Rebooting Your Servers is Making You Insecure and Noncompliant

(and it's a matter of time until you discover this the hard way)

KernelCare

KernelCare

Rebooting Your Servers is Making You Insecure and Noncompliant

(and it's a matter of time until you discover this the hard way)

Within IT organisations, there are processes so routine, and practices so well-established, that they are effectively invisible. It doesn't matter if such processes and practices are flawed, or if there exists a better way: Habits die hard, and if something has generally worked for a few years, people stop looking for alternatives.

This perfectly describes current approaches to kernel patching. **Kernel patching** is a background activity, carried out by SysAdmins without too much fuss. It is generally ignored altogether by those responsible for security and compliance. However:

This is a dangerous oversight. The standard approach to kernel patching exposes servers to malicious intent by threat actors on multiple attack vectors, putting IT organisations at risk of major security issues. Anyone tasked with keeping their organisation safe should be seeking a better way.

Linux was first released in 1991. Today, it is the go-to option for enterprise hosting, with more than half of all known websites running on Linux. Linux is a brilliantly built and very stable OS, but it is a large body of advanced software that has to run on a complicated kernel. A very complicated kernel. The master branch of the Linux kernel git repository contains more than 20,000,000 lines of human-written code.

As any SysAdmin can attest, with this much complexity comes bugs. And with bugs come vulnerabilities. More than 450 Linux vulnerabilities were found in 2017, up from over 200 a year before.[1] This increase continued through 2018, with some of the vulnerabilities being very threatening indeed.[2]

To counter such vulnerabilities, Linux vendors are constantly providing partial patch updates for the kernel. These patches correct the bugs. Bugs keep coming, and vendors keep generating patches in response. Some bugs need multiple patches to be remedied, while others are so minor that many of them can be covered by a single patch. Some patches modify a single line of code, whereas others add missing checks, or change data structure or functions. The whole thing is very unpredictable, a whack-a-mole of Linux security.

---

[1] https://www.kernelcare.com/wp-content/uploads/2018/08/six-benefits-of-kernelcare.pdf
[2] https://resources.whitesourcesoftware.com/blog-whitesource/top-5-linux-kernel-vulnerabilities-in-2018

This **perpetual patching** has to be managed by all operations running on Linux. Right now, 99% of organisations patch the same way: By **rebooting** their servers. When the releases have piled up to the point where they can't be ignored anymore, the server is shut down and restarted. Because there was a 17-year gap between the creation of Linux and the creation of any alternatives, this flawed practice has become set in stone for many.

But SysAdmins are (understandably) very reluctant to do this rebooting until they absolutely have to. It can take a while, and it usually has to be done in the middle of the night, to minimize the impact on peaktime services. While the servers are being rebooted, the websites they host will go down, and display an error message. And even after rebooting, it can take a while for performance to stabilize. Sometimes, environments never properly come back up after a reboot.

Because rebooting is a headache, people put if off for as long as they can. Which means patches aren't applied as early as possible. This gap between patch issue and patch application means risk, and also malpractice.

Here's why:

- In an open source environment, as soon as a kernel vulnerability becomes public knowledge, it becomes public knowledge. Meaning that Linux operators know about it, but so do bad actors – hackers and other digital attackers.[3]
- Industry best practice is a maximum of around 30 days for kernel patching. Well-functioning organisations aim for half that as standard. But organisations' aversion to rebooting can see them delay for months. They bundle fixes together, meaning that the earliest ones can be available for a long time before they are actually applied. They might even ignore vulnerabilities with a high CVSS base score (7 or above), in order to buy themselves more time.

This delay in patching is what causes danger. Every day that a vulnerability is discovered but not patched is another day when you are at risk.

As if that wasn't bad enough, these delays could also be putting organisations at risk of **noncompliance.** Most companies' errors and omissions (E&O) insurance policies, and the clauses in their SLA contracts, define adherence to best practices (approximately 30 days for kernel patching). Usually, reflecting industry best-practices, this period is no more than one month. The pressures of rebooting mean that organisations frequently take (much) longer than a month, and so therefore in breach of their insurance policies.

[3] This is doubly true when it is common practice in the cybersecurity community to combine the announcement of a vulnerability with the release of a detailed case study. These studies are very useful to those working in security; but they also provide insight to hackers.

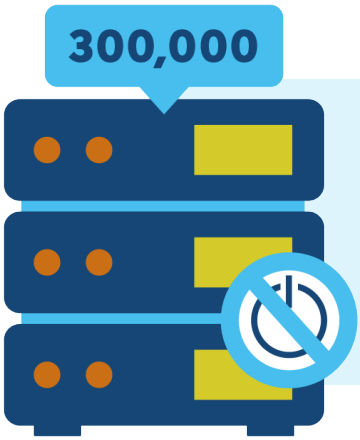
Reboot-centred kernel patching is a ticking time bomb. The solution? Live kernel patching.

At KernelCare, our kernel team monitors security mailing lists. When a vulnerability affecting supported kernels is announced, we prepare a patch as soon as technically possible. We compile each patch for that kernel and deploy it to our distribution servers. A KernelCare agent process running on your server synchronizes and checks with our distribution servers every four hours. When a new patch is available for the active kernel, the agent downloads it and applies it to the running kernel, right away.[4]

With this system, kernel updates are applied as quickly as possible, **protecting you from bad actors, and keeping you compliant.** This happens without a moment of kernel downtime or any disruption of its operation. There is no need to reboot, no service interruptions or packet drops, and no need to kill any processes or user sessions.

There will always be some lag between a vulnerability report and the creation of a fix, but live kernel patching turns this lag into days rather than months.

People might tell you that this isn't possible; that live kernel patching can't be done, or at least can't be done over the long-term. This is primarily because there have been previous attempts at live patching models that were built imperfectly. With these flawed services, live patching would work for a while, but then there would have to be a reboot within a year.



But at KernelCare, we have 300,000 servers that haven't needed to reboot in four years. Live kernel patching might have been a mirage once. But today, it is a reality. And it should be a feature of every responsible organisation's security posture.

There is a common objection to the idea that rebootless patching is a necessity; an objection that security folk often hear when they raise the issue of rebooting with their SysAdmins: "Oh, most kernel vulnerabilities are no big deal. And anyway, we've never been hacked, so our processes are clearly working." So:

---

[4] Kernelcare can write and test patches to the highest degree of skill.And, we can uninstall (rollback) patches, for those times when a SysAdmin suspects a patch has altered the functioning of a system, however subtly. This makes it easy to check what effect a patch is having, without having to reinstall the original kernel.

Yes, it is true that kernel vulnerabilities with the potential to trigger serious hacks are rare.[5] You might only a couple every year. But here's the thing: when they come, they are ruinous.

Vulnerabilities in the kernel are the worst kind of vulnerability an IT system can suffer. The kernel is the most important part of any Linux system. It provides vital low-level functions to the entire system. Any security issues detected within it jeopardize the whole server. Once a hacker has exploited the kernel, they can get anywhere, and access everything, including customer data, for months or years. It's like a thief getting into the safe-room. Once your kernel has been infiltrated, you're exposed to the worst possible digital damage.

So even if serious kernel vulnerabilities are few and far between, and even if your current process – a rebooting session every couple of months costing, 20 minutes – is just about manageable, it isn't worth the risk. If you are rebooting your servers for patch updates, then you are without doubt patching slower than you could be. Meaning you are **exposed**, in a way that is avoidable.

Rebootless kernel patching is like insurance: if you're lucky, you'll never find yourself in a bad situation. But if you do, you'll be damn glad that you have it.

And like insurance, rebootless kernel patching is not a nice-to-have; it is a absolute necessity for anyone who wants to stay safe.

Those responsible for the logistics of kernel care patching usually don't have security as their first priority. And like everywhere in life, they have a routine, and a habit. Even if it isn't perfect, it's worked, for the most part, and there have been no major disasters. (Yet.) So they stick with it. But:

If your job is to keep your organization secure and compliant, then you should be ensuring that you are never having to reboot your servers.

---

[5] The Common Vulnerability Scoring System (CVSS) assigns most vulnerabilities a Low base score, and Critical vulnerabilities make up only 5-10%. https://courses.cs.washington.edu/courses/cse484/14au/reading/25-years-vulnerabilities.pdf

KernelCare is simple, takes five minutes to install, and then ticks over in the background without anyone having to even think about it. Most days, you won't really need it. But on that day when you do need it, you'll be thankful.

To try KernelCare free for 30 days, head to kernelcare.com