro
      - /opt/certs/botum.ca.key:/etc/grafana/ssl/key.pem:ro
    environment:
      GF_SERVER_PROTOCOL: https
      GF_SERVER_CERT_FILE: /etc/grafana/ssl/cert.pem
      GF_SERVER_CERT_KEY: /etc/grafana/ssl/key.pem

# Certificate sync script (cron on Docker host):
# /opt/scripts/sync_certs.sh
#!/bin/bash
SRC="admin@192.168.10.1:/var/etc/acme-client/home/wildcard-botum-ca"
scp -i /root/.ssh/id_ed25519 $SRC/fullchain.cer /opt/certs/botum.ca.crt
scp -i /root/.ssh/id_ed25519 $SRC/*.key         /opt/certs/botum.ca.key
chmod 640 /opt/certs/botum.ca.key
docker compose -f /opt/stacks/monitoring/docker-compose.yml restart grafana wazuh
echo "Certs synced: $(date)" >> /var/log/cert_sync.log

# Crontab (sync day after ACME renewal):
# 0 3 * * * /opt/scripts/sync_certs.sh >> /var/log/cert_sync.log 2>&1
```

6. Automatic Renewal (Native ACME Cron)

The os-acme-client plugin includes a **native cron** that automatically checks and renews certificates. Let's Encrypt issues certificates valid for **90 days**; renewal triggers 30 days before expiry.

```
# Enable ACME cron:
# Services > ACME Client > Settings
# [x] Enable Cron Job
# Run interval : Daily (recommended)
# Scheduled at : 03:30 (overnight to minimize impact)

# What the cron does automatically:
# 1. Checks all configured certificates
# 2. If expiry < 30 days -> automatic renewal
# 3. Performs DNS-01 challenge (Cloudflare/OVH/Gandi)
# 4. Downloads new certificate
# 5. Triggers post-renewal Automations
# 6. Reloads OPNsense with new certificate

# Check ACME logs:
# Services > ACME Client > Log File
# or OPNsense SSH:
tail -100 /var/log/acme.log | grep -E "SUCCESS|ERROR|Renew"

# Manual renewal test (without actually renewing):
# Services > ACME Client > Certificates > [cert] > Simulate
# Or force immediate renewal:
# [button] Issue / Renew
```

7. Expiry Alerting (Monitoring Integration from Episode 9)

Even with automatic renewal, proactive certificate expiry alerts are essential — renewal can fail (DNS API unavailable, Let's Encrypt rate limit, etc.).

7.1 Certificate Monitoring Script

```
#!/bin/bash
# /opt/scripts/check_cert_expiry.sh
# Checks certificate expiry and alerts via Telegram

TELEGRAM_TOKEN="[your-token]"
TELEGRAM_CHAT="-1001234567890"
WARN_DAYS=14
CRIT_DAYS=7

alert_telegram() {
    curl -s "https://api.telegram.org/bot${TELEGRAM_TOKEN}/sendMessage"          -d "chat_id=${TELEGRAM_CHAT}"          -
}

for cert in grafana.botum.ca wazuh.botum.ca adguard.botum.ca fw.botum.ca; do
    EXPIRY=$(echo | openssl s_client -connect ${cert}:443 -servername ${cert} 2>/dev/null              | openssl x509 -no
    if [ -z "$EXPIRY" ]; then
        alert_telegram "CERT ALERT: Cannot verify ${cert}"
        continue
    fi
    EXPIRY_EPOCH=$(date -d "$EXPIRY" +%s)
    NOW_EPOCH=$(date +%s)
    DAYS_LEFT=$(( (EXPIRY_EPOCH - NOW_EPOCH) / 86400 ))

    if [ $DAYS_LEFT -le $CRIT_DAYS ]; then
        alert_telegram "CRITICAL: Cert ${cert} expires in ${DAYS_LEFT} days!"
    elif [ $DAYS_LEFT -le $WARN_DAYS ]; then
        alert_telegram "WARNING: Cert ${cert} expires in ${DAYS_LEFT} days"
    fi
    echo "${cert}: ${DAYS_LEFT} days remaining"
done

# Crontab (daily at 08:00):
# 0 8 * * * /opt/scripts/check_cert_expiry.sh
```

7.2 Prometheus / Grafana Integration (Episode 9)

```
# Export certificate metrics via ssl_exporter
# docker-compose.yml
services:
  ssl-exporter:
    image: ribbybibby/ssl-exporter:latest
    ports:
      - "9219:9219"
    volumes:
      - ./ssl_targets.yml:/ssl_targets.yml:ro

# ssl_targets.yml
- targets:
    - grafana.botum.ca:443
    - wazuh.botum.ca:443
    - adguard.botum.ca:443
    - fw.botum.ca:443

# Prometheus alert rules (rules/certs.yml)
groups:
  - name: cert-expiry
    rules:
      - alert: CertExpiryWarning
        expr: ssl_cert_not_after - time() < 86400 * 14
        labels: { severity: warning }
        annotations:
          summary: "Cert {{ $labels.target }} expiring soon"
      - alert: CertExpiryCritical
        expr: ssl_cert_not_after - time() < 86400 * 7
        labels: { severity: critical }
        annotations:
          summary: "CRITICAL: Cert {{ $labels.target }} expires in < 7 days!"
      - alert: CertExpired
        expr: ssl_cert_not_after - time() < 0
        labels: { severity: critical }
        annotations:
          summary: "EXPIRED: {{ $labels.target }} certificate is expired!"
```

8. Use Cases: AdGuard DoH, Grafana, Wazuh Dashboard

Let's Encrypt certificates directly improve three key services in our stack.

8.1 AdGuard Home DNS-over-HTTPS (Episode 8)

```
# AdGuard Home > Settings > Encryption Settings
Enable encryption : Yes
Server name       : adguard.botum.ca
HTTPS port        : 443
DNS-over-HTTPS    : https://adguard.botum.ca/dns-query
TLS certificate   : /etc/adguard/ssl/botum.ca.crt  (synced from OPNsense)
Private key       : /etc/adguard/ssl/botum.ca.key

# On clients (browsers, iOS, Android):
# DNS-over-HTTPS provider: https://adguard.botum.ca/dns-query
# End-to-end encryption without certificate warning
```

8.2 Grafana (Episode 9)

```
# grafana.ini (or Docker env variables):
[server]
protocol         = https
cert_file        = /etc/grafana/ssl/cert.pem
cert_key         = /etc/grafana/ssl/key.pem
domain           = grafana.botum.ca
root_url         = https://grafana.botum.ca/
enforce_domain   = true

# Result: https://grafana.botum.ca -> green padlock
# Automatic HSTS -> cannot downgrade to HTTP
# Prometheus -> Grafana API calls encrypted
```

8.3 Wazuh Dashboard (Episode 11)

```
# /etc/wazuh-dashboard/opensearch_dashboards.yml
server.host: 0.0.0.0
server.port: 5601
server.ssl.enabled: true
server.ssl.certificate: /etc/wazuh-dashboard/certs/wazuh-dashboard.pem
server.ssl.key: /etc/wazuh-dashboard/certs/wazuh-dashboard-key.pem

# Replace Wazuh self-signed certs with wildcard botum.ca:
# Copy botum.ca.crt -> wazuh-dashboard.pem
# Copy botum.ca.key -> wazuh-dashboard-key.pem
# Restart: systemctl restart wazuh-dashboard
# Result: https://wazuh.botum.ca:5601 -> valid certificate, no warning
```

9. Conclusion: A Unified and Automated TLS Stack

With Episode 14, **all services in the BOTUM OPNsense Stack are secured with a valid TLS certificate**, automatically renewed, with zero manual intervention:

- **os-acme-client** — Native OPNsense plugin, Let's Encrypt integration
- **DNS-01 challenge** — Validation without port 80, Cloudflare/OVH/Gandi support
- **Wildcard *.botum.ca** — One certificate for all subdomains
- **HAProxy + Nginx** — Centralized TLS termination
- **Docker services** — Certificates auto-synchronized
- **ACME cron** — Automatic renewal 30 days before expiry
- **Telegram + Prometheus alerting** — Proactive expiry monitoring

Coming next (Episode 15/15): Netflow + ntopng on OPNsense — real-time network traffic analysis, flow visualization by application, anomaly detection. The final episode of the series!

Full article: blog.botum.ca/opnsense-lets-encrypt-acme-certificates

Website: www.botum.ca | contact@botum.ca