


KubeCon | CloudNativeCon

Europe 2020

Virtual

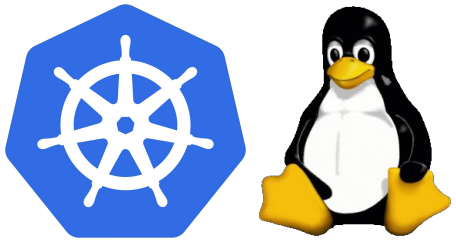
eBPF and Kubernetes: Little Helper Minions for Scaling Microservices

Daniel Borkmann (Isovalent), eBPF kernel co-maintainer

Kubernetes is eating the world

Kubernetes widely regarded as the **Cloud OS** these days.

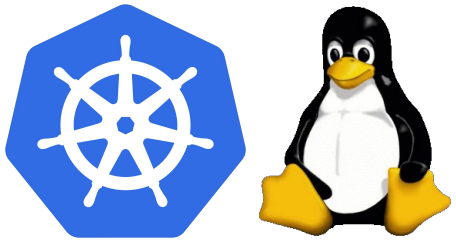
- ➔ Linux kernel as a base foundation to provide all building blocks
- ➔ Major core parts of critical path delivered via CNI networking plugin



Kubernetes is eating the world

Kubernetes widely regarded as the **Cloud OS** these days.

- ➔ Linux kernel as a base foundation to provide all building blocks
- ➔ Major core parts of critical path delivered via CNI networking plugin
  - ➔ General Pod connectivity
  - ➔ IP Address Management (IPAM)
  - ➔ Service handling and load balancing
  - ➔ Network policy enforcement
  - ➔ Monitoring and troubleshooting



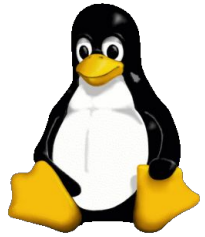
Kubernetes is eating the world

Kubernetes widely regarded as the **Cloud OS** these days.

- ➔ Linux kernel as a base foundation to provide all building blocks
- ➔ Major core parts of critical path delivered via CNI networking plugin
  - ➔ General Pod connectivity
  - ➔ IP Address Management (IPAM)
  - ➔ Service handling and load balancing
  - ➔ Network policy enforcement
  - ➔ Monitoring and troubleshooting

Clear trends: increasing Pod density and decreasing Pod lifespans

CNCF'19 survey report: https://www.cncf.io/wp-content/uploads/2020/03/CNCF_Survey_Report.pdf
sysdig'19 container usage report: https://sysdig.com/blog/sysdig-2019-container-usage-report/



Challenges from OS kernel side

Keeping up with **performance** & **scalability** requirements while having ever-increasing **complexity** of subsystems.

Partially due to “never break user space” paradigm.

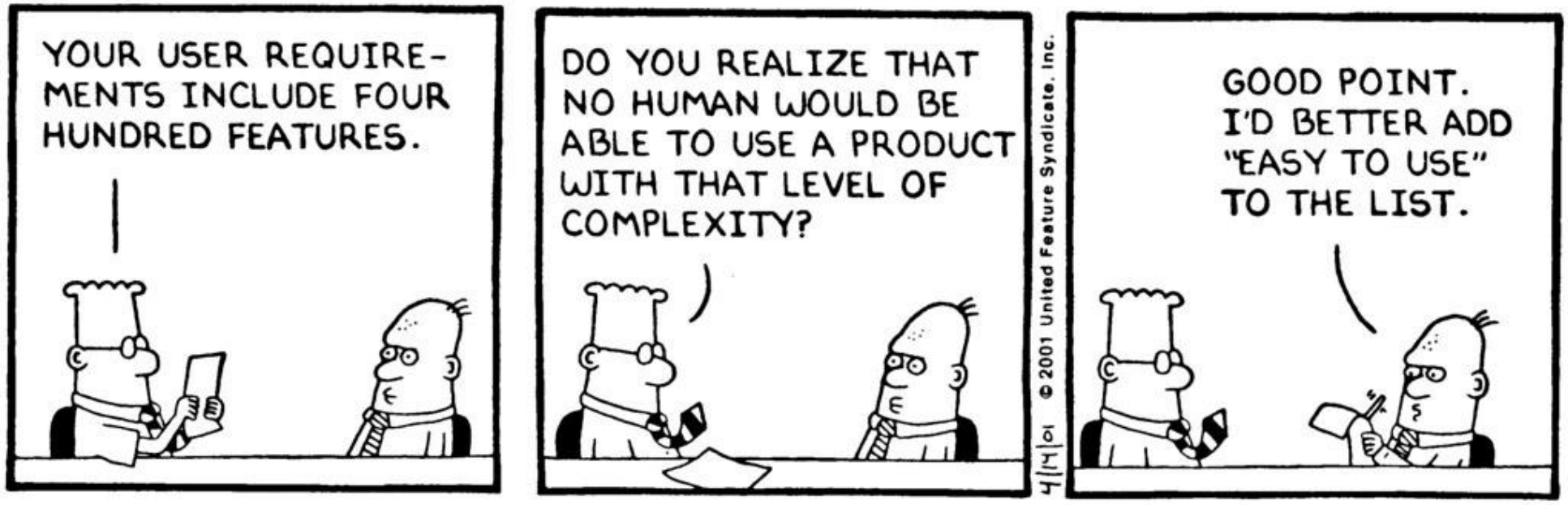
The kernel often suffers a feature creeping normality.

Challenges from OS kernel side

Keeping up with **performance** & **scalability** requirements while having ever-increasing **complexity** of subsystems.

Partially due to “never break user space” paradigm.

The kernel often suffers a feature creeping normality.



Challenges from OS kernel side

Def. **creeping normality**:

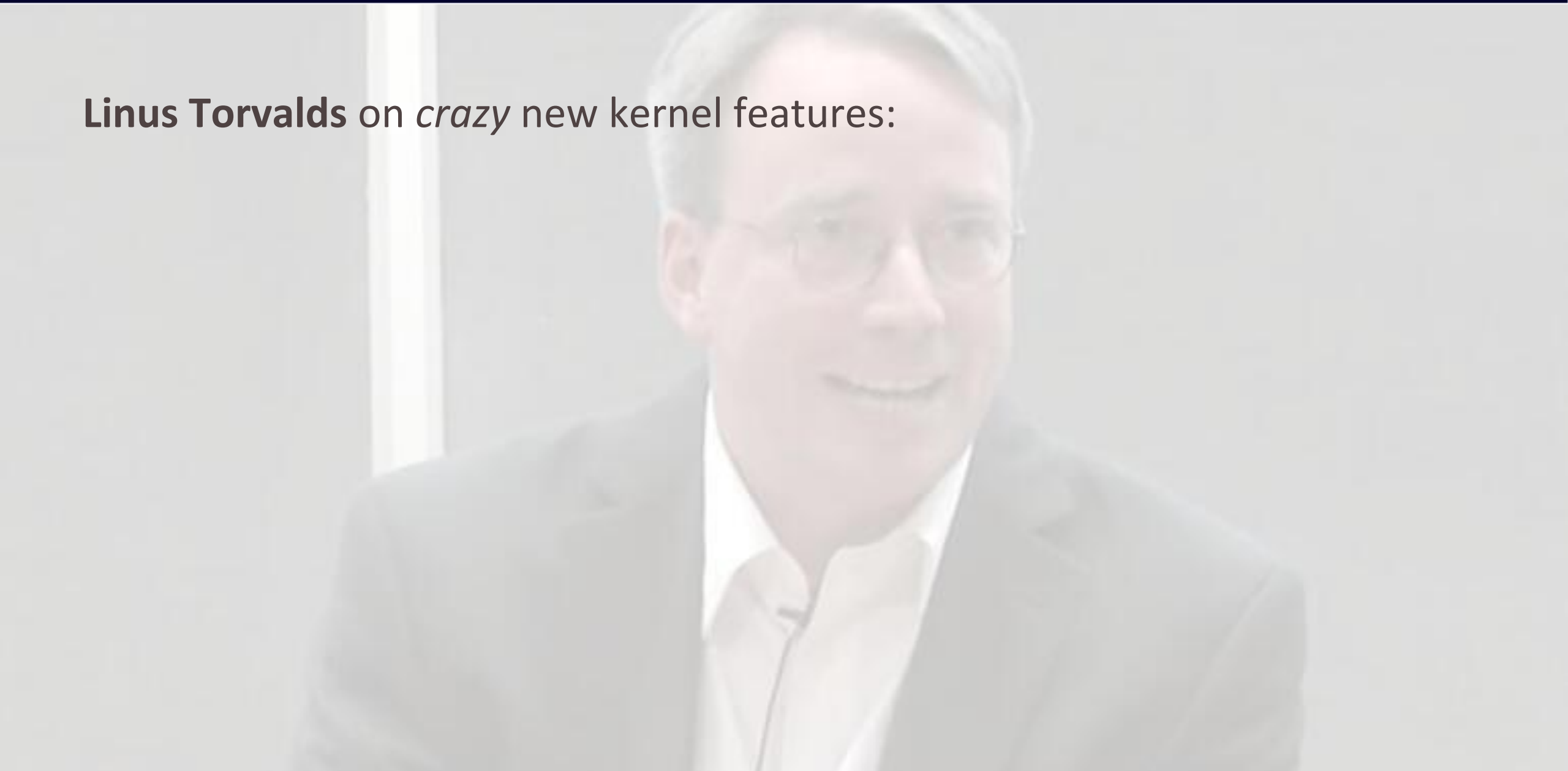
Challenges from OS kernel side

Def. **creeping normality**: … is a process by which a major change can be accepted as normal and acceptable if it happens slowly through small, often unnoticeable, increments of change. The change could otherwise be regarded as objectionable if it took place in a single step or short period.

https://en.wikipedia.org/w/index.php?title=Death_by_a_thousand_cuts_(psychology)

Challenges from OS kernel side

Linus Torvalds on *crazy* new kernel features:



Challenges from OS kernel side

Linus Torvalds on *crazy* new kernel features:

> So I can work with crazy people, that's not the problem. They just need to _sell_ their crazy stuff to me using non-crazy arguments, and in small and well-defined pieces. When I ask for killer features, I want them to lull me into a safe and cozy world where the stuff they are pushing is actually useful to mainline people _first_.
>
> In other words, every new crazy feature should be hidden in a nice solid "Trojan Horse" gift: something that looks _obviously_ good at first sight.

Linus Torvalds, https://lore.kernel.org/lkml/alpine.LFD.2.00.1001251002430.3574@localhost.localdomain/

eBPF - beginning to present

This is what actually happened to eBPF back in the initial days when we tried to get it merged.

Except: It's a crazy new kernel feature which ...

eBPF - beginning to present

This is what actually happened to eBPF back in the initial days when we tried to get it merged.

Except: It's a crazy new kernel feature which ...

➔ Reduces the kernel's feature creeping normality

eBPF - beginning to present

This is what actually happened to eBPF back in the initial days when we tried to get it merged.

Except: It's a crazy new kernel feature which ...

- ➔ Reduces the kernel's feature creeping normality
- ➔ Keeps the kernel's fast-path fast

eBPF - beginning to present

This is what actually happened to eBPF back in the initial days when we tried to get it merged.

Except: It's a crazy new kernel feature which ...

- ➔ Reduces the kernel's feature creeping normality
- ➔ Keeps the kernel's fast-path fast
- ➔ Allows users to fully & safely make the kernel programmable to solve their real-world production problems

eBPF for networking in a nutshell

KubeCon CloudNativeCon Europe 2020 Virtual


```
section(“to_netdev”)
int handle(struct sk_buff *skb) {
    …
    if (tcp->dport == 80)
        redirect(lxc0);
    …
    return DROP_PACKET;
}
```

userspace kernelspace

eBPF for networking in a nutshell


```
section("to_netdev")
int handle(struct sk_buff *skb) {
    ...
    if (tcp->dport == 80)
        redirect(lxc0);
    ...
    return DROP_PACKET;
}
```

agent
(e.g. Cilium)

userspace | kernelspace

eBPF for networking in a nutshell


```
section("to_netdev")
int handle(struct sk_buff *skb) {
    ...
    if (tcp->dport == 80)
        redirect(lxc0);
    ...
    return DROP_PACKET;
}
```

agent
(e.g. Cilium)

clang -target bpf [...]

foo.o

userspace kernelspace

eBPF for networking in a nutshell


```
section("to_netdev")
int handle(struct sk_buff *skb) {
    ...
    if (tcp->dport == 80)
        redirect(lxc0);
    ...
    return DROP_PACKET;
}
```

agent
(e.g. Cilium)

clang -target bpf [...]

foo.o

BPF
loader

userspace | kernelspace

eBPF for networking in a nutshell


```
section("to_netdev")
int handle(struct sk_buff *skb) {
    ...
    if (tcp->dport == 80)
        redirect(lxc0);
    ...
    return DROP_PACKET;
}
```

agent (e.g. Cilium)

clang -target bpf [...]

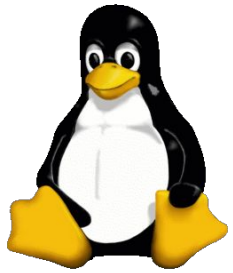
LLVM

foo.o

BPF loader

bpf(BPF_PROG_LOAD, ...)

BPF verifier



userspace kernelspace

eBPF for networking in a nutshell


```
section("to_netdev")
int handle(struct sk_buff *skb) {
    ...
    if (tcp->dport == 80)
        redirect(lxc0);
    ...
    return DROP_PACKET;
}
```

agent (e.g. Cilium)

clang -target bpf [...]

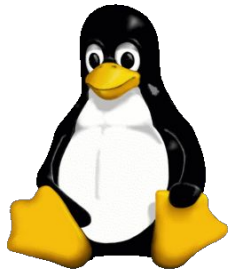
foo.o

BPF loader

bpf(BPF_PROG_LOAD, ...)

JIT

BPF verifier



userspace kernelspace

eBPF for networking in a nutshell


```
section("to_netdev")
int handle(struct sk_buff *skb) {
    ...
    if (tcp->dport == 80)
        redirect(lxc0);
    ...
    return DROP_PACKET;
}
```

agent (e.g. Cilium)

clang -target bpf [...]

foo.o

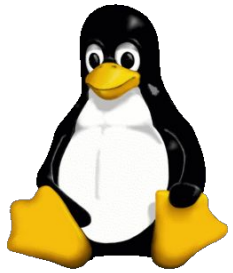
BPF loader

bpf(BPF_PROG_LOAD, ...)

native code

JIT

BPF verifier



userspace kernelspace

eBPF for networking in a nutshell


```
section("to_netdev")
int handle(struct sk_buff *skb) {
    ...
    if (tcp->dport == 80)
        redirect(lxc0);
    ...
    return DROP_PACKET;
}
```

agent (e.g. Cilium)

bpf(BPF_MAP...)

BPF maps

native code

JIT

BPF verifier

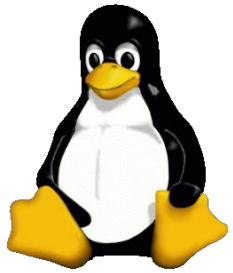
clang -target bpf [...]

foo.o

BPF loader

bpf(BPF_PROG_LOAD, ...)

LLVM



userspace kernelspace

eBPF for networking in a nutshell

KubeCon CloudNativeCon Europe 2020 Virtual


```
section("to_netdev")
int handle(struct sk_buff *skb) {
    ...
    if (tcp->dport == 80)
        redirect(lxc0);
    ...
    return DROP_PACKET;
}
```

clang -target bpf [...]

foo.o

BPF loader

bpf(BPF_PROG_LOAD, ...)

agent (e.g. Cilium)

bpf(BPF_MAP...)

BPF maps

event: packet

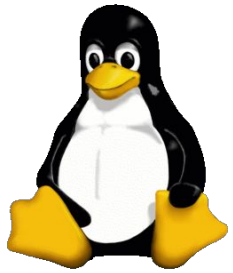
eth0

BPF_PROG_RUN(code)

native code

JIT

BPF verifier



userspace kernelspace

eBPF for networking in a nutshell


```
section("to_netdev")
int handle(struct sk_buff *skb) {
    ...
    if (tcp->dport == 80)
        redirect(lxc0);
    ...
    return DROP_PACKET;
}
```

agent (e.g. Cilium)

bpf(BPF_MAP...)

BPF maps

native code

BPF_PROG_RUN(code)

event: packet

eth0

Stack (App)

JIT

BPF verifier

lxc0

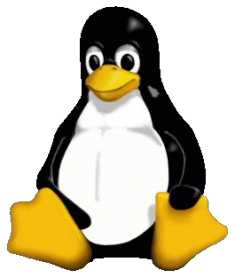
clang -target bpf [...]

foo.o

BPF loader

bpf(BPF_PROG_LOAD, ...)

LLVM



userspace kernelspace

eBPF for networking in a nutshell

KubeCon CloudNativeCon Europe 2020 Virtual

- cannot crash the kernel
- as fast as kernel module
- stable API guarantees

event: packet

eth0

BPF_PROG_RUN(code)

```
section("to_netdev")
int handle(struct sk_buff *skb) {
    ...
    if (tcp->dport == 80)
        redirect(lxc0);
    ...
    return DROP_PACKET;
}
```

agent (e.g. Cilium)

bpf(BPF_MAP...)

BPF maps

native code

Stack (App)

JIT

BPF verifier

clang -target bpf [...]

foo.o

BPF loader

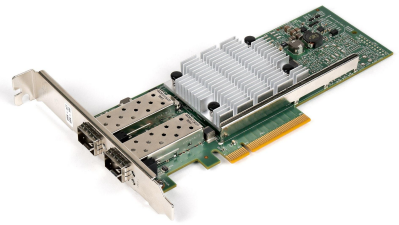
bpf(BPF_PROG_LOAD, ...)

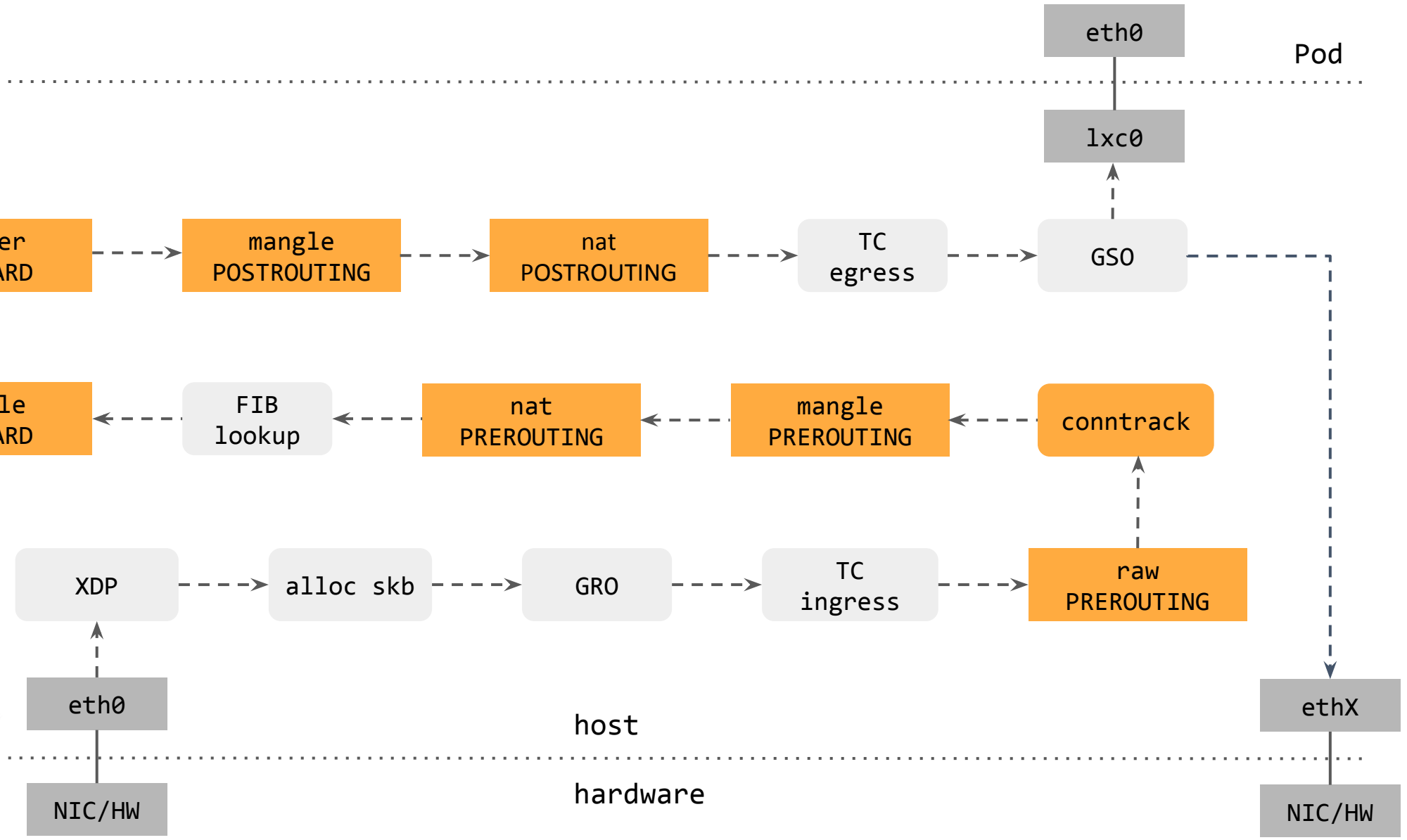
lxc0

LLVM

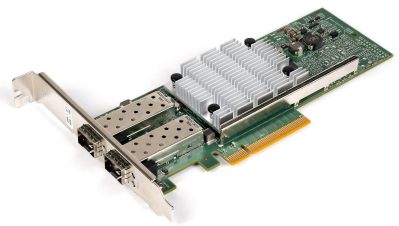
userspace kernelspace

Packet flow for kube-proxy

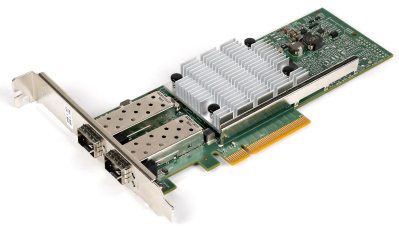




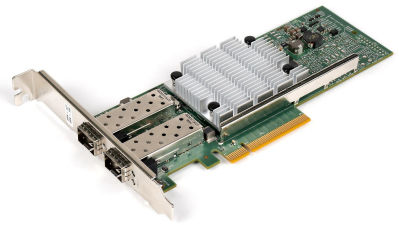
Packet flow for kube-proxy



Packet flow for kube-proxy

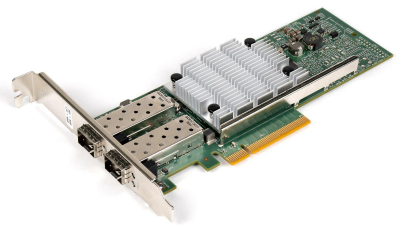


Packet flow for eBPF (Cilium)

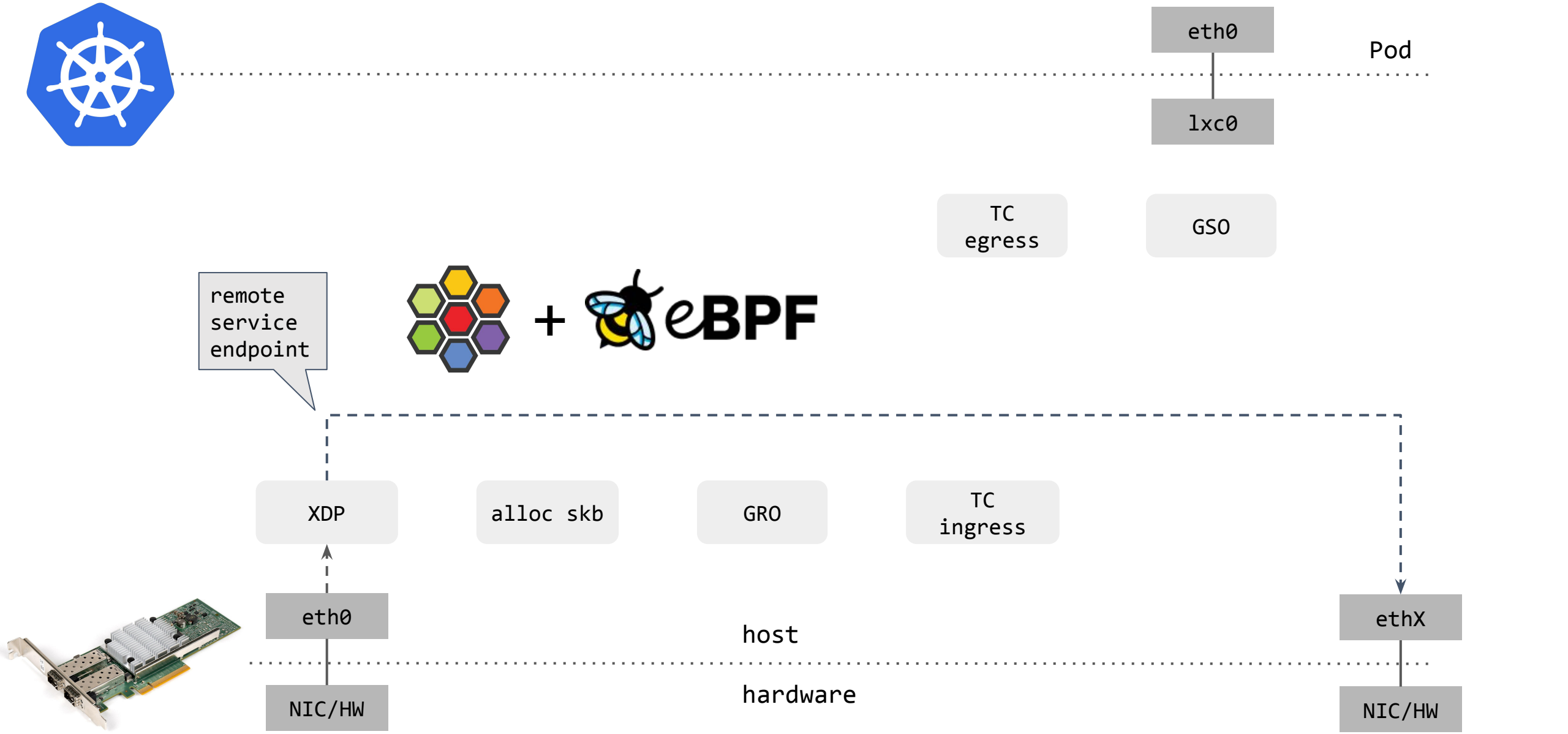


Packet flow for eBPF (Cilium)

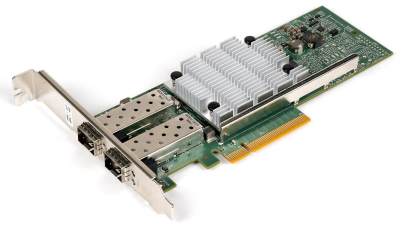
+ eBPF

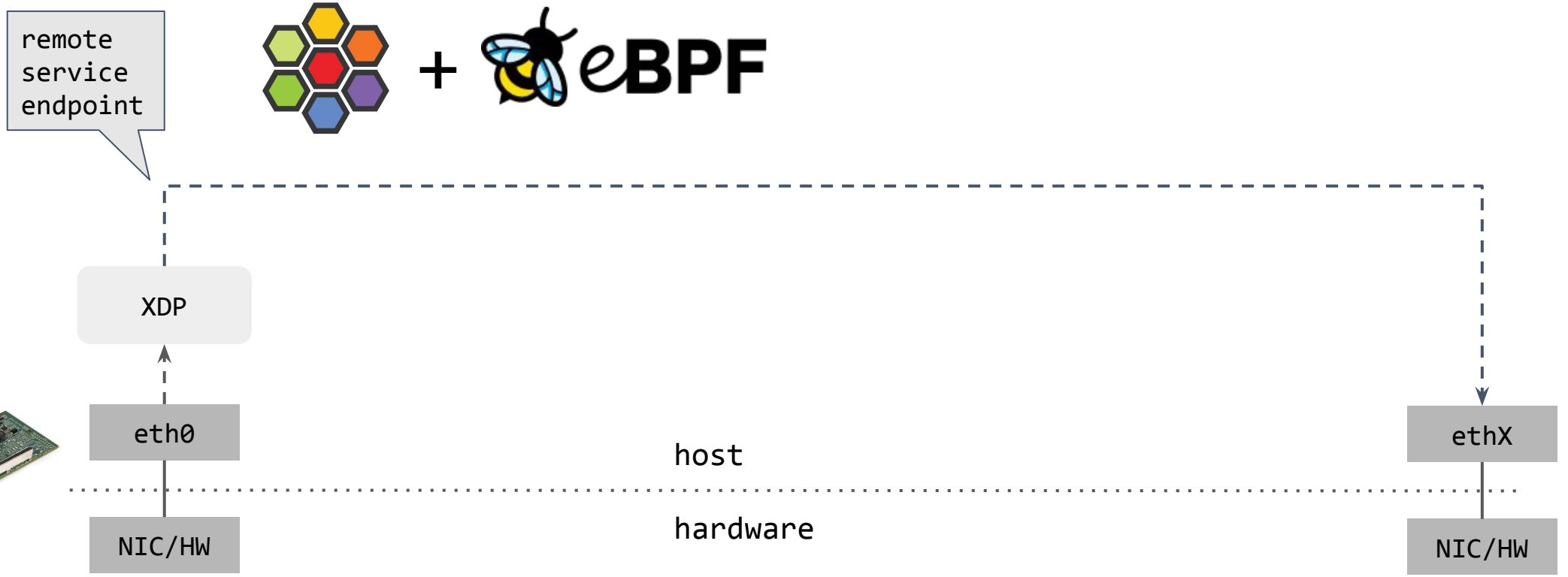


Packet flow for eBPF (Cilium)



Packet flow for eBPF (Cilium)





eBPF - beginning to present

Old “SDN” landscape at the time ...

Open vSwitch (**ovs**), traffic control (**tc**), netfilter (**iptables**, **ipvs**, **nftables**) to ‘program’ datapath

BPF at the time used for tcpdump, seccomp and misc other areas

< 2013

David S. Miller et al, https://lore.kernel.org/netdev/20150311.164550.2247000676434784013.davem@davemloft.net/

eBPF - beginning to present

Old “SDN” landscape at the time ...

Open vSwitch (**ovs**), traffic control (**tc**), netfilter (**iptables**, **ipvs**, **nftables**) to ‘program’ datapath

BPF at the time used for tcpdump, seccomp and misc other areas

Feature creep and often **duplication** between tc and netfilter

< 2013

David S. Miller et al, https://lore.kernel.org/netdev/20150311.164550.2247000676434784013.davem@davemloft.net/

eBPF - beginning to present

Old “SDN” landscape at the time ...

Open vSwitch (**ovs**), traffic control (**tc**), netfilter (**iptables**, **ipvs**, **nftables**) to ‘program’ datapath

BPF at the time used for tcpdump, seccomp and misc other areas

Feature creep and often **duplication** between tc and netfilter

ovs the most advanced data plane in the kernel back then which did many things right, but seen as ‘Frankenstein’ by core networking devs since it didn’t integrate well with rest of networking

< 2013

David S. Miller et al, https://lore.kernel.org/netdev/20150311.164550.2247000676434784013.davem@davemloft.net/

eBPF - beginning to present

Old “SDN” landscape at the time ...

Open tapath

BPF a

Featu

ovs th but seen

as ‘Fr vorking

How do they compare to eBPF? Think of it this way:

- ➔ ovs, tc, netfilter allow you to “program” the datapath, but *only* if the datapath knows what you want to do.
- ➔ eBPF lets you *create* the datapath instead.

< 2013

David S. Miller et al, https://lore.kernel.org/netdev/20150311.164550.2247000676434784013.davem@davemloft.net/

eBPF - beginning to present

Initial big patch bomb to propose ‘extended BPF’

RFCs caused plenty of discussion, too intrusive & nftables (inspired by BPF) was on the rise

Yet another interpreter for BPF itself that needed to be maintained

Eventually got stuck in discussions, and rejected

2013

Alexei Starovoitov, https://lore.kernel.org/netdev/1380589468-4656-1-git-send-email-ast@plumgrid.com/

eBPF - beginning to present

Initial big patch bomb to propose ‘extended BPF’

RFCs caused plenty of discussion, too intrusive & nftables (inspired by BPF) was on the rise

Yet another interpreter for BPF itself that needed to be maintained

Eventually got stuck in discussions, and rejected

2013

Alexei Starovoitov, https://lore.kernel.org/netdev/1380589468-4656-1-git-send-email-ast@plumgrid.com/

eBPF - beginning to present

First eBPF patch set that was merged into the kernel

Incremental pieces fully **replacing** old BPF interpreter with new one

Instruction set heavily extended, in-kernel translation from old to new

2013 2014

Alexei Starovoitov & Daniel Borkmann, https://lore.kernel.org/netdev/1396029506-16776-1-git-send-email-dborkman@redhat.com/

eBPF - beginning to present

First eBPF patch set that was merged into the kernel

Incremental pieces fully **replacing** old BPF interpreter with new one

Instruction set heavily extended, in-kernel translation from old to new

Later patches exposed it to UAPI, added **verifier** core and **JIT** pieces

Lots of follow-up work on basic infra to fade out old BPF from the core

2013 2014

Alexei Starovoitov & Daniel Borkmann, https://lore.kernel.org/netdev/1396029506-16776-1-git-send-email-dborkman@redhat.com/

eBPF - beginning to present

First eBPF patch set that was merged into the kernel

Incremental pieces fully **replacing** old BPF interpreter with new one

Instruction set heavily extended, in-kernel translation from old to new

Later patches exposed it to UAPI, added **verifier** core and **JIT** pieces

Lots of follow-up work on basic infra to fade out old BPF from the core

We effectively became **maintainers** overseeing BPF development since then

2013 2014

Alexei Starovoitov & Daniel Borkmann, https://lore.kernel.org/netdev/1396029506-16776-1-git-send-email-dborkman@redhat.com/

eBPF - be

First eBPF patch

- Incremental pieces
- Instruction set hea
- Later patches expo
- Lots of follow-up w
- We effectively bec

then

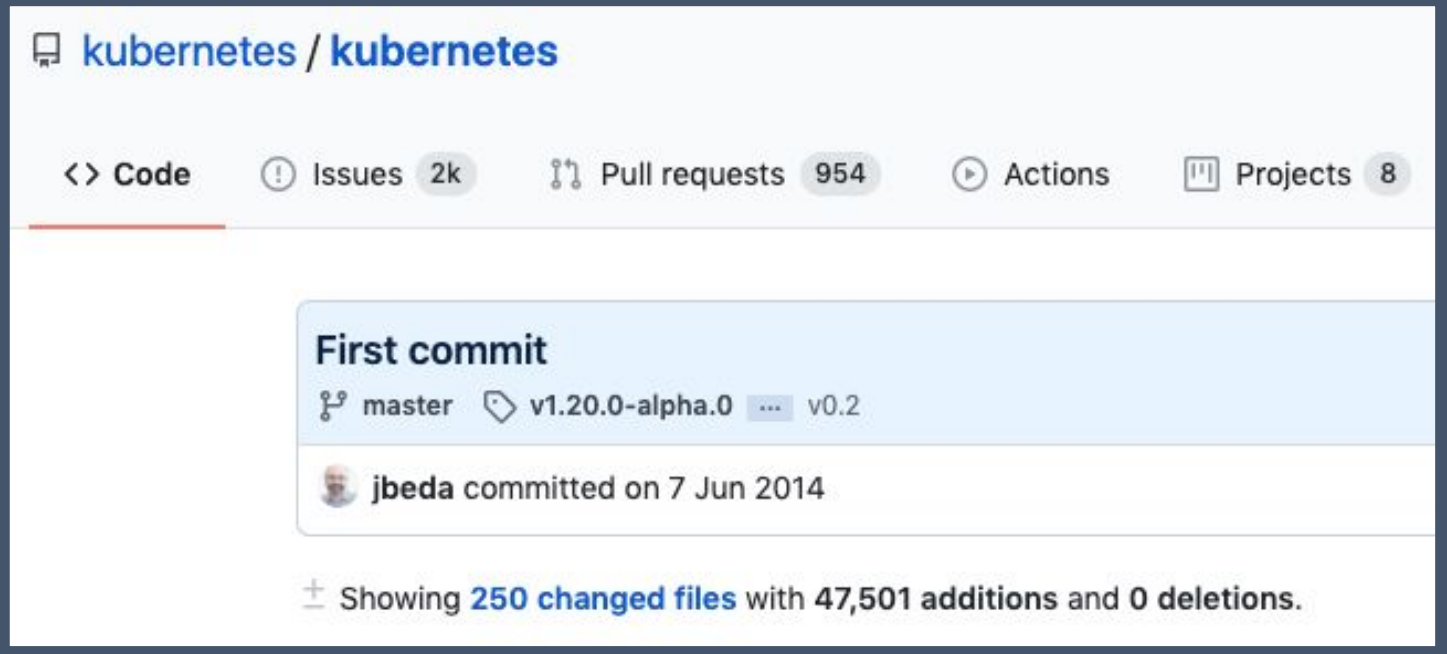
2013 2014

```
From: David Miller <davem@davemloft.net>
To: dborkman@redhat.com
Cc: ast@plumgrid.com, netdev@vger.kernel.org
Subject: Re: [PATCH net-next v4 0/9] BPF updates
Date: Mon, 31 Mar 2014 00:46:16 -0400 (EDT)
Message-ID: <20140331.004616.1557779324959569233.davem@davemloft.net> (raw)
In-Reply-To: <1396029506-16776-1-git-send-email-dborkman@redhat.com>

From: Daniel Borkmann <dborkman@redhat.com>
Date: Fri, 28 Mar 2014 18:58:17 +0100

> 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

Ok, applied, thanks a lot everyone.
```

eBPF - be


```
From: David Miller <davem@davemloft.net>
To: dborkman@redhat.com
```

What else happened in that year?



First eBPF pat

- Incremental pie
- Instruction set
- Later patches e
- Lots of follow
- We effective

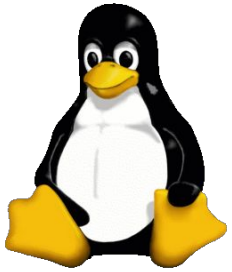
```
> We did extensive testing of BPF and seccomp on the new interpreter
> itself and also on the user ABIs and could not find any issues; new
> performance numbers as posted in patch 8 are also still the same.
>
> Please find more details in the patches themselves.
>
> For all the previous history from 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

Ok, applied, thanks a lot everyone.
```

2013 2014

eBPF - beginning to present

eBPF moved into two directions in parallel: networking & tracing

eBPF backend merged into upstream **LLVM** 3.7



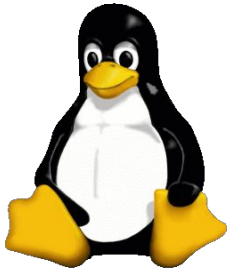
Alexei Starovoitov, https://lore.kernel.org/netdev/1425252465-27527-1-git-send-email-ast@plumgrid.com/
Daniel Borkmann, https://lore.kernel.org/netdev/cover.1425208501.git.daniel@iogearbox.net/
Daniel Borkmann, https://lore.kernel.org/netdev/61198814638d88ce3555dbecf8ef875523b95743.1452197856.git.daniel@iogearbox.net/
Brendan Gregg, http://www.brendangregg.com/blog/2015-09-22/bcc-linux-4.3-tracing.html

eBPF - beginning to present

eBPF moved into two directions in parallel: networking & tracing

eBPF backend merged into upstream **LLVM** 3.7

Ability to attach eBPF to kprobes as first **tracing** use case



Alexei Starovoitov, https://lore.kernel.org/netdev/1425252465-27527-1-git-send-email-ast@plumgrid.com/
Daniel Borkmann, https://lore.kernel.org/netdev/cover.1425208501.git.daniel@iogearbox.net/
Daniel Borkmann, https://lore.kernel.org/netdev/61198814638d88ce3555dbecf8ef875523b95743.1452197856.git.daniel@iogearbox.net/
Brendan Gregg, http://www.brendangregg.com/blog/2015-09-22/bcc-linux-4.3-tracing.html

eBPF - beginning to present

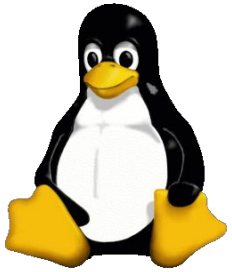
eBPF moved into two directions in parallel: networking & tracing

eBPF backend merged into upstream **LLVM** 3.7

Ability to attach eBPF to kprobes as first **tracing** use case

For **networking**, tc became fully programmable through eBPF via cls_bpf

tc got a lockless ingress & egress attach hook for eBPF



Alexei Starovoitov, https://lore.kernel.org/netdev/1425252465-27527-1-git-send-email-ast@plumgrid.com/
Daniel Borkmann, https://lore.kernel.org/netdev/cover.1425208501.git.daniel@iogearbox.net/
Daniel Borkmann, https://lore.kernel.org/netdev/61198814638d88ce3555dbecf8ef875523b95743.1452197856.git.daniel@iogearbox.net/
Brendan Gregg, http://www.brendangregg.com/blog/2015-09-22/bcc-linux-4.3-tracing.html

eBPF - beginning to present

eBPF moved into two directions in parallel: networking & tracing

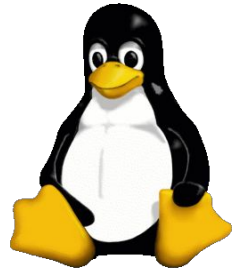
eBPF backend merged into upstream **LLVM** 3.7

Ability to attach eBPF to kprobes as first **tracing** use case

For **networking**, tc became fully programmable through eBPF via cls_bpf

tc got a lockless ingress & egress attach hook for eBPF

Plenty of verifier and eBPF helper work to make tracing and networking side useful



Alexei Starovoitov, https://lore.kernel.org/netdev/1425252465-27527-1-git-send-email-ast@plumgrid.com/
Daniel Borkmann, https://lore.kernel.org/netdev/cover.1425208501.git.daniel@iogearbox.net/
Daniel Borkmann, https://lore.kernel.org/netdev/61198814638d88ce3555dbecf8ef875523b95743.1452197856.git.daniel@iogearbox.net/
Brendan Gregg, http://www.brendangregg.com/blog/2015-09-22/bcc-linux-4.3-tracing.html

eBPF - beginning to present

eBPF moved into two directions in parallel: networking & tracing

eBPF backend merged into upstream **LLVM** 3.7

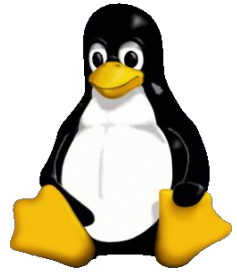
Ability to attach eBPF to kprobes as first **tracing** use case

For **networking**, tc became fully programmable through eBPF via cls_bpf

tc got a lockless ingress & egress attach hook for eBPF

Plenty of verifier and eBPF helper work to make tracing and networking side useful

Initial announcement of **bcc** project as tracing frontend for eBPF



Alexei Starovoitov, https://lore.kernel.org/netdev/1425252465-27527-1-git-send-email-ast@plumgrid.com/
Daniel Borkmann, https://lore.kernel.org/netdev/cover.1425208501.git.daniel@iogearbox.net/
Daniel Borkmann, https://lore.kernel.org/netdev/61198814638d88ce3555dbecf8ef875523b95743.1452197856.git.daniel@iogearbox.net/
Brendan Gregg, http://www.brendangregg.com/blog/2015-09-22/bcc-linux-4.3-tracing.html

eBPF - beginning to present

eBPF received a new networking fast-path, Cilium project first announced

eXpress DataPath (**XDP**) merged for attaching eBPF programs at driver's ingress layer

nfp as first NIC and driver to offload eBPF programs at cls_bpf & XDP hook

Brendan Blanco et al, https://lore.kernel.org/netdev/1468955817-10604-1-git-send-email-bblanco@plumgrid.com/
Thomas Graf et al, https://www.slideshare.net/ThomasGraf5/cilium-fast-ipv6-container-networking-with-bpf-and-xdp
Jakub Kicinski, https://lore.kernel.org/netdev/1478193129-23476-1-git-send-email-jakub.kicinski@netronome.com/

eBPF - beginning to present

eBPF received a new networking fast-path, Cilium project first announced

eXpress DataPath (**XDP**) merged for attaching eBPF programs at driver's ingress layer

nfp as first NIC and driver to offload eBPF programs at cls_bpf & XDP hook

Initial announcement of **Cilium** for container networking, mainly targeted at Docker back then

- Efficient label-based policy, NAT64, tunnel mesh, container connectivity all via eBPF
- Entire datapath & forwarding logic in eBPF, no more Docker or ovs bridge devices

cilium

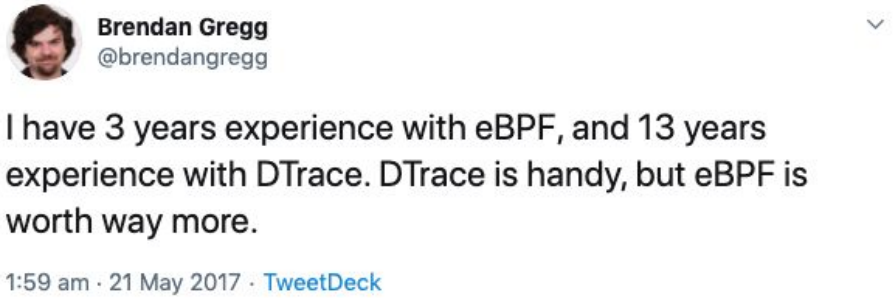
2013 2014 2015 2016

Brendan Blanco et al, https://lore.kernel.org/netdev/1468955817-10604-1-git-send-email-bblanco@plumgrid.com/
Thomas Graf et al, https://www.slideshare.net/ThomasGraf5/cilium-fast-ipv6-container-networking-with-bpf-and-xdp
Jakub Kicinski, https://lore.kernel.org/netdev/1478193129-23476-1-git-send-email-jakub.kicinski@netronome.com/

eBPF - beginning to present

eBPF taking over in production environments

Netflix on eBPF for tracing: ‘Linux BPF superpowers’



Brendan Gregg, https://www.slideshare.net/brendangregg/linux-bpf-superpowers
Nikita Shirokov et al, https://netdevconf.info/2.1/slides/apr6/zhou-netdev-xdp-2017.pdf & http://vger.kernel.org/lpc_net2018_talks/LPC_XDP_Shirokov_v2.pdf
Gilberto Bertin, https://netdevconf.info/2.1/slides/apr6/bertin_Netdev-XDP.pdf &
https://blog.cloudflare.com/cloudflare-architecture-and-how-bpf-eats-the-world/

eBPF - beginning to present

eBPF taking over in production environments

Netflix on eBPF for tracing: ‘Linux BPF superpowers’

Facebook announces XDP + eBPF production use (DDoS & LB)

- Replacing their ipvs load-balancing infrastructure with eBPF for up to 10x performance gain
- eBPF battle-tested: Every packet to facebook.com passes through XDP & eBPF since 2017

Brendan Gregg, https://www.slideshare.net/brendangregg/linux-bpf-superpowers
Nikita Shirokov et al, https://netdevconf.info/2.1/slides/apr6/zhou-netdev-xdp-2017.pdf & http://vger.kernel.org/lpc_net2018_talks/LPC_XDP_Shirokov_v2.pdf
Gilberto Bertin, https://netdevconf.info/2.1/slides/apr6/bertin_Netdev-XDP.pdf &
https://blog.cloudflare.com/cloudflare-architecture-and-how-bpf-eats-the-world/

eBPF - beginning to present

eBPF taking over in production environments

Netflix on eBPF for tracing: ‘Linux BPF superpowers’

Facebook announces XDP + eBPF production use (DDoS & LB)

- Replacing their ipvs load-balancing infrastructure with eBPF for up to 10x performance gain
- eBPF battle-tested: Every packet to facebook.com passes through XDP & eBPF since 2017

Cloudflare integrating XDP + BPF into their DDoS mitigation

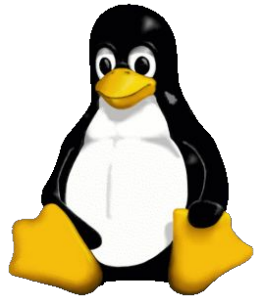
- Successively migrating their components from netfilter over to eBPF
- Fully rolled out into production along with their XDP load-balancer in 2018

Brendan Gregg, https://www.slideshare.net/brendangregg/linux-bpf-superpowers
Nikita Shirokov et al, https://netdevconf.info/2.1/slides/apr6/zhou-netdev-xdp-2017.pdf & http://vger.kernel.org/lpc_net2018_talks/LPC_XDP_Shirokov_v2.pdf
Gilberto Bertin, https://netdevconf.info/2.1/slides/apr6/bertin_Netdev-XDP.pdf &
https://blog.cloudflare.com/cloudflare-architecture-and-how-bpf-eats-the-world/

eBPF - beginning to present

eBPF community and continued feature growth

eBPF becomes its **own kernel subsystem** to ease continuously growing patch management
- We apply patches to bpf & bpf-next kernel trees on git.kernel.org
- Separate bpf kernel mailing list bpf@vger.kernel.org (archive at: lore.kernel.org/bpf/)
- Our pull requests go to Linus Torvalds via David S. Miller (networking maintainer)



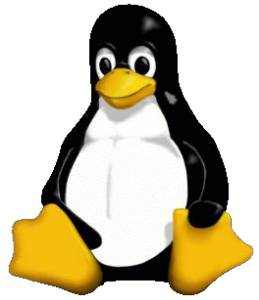
Daniel Borkmann & John Fastabend, http://vger.kernel.org/lpc_net2018_talks/ktls_bpf.pdf
Jakub Kicinski, https://lore.kernel.org/netdev/20170926153522.31500-1-jakub.kicinski@netronome.com/
Alexei Starovoitov, https://lore.kernel.org/netdev/20171215015517.409513-1-ast@kernel.org/

eBPF - beginning to present

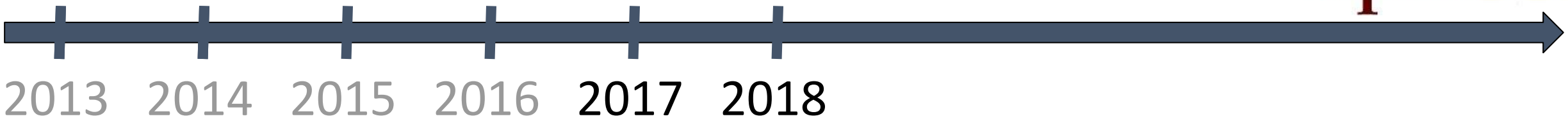
eBPF community and continued feature growth

eBPF becomes its **own kernel subsystem** to ease continuously growing patch management
- We apply patches to bpf & bpf-next kernel trees on git.kernel.org
- Separate bpf kernel mailing list bpf@vger.kernel.org (archive at: lore.kernel.org/bpf/)
- Our pull requests go to Linus Torvalds via David S. Miller (networking maintainer)

kTLS & eBPF for introspection and ability for in-kernel TLS policy enforcement
- Fully integrated and supported by openssl these days



OpenSSL

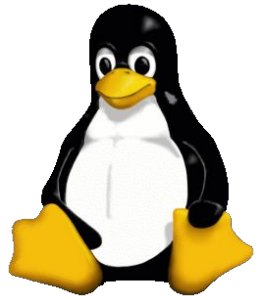


Daniel Borkmann & John Fastabend, http://vger.kernel.org/lpc_net2018_talks/ktls_bpf.pdf
Jakub Kicinski, https://lore.kernel.org/netdev/20170926153522.31500-1-jakub.kicinski@netronome.com/
Alexei Starovoitov, https://lore.kernel.org/netdev/20171215015517.409513-1-ast@kernel.org/

eBPF - beginning to present

eBPF community and continued feature growth

eBPF becomes its **own kernel subsystem** to ease continuously growing patch management
- We apply patches to bpf & bpf-next kernel trees on git.kernel.org
- Separate bpf kernel mailing list bpf@vger.kernel.org (archive at: lore.kernel.org/bpf/)
- Our pull requests go to Linus Torvalds via David S. Miller (networking maintainer)



kTLS & eBPF for introspection and ability for in-kernel TLS policy enforcement
- Fully integrated and supported by openssl these days

bpftool & libbpf for introspection, debugging and user space API for applications

BPF supporting BPF to BPF function calls

Daniel Borkmann & John Fastabend, http://vger.kernel.org/lpc_net2018_talks/ktls_bpf.pdf
Jakub Kicinski, https://lore.kernel.org/netdev/20170926153522.31500-1-jakub.kicinski@netronome.com/
Alexei Starovoitov, https://lore.kernel.org/netdev/20171215015517.409513-1-ast@kernel.org/

eBPF - beginning to present

Cilium 1.0 and eBPF's next big step: BTF (BPF Type Format)

Cilium 1.0: bringing the BPF revolution to K8s networking & security
- K8s CNI integration, identity-based L3-L7 policies, ClusterIP services, ...

2013 2014 2015 2016 2017 **2018**

Thomas Graf et al, https://cilium.io/blog/2018/04/24/cilium-10
Martin KaFai Lau, https://lore.kernel.org/netdev/20180418225606.2771620-1-kafai@fb.com/
Alexei Starovoitov & Daniel Borkmann, http://vger.kernel.org/lpc-bpf2018.html; http://vger.kernel.org/lpc-networking2018.html
Andrii Nakryiko, https://facebookmicrosites.github.io/bpf/blog/2018/11/14/btf-enhancement.html
David S. Miller & Alexei Starovoitov & Daniel Borkmann, https://cilium.io/blog/2018/04/17/why-is-the-kernel-community-replacing-iptables/

eBPF - beginning to present

Cilium 1.0 and eBPF's next big step: BTF (BPF Type Format)

Cilium 1.0: bringing the BPF revolution to K8s networking & security
- K8s CNI integration, identity-based L3-L7 policies, ClusterIP services, ...

BTF as efficient meta data format (100x smaller in size than DWARF)
- Kernel becomes self-descriptive, base for future features like CO-RE, live-patching, global data

Thomas Graf et al, https://cilium.io/blog/2018/04/24/cilium-10
Martin KaFai Lau, https://lore.kernel.org/netdev/20180418225606.2771620-1-kafai@fb.com/
Alexei Starovoitov & Daniel Borkmann, http://vger.kernel.org/lpc-bpf2018.html; http://vger.kernel.org/lpc-networking2018.html
Andrii Nakryiko, https://facebookmicrosites.github.io/bpf/blog/2018/11/14/btf-enhancement.html
David S. Miller & Alexei Starovoitov & Daniel Borkmann, https://cilium.io/blog/2018/04/17/why-is-the-kernel-community-replacing-iptables/

eBPF - beginning to present

Cilium 1.0 and eBPF's next big step: BTF (BPF Type Format)

Cilium 1.0: bringing the BPF revolution to K8s networking & security
- K8s CNI integration, identity-based L3-L7 policies, ClusterIP services, ...

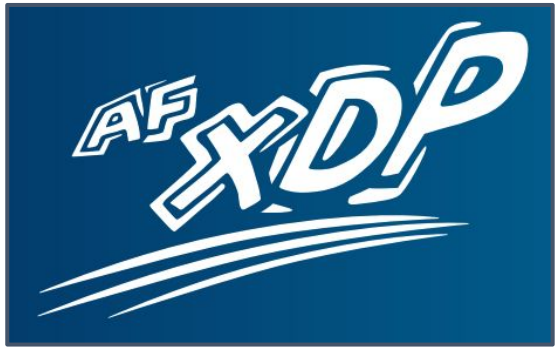
BTF as efficient meta data format (100x smaller in size than DWARF)
- Kernel becomes self-descriptive, base for future features like CO-RE, live-patching, global data

First **Linux Plumbers** BPF conference & half of Networking Track filled with BPF and XDP topics

Thomas Graf et al, https://cilium.io/blog/2018/04/24/cilium-10
Martin KaFai Lau, https://lore.kernel.org/netdev/20180418225606.2771620-1-kafai@fb.com/
Alexei Starovoitov & Daniel Borkmann, http://vger.kernel.org/lpc-bpf2018.html; http://vger.kernel.org/lpc-networking2018.html
Andrii Nakryiko, https://facebookmicrosites.github.io/bpf/blog/2018/11/14/btf-enhancement.html
David S. Miller & Alexei Starovoitov & Daniel Borkmann, https://cilium.io/blog/2018/04/17/why-is-the-kernel-community-replacing-iptables/

eBPF - beginning to present

Cilium 1.0 and eBPF's next big step: BTF (BPF Type Format)



Cilium 1.0: bringing the BPF revolution to K8s networking & security
- K8s CNI integration, identity-based L3-L7 policies, ClusterIP services, ...

BTF as efficient meta data format (100x smaller in size than DWARF)
- Kernel becomes self-descriptive, base for future features like CO-RE, live-patching, global data

First **Linux Plumbers** BPF conference & half of Networking Track filled with BPF and XDP topics

AF_XDP merged into the kernel for DPDK performance at XDP layer with in-kernel drivers

bpfilter prototype to translate iptables rulesets into BPF via user mode driver

Thomas Graf et al, https://cilium.io/blog/2018/04/24/cilium-10
Martin KaFai Lau, https://lore.kernel.org/netdev/20180418225606.2771620-1-kafai@fb.com/
Alexei Starovoitov & Daniel Borkmann, http://vger.kernel.org/lpc-bpf2018.html; http://vger.kernel.org/lpc-networking2018.html
Andrii Nakryiko, https://facebookmicrosites.github.io/bpf/blog/2018/11/14/btf-enhancement.html
David S. Miller & Alexei Starovoitov & Daniel Borkmann, https://cilium.io/blog/2018/04/17/why-is-the-kernel-community-replacing-iptables/

eBPF - beginning to present

bpftrace, first books on eBPF, live-patching and Cilium replacing kube-proxy

bpftrace announced as DTrace 2.0 for Linux

2013 2014 2015 2016 2017 2018 2019

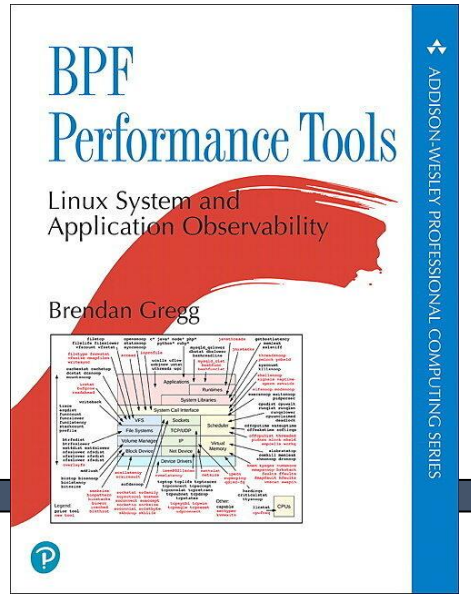
Brendan Gregg, http://www.brendangregg.com/blog/2018-10-08/dtrace-for-linux-2018.html, https://lwn.net/Articles/787131/
Brendan Gregg, http://www.brendangregg.com/bpf-performance-tools-book.html, http://www.brendangregg.com/systems-performance-2nd-edition-book.html
Thomas Graf et al, https://cilium.io/blog/2019/08/20/cilium-16
Alexei Starovoitov & Daniel Borkmann, http://vger.kernel.org/bpfconf2019.html
Daniel Borkmann, https://lore.kernel.org/bpf/cover.1574452833.git.daniel@iogearbox.net/

eBPF - beginning to present

bpftrace, first books on eBPF, live-patching and Cilium replacing kube-proxy

bpftrace announced as DTrace 2.0 for Linux

880-page **book** on BPF tracing from Brendan Gregg (next one on BPF as well for 2020)



2013 2014 2015 2016 2017 2018 2019

Brendan Gregg, http://www.brendangregg.com/blog/2018-10-08/dtrace-for-linux-2018.html, https://lwn.net/Articles/787131/
Brendan Gregg, http://www.brendangregg.com/bpf-performance-tools-book.html, http://www.brendangregg.com/systems-performance-2nd-edition-book.html
Thomas Graf et al, https://cilium.io/blog/2019/08/20/cilium-16
Alexei Starovoitov & Daniel Borkmann, http://vger.kernel.org/bpfconf2019.html
Daniel Borkmann, https://lore.kernel.org/bpf/cover.1574452833.git.daniel@iogearbox.net/

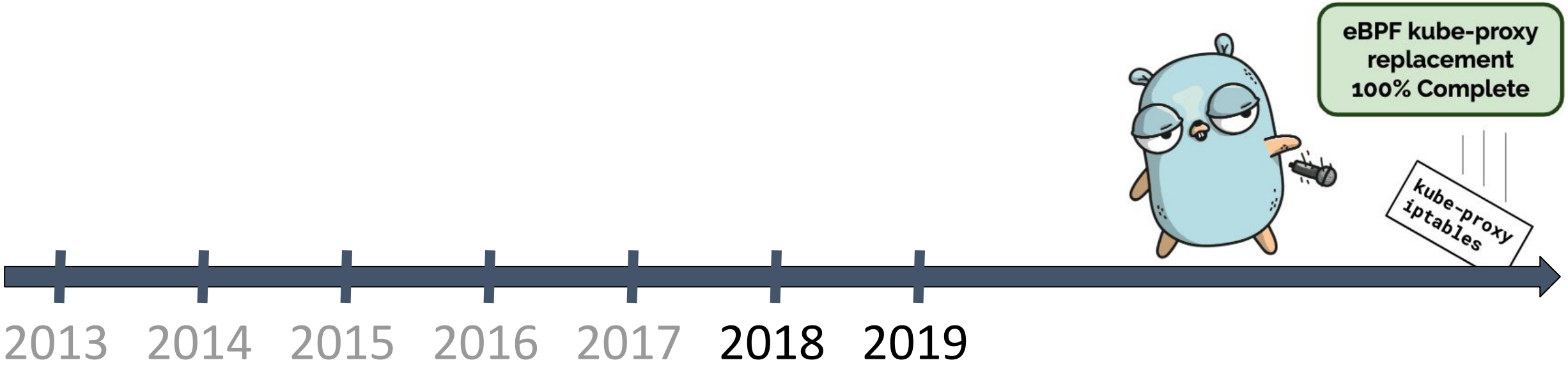
eBPF - beginning to present

bpftrace, first books on eBPF, live-patching and Cilium replacing kube-proxy

bpftrace announced as DTrace 2.0 for Linux

880-page **book** on BPF tracing from Brendan Gregg (next one on BPF as well for 2020)

Cilium 1.6 replacing all of iptables-based kube-proxy through BPF for the first time



Brendan Gregg, http://www.brendangregg.com/blog/2018-10-08/dtrace-for-linux-2018.html, https://lwn.net/Articles/787131/
Brendan Gregg, http://www.brendangregg.com/bpf-performance-tools-book.html, http://www.brendangregg.com/systems-performance-2nd-edition-book.html
Thomas Graf et al, https://cilium.io/blog/2019/08/20/cilium-16
Alexei Starovoitov & Daniel Borkmann, http://vger.kernel.org/bpfconf2019.html
Daniel Borkmann, https://lore.kernel.org/bpf/cover.1574452833.git.daniel@iogearbox.net/

eBPF - beginning to present

bpftrace, first books on eBPF, live-patching and Cilium replacing kube-proxy

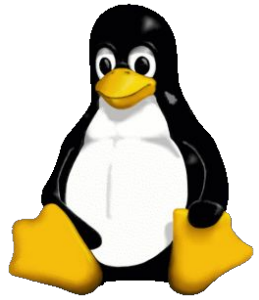
bpftrace announced as DTrace 2.0 for Linux

880-page **book** on BPF tracing from Brendan Gregg (next one on BPF as well for 2020)

Cilium 1.6 replacing all of iptables-based kube-proxy through BPF for the first time

BPF performs **live-patching** on the kernel (tail calls, BPF trampolines, etc)

First invite-only **bpfconf** conference among BPF kernel experts
- Alternating with Linux Plumbers BPF tracks on half-a-year cadence since then



Brendan Gregg, http://www.brendangregg.com/blog/2018-10-08/dtrace-for-linux-2018.html, https://lwn.net/Articles/787131/
Brendan Gregg, http://www.brendangregg.com/bpf-performance-tools-book.html, http://www.brendangregg.com/systems-performance-2nd-edition-book.html
Thomas Graf et al, https://cilium.io/blog/2019/08/20/cilium-16
Alexei Starovoitov & Daniel Borkmann, http://vger.kernel.org/bpfconf2019.html
Daniel Borkmann, https://lore.kernel.org/bpf/cover.1574452833.git.daniel@iogearbox.net/

eBPF - beginning to present

bpftrace, first books on eBPF, live-patching and Cilium replacing kube-proxy

bpftrace announced as DTrace 2.0 for Linux

880-page **book** on BPF tracing from Brendan Gregg (next one on BPF as well for 2020)

Cilium 1.6 replacing all of iptables-based kube-proxy through BPF for the first time

BPF performs **live-patching** on the kernel (tail calls, BPF trampolines, etc)

First invite-only **bpfconf** conference among BPF kernel experts
- Alternating with Linux Plumbers BPF tracks on half-a-year cadence since then

BPF backend for **GCC** finally merged, BPF gets bounded loops support

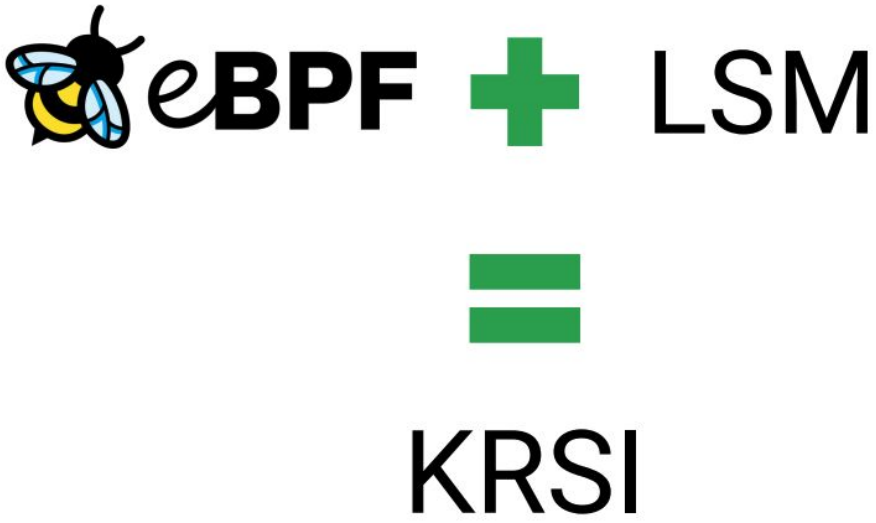
2013 2014 2015 2016 2017 2018 2019

Brendan Gregg, http://www.brendangregg.com/blog/2018-10-08/dtrace-for-linux-2018.html, https://lwn.net/Articles/787131/
Brendan Gregg, http://www.brendangregg.com/bpf-performance-tools-book.html, http://www.brendangregg.com/systems-performance-2nd-edition-book.html
Thomas Graf et al, https://cilium.io/blog/2019/08/20/cilium-16
Alexei Starovoitov & Daniel Borkmann, http://vger.kernel.org/bpfconf2019.html
Daniel Borkmann, https://lore.kernel.org/bpf/cover.1574452833.git.daniel@iogearbox.net/

eBPF - beginning to present

Relentless growth & third major direction for eBPF: Linux security modules

Google upstreaming BPF **LSM** support for their server fleet



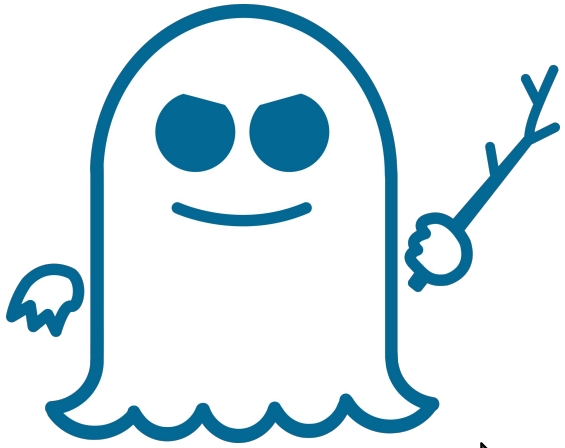
KP Singh, https://lore.kernel.org/bpf/20200329004356.27286-1-kpsingh@chromium.org/
Daniel Borkmann, https://lore.kernel.org/netdev/20190102235835.3311-1-daniel@iogearbox.net/
Thomas Graf et al, https://cilium.io/blog/2020/06/22/cilium-18
Martin KaFai Lau, https://lore.kernel.org/bpf/20200109003453.3854769-1-kafai@fb.com/

eBPF - beginning to present

Relentless growth & third major direction for eBPF: Linux security modules

Google upstreaming BPF **LSM** support for their server fleet

BPF verifier protecting against **Spectre**, even verifying safety on speculative program paths



KP Singh, https://lore.kernel.org/bpf/20200329004356.27286-1-kpsingh@chromium.org/
Daniel Borkmann, https://lore.kernel.org/netdev/20190102235835.3311-1-daniel@iogearbox.net/
Thomas Graf et al, https://cilium.io/blog/2020/06/22/cilium-18
Martin KaFai Lau, https://lore.kernel.org/bpf/20200109003453.3854769-1-kafai@fb.com/

eBPF - beginning to present

Relentless growth & third major direction for eBPF: Linux security modules

Google upstreaming BPF **LSM** support for their server fleet

BPF verifier protecting against **Spectre**, even verifying safety on speculative program paths

XDP support via SRIOV on major **cloud providers**: AWS (ena driver), Azure (hv_netvsc driver), ...

KP Singh, https://lore.kernel.org/bpf/20200329004356.27286-1-kpsingh@chromium.org/
Daniel Borkmann, https://lore.kernel.org/netdev/20190102235835.3311-1-daniel@iogearbox.net/
Thomas Graf et al, https://cilium.io/blog/2020/06/22/cilium-18
Martin KaFai Lau, https://lore.kernel.org/bpf/20200109003453.3854769-1-kafai@fb.com/

eBPF - beginning to present

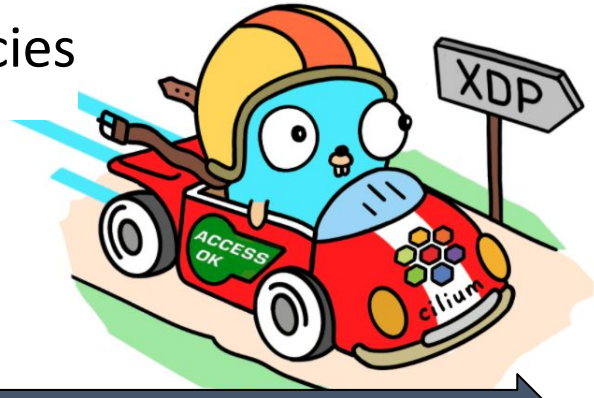
Relentless growth & third major direction for eBPF: Linux security modules

Google upstreaming BPF **LSM** support for their server fleet

BPF verifier protecting against **Spectre**, even verifying safety on speculative program paths

XDP support via SRIOV on major **cloud providers**: AWS (ena driver), Azure (hv_netvsc driver), ...

Cilium 1.8 adds XDP-based load-balancing for services & host network policies



KP Singh, https://lore.kernel.org/bpf/20200329004356.27286-1-kpsingh@chromium.org/
Daniel Borkmann, https://lore.kernel.org/netdev/20190102235835.3311-1-daniel@iogearbox.net/
Thomas Graf et al, https://cilium.io/blog/2020/06/22/cilium-18
Martin KaFai Lau, https://lore.kernel.org/bpf/20200109003453.3854769-1-kafai@fb.com/

eBPF - beginning to present

KubeCon CloudNativeCon Europe 2020 Virtual

Relentless growth & third major dire

Google upstreaming BPF **LSM** support for t

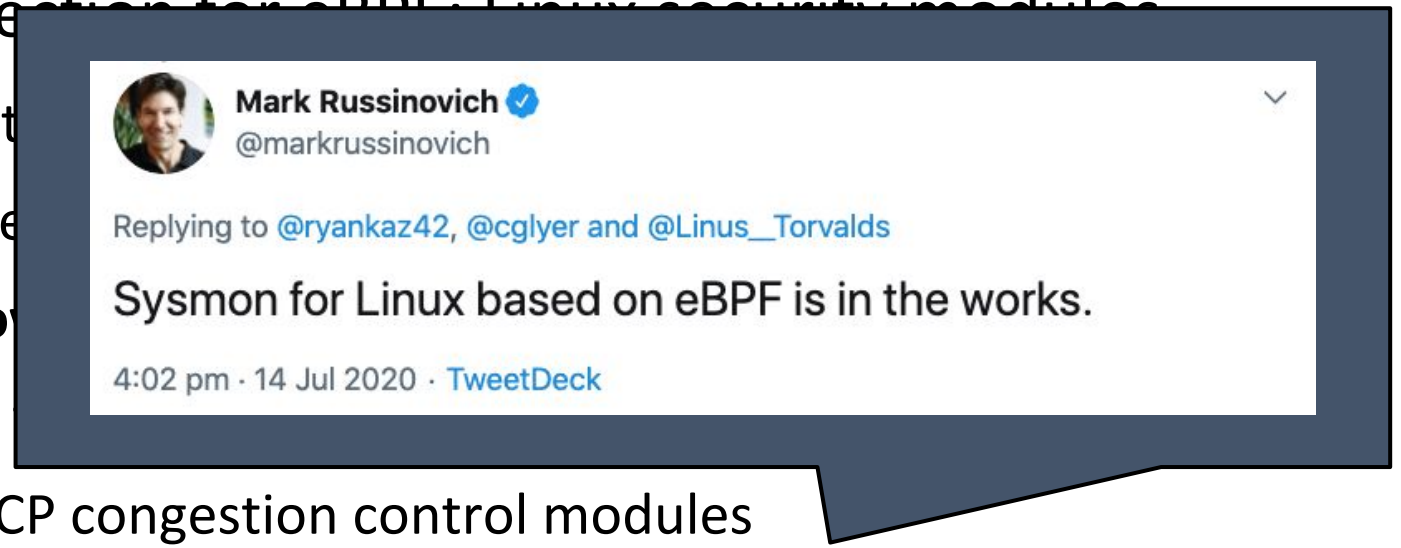
BPF verifier protecting against **Spectre**, eve

XDP support via SRIOV on major **cloud pro**

Cilium 1.8 adds XDP-based load-balancing

Facebook adding support for BPF-based TCP congestion control modules

Microsoft converting their Windows monitoring tools over to Linux based on BPF



KP Singh, https://lore.kernel.org/bpf/20200329004356.27286-1-kpsingh@chromium.org/
Daniel Borkmann, https://lore.kernel.org/netdev/20190102235835.3311-1-daniel@iogearbox.net/
Thomas Graf et al, https://cilium.io/blog/2020/06/22/cilium-18
Martin KaFai Lau, https://lore.kernel.org/bpf/20200109003453.3854769-1-kafai@fb.com/

eBPF - major OS change in 50yrs

KubeCon CloudNativeCon Europe 2020 Virtual

ZDNet

CENTRAL EUROPE MIDDLE EAST SCANDINAVIA AFRICA UK ITALY SPAIN MORE NEWSLETTERS ALL WRITE

MUST READ: Ransomware attack locked a football club's turnstiles

Netflix: BPF is a new type of software we use to run Linux apps securely in the kernel

A Netflix performance architect says BPF promises a fundamental change to a 50-year-old kernel model.

ZDNet, https://www.zdnet.com/article/netflix-bpf-is-a-new-type-of-software-we-use-to-run-linux-apps-securely-in-the-kernel/

eBPF - major OS change in 50yrs

KubeCon CloudNativeCon Europe 2020 Virtual

ZDNet

CENTRAL EUROPE MIDDLE EAST SCANDINAVIA AFRICA UK ITALY SPAIN MORE NEWSLETTERS ALL WRITE

MUST READ: Ransomware attack locked a football club's turnstiles

Netflix: BPF is a new type of software we use to run Linux apps securely in the kernel

A Netflix performance architect says BPF promises a fundamental change to a 50-year-old kernel model.

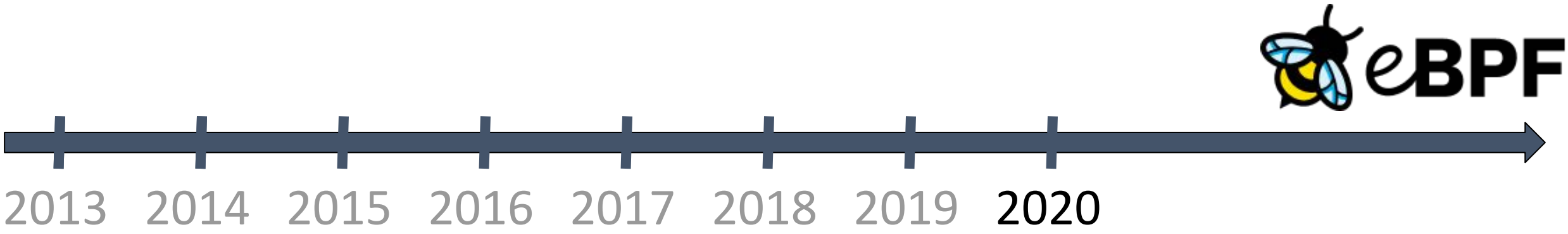
"BPF is the biggest operating systems change I've seen in my career, and it's thrilling to be a part of it," wrote Gregg.

ZDNet, https://www.zdnet.com/article/netflix-bpf-is-a-new-type-of-software-we-use-to-run-linux-apps-securely-in-the-kernel/

eBPF - in numbers (July 2020) ...

eBPF kernel community today:

4,935 patches in total went into BPF subsystem from **347 contributors**

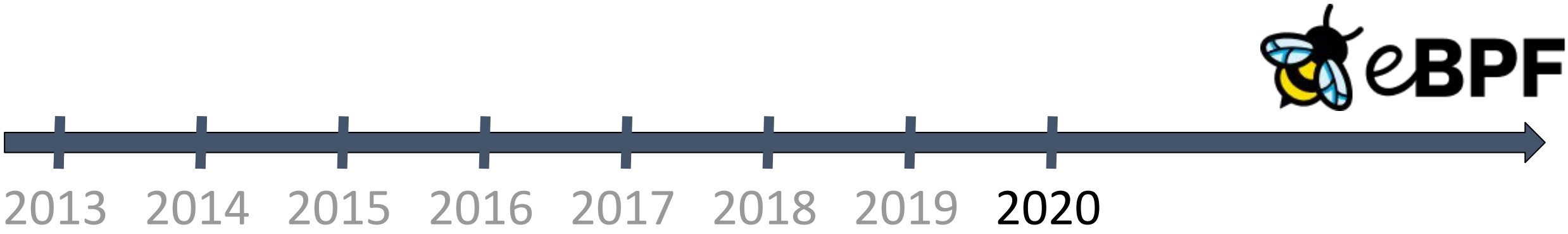


eBPF - in numbers (July 2020) ...

eBPF kernel community today:

4,935 patches in total went into BPF subsystem from **347 contributors**

Around **50 new mails on average every day** on BPF kernel list (often peaking > 100)
- 23,395 mails since mailing list git archive was added in Feb 2019



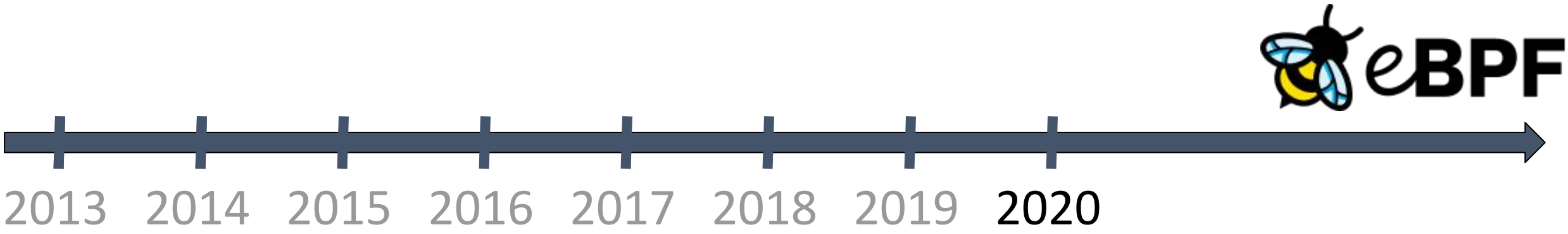
eBPF - in numbers (July 2020) ...

eBPF kernel community today:

4,935 patches in total went into BPF subsystem from **347 contributors**

Around **50 new mails on average every day** on BPF kernel list (often peaking > 100)
- 23,395 mails since mailing list git archive was added in Feb 2019

4 new BPF kernel patches applied every day. Approx 30% of *total* submitted patches applied.



eBPF - in numbers (July 2020) ...

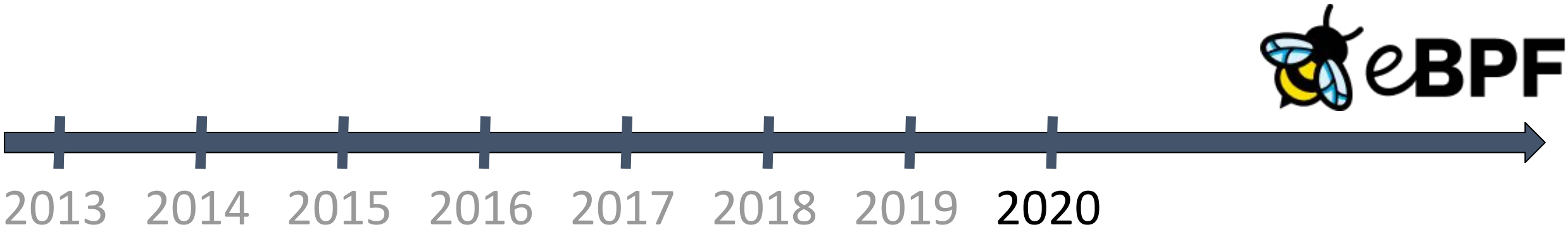
eBPF kernel community today:

4,935 patches in total went into BPF subsystem from **347 contributors**

Around **50 new mails on average every day** on BPF kernel list (often peaking > 100)
- 23,395 mails since mailing list git archive was added in Feb 2019

4 new BPF kernel patches applied every day. Approx 30% of *total* submitted patches applied.

30 different program, 27 different map types, 141 different BPF helpers and **over 3,500 tests**



eBPF - in numbers (July 2020) ...

eBPF kernel community today:

4,935 patches in total went into BPF subsystem from **347 contributors**

Around **50 new mails on average every day** on BPF kernel list (often peaking > 100)
- 23,395 mails since mailing list git archive was added in Feb 2019

4 new BPF kernel patches applied every day. Approx 30% of *total* submitted patches applied.

30 different program, 27 different map types, 141 different BPF helpers and **over 3,500 tests**

2 BPF kernel maintainers & team of 6 senior core reviewers to keep up with the patch load
- Team comprises major contributors from **Isovalent**, **Facebook** and **Google**

eBPF - in numbers (July 2020) ...

eBPF kernel community today:

4,935 patches in total went into BPF subsystem from **347 contributors**

Around **50 new mails on average every day** on BPF kernel list (often peaking > 100)
- 23,395 mails since mailing list git archive was added in Feb 2019

4 new BPF kernel patches applied every day. Approx 30% of *total* submitted patches applied.

30 different program, 27 different map types, 141 different BPF helpers and **over 3,500 tests**

2 BPF kernel maintainers & team of 6 senior core reviewers to keep up with the patch load
- Team comprises major contributors from **Isovalent**, **Facebook** and **Google**

One of the fastest growing subsystem in the Linux kernel ...

eBPF - the industry shift

eBPF

347 contributors (Jan 2014 to Jul 2020):

- 588 Daniel Borkmann (Isovalent; maintainer)
- 421 Andrii Nakryiko (Facebook)
- 401 Alexei Starovoitov (Facebook; maintainer)
- 224 Yonghong Song (Facebook)
- 209 Jakub Kicinski (Facebook)
- 183 Martin KaFai Lau (Facebook)
- 179 Stanislav Fomichev (Google)
- 165 John Fastabend (Isovalent)
- 161 Quentin Monnet (Isovalent)
- 130 Jesper Dangaard Brouer (Red Hat)
- 117 Andrey Ignatov (Facebook)
- [...]

Large scale BPF production users:

Bringing eBPF revolution to K8s

Bringing eBPF revolution to K8s

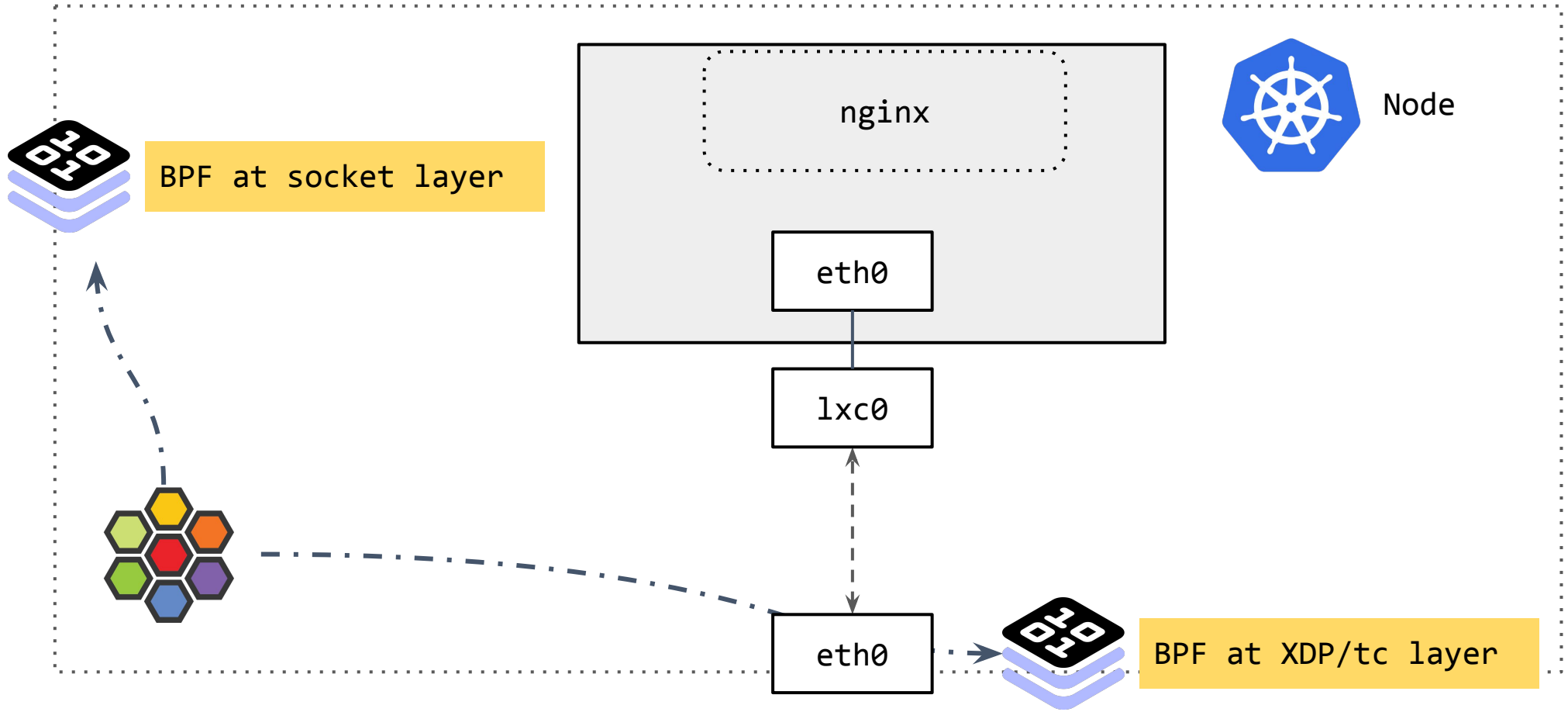
KubeCon CloudNativeCon Europe 2020 Virtual


```
:OUTPUT ACCEPT [2893:671491]
:DOCKER - [0:0]
:DOCKER-ISOLATION-STAGE-1 - [0:0]
:DOCKER-ISOLATION-STAGE-2 - [0:0]
:DOCKER-USER - [0:0]
:KUBE-EXTERNAL-SERVICES - [0:0]
:KUBE-FIREWALL - [0:0]
:KUBE-FORWARD - [0:0]
:KUBE-SERVICES - [0:0]
-A INPUT -m conntrack --ctstate NEW -m comment --comment "kubernetes service portals" -j KUBE-SERVICES
-A INPUT -m conntrack --ctstate NEW -m comment --comment "kubernetes externally-visible service portals" -j KUBE-EXTERNAL-SERVICES
-A INPUT -j KUBE-FIREWALL
-A FORWARD -m comment --comment "kubernetes forwarding rules" -j KUBE-FORWARD
-A FORWARD -m conntrack --ctstate NEW -m comment --comment "kubernetes service portals" -j KUBE-SERVICES
-A FORWARD -j DOCKER-USER
-A FORWARD -j DOCKER-ISOLATION-STAGE-1
-A FORWARD -o docker0 -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -o docker0 -j DOCKER
-A FORWARD -i docker0 ! -o docker0 -j ACCEPT
-A FORWARD -i docker0 -o docker0 -j ACCEPT
-A FORWARD -o docker_gwbridge -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -o docker_gwbridge -j DOCKER
-A FORWARD -i docker_gwbridge ! -o docker_gwbridge -j ACCEPT
-A FORWARD -i docker_gwbridge -o docker_gwbridge -j DROP
-A OUTPUT -m conntrack --ctstate NEW -m comment --comment "kubernetes service portals" -j KUBE-SERVICES
-A OUTPUT -j KUBE-FIREWALL
-A DOCKER-ISOLATION-STAGE-1 -i docker0 ! -o docker0 -j DOCKER-ISOLATION-STAGE-2
-A DOCKER-ISOLATION-STAGE-1 -i docker_gwbridge ! -o docker_gwbridge -j DOCKER-ISOLATION-STAGE-2
-A DOCKER-ISOLATION-STAGE-1 -j RETURN
-A DOCKER-ISOLATION-STAGE-2 -o docker0 -j DROP
-A DOCKER-ISOLATION-STAGE-2 -o docker_gwbridge -j DROP
-A DOCKER-ISOLATION-STAGE-2 -j RETURN
-A DOCKER-USER -j RETURN
-A KUBE-FIREWALL -m comment --comment "kubernetes firewall for dropping marked packets" -m mark --mark 0x8000/0x8000 -j DROP
-A KUBE-FORWARD -m conntrack --ctstate INVALID -j DROP
-A KUBE-FORWARD -m comment --comment "kubernetes forwarding rules" -m mark --mark 0x4000/0x4000 -j ACCEPT
-A KUBE-FORWARD -s 10.217.0.0/16 -m comment --comment "kubernetes forwarding conntrack pod source rule" -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
-A KUBE-FORWARD -d 10.217.0.0/16 -m comment --comment "kubernetes forwarding conntrack pod destination rule" -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
-A KUBE-SERVICES -d 10.99.38.155/32 -p tcp -m comment --comment "default/nginx-59: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.96.61.252/32 -p tcp -m comment --comment "default/nginx-64: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.104.166.10/32 -p tcp -m comment --comment "default/nginx-67: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.98.85.41/32 -p tcp -m comment --comment "default/nginx-9: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.97.138.144/32 -p tcp -m comment --comment "default/nginx-17: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.106.49.80/32 -p tcp -m comment --comment "default/nginx-37: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.104.164.205/32 -p tcp -m comment --comment "default/nginx-5: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.104.25.150/32 -p tcp -m comment --comment "default/nginx-19: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.106.234.213/32 -p tcp -m comment --comment "default/nginx-88: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.109.209.136/32 -p tcp -m comment --comment "default/nginx-33: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.106.196.105/32 -p tcp -m comment --comment "default/nginx-49: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.111.101.6/32 -p tcp -m comment --comment "default/nginx-53: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.110.226.230/32 -p tcp -m comment --comment "default/nginx-79: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.98.99.136/32 -p tcp -m comment --comment "default/nginx-6: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.99.75.233/32 -p tcp -m comment --comment "default/nginx-7: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.108.41.202/32 -p tcp -m comment --comment "default/nginx-14: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.97.36.249/32 -p tcp -m comment --comment "default/nginx-99: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.98.213.37/32 -p tcp -m comment --comment "default/nginx-77: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.107.229.31/32 -p tcp -m comment --comment "default/nginx-92: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.98.64.251/32 -p tcp -m comment --comment "default/nginx-16: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.101.88.159/32 -p tcp -m comment --comment "default/nginx-31: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.105.71.74/32 -p tcp -m comment --comment "default/nginx-41: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.108.92.226/32 -p tcp -m comment --comment "default/nginx-63: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.109.252.234/32 -p tcp -m comment --comment "default/nginx-18: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.104.118.66/32 -p tcp -m comment --comment "default/nginx-30: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.106.224.55/32 -p tcp -m comment --comment "default/nginx-83: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.109.16.199/32 -p tcp -m comment --comment "default/nginx-100: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.109.231.213/32 -p tcp -m comment --comment "default/nginx-61: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.98.27.250/32 -p tcp -m comment --comment "default/nginx-95: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.105.42.108/32 -p tcp -m comment --comment "default/nginx-12: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.99.35.236/32 -p tcp -m comment --comment "default/nginx-20: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.111.42.123/32 -p tcp -m comment --comment "default/nginx-21: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.99.47.225/32 -p tcp -m comment --comment "default/nginx-22: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
-A KUBE-SERVICES -d 10.104.184.242/32 -p tcp -m comment --comment "default/nginx-51: has no endpoints" -m tcp --dport 80 -j REJECT --reject-with icmp-port-unreachable
```

Example: s/kube-proxy/eBPF/

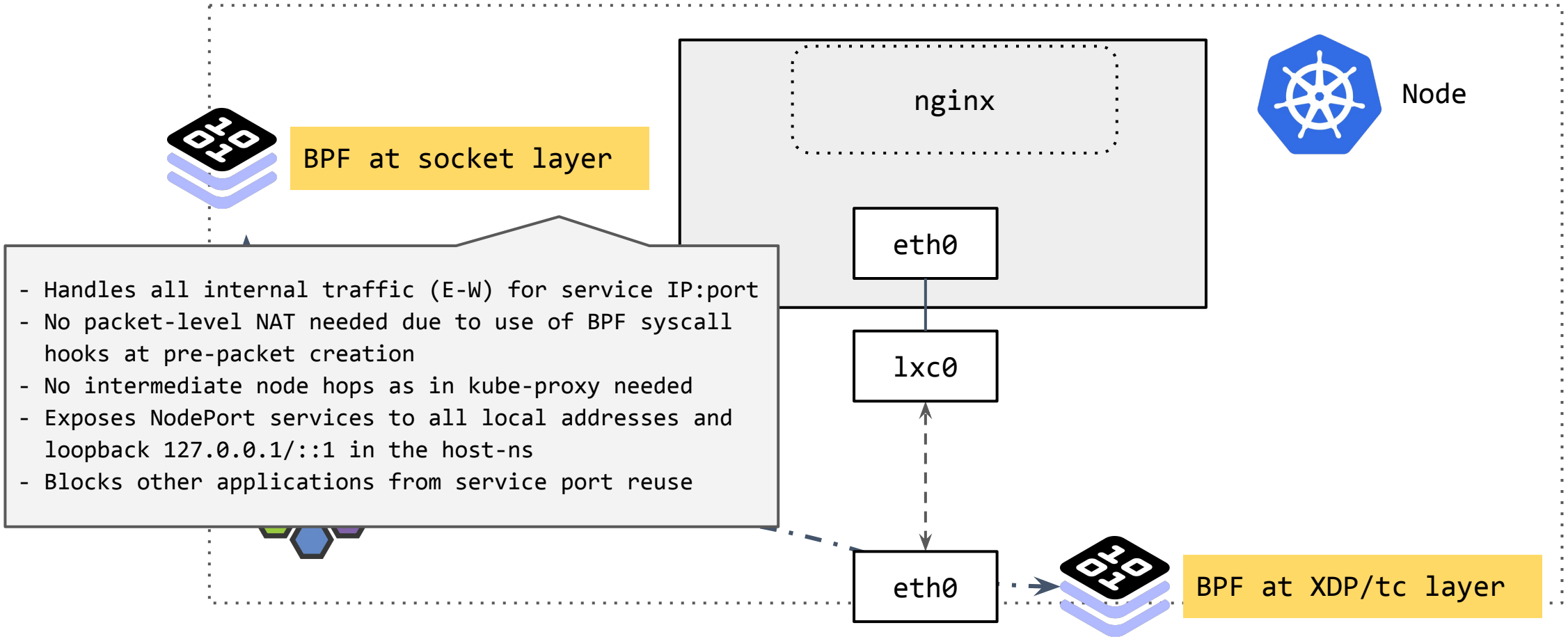

```
$ kubectl -n kube-system delete ds kube-proxy
```

Cilium's service load balancing



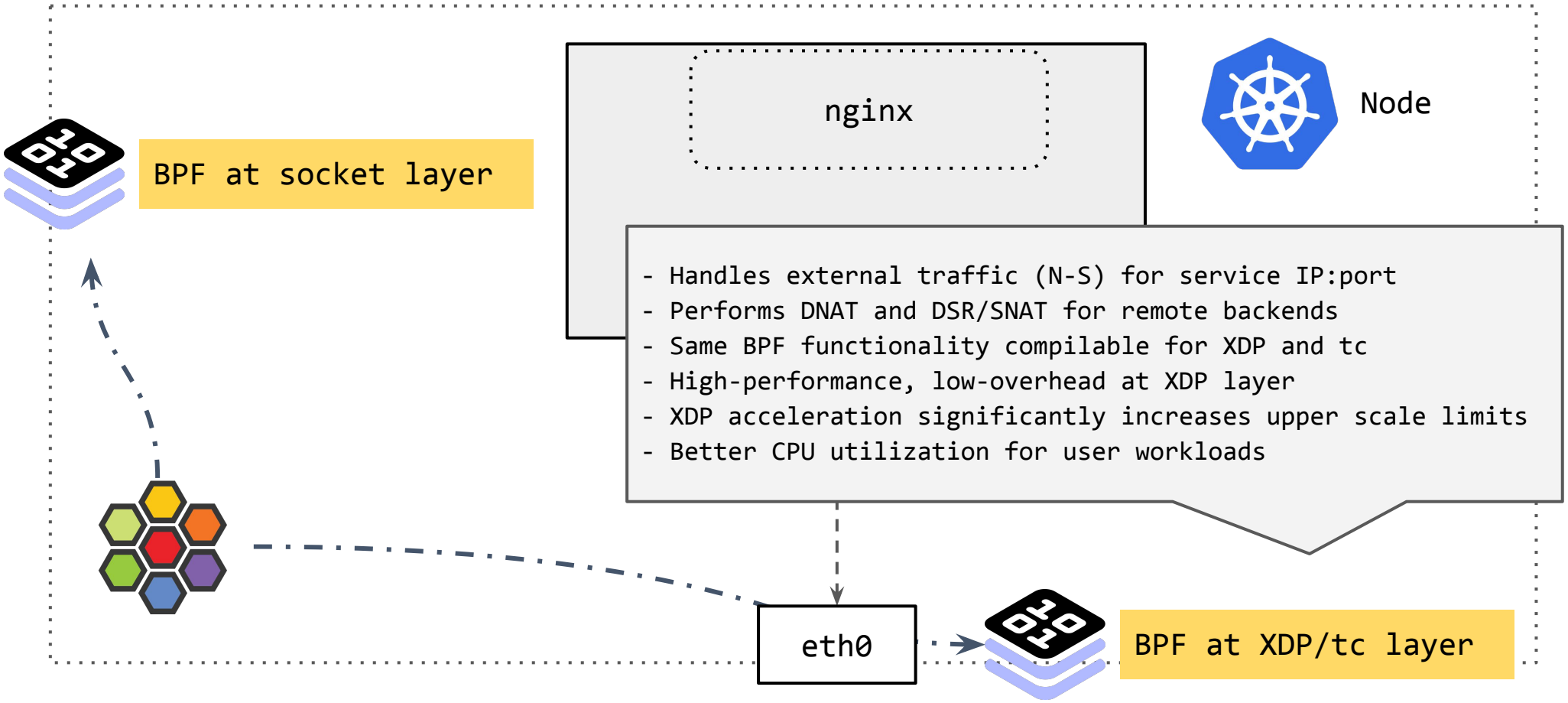
Main principle: Operating as close as possible to the socket for E-W and as close as possible to the driver for N-S.

Cilium's service load balancing



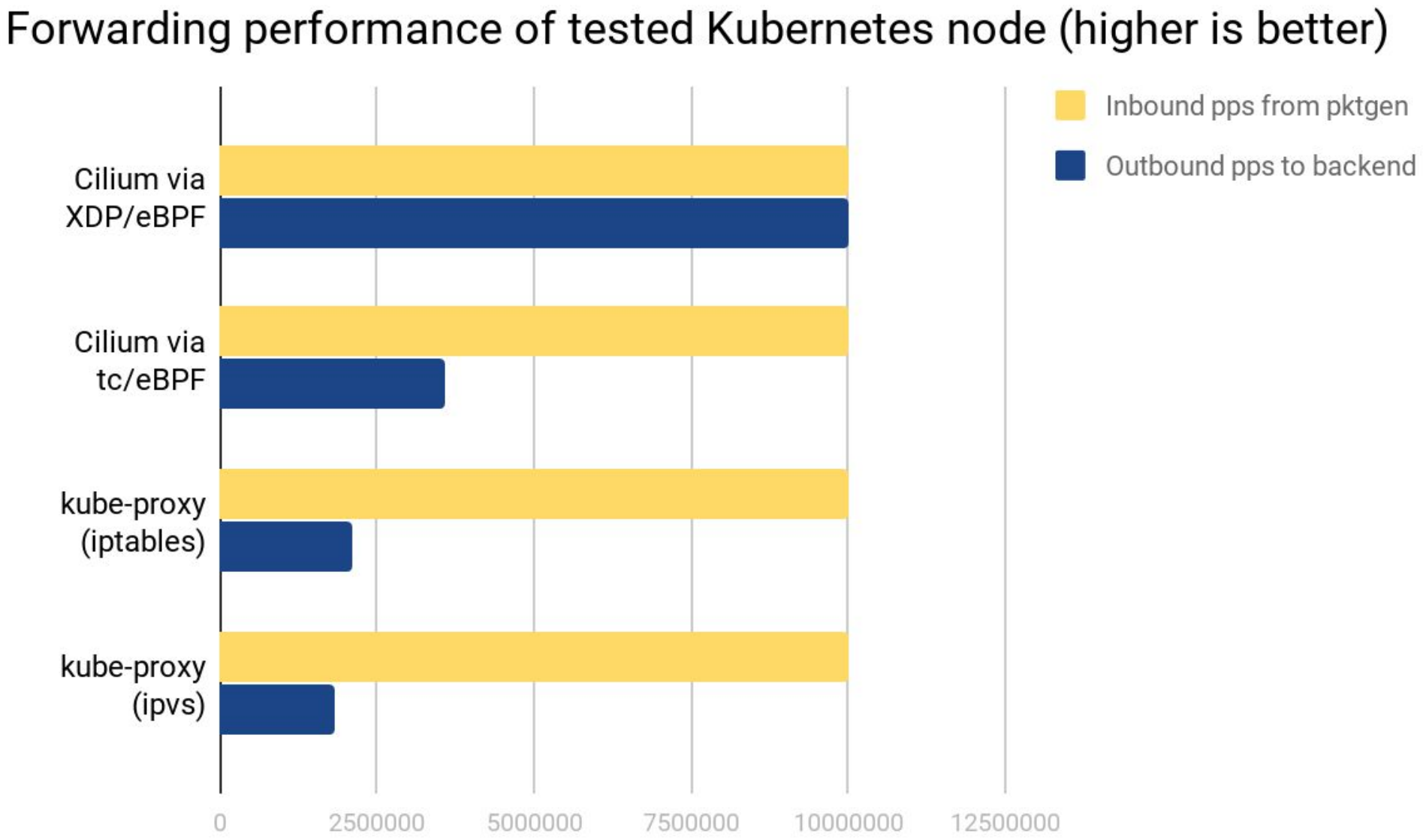
Main principle: Operating as close as possible to the socket for E-W and as close as possible to the driver for N-S.

Cilium's service load balancing

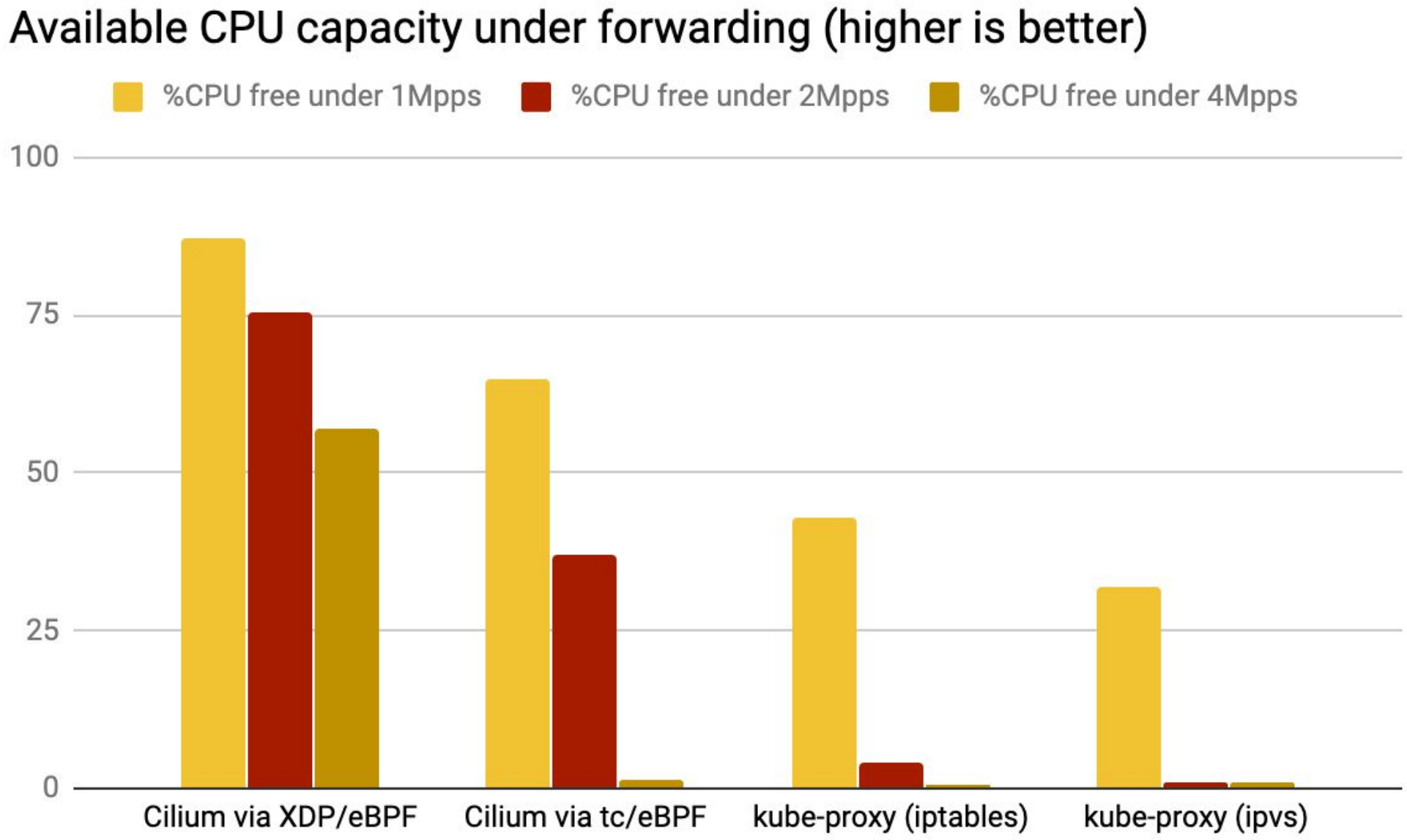


Main principle: Operating as close as possible to the socket for E-W and as close as possible to the driver for N-S.

XDP & eBPF vs kube-proxy



XDP & eBPF vs kube-proxy



eBPF & Kubernetes - the future

“The Linux kernel continues its march towards becoming BPF runtime-powered microkernel.”

Tiny core kernel with user definable kernel functionality in BPF (instead of kernel modules)

Toke Høiland-Jørgensen, https://lwn.net/ml/bufferbloat/87bls8bnsm.fsf@toke.dk/

eBPF & Kubernetes - the future

KubeCon CloudNativeCon Europe 2020 Virtual

“The Linux kernel continues its march towards becoming BPF runtime-powered microkernel.”

Tiny core kernel with user definable kernel functionality in BPF (instead of kernel modules)

Less security bugs & kernel crashes due to **smaller attack surface** and **safety-verified code**

Drastic reduction of ‘static’ feature creep for **better resource efficiency**

Toke Høiland-Jørgensen, https://lwn.net/ml/bufferbloat/87bls8bnsm.fsf@toke.dk/

eBPF & Kubernetes - the future

KubeCon CloudNativeCon Europe 2020 Virtual

“The Linux kernel continues its march towards becoming BPF runtime-powered microkernel.”

Tiny core kernel with user definable kernel functionality in BPF (instead of kernel modules)

Less security bugs & kernel crashes due to **smaller attack surface** and **safety-verified code**

Drastic reduction of ‘static’ feature creep for **better resource efficiency**

Kubernetes then ships **custom BPF**-tailored extensions to optimize needs for user workloads

Toke Høiland-Jørgensen, https://lwn.net/ml/bufferbloat/87bls8bnsm.fsf@toke.dk/

eBPF & Kubernetes - the future

KubeCon CloudNativeCon Europe 2020 Virtual

“The Linux kernel continues its march towards becoming BPF runtime-powered microkernel.”

Tiny core kernel with user definable kernel functionality in BPF (instead of kernel modules)

Less security bugs & kernel crashes due to **smaller attack surface** and **safety-verified code**

Drastic reduction of ‘static’ feature creep for **better resource efficiency**

Kubernetes then ships **custom BPF**-tailored extensions to optimize needs for user workloads

Today’s kube-proxy replacement through BPF is just a tiny dot in that universe ...

Toke Høiland-Jørgensen, https://lwn.net/ml/bufferbloat/87bls8bnsm.fsf@toke.dk/

Thanks a lot! Questions?

KubeCon CloudNativeCon Europe 2020 Virtual

- ➔ Pod-to-Pod Network Connectivity (CNI)
- ➔ Service-based load balancing
- ➔ Security Enforcement (NetworkPolicy)
- ➔ Transparent Encryption
- ➔ Observability, Metrics & Troubleshooting

Try it out: https://cilium.link/kubeproxy-free

Contribute: https://github.com/cilium/cilium

See also: Hubble - eBPF Based Observability for Kubernetes – Sebastian Wicki


KubeCon
CloudNativeCon
Europe 2020
Virtual
HELM
KEEP CLOUD NATIVE
CONNECTED