Ten years isn't long enough for maximum age settings

January 3, 2024

The recent news of the time interval is that a ten year certificate built into every Keybase client expired at the end of 2023. I had a little reaction to this:

> So much quiet damage has been done to so many people's pinned cert or private CA deployments by ten-year defaults. So much.
>
> (It happened to us so now our private OpenVPN CA root is much, much longer.)

If you're making an internal thing good for ten years, don't (whether it is a TLS certificate or, for example, setting a retention duration for some database). Ten years is either not long enough or too long. If you're writing an example and use a ten year validity period, please don't (people are sure to copy it). Ten years can sound like an implausibly long time when you're setting something up, but as we all know there's nothing quite as permanent as a quick hack and there you are ten years later with some problems. This happened to us with OpenVPN and we failed to find a solution, to our pain.

If you don't have a plan to either roll over and renew whatever it is or definitely take it out of service, ten years is far too short. If you want to keep something active until you change your mind, set your not-after date, retention duration, or whatever to as high as you can. Ten years is a terrible value for this; it looks and feels long enough but it isn't anywhere near sufficient.

(In some cases it may be a little bit of a problem that we're now only fourteen years away from the year 2038 issue. But if you have a problem there, it's better to find out about it early.)

If you want to definitely take your thing out of service after a certain period of time, ten years is far too long. If your thing is still in use as ten years approaches, it's almost guaranteed to have wormed its way into all sorts of places, so that taking it out of service will cause explosions and the possibility of this will get people to show up demanding that its lifetime be extended (assuming that people even remember where it's used and what depends on it).

If you have a plan to roll over, extend, or renew the duration, ten years is also far too long. As everyone has found out through painful experience with TLS certificates, doing something only once every few years is a recipe for problems (and for forgetting that it needs to be done at all). To make sure you keep in practice and the process still works, you need to do this much more frequently than once every ten years. At this point I'd probably try to do it twice a year.

As it happens I'm not without sin, because the retention time for our Prometheus database is currently '3650 days' from late November 2018. We may well run out of disk space before then and when we get there we may decide we don't need ten years (or more than ten years) after all, but I should probably bump that up anyway since our intention is 'keep as much as we have disk space for and maybe get more disk space if we're running short'. I should definitely have some sort of calendar entry for it, as a reminder just in case.

(We do now check all of our long-lived internal TLS certificates and alert if their expiry time is getting close. It's probably overkill but it doesn't hurt. And some of them are less than a decade away at this point.)

These are my WanderingThoughts
(About the blog)

Full index of entries
Recent comments

This is part of CSpace, and is written by ChrisSiebenmann.
Mastodon: @cks
~~Twitter~~ @thatcks

* * *

Categories: links, linux, programming, python, snark, solaris, spam, sysadmin, tech, unix, web
Also: (Sub)topics

This is a DWiki.
GettingAround
(Help)

Search:

(This is related to a much older discovery that '999 days' is not forever, and for that matter neither is 9999 days, although that's 27 years and change so it's closer. 99999 days is long enough for almost everyone to not worry about it, though.)

(One comment.)

Written on 03 January 2024.

« Why Unix's `lseek()` has that name instead of '`seek()`'

Page tools: View Source, Add Comment.

Search: Login: Password: Login

Atom Syndication: Recent Comments.