Linux and Secure Boot certificate expiration

[LWN subscriber-only content]

By **Jake Edge**
July 16, 2025

Linux users who have Secure Boot enabled on their systems knowingly or unknowingly rely on a key from Microsoft that is set to expire in September. After that point, Microsoft will no longer use that key to sign the shim first-stage UEFI bootloader that is used by Linux distributions to boot the kernel with Secure Boot. But the replacement key, which has been available since 2023, may not be installed on many systems; worse yet, it may require the hardware vendor to issue an update for the system firmware, which may or may not happen. It seems that the vast majority of systems will not be lost in the shuffle, but it may require extra work from distributors and users.

Mateus Rodrigues Costa raised the issue on the Fedora devel mailing list on July 8. He had noticed a warning that came with "this month's Windows 11 cumulative update"; it talked about Secure Boot certificates that are scheduled to expire starting in June 2026. Those particular certificates are separate from the one used for shim, which expires much sooner. In any case, the problem of certificate expiration is one that the Linux world will need to tackle.

The situation is rather complicated. Daniel P. Berrangé pointed to a page at the Linux Vendor Firmware Service (LVFS) site that describes it. LVFS is the home of fwupd and other tools that are used to update system firmware from Linux. LVFS and fwupd are the subject of an LWN article from 2020.

There are multiple moving parts to the problem. In order to do a Secure Boot into the Linux kernel, the UEFI boot process requires the first-stage bootloader to be signed with a key in the firmware database that has not expired. Those keys are contained in certificates, which have other information, such as an expiration date and signature(s). The certificate expiration should largely only be a problem when installing a new distribution on a Secure Boot system; the shim that gets installed will have distribution-specific keys and can act as the root of trust for running other programs (e.g. GRUB) using those keys.

Currently, shim is signed with a Microsoft key from 2011 that expires on September 11. Past that point, installation media will no longer boot unless it has an updated shim that is signed with the Microsoft 2023 UEFI key for third-parties (which is different than the specific key mentioned in the Windows update). Any installed distribution should have a bootloader signed with its own key that will continue to boot.

But there are lots of systems out there with a firmware database that lacks Microsoft's new key, some have both old and new keys, while there are likely some that only have the new key and cannot Secure Boot Linux installation media at all at this point. Vendors can (and hopefully most will) provide firmware updates that add the new key, and installation media can be created with a shim signed by it, but those updates have to be installed on systems. That's where LVFS and fwupd come in.

LVFS is a repository of vendor-firmware updates of various sorts, which fwupd and other tools can use to install the pieces that are needed into the firmware from Linux. Berrangé noted that older versions of fwupd were unable to solve all of the problems, "but recent releases have been enhanced to handle the updates that Linux users will need to see, which should mitigate the worst of the impact". There may still be a bit of a bumpy ride, however: "Users should be 'aware' of the potential for trouble, but hopefully the worst of the 'worry' part is handled by the OS vendors and maintainers."

LVFS creator and maintainer Richard Hughes agreed, noting that there were various ways that people's systems would be able to get updated Secure Boot functionality. A full firmware update might be provided by the vendor, which would (presumably) add the new database, including the new Microsoft key. Another avenue would be a "key exchange key" (KEK) update, which is a vendor-specific key that is signed by the Microsoft key; it can be used by fwupd to update the database with the new key. But there are some caveats:

> The KEK updates are going out at ~98% success, and db update is ~99% success -- but even 1% multiplied by millions of people is a fair few failures to deploy -- the "failed to write efivarfs" thing. What fixes it for some people is rebooting and clearing the BIOS to factory defaults -- this has the effect of triggering a "de-fragmentation" of the available efivar space so that there's enough contiguous space to deploy the update. The older your BIOS the more likely you are to hit this.

Hughes is referring to a known problem with space for new EFI variables.

For systems where the vendor provides no updates, disabling Secure Boot may be the only option to allow a new install. In a few short months, all existing installation images and media will not be installable with Secure Boot—that may already be true for systems that only have the new key. Secure Boot installation just got that much more complicated.

Beyond that, though, is the possibility of mistakes or problems with the vendor updates. Hughes pointed out that at least one manufacturer has lost access to the private part of its platform key (PK), which is a vendor-specific key burned into the hardware when it is made. That means the platform keys in the hardware need to be changed, which is uncharted water and "a terrible idea from an attestation point of view". In addition, as Gerd Hoffman pointed out, the KEK update process is new as well: "a KEK update has never happened before so there are chances that bios vendors messed up things and updating the KEK doesn't work".

The thread has multiple reports on the Secure Boot certificates on various hardware models, as well as reports of updates to the KEK and database. One thing that is not entirely clear is whether the firmware implementations will actually enforce the expiration date on the 2011 key. A working system with a functional trust-chain based on that key might continue to work with a shim signed with that key, even after September. Any shim updates, for, say, security problems, would not be able to be signed with the old key, however—Microsoft will not sign anything using the expired key. That may lead to a "solution" of sorts, as Adam Williamson said:

> In theory wouldn't we also have the option to ship an old shim for such cases? If the whole chain is old it should work, right? Of course, we'd then need some heuristic to figure out if we're on the old MS cert and install the old shim...

He said that it may not really make sense and users should just disable Secure Boot. Hoffman agreed with all of that, but pointed out the problem with shim updates: "Continuing running shim with known security bugs makes it [kinda] pointless to have secure boot turned on".

All in all, it seems like the Linux world is doing the best it can under the circumstances—as is so often the case when it comes to hardware from vendors that mostly care about Windows. Given that the Secure Boot root-of-trust keys (both the platform key and the signing key) are under the control of vendors—Microsoft and the hardware makers—it is always going to be a bit of a struggle to keep up. Since older hardware is something that Linux and distributions explicitly support, while the other vendors have long since moved on to the latest shiny, it was clearly going to lead to some tension there. One can only hope that the ride is as smooth as it can be.

[–] Press X to doubt

Posted Jul 16, 2025 18:31 UTC (Wed) by **bluca** (subscriber, #118303) [Link] (4 responses)

> One thing that is not entirely clear is whether the firmware implementations will actually enforce the expiration date on the 2011 key.

Based on what I've seen over the years, I have a sneaking suspicion that the majority of firmwares actually won't check this

Reply to this comment

[–] Press X to doubt

Posted Jul 16, 2025 19:39 UTC (Wed) by **kraxel** (subscriber, #49444) [Link] (2 responses)

tianocore edk2 sets the NO_CHECK_TIME for pkcs7 signature verification.
https://github.com/tianocore/edk2/blob/master/CryptoPkg/L...

Reply to this comment

[–] Press X to doubt

Posted Jul 17, 2025 15:32 UTC (Thu) by **pjones** (subscriber, #31722) [Link] (1 responses)

You're right, but also the reason for this is that it's neither helpful nor at all reasonable for anything to check the validation windows on certs in Secure Boot, for two reasons.

One is that it doesn't help guarantee any security - the general threat being protected against is compromised administrative accounts escalating to have any of several more advanced forms of persistence. In that kind of attack, the attacker has total control of the clock. Also, RTCs drift quite badly or even reset sometimes without power, and often (especially on servers) need to be corrected during the first OS installation or boot.

But also it's not just OSes - if the validation window is honored, then on 27-Jun-2026 (or whenever an RTC drifts sufficiently during shipping) option ROMs on PCIe video cards, NICs, and HBAs all stop POSTing.

It'd be a total disaster.

Reply to this comment

[–] Press X to doubt

Posted Jul 19, 2025 13:51 UTC (Sat) by **patrakov** (subscriber, #97174) [Link]

I am not sure about the date. As far as I understand, option ROMs are signed using the same key as the shim.

Reply to this comment

[–] Press X to doubt

Posted Jul 16, 2025 21:33 UTC (Wed) by **stevem** (subscriber, #1512) [Link]

Yeah, most likely.

But we all know that some firmware implementations are likely to include flying monkeys too. Or worse...

Reply to this comment

[–] What about custom keys?

Posted Jul 16, 2025 18:53 UTC (Wed) by **das_j** (subscriber, #143082) [Link] (2 responses)

> Linux users who have Secure Boot enabled on their systems knowingly or unknowingly rely on a key from Microsoft that is set to expire in September

That sounds like a really general statement which is surprising to me. I'm not sure how this is handled on other distributions, but for NixOS, for example, people mostly use sbctl to upload the keys to the UEFI. Is that not more common?

Reply to this comment

[–] What about custom keys?

Posted Jul 16, 2025 19:11 UTC (Wed) by **daroc** (editor, #160859) [Link]

I have locally generated SecureBoot keys that I use on my computers, but my anecdotal impression is that it's not common. For the first decade or so that I used Linux, I relied on whatever it was my distribution was doing by default.

For the case of installers, though, it's sort of a chicken and egg problem, and an additional barrier to new users who may not be familiar with all the details.

Reply to this comment

[–] What about custom keys?

Posted Jul 16, 2025 19:20 UTC (Wed) by **heftig** (subscriber, #73632) [Link]

No, it's rare. The overwhelming number of Linux installs with Secure Boot enabled use the shim because it does not require the user to put the system into Secure Boot's setup mode.

Reply to this comment

[–] System time

Posted Jul 16, 2025 19:14 UTC (Wed) by **tux3** (subscriber, #101245) [Link] (9 responses)

I have (not so) fond memories of old BIOS systems where removing the battery would reset both the settings and clock.

I'm not sure if this is still a thing. Does Secure Boot prevent an attacker from turning the clock back; is there maybe some internal clock that cannot be tampered with? Can one not boot a system the firmware accepts today, reset this firmware clock, and then merrily go on to boot payloads signed with the expired key?

Some embedded systems burn fuses to prevent firmware rollbacks, but that's based on version numbers of a fixed boot chain, instead of certificates that might sign arbitrary payloads. I can't see how this sort of hard anti-rollback would work for secure boot, but I'm not sure how much certificate expiration is worth if you don't have a trusted clock.

Reply to this comment

[–] System time

Posted Jul 17, 2025 0:02 UTC (Thu) by **mathstuf** (subscriber, #69389) [Link] (1 responses)

> I'm not sure how much certificate expiration is worth if you don't have a trusted clock.

The firmware could have a "build date" baked into the signed part of its firmware and do `max(curtime, fwtime)` upon initialization instead of blindly trusting the hardware clock.

Reply to this comment

[–] System time

Posted Jul 17, 2025 8:02 UTC (Thu) by **taladar** (subscriber, #68407) [Link]

That would only work if you issue two firmware updates, one for legitimate users before the old key expires to avoid a window where neither key works and one issued after that uses the trick you describe to prevent setting the clock back to the time when the old key was still valid.

Reply to this comment

[–] System time

Posted Jul 17, 2025 8:32 UTC (Thu) by **aaribaud** (subscriber, #87304) [Link] (6 responses)

> I have (not so) fond memories of old BIOS systems where removing the battery would reset
> both the settings and clock.
>
> I'm not sure if this is still a thing.

It is not.

For decades now, the BIOS settings are not stored in volatile memory but in the BIOS Flash chip, where they survive as long as they are not reset through a BIOS menu command -- and if that BIOS has a master password set and you don't know it, then you can't enter the BIOS menu, and the only way out is either physically reprogramming the flash chip with a "blank" BIOS, or buying a new motherboard (ask me how I know. Better yet, don't ask).

Plus, about a decade ago, at least one BIOS provider, AMI, provided support for protecting the master password with the TPM, which--if applied by the BIOS vendor--basically removes the "physically reprogramming the flash chip" option.

Reply to this comment

[–] System time

Posted Jul 18, 2025 0:35 UTC (Fri) by **marcH** (subscriber, #57642) [Link] (5 responses)

I recently lost all video outputs after trying some "unusual" BIOS settings (lesson learned...)

I realize it's different from a master password lockdown, but I found in the manual a way to reset settings with a jumper (no jumper actually needed, any small piece of metal worked). It worked!

Reply to this comment

[–] System time

Posted Jul 18, 2025 21:03 UTC (Fri) by **Lwnkhz** (guest, #178382) [Link] (4 responses)

Not possible on TPM protected systems

Reply to this comment

[–] System time

Posted Jul 18, 2025 21:18 UTC (Fri) by **mjg59** (subscriber, #23239) [Link] (3 responses)

The TPM has nothing whatsoever to do with firmware settings or whether they can be reset.

Reply to this comment

[–] System time

Posted Jul 18, 2025 22:06 UTC (Fri) by **aaribaud** (subscriber, #87304) [Link] (2 responses)

> The TPM has nothing whatsoever to do with firmware settings or whether they can be reset.

This rather depends on the context, notably whether and how the BIOS is TPM-protected.

Reply to this comment

[–] System time

Posted Jul 18, 2025 22:54 UTC (Fri) by **mjg59** (subscriber, #23239) [Link] (1 responses)

No, it doesn't. There's no such thing as a TPM-protected BIOS and even if there were that would have nothing to do with the firmware variables which are inherently mutable.

Reply to this comment

[–] System time

Posted Jul 19, 2025 4:48 UTC (Sat) by **aaribaud** (subscriber, #87304) [Link]

> No, it doesn't. There's no such thing as a TPM-protected BIOS and even if there were that would have nothing to do with the firmware variables which are inherently mutable.

There appears to be such thing as a TPM-protected BIOS:
https://trustedcomputinggroup.org/american-megatrends-sup...

And if there is such a BIOS, then when the master password is set, many settings become immutable.

Reply to this comment

[–] Multiple Microsoft secure boot keys expiring in 2026

Posted Jul 16, 2025 19:48 UTC (Wed) by **ewen** (subscriber, #4772) [Link] (1 responses)

FTR multiple Microsoft secure boot related CA keys are expiring in 2026:

https://support.microsoft.com/en-us/topic/windows-secure-...

Including the key used for updating the other keys (the Key Exchange Key aka KEK).

And Microsoft only really started on this replacement rollout in 2023, which means even systems bought as recently as 2023 or 2024 probably have to go through the key update / replacement process.

Also note that if you have a dual boot system with Microsoft Windows using a TPM unlocked Bitlocker (ie automagically) then that is tied to the secure boot measurements, and thus will change when the keys used also change. Microsoft Windows supposedly can handle the new key / measurements expected… but only if it updates the secure boot keys itself, and thus updates the expected measurements to unlock the Bitlocker encryption.

I suspect this will be a “fun” transition for any systems not on the happy path of very recent hardware, active BIOS updates / vendor, single OS boot, update managed by that OS vendor. And anyone with TPM / secure boot secured encrypted disks would be wise to have good backups (of the recovery keys and disk contents).

Ewen

Reply to this comment

[–] Multiple Microsoft secure boot keys expiring in 2026

Posted Jul 16, 2025 19:59 UTC (Wed) by **ewen** (subscriber, #4772) [Link]

There’s some back story to these Microsoft key rollover plans in a 2023 UEFI plugfest presentation:

https://uefi.org/sites/default/files/resources/Evolving%2...

https://m.youtube.com/watch?v=o7kg1gX-KNc

And fortunately from the Microsoft collection of vendor (Platform Key signed) key updates it does at least look like Microsoft have managed to get pretty much all major platform vendors involved in the rollover process by now.

But it’s still going to be “fun” that the Linux secure boot shim signing is going to be one of the first things pushed through only being signed by the new Microsoft secure boot CA trust chain :-/

Ewen

Reply to this comment

[–] Why?

Posted Jul 16, 2025 20:15 UTC (Wed) by **mb** (subscriber, #50428) [Link] (7 responses)

What problem does boot certificate expiration solve? Why do these keys expire at all?

Reply to this comment

[–] Why?

Posted Jul 16, 2025 22:56 UTC (Wed) by **NYKevin** (subscriber, #129325) [Link] (5 responses)

The purpose of any certificate expiring is nearly always the same: To protect against undetected compromise of the secret key.

(If you don't consider key compromise a Bad Thing, then the certificate is objectively worthless and provides no benefit, so you should not be checking it in the first place.)

Reply to this comment

[–] Why?

Posted Jul 17, 2025 4:57 UTC (Thu) by **kraxel** (subscriber, #49444) [Link]

In firmware context the problem is that there is no time source available (other than the cmos real time clock which can be changed easily). So, yes, expiring certificates doesn't make much sense in this specific case. You can't create x509 certificates without expiry date though, so the firmware goes turn off time checks instead.

Reply to this comment

[–] Why?

Posted Jul 17, 2025 6:17 UTC (Thu) by **mb** (subscriber, #50428) [Link] (1 responses)

Well, but does the expiration really solve anything in the context of booting the machine?

The key is not compromised but expired -> My machine is broken. -> Obviously bad.

The key is compromised and expired -> My machine is broken -> Why is that better than a booting the machine with a compromised key? It could show a warning about the expired key after booting instead. That would be useful for the user. But a broken machine is pretty much worst case useless and it basically protects me from nothing. A compromised key is *far* from an actually compromised machine.

Bricking the device at a specific expiry date is just a ticking time bomb.

(The problem with the clock has already been addressed in a parallel post. Thanks!)

Reply to this comment

[–] Why?

Posted Jul 19, 2025 7:50 UTC (Sat) by **epa** (subscriber, #39769) [Link]

As I understand it from the article, device doesn’t become bricked when the certificate expires, but you can no longer change the OS that gets booted. (Though I don’t understand why it’s so)

Reply to this comment

[–] Why?

Posted Jul 17, 2025 8:45 UTC (Thu) by **epa** (subscriber, #39769) [Link]

If the certificate expires after a short time then it does provide some protection against the secret key being leaked. But after ten years? If the key were compromised then you could have years left to run. There's no way that could be adequate protection, if the key is guarding anything important.

Personal computing devices have a short lifespan and many could be nearly obsolete at ten years old. It would make more sense for the life of the certificate to match the life of the device. If you are still using the device after a couple of decades, it's clearly a museum piece by that point, and no purpose is served by having a certificate expire so you can no longer change the OS.

Reply to this comment

[–] Why?

Posted Jul 17, 2025 15:00 UTC (Thu) by **jem** (subscriber, #24231) [Link]

I would say a more important reason for expiration is to protect against insecure algorithms. When issuing a certificate, you never know what the world looks like in five years, but if you don't limit the validity you can be sure the certificate is still in use after 10 years. Without an expiration data on certificates, we'd still have valid 512-bit certificates.

Reply to this comment

[–] Why?

Posted Jul 17, 2025 18:03 UTC (Thu) by **wtarreau** (subscriber, #51152) [Link]

> What problem does boot certificate expiration solve? Why do these keys expire at all?

How do you want to force end-users to replace their hardware nowadays without this ? Hardware vendors are starving, 10+ year-old PCs are still very common in the field everywhere users just need something to access the net to check their bank account and do a few simple things (and who don't need windows 11 which already tried to force them to upgrade the PC until they realized they were still on windows 7 and have no care for programmed obsolescence).

Reply to this comment

[–] Installers

Posted Jul 16, 2025 22:03 UTC (Wed) by **comex** (subscriber, #71521) [Link] (4 responses)

The article focuses mostly on existing Linux installs. But what about new ones? What happens if you don’t have a working OS, your firmware only has the old keys, and you try to boot from new Linux installation media – or for that matter new Windows installation media? It sounds like the firmware should refuse to run the installer.

Does UEFI have some magic system where the firmware can update its keys from the install media before actually running the installer? Or are you just out of luck unless you turn off Secure Boot?

Reply to this comment

[–] Installers

Posted Jul 17, 2025 1:22 UTC (Thu) by **jreiser** (subscriber, #11027) [Link]

Many ASUS motherboards within the last several years can update the board firmware directly from a file on USB flash memory (in exFAT format) without booting or using any OS at all.

Reply to this comment

[–] Installers

Posted Jul 17, 2025 11:00 UTC (Thu) by **jengelh** (subscriber, #33263) [Link] (1 responses)

Shouldn't firmwares allow you to manually input keys? Then you could add a current one, and then boot off a current OS.

Reply to this comment

[–] Installers

Posted Jul 18, 2025 6:50 UTC (Fri) by **kraxel** (subscriber, #49444) [Link]

Some firmwares offer that functionality somewhere in the firmware setup menus. Others do not. And even for those who do there is no standard way to do so, so it is pretty hard to support that workflow.

BTW: Microsoft has released signed DB updates which add the 2023 code signing keys meanwhile.
https://github.com/microsoft/secureboot_objects/tree/main...
They are signed with the old (2011) KEK key, so there is no need to enroll the new (2023) KEK key to apply those updates.

LWN.net
News from the source

Content
Weekly Edition
Archives
Search
Kernel
Security
Events calendar
Unread comments

LWN FAQ
Write for us

Edition
Return to the Front page

These can be applied by standard EFI variable updates, using the efi-updatevar utility for example. I expect fwupd will support that soon too.

Reply to this comment

[–] Installers

Posted Jul 18, 2025 13:50 UTC (Fri) by **pjones** (subscriber, #31722) [Link]

Our (Fedora, etc) plan right now is to make special remediation boot media, so you can boot it with an older bootloader and it'll run fwupd to update the enrolled certificates. Obviously even that can only be so successful.

We're also going to try some experiments with making that a secondary boot entry on the primary media, with the hopes that at least some firmwares will correctly attempt it after the newer boot target, but it's yet to be seen how effective that will be. We'll also do our best to make sure EDK2 supports that correctly, and try to get Red Hat's hardware partners to make sure they have that support.

Reply to this comment

[–] Vendor-specificity of KEK

Posted Jul 17, 2025 13:19 UTC (Thu) by **linuxtardis** (subscriber, #178362) [Link] (2 responses)

Hello, I would like to discuss the following part of the article:

> Another avenue would be a "key exchange key" (KEK) update, which is a vendor-specific key that is signed by the Microsoft key; it can be used by fwupd to update the database with the new key.

I believe the situation is somewhat the opposite (see below for my interpretation of PK, KEK, db, dbx):

- The KEK certificate that needs to be renewed/updated is generic and is owned by Microsoft.
- The update must be signed by a PK (platform key), and the PK is vendor-specific.
- Therefore the KEK *update package* is vendor-specific due to being signed by the vendor's key, but the KEK itself is not.

This, in effect, could mean that Microsoft and LVFS can still roll out an update to "db" (to add the "Microsoft UEFI CA 2023") using the old Microsoft KEK. This would make old systems trust Linux shims signed by the new Microsoft "db" certificate. Only the KEK update needs the cooperation of hardware vendors; the KEK needs to be updated for Microsoft to be able to update "db" and "dbx" (e.g. to block vulnerable bootloaders) after 2026.

I am not 100% sure my claim is correct, but it seems to agree with Microsoft's documentation at [1]. I would be happy if someone else can confirm or refute my hypothesis.

My interpretation of the roles of the various UEFI keys is the following (refer also to the image https://learn.microsoft.com/en-us/windows-hardware/manufa...):

- "PK" ("platform key") is the motherboard manufacturer's key/certificate. Updates to KEK must be signed using this (vendor-specific) key.
- "KEK" ("key exchange key") list includes a Microsoft KEK certificate (common to all vendors). Through this certificate, Microsoft can issue and sign updates to "db" and "dbx" (the updates of "db" and "dbx" must be signed by a trusted PK or KEK).
- "db" is checked when executing UEFI bootloaders. When Secure Boot is enabled, each executed binary must be signed by at least one certificate in this list. This list typically includes three Microsoft keys/certificates: the key for the Windows bootloader, the key for third-party bootloaders (e.g. shim) and newly a key for Option ROMs.
- "dbx" is used to revoke signatures of vulnerable bootloaders that the firmware would otherwise trust via "db".

[1]: https://learn.microsoft.com/en-us/windows-hardware/manufa...

Reply to this comment

[–] Vendor-specificity of KEK

Posted Jul 18, 2025 1:08 UTC (Fri) by **marcH** (subscriber, #57642) [Link]

Trying to understand KEK, PK and their friends is futile: it's security through security. Even the names are cryptic!

Could not resist sorry.

Reply to this comment

[–] Vendor-specificity of KEK

Posted Jul 19, 2025 10:08 UTC (Sat) by **linuxtardis** (subscriber, #178362) [Link]

I found a brief overview of the different keys in a Matthew Garrett's presentation from 2012: https://youtu.be/V2aq5M3Q76U?si=hnMbw_H8LzaiMi1I&t=1832

Reply to this comment

[–] September expiration date

Posted Jul 17, 2025 17:30 UTC (Thu) by **linuxtardis** (subscriber, #178362) [Link]

Hello, I am a bit confused about the September 2025 expiration date. I've found conflicting information:

* The LVFS wiki page at https://fwupd.github.io/libfwupdplugin/uefi-db.html claims that the expiration date of some certificate is 11th September 2025.

* The shim currently shipped in Ubuntu 22.04 is signed by an intermediate "Microsoft Windows UEFI Driver Publisher" certificate that already expired (!) on 16th October 2024. This intermediate is signed using the root "Microsoft Corporation UEFI CA 2011" certificate that expires on 27th June 2026. This root certificate is typically trusted by the firmware.

* Shims in some other distributions (Fedora, AlmaLinux) also seem to be signed by the expired intermediate certificate. OTOH, the CentOS Stream 10 shim is signed by an updated intermediate that expires on 15th May 2026.

* My ThinkPad L13 Gen2 can boot into Ubuntu even though their intermediate is expired.

* In https://github.com/rhboot/shim-review/issues/454#issuecom... I've found the information that Microsoft will *start* signing binaries with their new CA certificate in cca. October 2025.

I have a (speculative and unproven) theory on how to fit this together: starting with September or October 2025 Linux distros may not be able to obtain shims signed by the old 2011 MS certificate. This may prevent new installation media from booting on systems without the new "Microsoft UEFI CA 2023" certificate in their UEFI "db". Old media may remain working until June 27, 2026, when the old "Microsoft Corporation UEFI CA 2011" expires. This assumes that UEFI implementations check the certificate expiration date, which seems not to always be the case.

Could anyone please confirm or refute this?

Thank you, Jakub

Reply to this comment

[–] Old hardware?

Posted Jul 18, 2025 4:55 UTC (Fri) by **pabs** (subscriber, #43278) [Link] (3 responses)

What happens with old hardware where the vendor isn't in LVFS and no longer provides any manual firmware updates?

Will Microsoft be providing per-vendor KEK updates to LVFS for those devices?

Or will such folks just have to boot in BIOS mode, or with Secure Boot disabled, if they can do that and can figure out how to do that?

Distros generally don't enable dual BIOS + UEFI booting on installed systems, so there are going to be a number of confused folks at some point.

Reply to this comment

[–] Old hardware?

Posted Jul 18, 2025 13:57 UTC (Fri) by **pjones** (subscriber, #31722) [Link] (1 responses)

> What happens with old hardware where the vendor isn't in LVFS and no longer provides any manual firmware updates?

Microsoft is sharing their partners' certificate updates with us for both the vendor-signed KEK updates and the MS KEK-signed db updates. So for the vendors that are competent enough to actually be able to sign KEK updates, and therefore don't need firmware updates for that, those will still be in LVFS even if the vendor doesn't provide firmware updates there.

Reply to this comment

[–] Old hardware?

Posted Jul 18, 2025 23:39 UTC (Fri) by **pabs** (subscriber, #43278) [Link]

So hardware with vendors who went out of business before now, or with incompetent vendors, will need to disable SecureBoot permanently. And hardware that wasn't updated before the expiry will need to temporarily disable SecureBoot to do the update, and then re-enable it. Is that correct?

Reply to this comment

[–] Old hardware?

Posted Jul 19, 2025 8:30 UTC (Sat) by **linuxtardis** (subscriber, #178362) [Link]

You may still be able to manually load the new Microsoft KEK into UEFI through the Setup UI even without it being signed by the vendor's platform key. In [1] they have the KEK certificate ("Microsoft Corporation KEK 2K CA 2023") and the Owner GUID that has to be entered into UEFI. I remember doing something similar to make my ThinkPad trust some helper utilities signed by me (I had to add my own signing key to the laptop's UEFI "db").

You may also be able to install the new "db" certificate that Microsoft will likely use to sign shim in the future. In [1] it is the "Microsoft UEFI CA 2023" certificate. This is IMO the more important certificate in the short-term, as adding it will allow you to run newly signed bootloaders. Luckily, this is also the certificate that LVFS can potentially update even without the help from vendors. This is because the update package for this certificate is already published and is signed by the old, commonly trusted Microsoft KEK (see [2]).

[1]: https://learn.microsoft.com/en-us/windows-hardware/manufa...
[2]: https://github.com/microsoft/secureboot_objects/blob/main...

Reply to this comment