Automating Certificate Renewal with ACME and Certbot

Of all the tools that made automated certificate renewal a practical reality for millions of websites, Certbot is arguably the most widely deployed. Built as the reference client implementation for the ACME protocol, it turned what used to be a manual, once-a-year chore into a background process most administrators never have to think about again. This article covers how ACME and Certbot actually work together, and how to set up renewal automation that genuinely holds up over time.

The Protocol Underneath the Tool

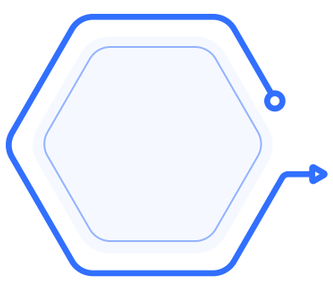
ACME, the Automatic Certificate Management Environment protocol, defines a standardized way for a server to prove domain control to a Certificate Authority and receive a certificate in return, entirely through automated network requests rather than a human filling out a web form. Certbot is a client implementation of that protocol, originally developed alongside Let's Encrypt but compatible with any CA that supports the ACME standard, which by now includes most major public CAs offering automated issuance.

How the Validation Challenge Actually Works

When Certbot requests a certificate, the CA issues a challenge that must be satisfied to prove domain control. The two most common challenge types are HTTP-01, where Certbot places a specific file at a defined path on the web server for the CA to retrieve and verify, and DNS-01, where Certbot creates a specific TXT record in the domain's DNS zone. HTTP-01 is simpler to set up for a single server serving public web traffic, while DNS-01 is the only option that works for wildcard certificates and is often preferred in environments where the web server itself is not directly reachable from the public internet, such as internal services validated through a split-horizon DNS setup.

Setting Up Genuinely Reliable Automated Renewal

Certbot installs a renewal check, typically run twice daily through a scheduled task or systemd timer, that examines every certificate it manages and renews any approaching its expiration window, commonly triggering renewal around thirty days before expiry. This built-in scheduling removes the single most common point of human failure, forgetting to renew, but it depends on the underlying system's scheduler actually running reliably, which is worth monitoring independently rather than assuming silently.



A renewal succeeding technically is not the same as a renewal being fully effective; Certbot supports post-renewal hooks that can automatically reload or restart the web server, load balancer, or other service consuming the certificate, ensuring the newly issued certificate is actually picked up rather than sitting on disk while the running service continues serving the old one until its next unrelated restart.

Monitoring the Automation Itself

Automation removes manual renewal labor, but it introduces a new failure mode worth guarding against: an automation pipeline that silently stops working without anyone noticing until a certificate actually expires. Production deployments relying on Certbot should pair it with independent monitoring that checks actual certificate expiration dates on live endpoints, separate from trusting that the renewal cron job or systemd timer ran successfully, since a permissions change, a DNS misconfiguration, or an unnoticed rate limit can quietly break automated renewal well before anyone would otherwise find out.

Scaling Beyond a Single Server

Certbot handles single-server and small-fleet deployments well, but organizations running certificates across many servers, containers, or a broader infrastructure-as-code pipeline often integrate ACME clients directly into their deployment automation rather than running Certbot interactively on each host, or adopt a dedicated certificate lifecycle management platform that centralizes ACME-based issuance across the entire fleet, discussed in more depth elsewhere in this series.

ACME Automation for AI Infrastructure Endpoints

The same ACME and Certbot patterns that secure a typical web server apply directly to the API endpoints, load balancers, and reverse proxies fronting AI model-serving infrastructure, which frequently need certificates issued and renewed at a pace well beyond what manual processes could support. Teams standing up new AI-serving endpoints benefit from baking ACME-based automation into the deployment template from day one, so every new endpoint inherits working certificate automation automatically rather than needing it configured as an afterthought.

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

Certbot's built-in automated renewal, running well ahead of expiration, is exactly the behavior every certificate will need to have as the schedule below compresses maximum lifetimes down to 47 days, making ACME-based tooling like this the practical baseline rather than an optional convenience.

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.