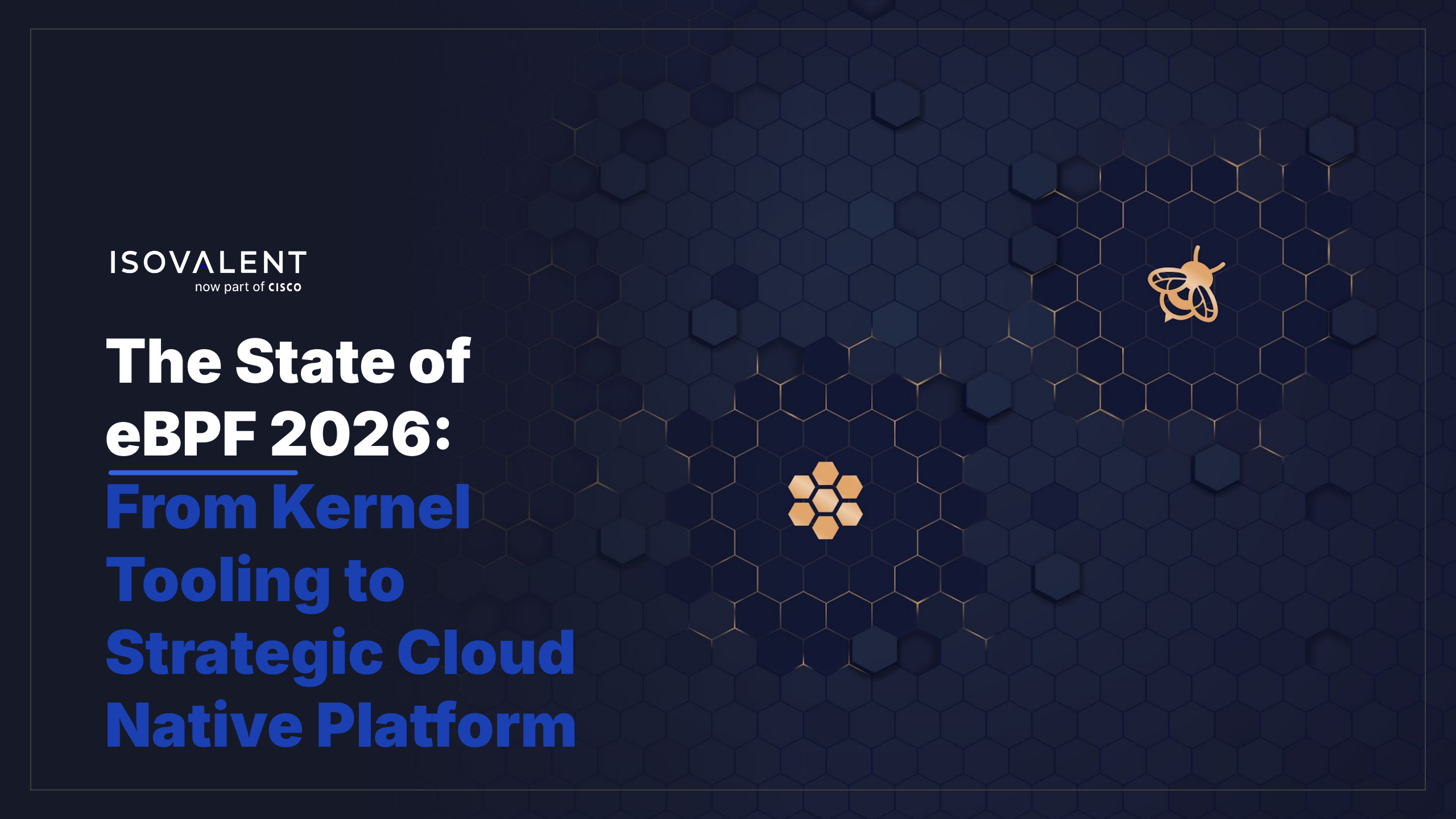
ISOVALENT
now part of cisco

The State of eBPF 2026: From Kernel Tooling to Strategic Cloud Native Platform



RFC 9669: BPF Instruction Set Architecture (ISA)

D. Thaler, Ed.

PROPOSED STANDARD

Abstract

eBPF (which is no longer an acronym for anything), also commonly referred to as BPF, is a technology with origins in the Linux kernel that can run untrusted programs in a privileged context such as an operating system kernel. This document specifies the BPF instruction set architecture (ISA).

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9669.

A Vision for eBPF Inside

ISOVALENT

eBPF in 2026

Happy 12th Birthday 🥳

From: Daniel Borkmann <dborkman@redhat.com>
To: davem@davemloft.net
Cc: ast@plumgrid.com, netdev@vger.kernel.org
Subject: [PATCH net-next v4 0/9] BPF updates
Date: Fri, 28 Mar 2014 18:58:17 +0100 [thread overview]
Message-ID: <1396029504-16776-1-git-send-email-dborkman@redhat.com> (raw)

We sat down and have heavily reworked the whole previous patchset
from v10 [1] to address all comments/concerns. This patchset therefore
replaces the internal BPF interpreter with the new layout as
discussed in [1], and migrates some exotic callers to properly use the
BPF API for a transparent upgrade. All other callers that already use
the BPF API in a way it should be used, need no further changes to run
the new internals. We also removed the sysctl knob entirely, and do not
expose any structure to userland, so that implementation details only
reside in kernel space. Since we are replacing the interpreter we had
to migrate seccomp in one patch along with the interpreter to not break
anything. When attaching a new filter, the flow can be described as
following: i) test if jit compiler is enabled and can compile the user
BPF, ii) if so, then go for it, iii) if not, then transparently migrate
the filter into the new representation, and run it in the interpreter.
Also, we have scratched the jit flag from the len attribute and made it
as initial patch in this series as Pablo has suggested in the last
feedback, thanks. For details, please refer to the patches themselves.

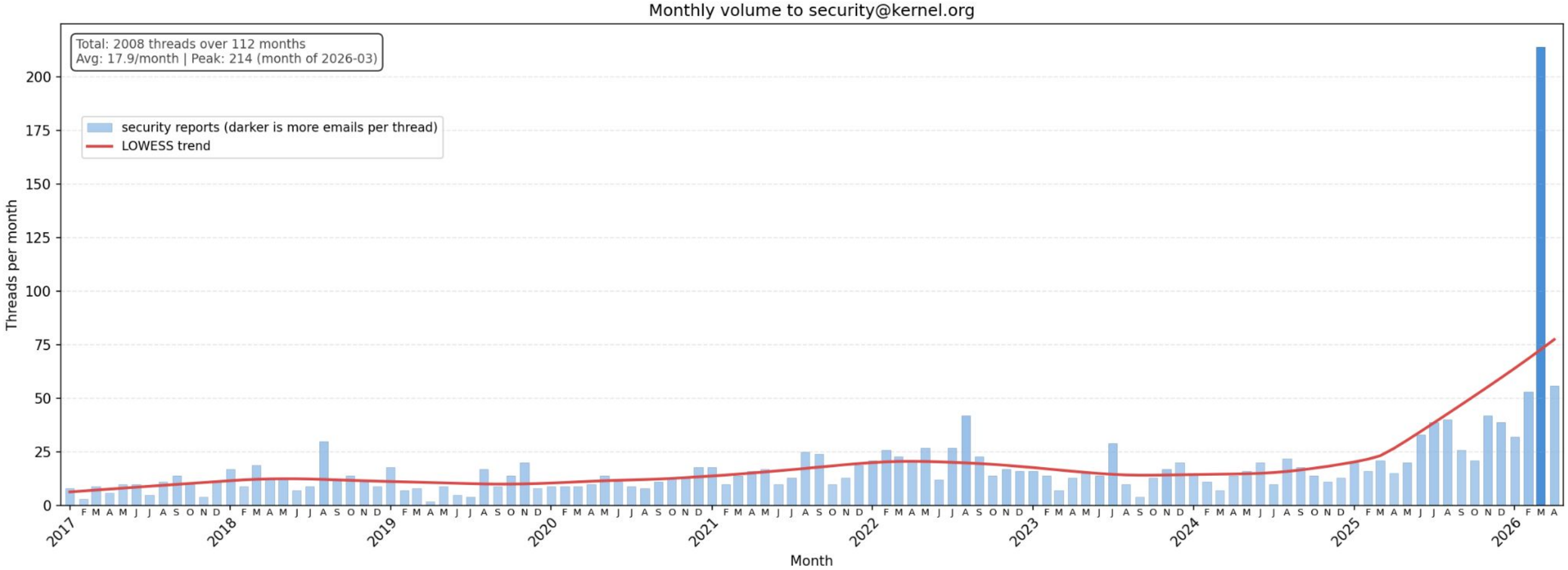
We did extensive testing of BPF and seccomp on the new interpreter
itself and also on the user ABIs and could not find any issues; new
performance numbers as posted in patch 8 are also still the same.

Please find more details in the patches themselves.

For all the previous history from v1 to v10, see [1]. We have decided
to drop the v11 as we have pedantically reworked the set, but of course,
included all previous feedback.

ISOVALENT

Observation: security@kernel.org



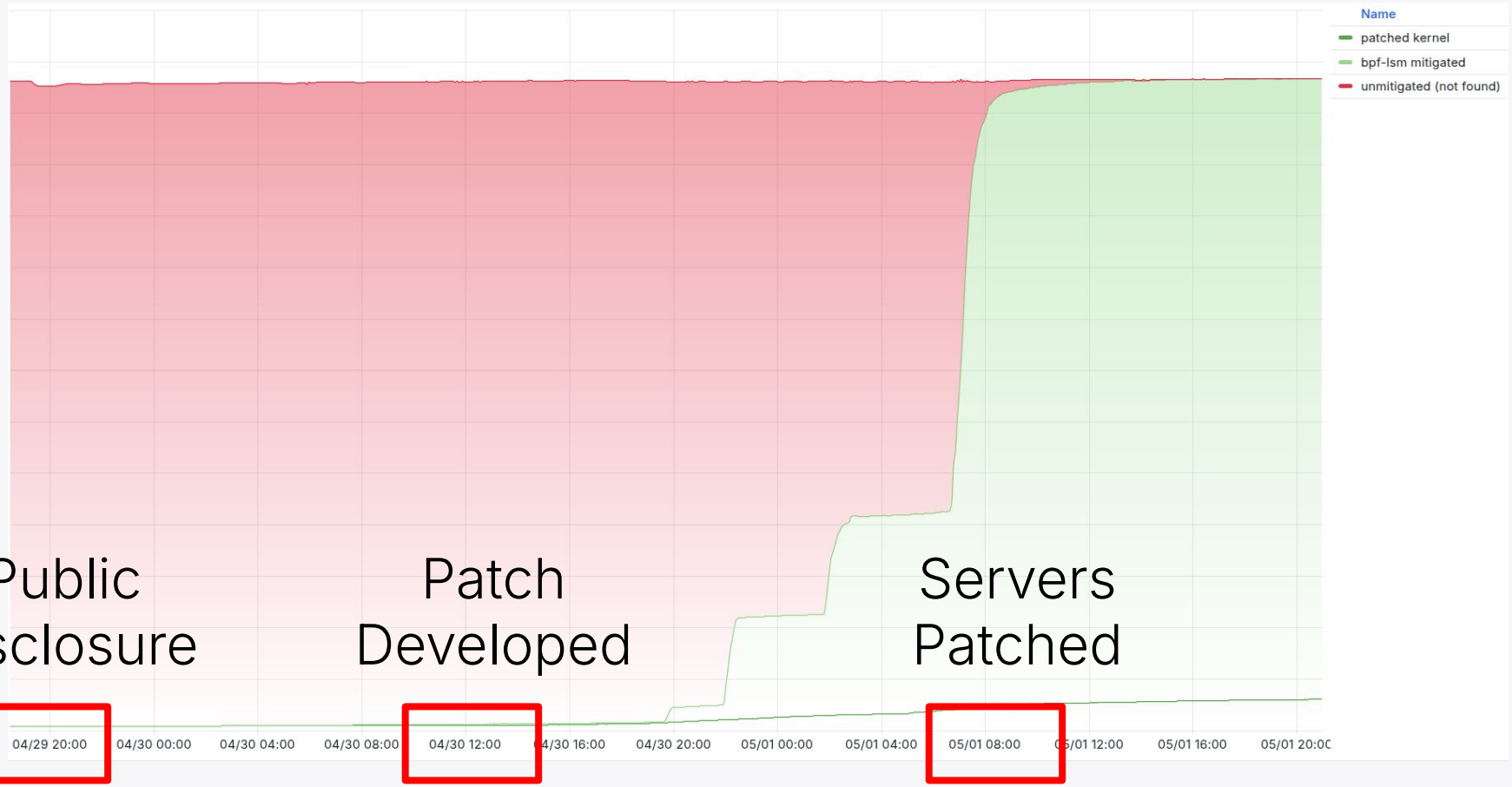
ISOVALENT

New poor-man “bandaid” approaches appear after CopyFail CVE:

```
Subject: [PATCH v3] killswitch: add per-function short-circuit mitigation primitive
Date: Sun, 17 May 2026 09:48:57 -0400   [thread overview]
Message-ID: <20260517134858.146569-1-sashal@kernel.org> (raw)
In-Reply-To: <20260508195749.1885522-1-sashal@kernel.org>

When a kernel (security) issue goes public, fleets stay exposed until a patched
kernel is built, distributed, and rebooted into.

For many such issues the simplest mitigation is to stop calling the buggy
function. Killswitch provides that. An admin writes:

    echo "engage af_alg_sendmsg -1" \
        > /sys/kernel/security/killswitch/control
```

eBPF + AI: Runtime Security

CLOUDFLARE®



ISOVALENT

Idea #1

Mitigation need to be created at the speed of agents.

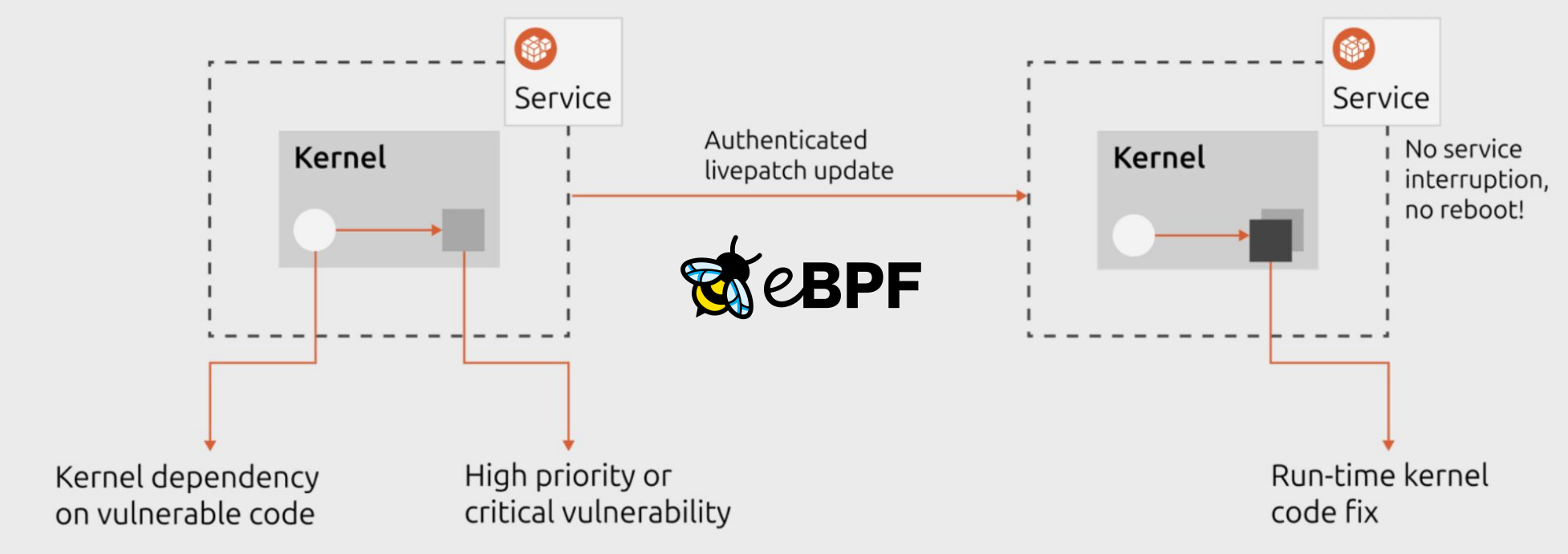
ISOVALENT

Idea #2

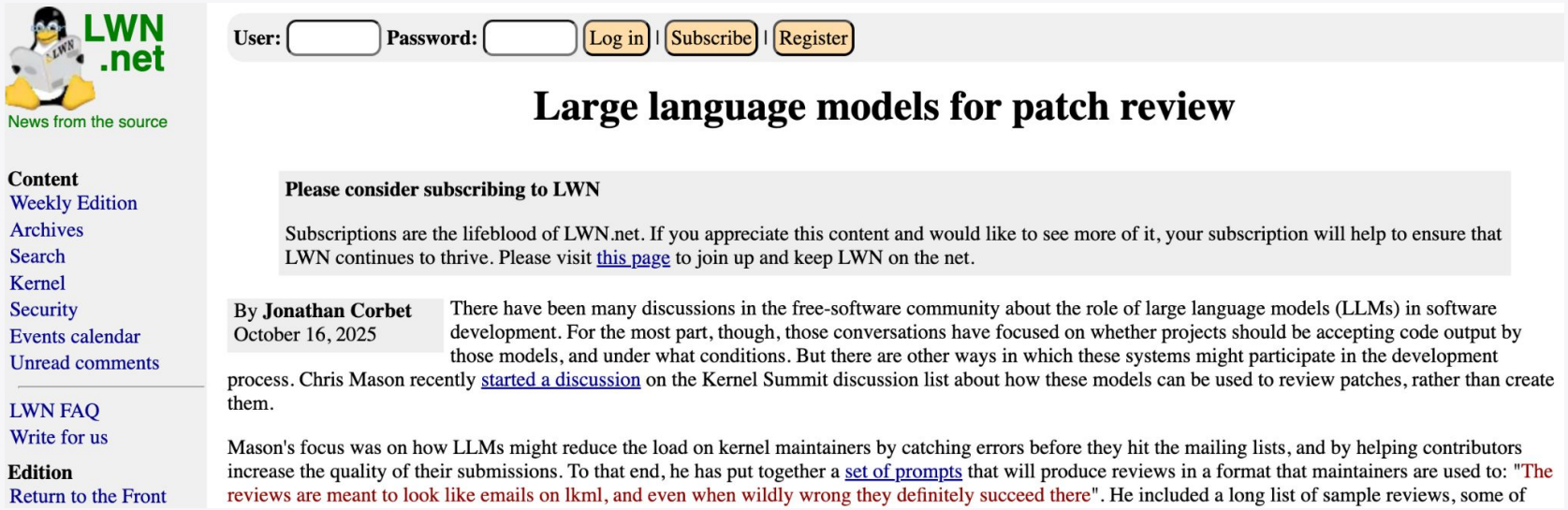
Consider **BPF Security** part of the **trusted** compute platform.

eBPF + AI: Runtime Security

- Cut off the maximum that is not needed from the kernel/workload
- Live mitigate vulnerabilities for used code both for kernel itself as well as user space applications



eBPF + AI: Coding



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

User: Password: Log in | Subscribe | Register

Large language models for patch review

Please consider subscribing to LWN

Subscriptions are the lifeblood of LWN.net. If you appreciate this content and would like to see more of it, your subscription will help to ensure that LWN continues to thrive. Please visit this page to join up and keep LWN on the net.

By **Jonathan Corbet**
October 16, 2025

There have been many discussions in the free-software community about the role of large language models (LLMs) in software development. For the most part, though, those conversations have focused on whether projects should be accepting code output by those models, and under what conditions. But there are other ways in which these systems might participate in the development process. Chris Mason recently started a discussion on the Kernel Summit discussion list about how these models can be used to review patches, rather than create them.

Mason's focus was on how LLMs might reduce the load on kernel maintainers by catching errors before they hit the mailing lists, and by helping contributors increase the quality of their submissions. To that end, he has put together a set of prompts that will produce reviews in a format that maintainers are used to: "The reviews are meant to look like emails on lkml, and even when wildly wrong they definitely succeed there". He included a long list of sample reviews, some of

AI used to review upstream patches

eBPF in the Agentic Era

Agents excel when they get tight loop feedback:

```
write code  →  interpret compiler errors  →  fix the code

run code  →  interpret runtime results  →  write code
```

eBPF in “existential crisis” mode right now:

Slow iteration

Need to boot VM, pass compiled BPF prog into VM, run veristat to get verifier log. Agents can't iterate fast.

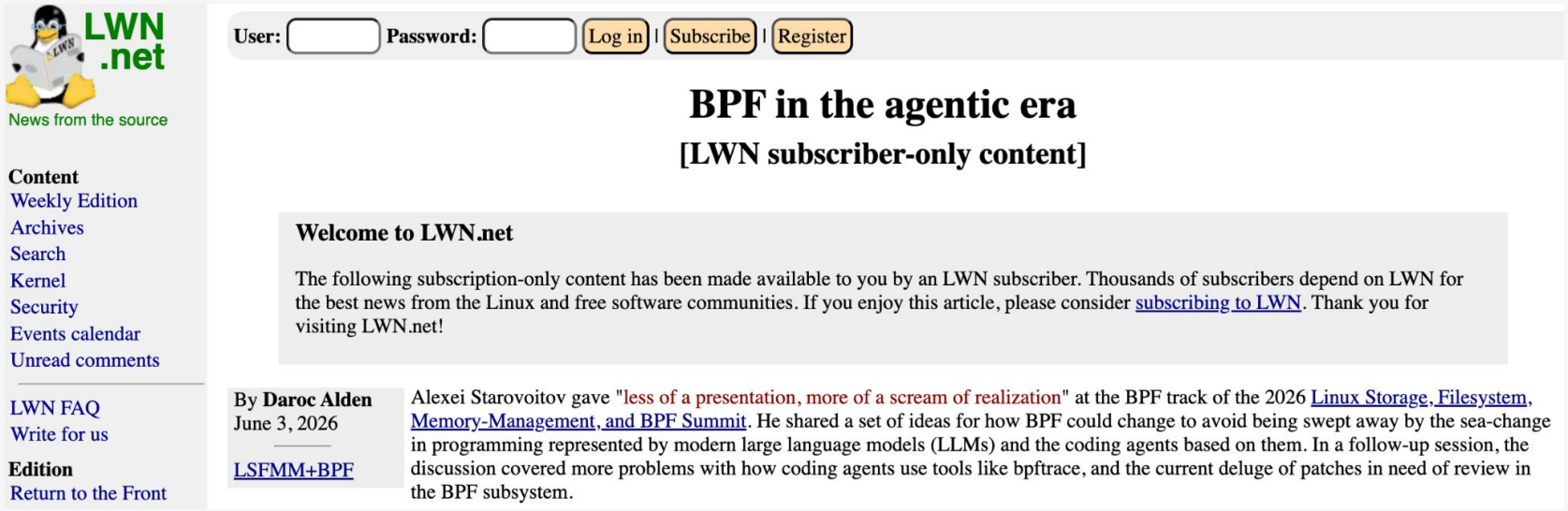
Unreadable errors

Multi-MB verifier logs that dump everything examined. Hard for humans, harder for agents.

"Fundamental limits"

Agents hit verifier rejections and treat them as platform limits. They give up instead of working around.

eBPF + AI: Coding



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

User: Password: Log in | Subscribe | Register

BPF in the agentic era

[LWN subscriber-only content]

Welcome to LWN.net

The following subscription-only content has been made available to you by an LWN subscriber. Thousands of subscribers depend on LWN for the best news from the Linux and free software communities. If you enjoy this article, please consider subscribing to LWN. Thank you for visiting LWN.net!

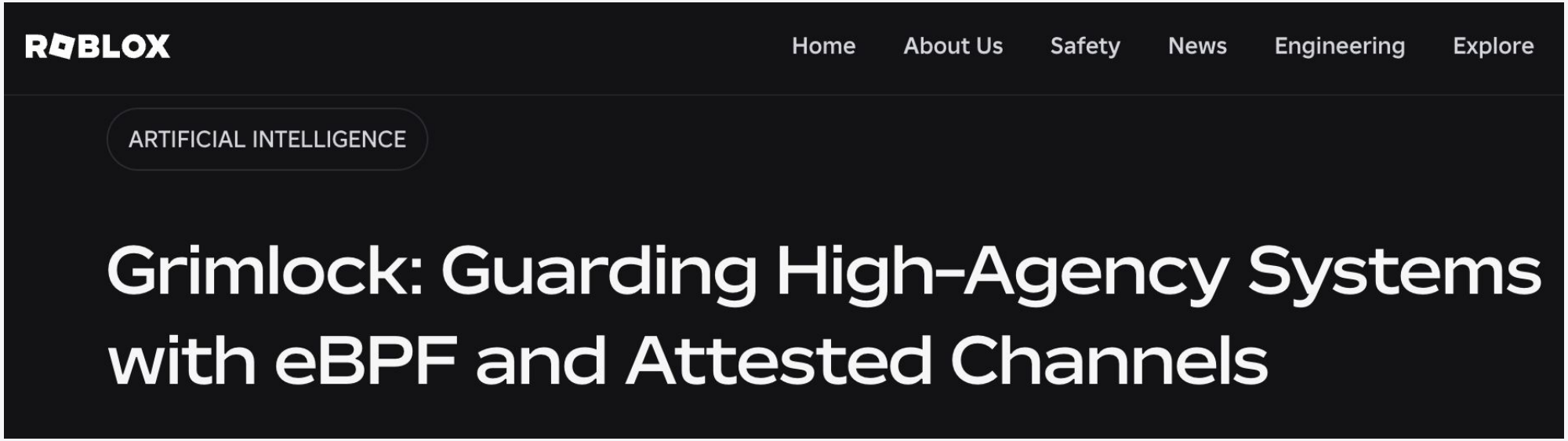
By **Daroc Alden**
June 3, 2026

LSFMM+BPF

Alexei Starovoitov gave "less of a presentation, more of a scream of realization" at the BPF track of the 2026 Linux Storage, Filesystem, Memory-Management, and BPF Summit. He shared a set of ideas for how BPF could change to avoid being swept away by the sea-change in programming represented by modern large language models (LLMs) and the coding agents based on them. In a follow-up session, the discussion covered more problems with how coding agents use tools like bpftrace, and the current deluge of patches in need of review in the BPF subsystem.

"there will be no fighting the verifier anymore"
- Alexei Starovoitov

eBPF + AI: Sandboxing



"Grimlock offers a path toward transparent, auditable, and scope-bound agent-to-agent communication across heterogeneous multi-cloud environments, using commodity Linux primitives and without requiring changes to user-layer orchestration code."

New Uses Cases - sched_ex

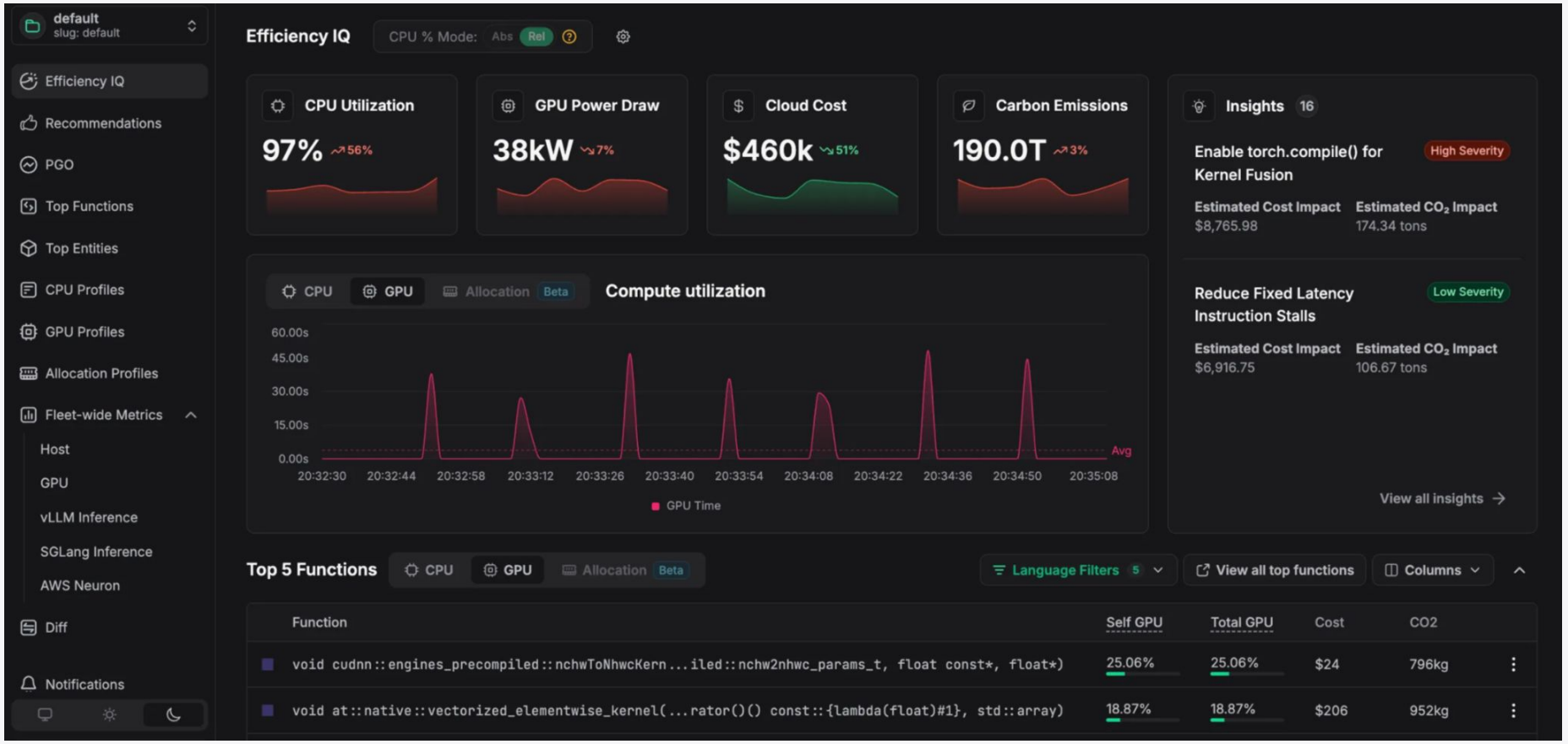
Hello eBPF: A scheduler controlled by sound (20)

Posted on **March 25, 2025**

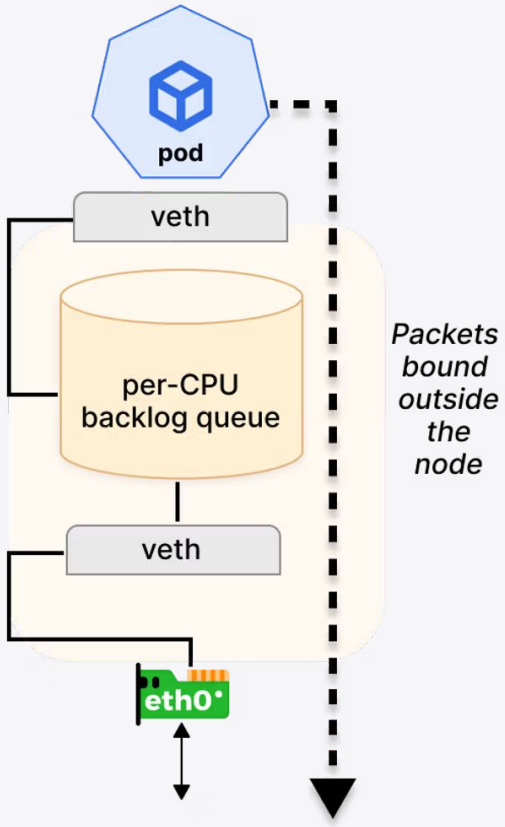
Welcome back to my hello-ebpf series. Last week, I was an attendee at a scheduling and power management summit (and to bring someone a sched-ext-themed birthday cake), and the Chemnitz Linux Days, so this blog post will be a bit shorter but cover a small scheduler that I wrote for Chemnitz.

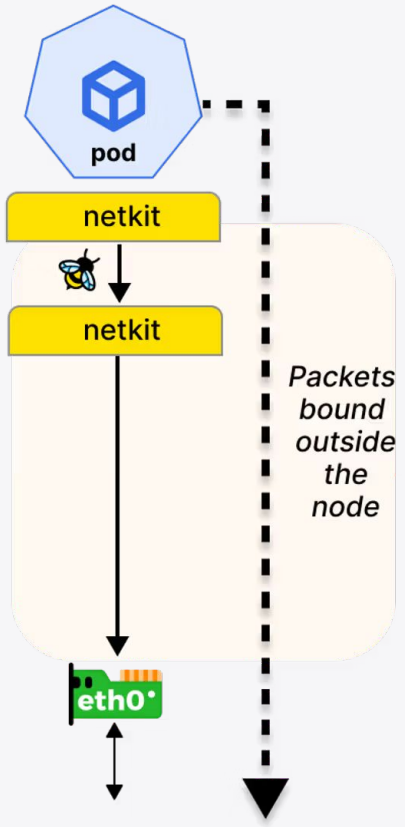
Shouting at my computer to make the game run faster

New Uses Cases - GPU Profiling

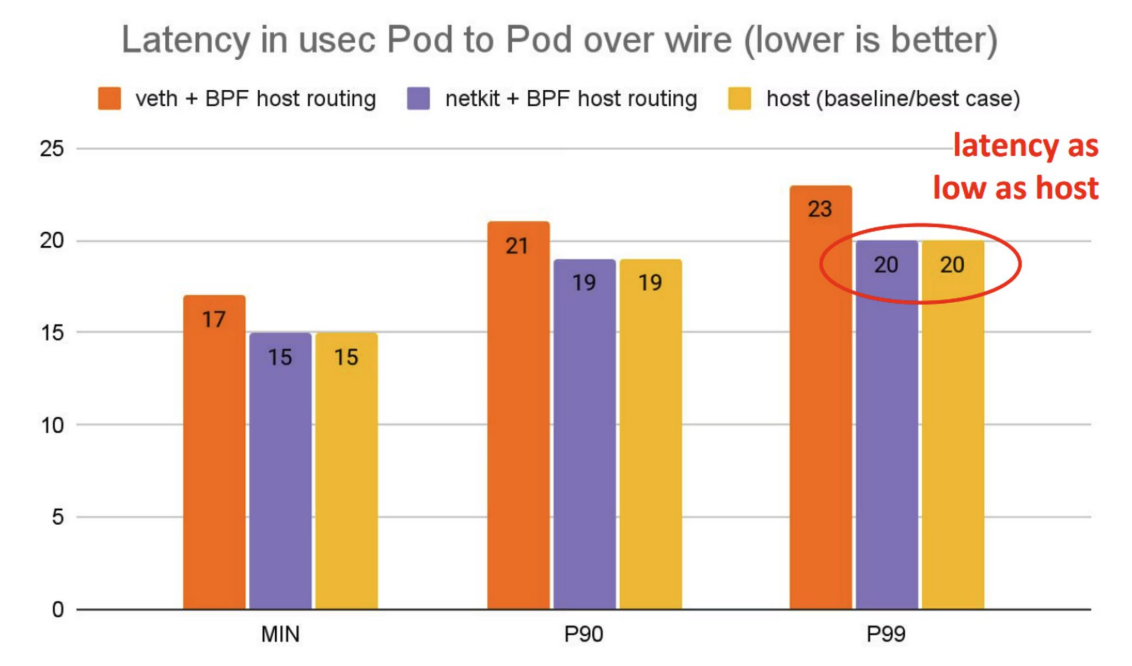


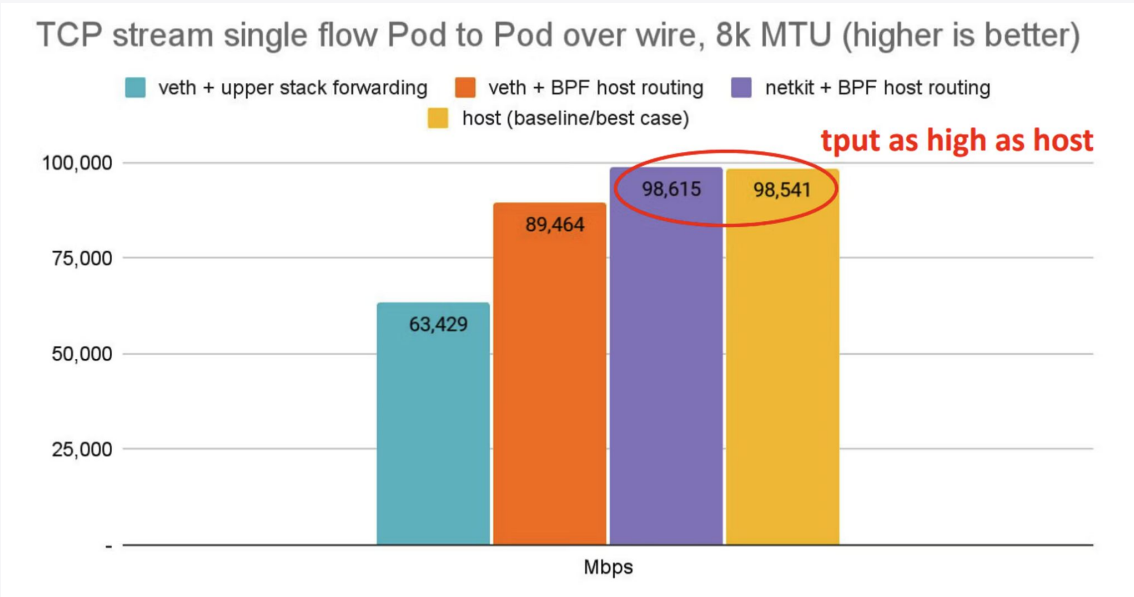
New Uses Cases - netkit





New Uses Cases - netkit





Upstream advances

- BPF Tokens
  - Safe privilege delegation
- Signing of programs
  - Only trusted programs can run
- Zero-Copy memory with bpf_arena
  - User space shares same memory pages as eBPF programs

eBPF Research Funding

eBPF FOUNDATION

Foundation Resources | Funding Opportunities | Blog | Events | Foundation | Join

Academic Research Grant Program

The eBPF Foundation is pleased to announce that applications for the 2026 Academic Research Grant Program will be open from June 15 – July 15, 2026. Recipients announced on September 1, 2026. Return to this page on June 15 to apply.

Academic Research Grant Program Purpose

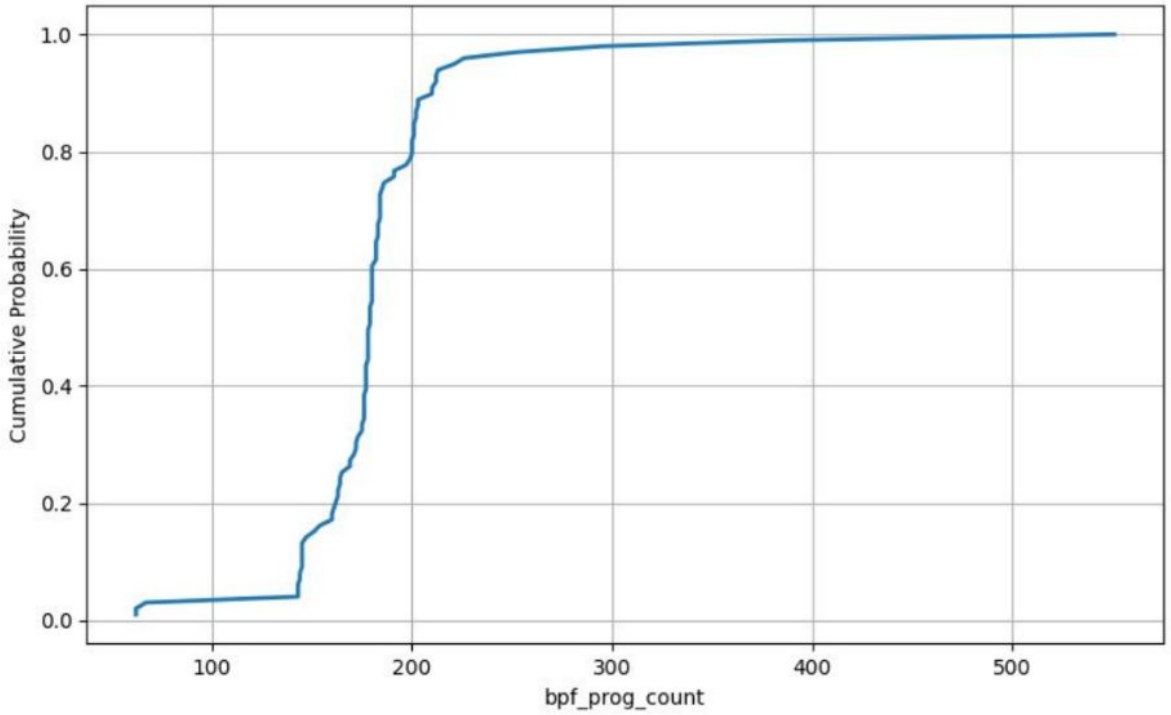
eBPF Foundation is pleased to invite university/research-institute faculty to respond to this call for research proposals on eBPF; unrestricted research grants of up to $50,000 will be available. We are interested in making awards on a wide range of eBPF research, including, but not limited to[1]:

- eBPF program verification, and formal verification of the verifier and JITs

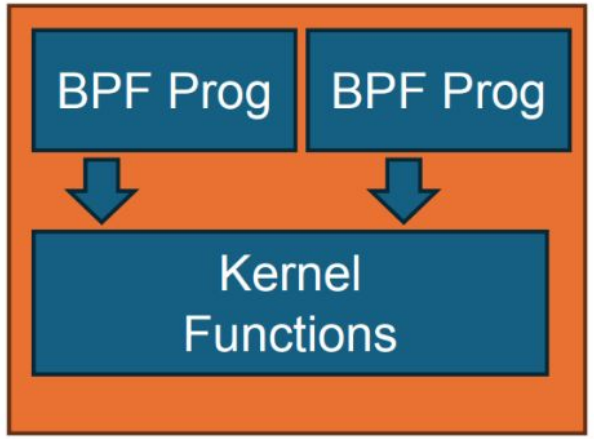
ISOVALENT

eBPF in Companies

Extensions to CI/CD for BPF Specifics Detecting program-2-program interactions

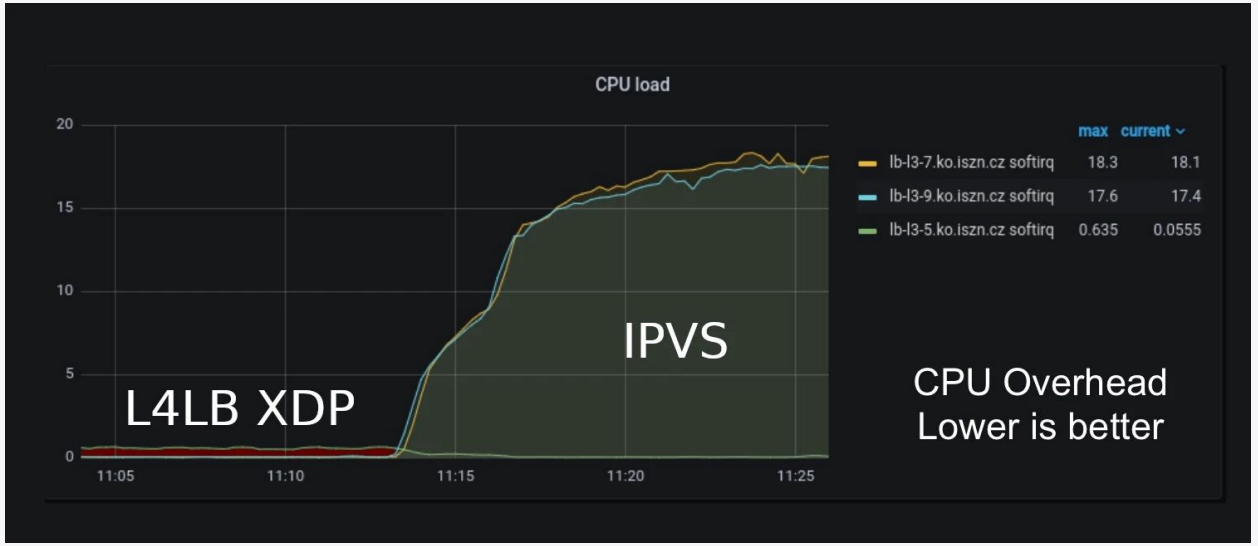


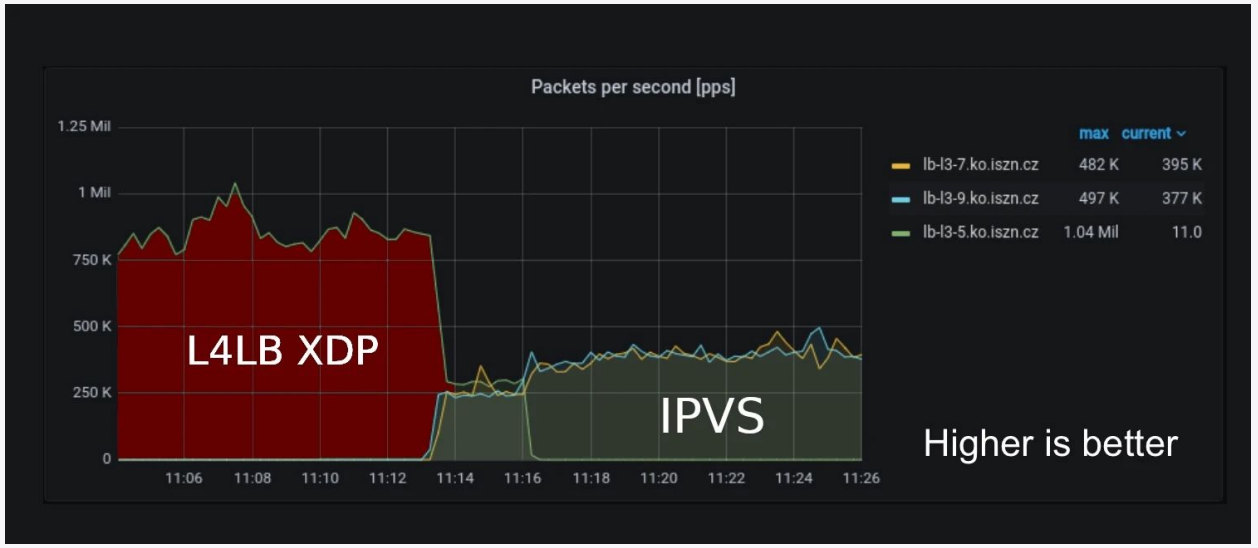
50% of the servers run over 180 BPF programs



- BPF Programs interact in in-direct ways via the kernel
  - Bugs in a BPF program can impact others
  - Large combination of BPF programs at Meta
- Introduce new testing phase
  - Deploy different combination of BPF programs
  - Test with shadow production traffic

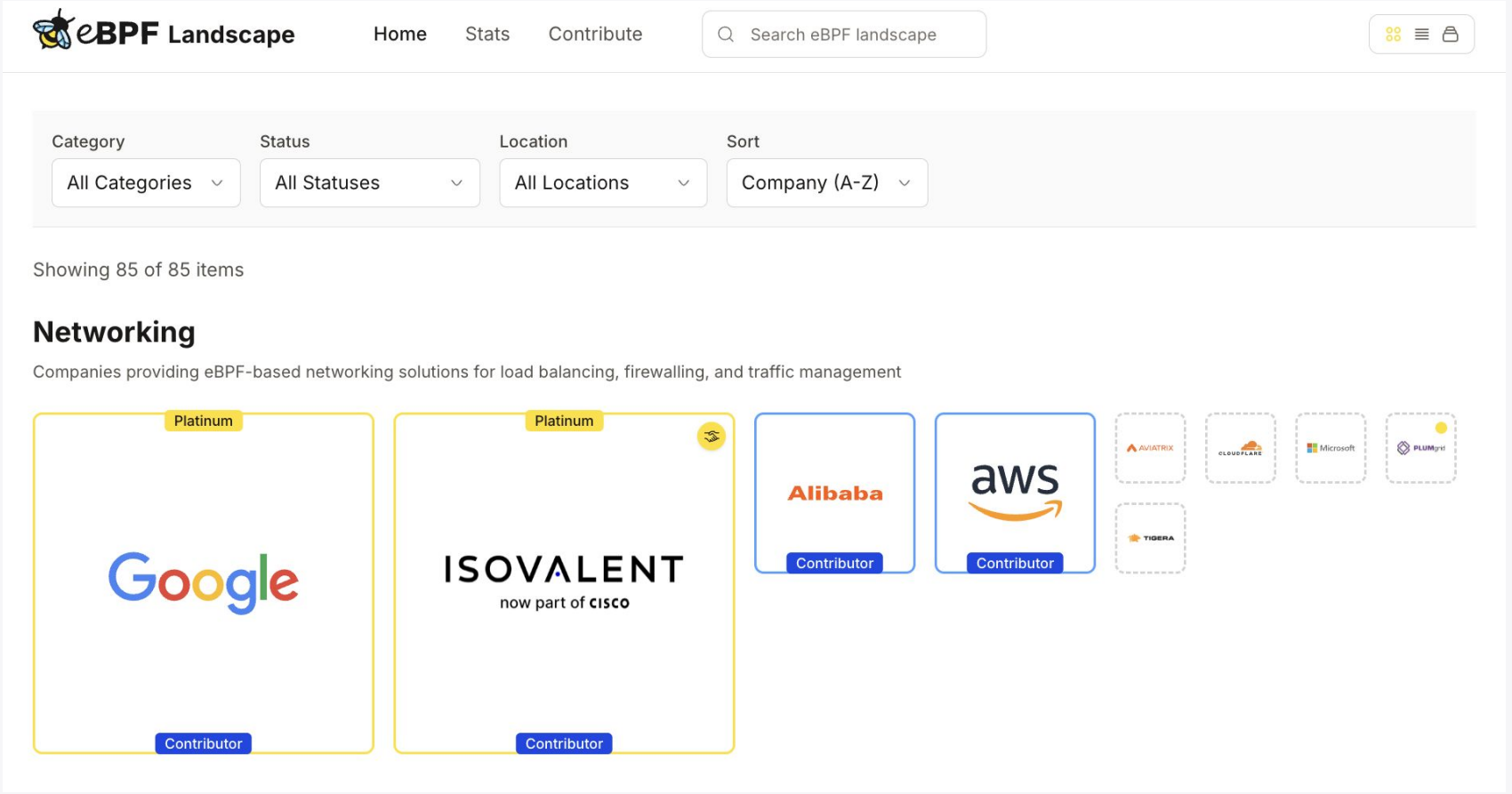
SEZNAM.CZ


CPU load
20
15
10
5
0
11:05
11:10
11:15
11:20
11:25
L4LB XDP
IPVS
max current
lb-l3-7.ko.iszn.cz softirq 18.3 18.1
lb-l3-9.ko.iszn.cz softirq 17.6 17.4
lb-l3-5.ko.iszn.cz softirq 0.635 0.0555
CPU Overhead
Lower is better


Packets per second [pps]
1.25 Mil
1 Mil
750 K
500 K
250 K
0
11:06
11:08
11:10
11:12
11:14
11:16
11:18
11:20
11:22
11:24
11:26
L4LB XDP
IPVS
max current
lb-l3-7.ko.iszn.cz 482 K 395 K
lb-l3-9.ko.iszn.cz 497 K 377 K
lb-l3-5.ko.iszn.cz 1.04 Mil 11.0
Higher is better


eBPF
FOUNDATION
eBPF
In Production
An Overview of Compelling Enterprise Outcomes Using eBPF
by Bill Doerrfeld
February 2026

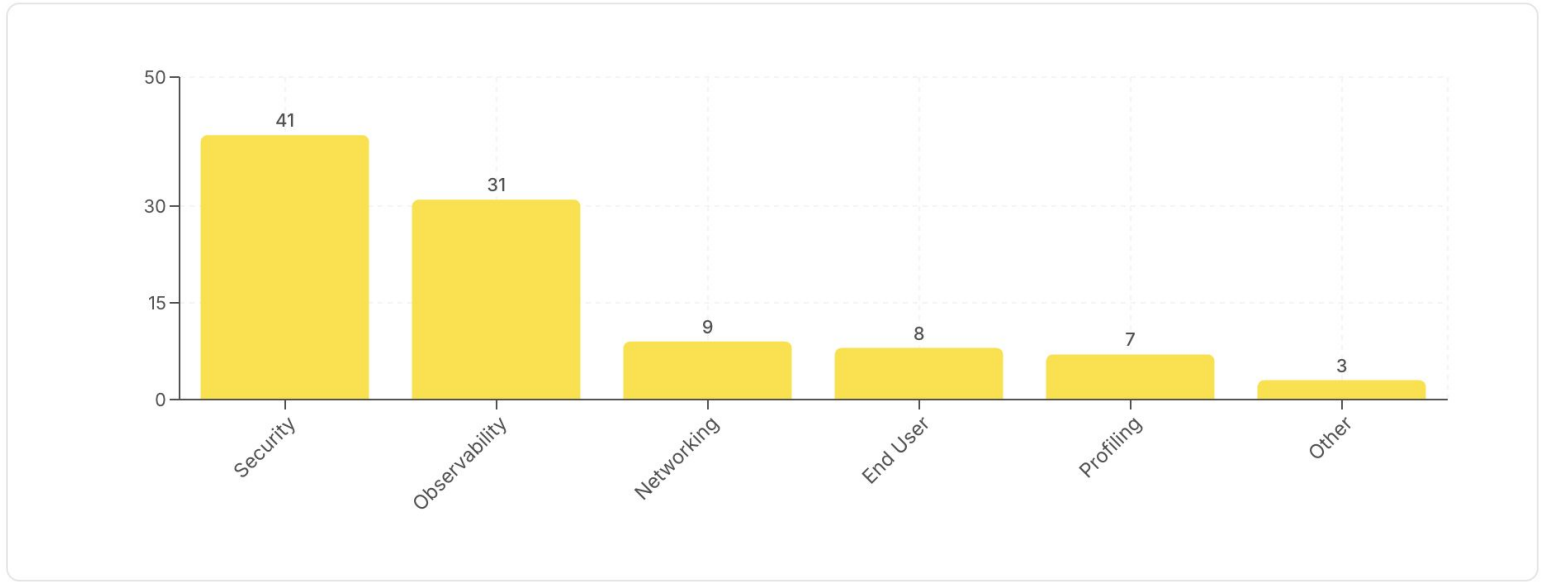
eBPF Company Landscape



eBPF Company Landscape

Distribution by Category

The eBPF ecosystem spans multiple categories, with companies often operating across multiple domains.



eBPF Acquisitions

- PLUMgrid ➡ VMware (December 2016) *Networking*
- Flowmill ➡ Splunk (November 2020) *Observability*
- StackRox ➡ RedHat (January 2021) *Security*
- Kinvolk ➡ Microsoft (April 2021) *Observability*
- Pixie ➡ New Relic (July 2021) *Observability*
- Cmd ➡ Elastic (October 2021) *Security*
- Optimyze ➡ Elastic (August 2021) *Profiling*
- Seekret ➡ Datadog (August 2022) *Observability*
- Pyroscope ➡ Grafana (March 2023) *Profiling*
- Akita ➡ Postman (July 2023) *Observability*
- Helios ➡ Snyk (January 2024) *Security*
- Isovalent ➡ Cisco (April 2024) *Networking, Observability, Security*
- Flow Security ➡ CrowdStrike *(March 2024)* *Security*
- StackState ➡ SUSE (June 2024) *Observability*
- Threatx ➡ A10 Networks (February 2025) *Security*
- Traceable ➡ Harness (February 2025) *Security*
- Wiz ➡ Google (March 2025) *Security*
- Nyx ➡ Upwind (April 2025) Security
- Otterize ➡ Cyera (June 2025) Security
- Protect AI ➡ Palo Alto Networks (July 2025) Security
- Red Canary ➡ Zscaler (August 2025) *Security*
- Prompt Security ➡ SentinelOne (August 2025) *Security*
- MantisNet ➡ F5 (August 2025) Observability
- Attribute ➡ DoiT (July 2026) FinOps

Feature & Sensor Upgrade

Platform & Community Land Grabs

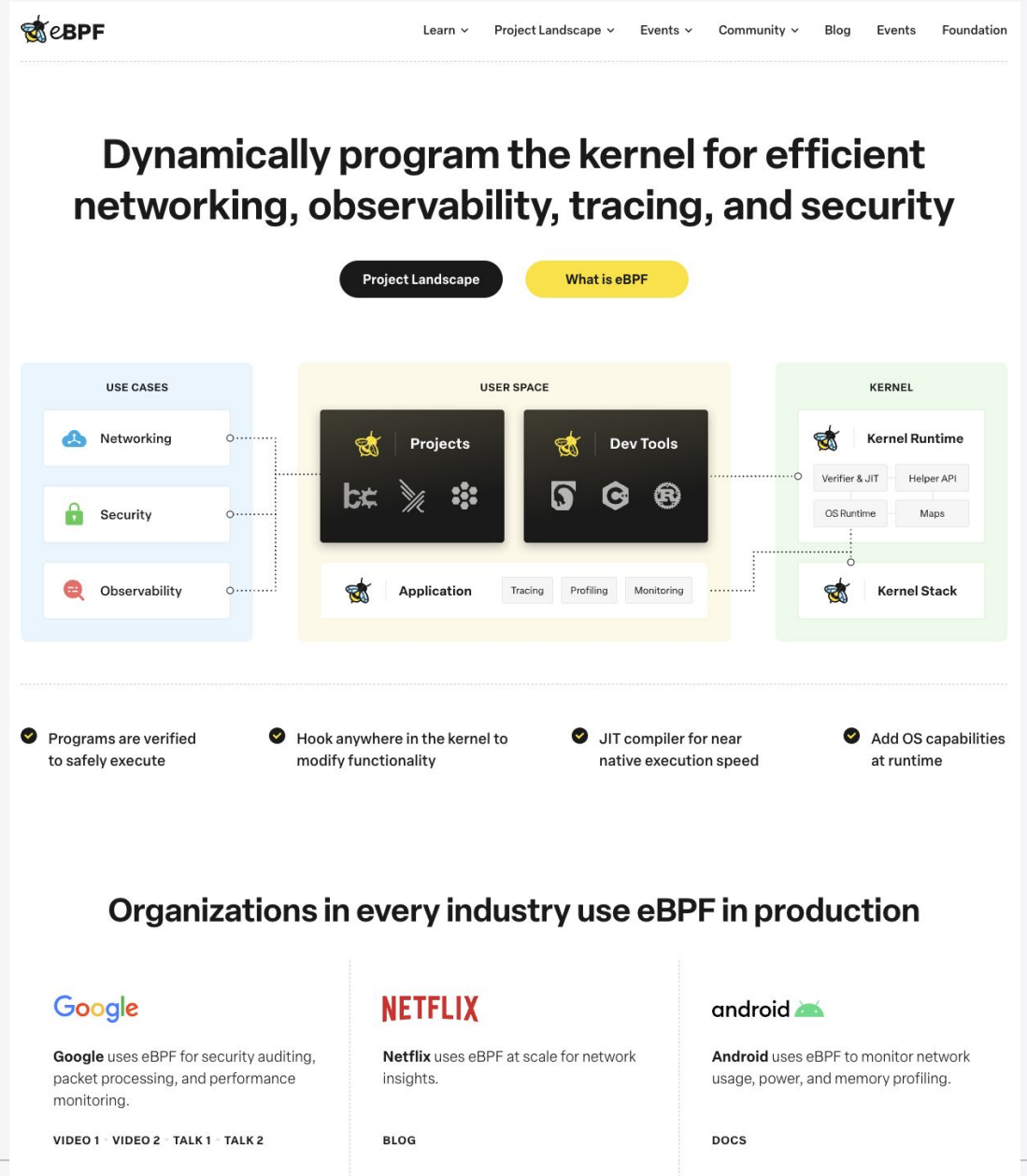
AI & Runtime Security Consolidation

Wave 1	2020	New Relic acquires Pixie	Observability
		Splunk acquires Flowmill	Observability
	2021	Red Hat acquires StackRox	Security
		Elastic acquires Cmd	Security
		Elastic acquires Optimyze	Observability
		Microsoft acquires Kinvolk	Observability
	2022	Datadog acquires Seekret	Observability
Wave 2	2023	Snyk acquires Helios	Security, Observability
		Postman acquires Akita	Observability, Security
		Grafana acquires Pyroscope	Observability
	2024	Cisco acquires Isovalent	Networking, Security, Observability
		SUSE acquires StackState	Observability
		CrowdStrike acquires Flow Security	Security
		Cisco acquires Splunk	Observability, Security
Wave 3	2025	Cyera acquires Otterize	Security
		Harness acquires Traceable	Security, Observability
		Upwind acquires Nyx	Security
		Zscaler acquires Red Canary	Security
		F5 acquires MantisNet	Networking, Observability
		Palo Alto Networks acquires Protect AI	AI security
		SentinelOne acquires Prompt Security	AI security
		A10 Networks acquires ThreatX Protect	Networking, Security
	2026	Google acquires Wiz	Security

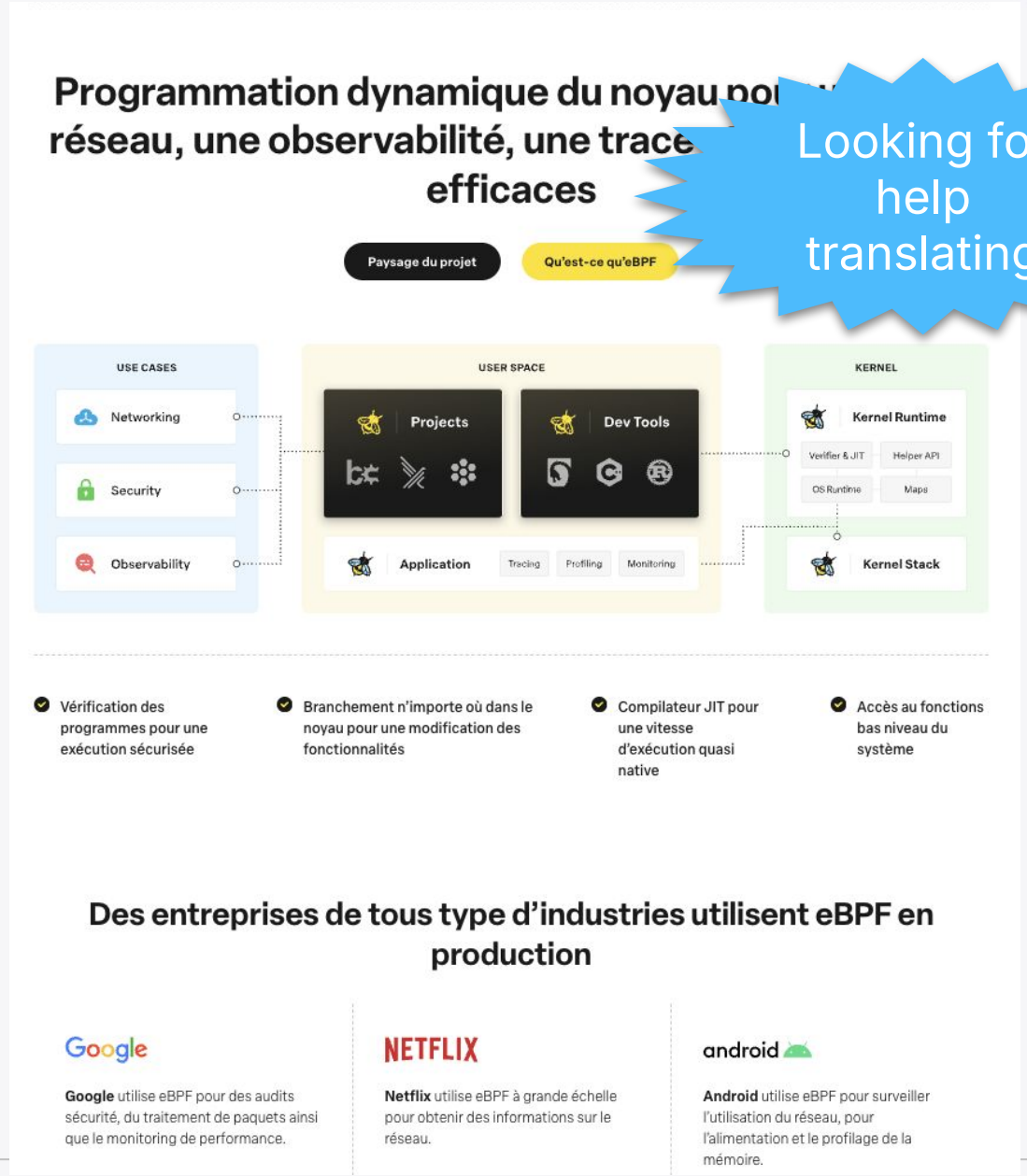
ISOVALENT

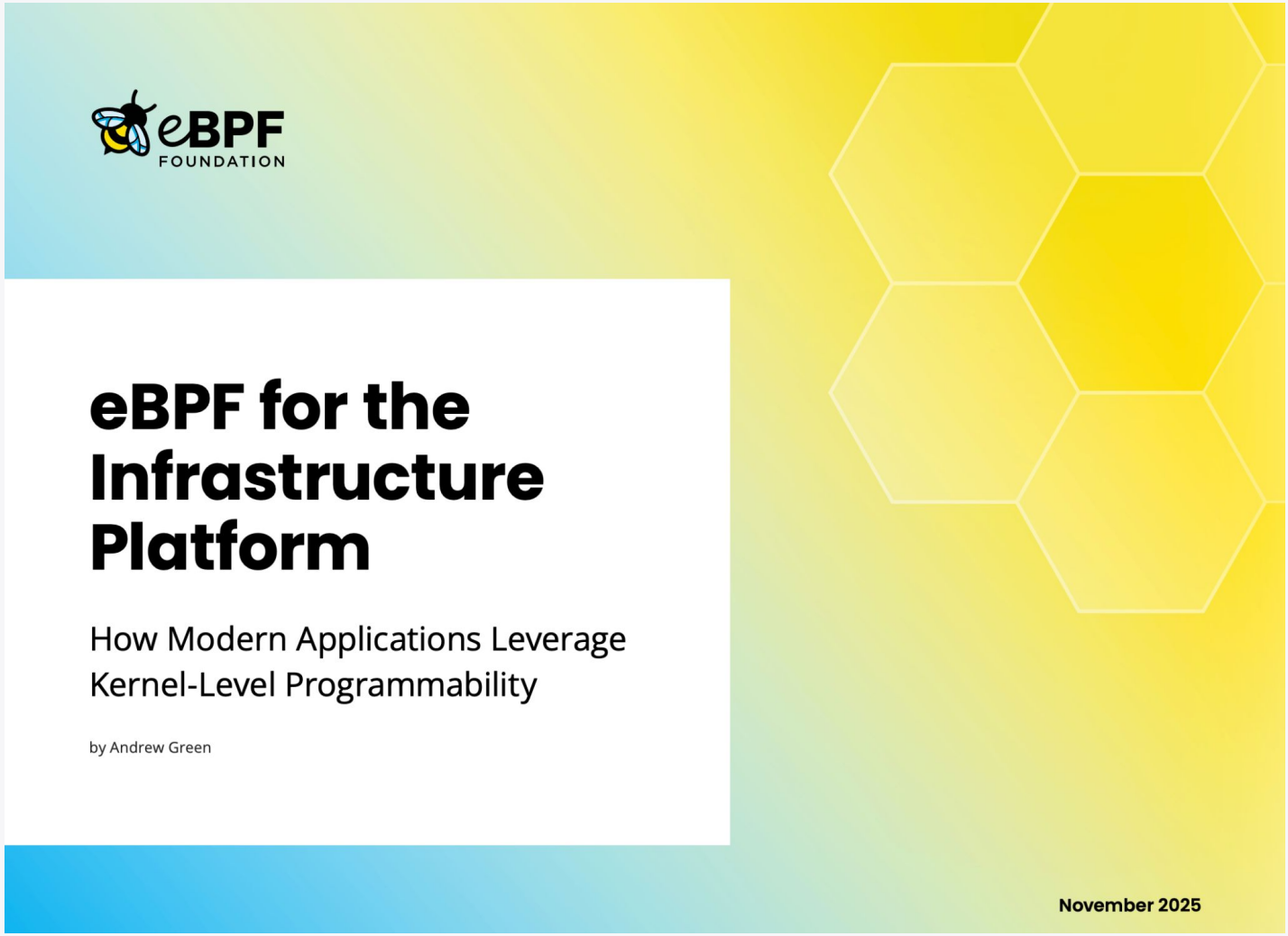
Where to go Next

ebpf.io


eBPF
Learn
Project Landscape
Events
Community
Blog
Events
Foundation
Dynamically program the kernel for efficient networking, observability, tracing, and security
Project Landscape
What is eBPF
USE CASES
Networking
Security
Observability
USER SPACE
Projects
Dev Tools
Application
Tracing
Profiling
Monitoring
KERNEL
Kernel Runtime
Verifier & JIT
Helper API
OS Runtime
Maps
Kernel Stack
Programs are verified to safely execute
Hook anywhere in the kernel to modify functionality
JIT compiler for near native execution speed
Add OS capabilities at runtime
Organizations in every industry use eBPF in production
Google
Google uses eBPF for security auditing, packet processing, and performance monitoring.
VIDEO 1 · VIDEO 2 · TALK 1 · TALK 2
NETFLIX
Netflix uses eBPF at scale for network insights.
BLOG
android
Android uses eBPF to monitor network usage, power, and memory profiling.
DOCS

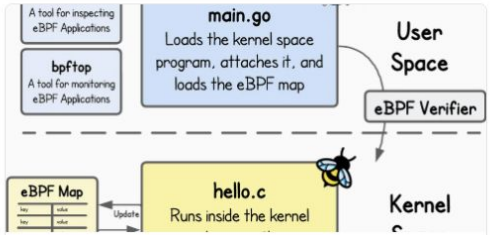
ebpf.io
en Frrançais
Italiano
Português
Swahili
Español
繁體中文
简体中文




eBPF
FOUNDATION
eBPF for the Infrastructure Platform
How Modern Applications Leverage Kernel-Level Programmability
by Andrew Green
November 2025

ebpf.io Labs

Dig into eBPF with interactive labs

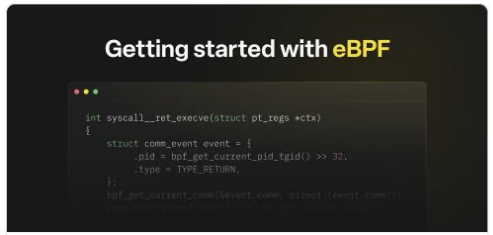


NETWORKING

eBPF Beginner Skill Path

Learn eBPF from the ground up — from writing and running your first program, to storing data in eBPF maps, inspecting and monitoring kernel activity using bpftool and bpftop, and understanding how the verifier ensures safe eBPF execution. Finish with a hands-on challenge to put your new skills to the test.

Start the labs

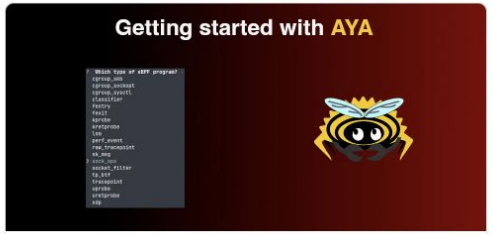


NETWORKING

Getting started with eBPF

eBPF is the new standard to program Linux kernel capabilities in a safe and efficient manner without requiring to change kernel source code or loading kernel modules. It has enabled a new generation of high performance tooling.

Start the lab



NETWORKING

Getting started with eBPF with Aya

Aya is a Rust framework for writing eBPF programs. These labs guide you through using it to develop eBPF programs. 🦀 + 🐝 = Aya

Start the labs

NETWORKING

Learning eBPF Tutorial

Learn how to write your first eBPF Hello World program and dive into all the key concepts and tools of eBPF such as eBPF maps, bytecode, bpftool, xdp and the eBPF verifier.

Start the lab

The Illustrated Children's Guide to eBPF

eBPF Documentary - ebpfdocumentary.com

A Vision for eBPF Inside