
ceph days
LONDON 2026

ceph days
LONDON 2026

No More Expired Certificates: Simplifying TLS Management in Ceph with Certmgr

ceph days
LONDON 2026

Redouane Kachach
rkachach@ibm.com

- Software Architect at IBM
- 20+ years of experience (Motorola, Nokia, Bell-Labs, Red Hat, IBM)
- Cephadm component lead and core contributor since 2022

ceph days LONDON 2026

Agenda

- Why cert lifecycle is hard in production clusters
- Certmgr responsibilities + Architecture
- Data model + certificate scopes
- Unified cephadm spec TLS fields
- Validation + status classification
- Decision tree: auto-rotate vs operator action
- Health reporting and operator UX

ceph days LONDON 2026

The problem Certmgr solves

Operational Complexity

- Many services, many TLS surfaces (RGW, NFS, Grafana, ...)
- Certs embedded inline in service specs (no central inventory)
- Plaintext keys in specs are a security risk (leak via git, backups, logs)
- Many services × hosts × scopes, each with its own lifecycle
- No central inventory: who owns what, where it's used, when it expires

Operational Risks

- Expired cert → service unreachable
- Leaked spec → forced emergency key rotation
- Manual hunt to locate cert usage and ownership
- Drift: out-of-band changes break cephadm reconciliation

ceph days LONDON 2026

What is Cephadm Certmgr?

Core Responsibilities

- Own Ceph cluster Root CA
- Issue / renew service certificates
- Validate cert/key material continuously
- Decide: auto-remediate vs operator action
- Publish health warnings/errors with context

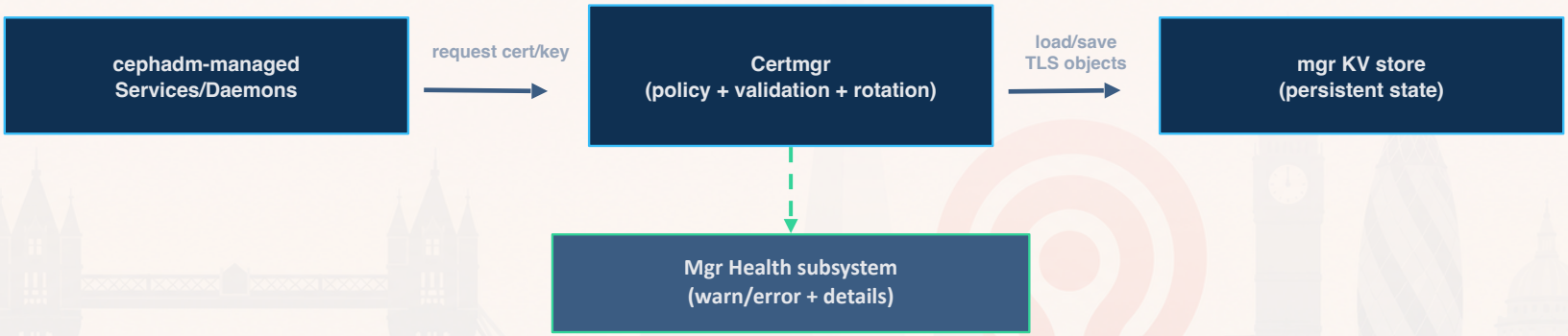
Success Criteria

- Secure-by-default TLS everywhere
- Auto-fix safe cases (generated certs)
- Escalate unsafe cases (user certs)
- Simple introspection for operators

Ops-safe defaults | Scopes: global/host/service | Auto-renew where safe | Actionable health alerts

ceph days LONDON 2026

Architecture Overview



→**Certmgr is a submodule of cephadm** (not a standalone CLI)

ceph days LONDON 2026

Data model: TLS objects + scopes

GLOBAL

→**Cluster-wide CA** / Shared material

```
Storage shape
entity → object

Examples:

   ●   Mgmt-gateway
   ●   Cephadm Root CA


→  One global instance
```

HOST

→Bound to a specific **node/host**

```
Storage shape
entity → target(host) → object

Examples:

   ●   Grafana
   ●   Prometheus
   ●   Alertmanager
   ●   …
→ per-host instances
```

SERVICE

→Bound to a **service instance**

```
Storage shape
entity → target(service) → object

Examples:

   ●   RGW
   ●   NFS
   ●   NvmeOF
   ●   …
→ per-service instances
```

ceph days LONDON 2026

Inconsistent TLS Fields Across Service Specs

Some examples from cephadm legacy Specs:

TLS Certificate/Key fields:

- rgw → rgw_frontend_ssl_certificate
- nvmeof → server_cert / server_key
- iscsi → ssl_cert / ssl_key
- …

TLS boolean enable/disable fields:

- rgw → ssl / generate_cert
- nvmeof → enable_auth
- iscsi → ssl
- …

→ No consistent support for cephadm-signed certificates across the services

ceph days LONDON 2026

Certmgr: Unified Spec Files Syntax

Cephadm Spec Fields:

- **ssl:** Boolean to enable/disable SSL/TLS
- **ssl_cert:** Certificate content (for inline)
- **ssl_key:** Key content (for inline)
- **certificate_source:**
  - **inline:** legacy embedded certs/keys in spec
  - **reference:** Reference to cert/key stored in the certmgr
  - **cephadm_signed:** generated by cephadm

Unified support for cephadm-signed certificates across all the services

Spec example:

```
service_type: rgw
service_id: my-rgw
spec:
  ssl: true
```

ceph days LONDON 2026

Naming & Ownership

Unified naming conventions

per-service certificates naming convention:

```
<service>_ssl_cert
<service>_ssl_key
```

cephadm-signed certificates naming convention:

```
cephadm-signed_<service>[_<label>]_cert
cephadm-signed_<service>[_<label>]_key
```

Ownership flags (per object)

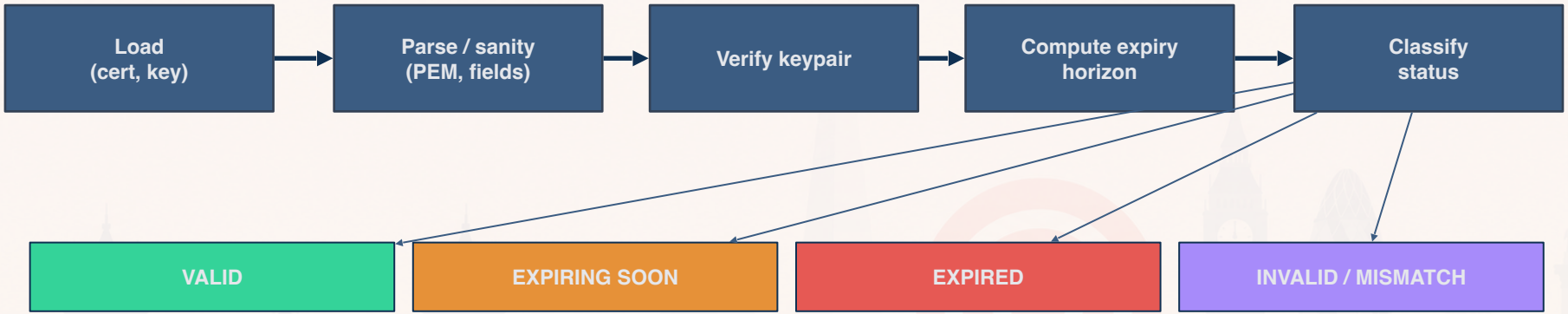
- **user_made = true** → operator-owned material
- **editable = true** → replacement allowed via mgmt interface
- **Policy:** user-made + invalid → alert (no auto-replace)
- **Policy:** generated + expiring → auto-renew (if enabled)

ceph days LONDON 2026

Certmgr Configuration Parameters

Option	Type	Default	Range	What it controls
certificate_check_period	int	1 day	0 - 30 days	**How often CertMgr checks** stored certificates for validity → 0 disables periodic checking.
certificate_automated_rotation_enabled	bool	True	—	Whether cephadm **automatically rotates** cephadm-signed certificates when they are expiring/expired.
certificate_renewal_threshold_days	int	30 days	10 - 90 days	**Lead time before expiry** to treat a cert as "expiring" and initiate renewal actions.
certificate_duration_days	int	3 years	90 - 3650 (10 years)	**Validity duration** for newly generated cephadm-signed certificates.

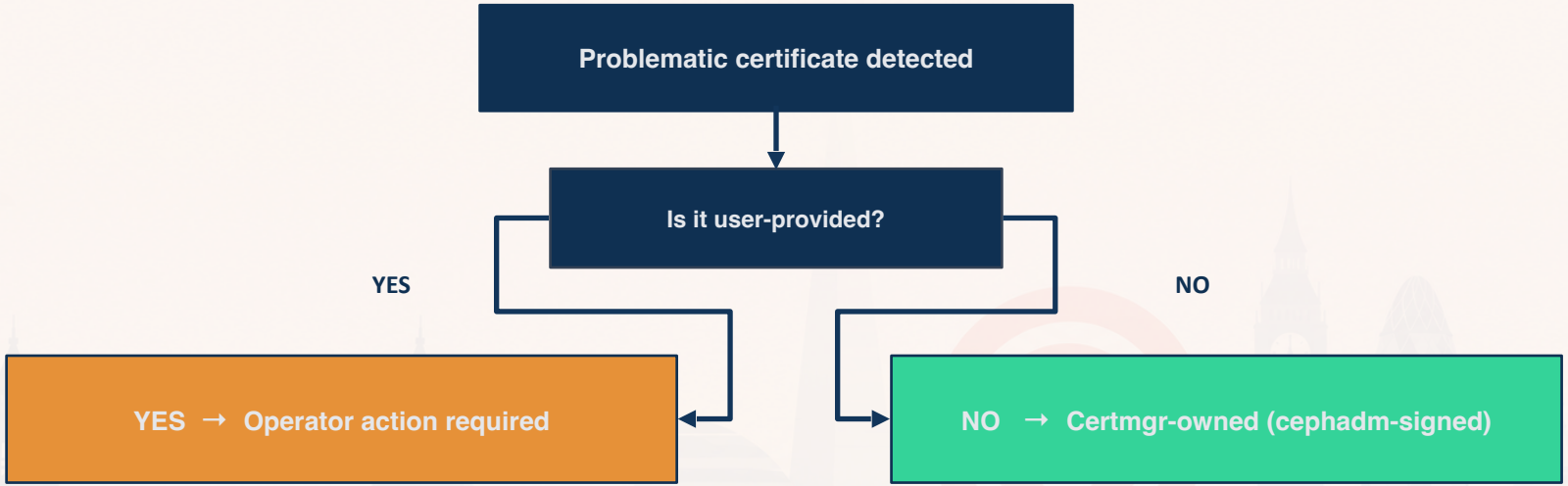
Validation Pipeline



- Operationally healthy → cryptographically valid AND not inside the “too close to expiry” window
- Expiring-soon certs remain usable but trigger HEALTH_WARN so renewal happens before expiry
- Status drives both remediation decisions and health messages

ceph days LONDON 2026

Remediation Decision Tree

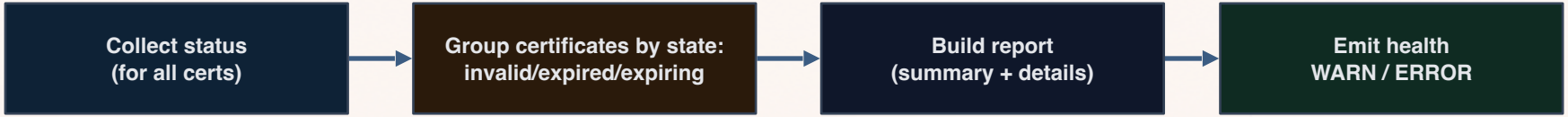


- Emit HEALTH_WARN/HEALTH_ERR
- Explain invalid/expired/expiring details
- Do not auto-replace user material

- If invalid → regenerate
- If expiring → renew
- Mark affected services "needs reconfig"

Note: when auto-rotation is disabled → expiring cephadm-signed certs become "operator attention" instead of auto-renew

Health Reporting



Example output:

```
{
  "severity": "HEALTH_ERR",
  "summary": {
    "message": "Detected 2 cephadm certificate(s) issues: 1 invalid, 1 expired",
    "count": 2
  },
  "detail": [
    {
      "message": "Certificate 'grafana_cert (ceph-node-0)' has expired."
    },
    {
      "message": "Invalid certificate 'mgmt_gateway_ssl_cert': Invalid cert/key pair: [('key values mismatch')]."
    }
  ],
  "muted": false
}
```

ceph days LONDON 2026

Operator UX: Inspection & Response

Inspection checklist

- List certs/keys (include scope & ownership flags)
- Filter by status: **expired / expiring / invalid**
- Confirm matching key exists for each cert
- For user-provided: verify chain + SANs

Fast response playbook

- HEALTH_ERR (**expired**):
  - generated → rotate
  - user → replace cert/key
- HEALTH_WARN (**expiring**): schedule renewal
- After changes: **redeploy/reconfig** affected services
- Confirm health clears

Operator UX: CLI

Basic certmgr command examples:

For listing certs:

- `ceph orch certmgr cert ls`
- `ceph orch certmgr cert ls --show-details`
- `ceph orch certmgr cert ls --include-cephadm-signed`
- `ceph orch certmgr cert ls --filter-by "scope=service,status=expiring"`
- `ceph orch certmgr cert ls --include-cephadm-signed --filter-by "name=rgw*,status=valid"`

For private keys:

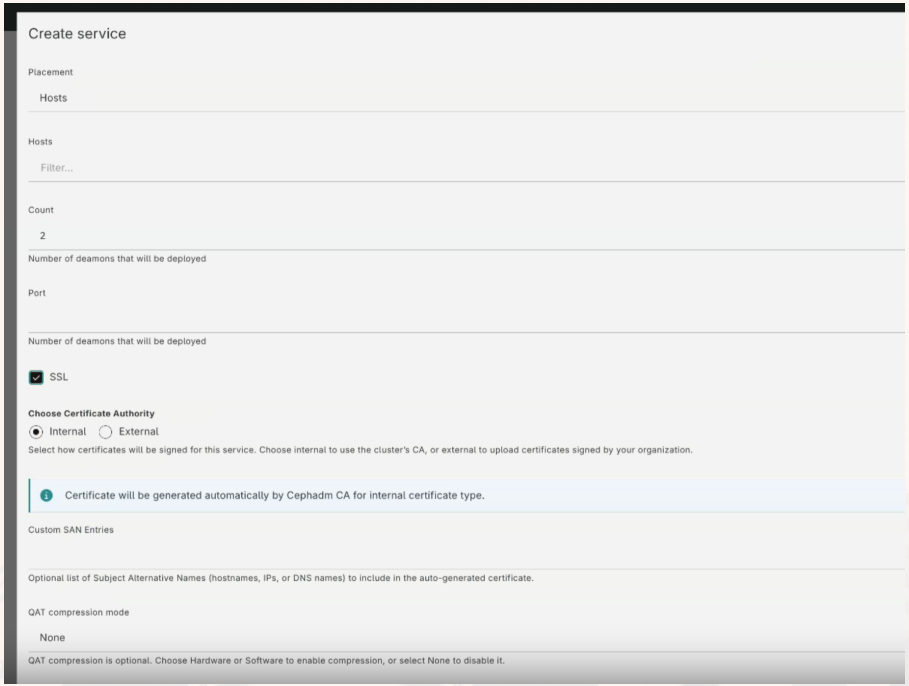
- `ceph orch certmgr key ls`
- `ceph orch certmgr key ls --include-cephadm-generated-keys`

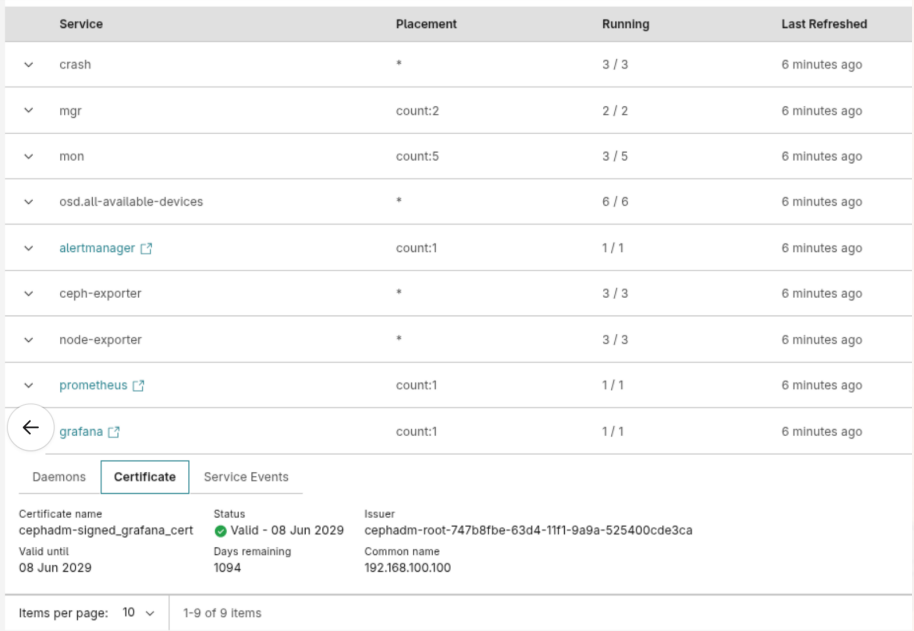
For setting certs/keys:

- `ceph orch bindings ls`
- `ceph orch certmgr cert set --cert-name <cert-name> --service-name <service-name> -i <cert-file>`
- `ceph orch certmgr key set --key-name <key-name> --service-name <service-name> -i <key-path>`

ceph days LONDON 2026

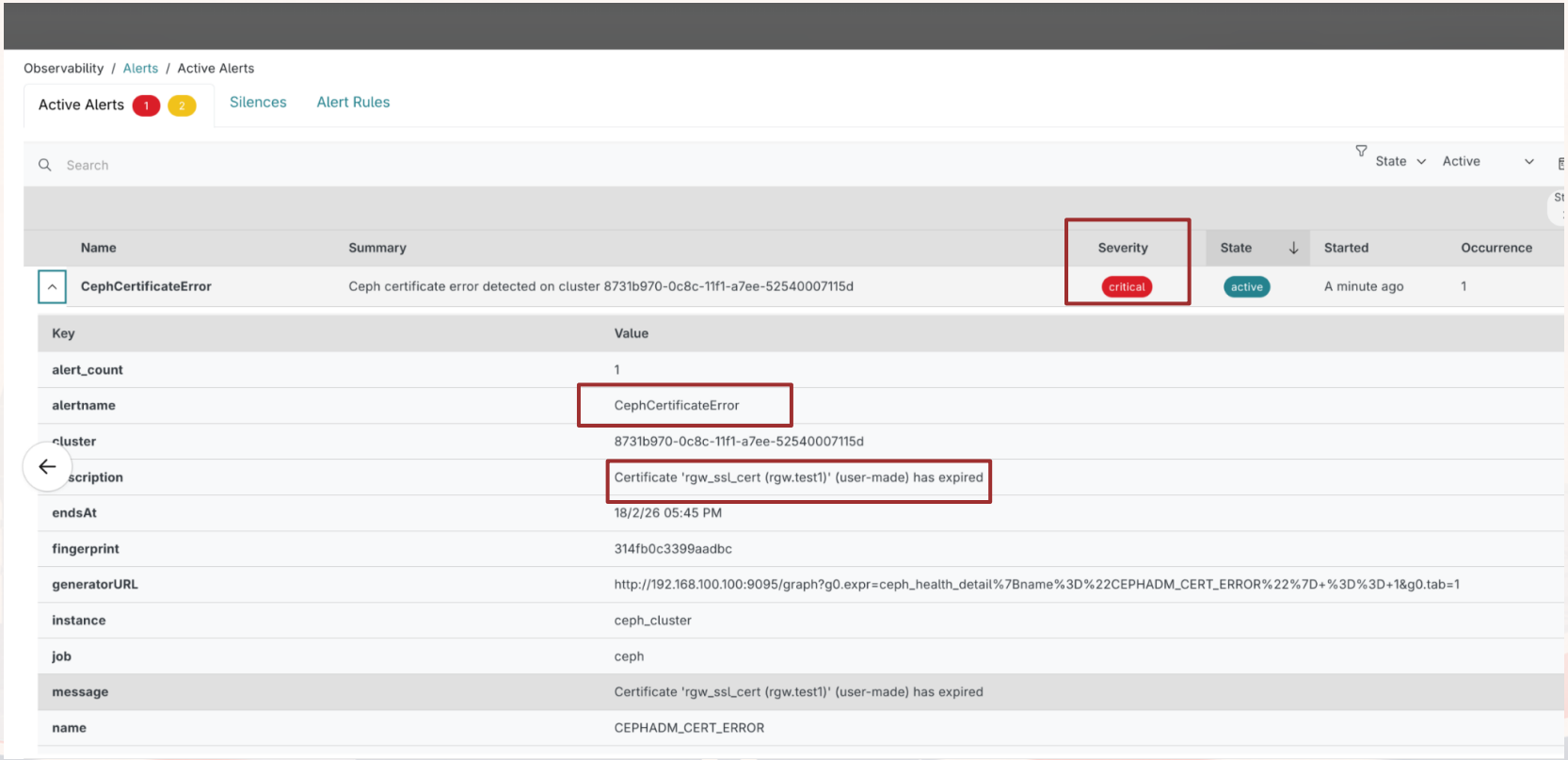
Dashboard Integration: Provisioning





ceph days LONDON 2026

Dashboard Integration: Monitoring



ceph days
LONDON 2026

Questions ?