Building Your Own Internal CA
for Private Certificates

Running a handful of OpenSSL commands by hand is a reasonable way to learn how a private Certificate Authority works, but it is not a viable way to run one at organizational scale. Once an internal PKI needs to issue, track, and rotate more than a small number of certificates, it needs to become a real, governed system: an internal Certificate Authority with defined policies, appropriate hardware protection, and automation baked in from the start. This article covers what that actually looks like in practice.

Why Organizations Build Their Own CA

Internal CAs exist because a large share of an organization's certificate needs are entirely internal: service-to-service authentication, VPN client certificates, internal web applications, device identity, and machine-to-machine credentials that have no business being validated by a public CA and no need for public trust at all. A private CA lets an organization issue exactly the certificates it needs, on its own schedule, without depending on external rate limits or validation turnaround times, while also allowing custom certificate policies tailored to internal risk tolerance rather than generic public standards.

The Two-Tier Architecture

Nearly every well-run internal CA follows the same structural pattern used by public CAs: a root CA that is used as rarely as possible and kept as isolated as possible, and one or more intermediate CAs that handle actual day-to-day issuance. This separation limits the damage of a compromise; if an intermediate is ever breached, it can be revoked and replaced without touching the root or re-establishing trust across the entire organization. Some larger organizations extend this to a three-tier model, adding a policy CA layer between root and issuing intermediates to further segment issuance by business unit or use case.

Writing a Certificate Policy

A mature internal CA operates under a documented Certificate Policy and Certification Practice Statement, even in an informal, internal-only form. These documents define what types of certificates the CA issues, what validation is required before issuance, how long certificates remain valid, key length and algorithm requirements, and the procedures for revocation. Skipping this step tends to produce an internal CA that works fine on day one and becomes an ungoverned mess within a year, as different teams request certificates for different purposes with no consistent standard behind any of them.

Protecting the Root Key

The root CA's private key deserves the strongest protection an organization can reasonably provide. Many organizations keep the root key in a hardware security module, physically air-gapped from any network connection, brought online only for the rare occasion of signing a new intermediate certificate. Smaller organizations without dedicated HSM budgets can still improve on a plain file on disk by using strong encryption, strict access controls, and split-knowledge procedures requiring more than one person to access the key material.

Choosing Tooling for Issuance and Automation

Manually running OpenSSL commands for every certificate does not scale past a very small environment. Purpose-built tools exist specifically to run an internal CA with automation in mind, ranging from lightweight open-source options like step-ca to full enterprise platforms like Microsoft Active Directory Certificate Services, HashiCorp Vault's PKI secrets engine, and dedicated commercial machine identity management platforms. Most of these support the ACME protocol internally, meaning services can request and renew certificates the same automated way they would from a public CA, just pointed at an internal endpoint instead.

Certificate Templates and Least Privilege

Rather than issuing every certificate manually with custom parameters, mature internal CAs define templates: pre-approved certificate profiles for common use cases, such as an internal web server template, a VPN client template, and a machine identity template scoped tightly for AI agents and automated service accounts. Templates enforce consistency and prevent an over-privileged certificate from being issued by accident, since the template itself restricts key usage, validity period, and allowed extensions before a human ever gets involved.

Issuing Identities to AI Agents and Automated Services

One of the fastest-growing categories of internal certificate demand comes from AI agents and the automated services that support them: model-serving endpoints, orchestration layers, retrieval systems, and the agents themselves when they act as clients calling other internal services. These identities benefit enormously from a dedicated template with a short validity window, tight scope, and full automation, since an AI-driven pipeline may provision and retire dozens of short-lived workloads in a single hour, each needing its own scoped identity rather than sharing one broad credential across every agent instance. Internal CAs designed with this use case in mind from the start avoid the common anti-pattern of issuing one long-lived, overly broad certificate to an entire AI platform and hoping nothing goes wrong.

Revocation and Ongoing Monitoring

An internal CA needs a working revocation mechanism just as much as a public one does, whether through a Certificate Revocation List, OCSP responder, or the revocation features built into modern PKI platforms. Monitoring should track every certificate's expiration date, flag anomalous issuance patterns, and alert on certificates approaching expiration well ahead of the deadline. Without this operational layer, an internal CA becomes a system nobody trusts to actually enforce revocation when it matters, which defeats much of the point of running one in the first place.

Bringing It All Together

Building an internal CA is a meaningfully bigger commitment than generating a few certificates with OpenSSL, but it is the right investment for any organization whose internal certificate needs have outgrown ad hoc management. A well-run internal CA, with a clear policy, protected root key, sensible templates, and strong automation, gives an organization the ability to issue trust on its own terms, at whatever scale its infrastructure, including its growing roster of AI-driven services, actually requires.

The Countdown Is Already Running: 200 Days, 100 Days, 47 Days

Every certificate conversation in 2026 eventually arrives at the same clock, and it is worth closing on it here. The CA/Browser Forum's Ballot SC-081v3 is not a proposal under discussion; it is an approved, already-in-motion schedule. Maximum public TLS certificate lifetimes fall from 398 days to 200 days on March 15, 2026. They fall again to 100 days on March 15, 2027. By March 15, 2029, they drop to just 47 days, with domain validation itself needing to be re-proven roughly every 10 days.

01

Translate that into operational terms and the picture gets stark quickly. An organization currently renewing certificates a few times a year will be handling renewal events on the order of every couple of weeks by the end of this countdown, across every endpoint it operates. Manual tracking, calendar reminders, and a spreadsheet somebody checks once a month will not survive contact with that cadence. What has always been an occasional chore is becoming a continuous, automated operation, whether an organization plans for it or not.

02

An internal CA built with templates and automation from day one is exactly the kind of system that will absorb the public certificate deadlines below without strain, since the same discipline of scoped, short-lived, automatically issued credentials applies on both sides of the trust boundary.

03

The 200-day, 100-day, and 47-day milestones are not distant hypotheticals; the first has already arrived. Organizations that build the automation loop now, generating keys, vaulting them securely, brokering issuance across Certificate Authorities through APIs, and rebinding certificates to live endpoints without manual intervention, will meet each deadline without disruption. Organizations that wait will be rebuilding their certificate operations under deadline pressure, with far less room for error and far less time to get it right. The countdown is the call to action. The only real decision left is whether to automate on your own schedule, or on the CA/Browser Forum's.