Axellio PacketXpress®

Drop-In libpcap Compatibility for Every Tool Your SOC Already Owns

Security teams don't evaluate a packet broker on throughput alone. They evaluate what they stand to lose. Swap the capture platform underneath your tools, and you risk everything built on top: tuned rules, custom scripts, dashboards, years of analyst muscle memory. PacketXpress was built to remove that risk.

Every tool your SOC runs, including Suricata, Zeek, Wireshark, tcpdump, tshark, scapy, and any libpcap-linked tool, works with PacketXpress with no recompile, no API change, no reconfiguration. Tools still believe they're reading from a network interface. In reality, they're reading from a zero-copy, shared-memory ring that PacketXpress has already filtered, distributed, and flow-controlled.

Why This Matters

Replacing a packet broker is rarely blocked by technology - it's blocked by risk. Security teams spend years tuning Suricata and Snort rules, writing custom Zeek scripts, building dashboards, and training analysts around the capture source they already have. Ripping that pipeline out to adopt a new platform threatens to undo all of it.

That risk typically comes down to three things:

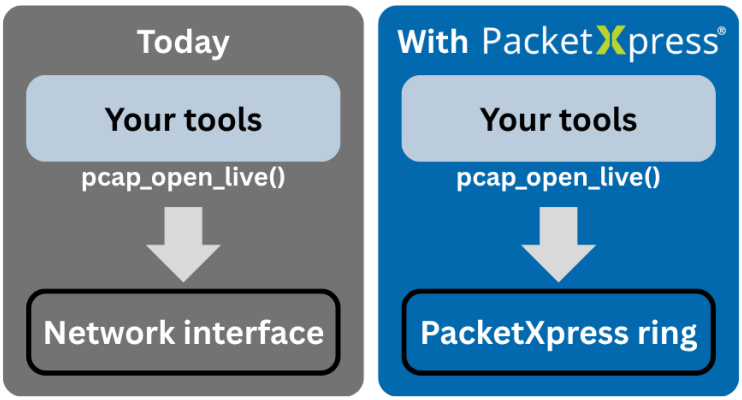
- **Tuned detection content:** Suricata and Snort rules, custom Zeek scripts, and mature dashboards built for a specific capture source.
- **Analyst muscle memory:** playbooks, alert workflows, and training built around today's tools.
- **Cutover risk:** months of parallel-running and requalification before a new platform is trusted.

For many SOCs, that risk alone is reason enough to keep running an aging, oversubscribed broker rather than replace it, even as traffic outgrows it.

PacketXpress With Drop-In Compatibility

PacketXpress answers this problem directly. Every rule, every signature, every script, every dashboard, and every playbook survives the migration unchanged, because adopting the platform does not disrupt the analytical investment already in place. Instead, it upgrades the data feeding it.

- **What changed**: the data behind the interface is now filtered, flow-controlled, retroactively queryable, and sourced from lossless capture. What the tool sees: nothing different at all.



Under the Hood: fx-libpcap

At the core of this compatibility is fx-libpcap, an open-source fork of standard libpcap with one addition: a capture backend for the Axellio shared-memory ring. Any tool linked against fx-libpcap sees the PacketXpress ring as just another capture device alongside the host's Ethernet interfaces. It still calls pcap_open_live(), pcap_create(), and pcap_activate() as always, still reads through pcap_next(), pcap_loop(), and pcap_dispatch(), with drop counters populated through pcap_stats(). BPF filters, non-blocking mode, and selectable-fd support all work exactly as they did on a network interface. The tool's model of the world never changes; everything else does.

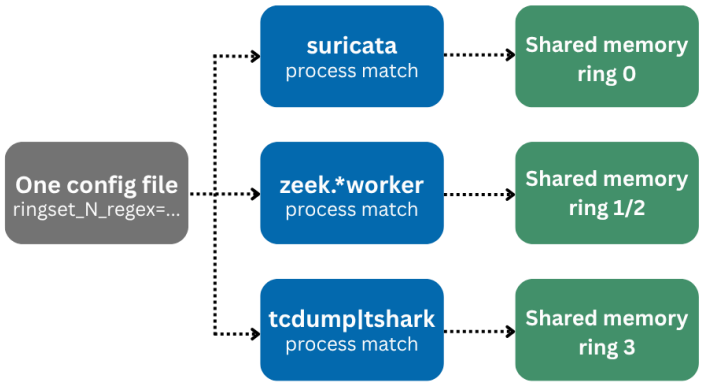
Zero-Copy, Cache-Line-Aware Delivery

Packets move from producer to consumer over a System V shared-memory segment: the producer, axRecv, maps the ring writable, every consuming tool maps it read-only, and both sides see the same physical pages. There is no packet copy on the delivery path, and no kernel system call per packet, and the ring layout keeps producer and consumer indices on separate cache lines to eliminate false sharing between processes. For tools processing many gigabits of traffic, this delivery model is measurably competitive with (and often faster than) kernel-delivered packets on high-end NICs. Nanosecond timestamps are preserved end-to-end, which matters for latency forensics, microburst analysis, and any investigation where timing is the entire point.

Regex-Based Shared Memory Ring Assignment

Consumers claim their shared memory rings automatically, by regex match on process name, with PID-based ownership as the fallback when no rule matches.

A single configuration file governs assignment for an entire SOC's tool fleet: starting Suricata automatically binds it to shared memory ring 0, with multiple instances sharing the same ring; Zeek workers bind to shared memory ring 1 or 2 depending on which regex matches their command line; and an analyst running ad-hoc tcpdump or tshark lands on shared memory ring 3 without touching anything. A custom tool with no rule configured simply receives an unused shared memory ring tied to its process ID, released automatically when it exits.



Keep Every Tool You've Already Tuned

PacketXpress is built on a simple premise: the platform under your tools shouldn't stop you from upgrading it. By preserving the standard libpcap interface, it lets a SOC modernize its capture and distribution layer without touching the rules, scripts, dashboards, or workflows on top — a platform upgrade, not a platform swap. Teams spend their time acting on better data, not rebuilding around it.

Axellio's mission is to control data overload in timeseries analysis systems that monitor for threats to our infrastructure through innovative storage and distribution solutions. www.Axellio.com