

## Alert Bulletin

**Bulletin N.:** 86

**Publication Date:** 07/08/2026

**Subject:** Alert 2026-86 Virtual Machine Escape with Host RCE in the Linux KVM Kernel ("Zapscape")

**Traffic Light Protocol (TLP):** White

## Affected product(s):

**Linux Kernel — KVM/x86 Subsystem** (Kernel-based Virtual Machine)

| Product                                    | Affected versions          | Fixed version         |
|--------------------------------------------|----------------------------|-----------------------|
| <b>Linux Kernel (mainline)</b>             | 5.9 through 7.1.5          | 7.1.6                 |
| <b>Linux Kernel (stable 6.18.x)</b>        | 6.18 through 6.18.41       | 6.18.42               |
| <b>Linux Kernel (stable 6.12.x)</b>        | 6.12 through 6.12.100      | 6.12.101              |
| <b>Linux Kernel (stable 6.6.x LTS)</b>     | 6.6 through 6.6.147        | 6.6.148               |
| <b>Red Hat / RHEL</b>                      | Versions with affected KVM | Update pending        |
| <b>Debian (bullseye, bookworm, trixie)</b> | All current branches       | Patch in distribution |
| <b>CloudLinux 7h, 8, 9, 10</b>             | Affected                   | Patch published       |

## Description

A *use-after-free* vulnerability has been disclosed in the **KVM/x86** subsystem of the Linux kernel. It has been identified as CVE-2026-64561 with a CVSSv3.1 score of 7.0 — Important; the researcher named it Zapscape,

The flaw allows an attacker with **root privileges inside a guest virtual machine (L1)**, on systems with **nested virtualization enabled**, to escape the VM isolation and execute arbitrary code with **root privileges on the physical host**.

Unlike previous similar vulnerabilities, Zapscape is exploitable on both **Intel (Ice Lake-SP and later)** and **AMD** processors, significantly expanding the attack surface.

The flaw resides in an **incorrect validation order** in the handling of guest page faults within KVM/x86's shadow MMU. When KVM attempts to reuse a shadow page, it does not properly check whether the *page root* had already been marked as invalid by the memory reclamation process (`make_mmu_pages_available()`).

If the reclaim invalidates the active root, KVM continues attempting to map memory within that invalid root. As a consequence, the child shadow pages created during the process inherit the invalid state from the parent, **violating KVM's internal invariant** which states that invalid pages must never be in the active MMU pages list. This generates a memory corruption condition exploitable for VM escape.

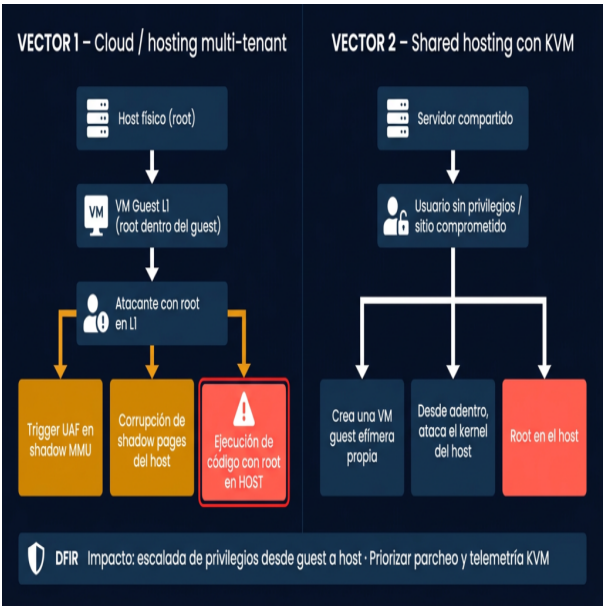


Figure 1 – Privilege Escalation in Virtualization: KVM Guest-to-Host Attack Vectors

Additionally, the **PoC (Proof of Concept) code has been published on GitHub** just days after the announcement, making the risk of active exploitation immediately elevated.

The patch moves the *stale root* check to run **after** `make_mmu_pages_available()`. If the reclaim invalidates the current root, KVM now restarts fault handling with `RET_PF_RETRY` instead of continuing to map under an invalid root.

## Solution

Update the kernel to the fixed versions:

| Branch              | Fixed version    |
|---------------------|------------------|
| Mainline            | 7.1.6 or 7.2-rc5 |
| Stable 6.18.x       | 6.18.42          |
| Stable 6.12.x (LTS) | 6.12.101         |
| Stable 6.6.x (LTS)  | 6.6.148          |

Kernel update in various distributions:

```
# Debian / Ubuntu
apt update && apt upgrade linux-image-$(uname -r)

# RHEL / CentOS / AlmaLinux / Rocky
```

```
dnf update kernel
```

```
# CloudLinux
```

```
yum update kernel
```

```
# Arch Linux
```

```
pacman -Syu linux
```

```
# Verify version after update
```

```
uname -r
```

## Mitigation

While planning the application of the kernel patch, you can follow these recommendations:

- **Disable nested virtualization** on all hosts where it is not strictly necessary. This is the most effective mitigation as it eliminates the main prerequisite of the attack:

```
# QEMU/KVM — disable nested virt per VM
```

```
-cpu ${CPU},vmx=off,svm=off
```

```
# Or at the kernel module level
```

```
echo "options kvm_intel nested=0" >> /etc/modprobe.d/kvm.conf
```

```
echo "options kvm_amd nested=0" >> /etc/modprobe.d/kvm.conf
```

```
modprobe -r kvm_intel && modprobe kvm_intel
```

- **Apply live patching** with tools such as **KSplice** or **kpatch** as a first-aid measure while scheduling a reboot to update the kernel.
- **Audit which VMs have access to nested virtualization** in the environment and restrict it exclusively to trusted guests.
- **In multi-tenant cloud environments:** review the configuration of `cpu_flags` exposed to guests and remove `vmx/svm` from untrusted third-party guests.
- **Monitor** unusual access attempts to the KVM subsystem from within VMs.

## Additional information:

- The Hacker News — Zapscape CVE-2026-64561: <https://thehackernews.com/2026/08/new-zapscape-kvm-flaw-could-let.html>
- NVD — CVE-2026-64561: <https://nvd.nist.gov/vuln/detail/CVE-2026-64561>
- Red Hat — CVE-2026-64561: <https://access.redhat.com/security/cve/cve-2026-64561>
- CloudLinux — Analysis and mitigation: <https://blog.cloudlinux.com/zapscape-cve-2026-64561>
- Debian Security Tracker: <https://security-tracker.debian.org/tracker/CVE-2026-64561>

- Hispasec — Technical analysis in Spanish: <https://unaaldia.hispasec.com>
- Upstream fix commit: 2abd5287f083 — linux/kernel/git/torvalds/linux

<https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=2abd5287f083>