

OBSERVABILITY SUMMIT EUROPE 2026

# Where did the CPU go?



**Nayef Ghattas**

Staff Engineer, Profiling/OpenTelemetry

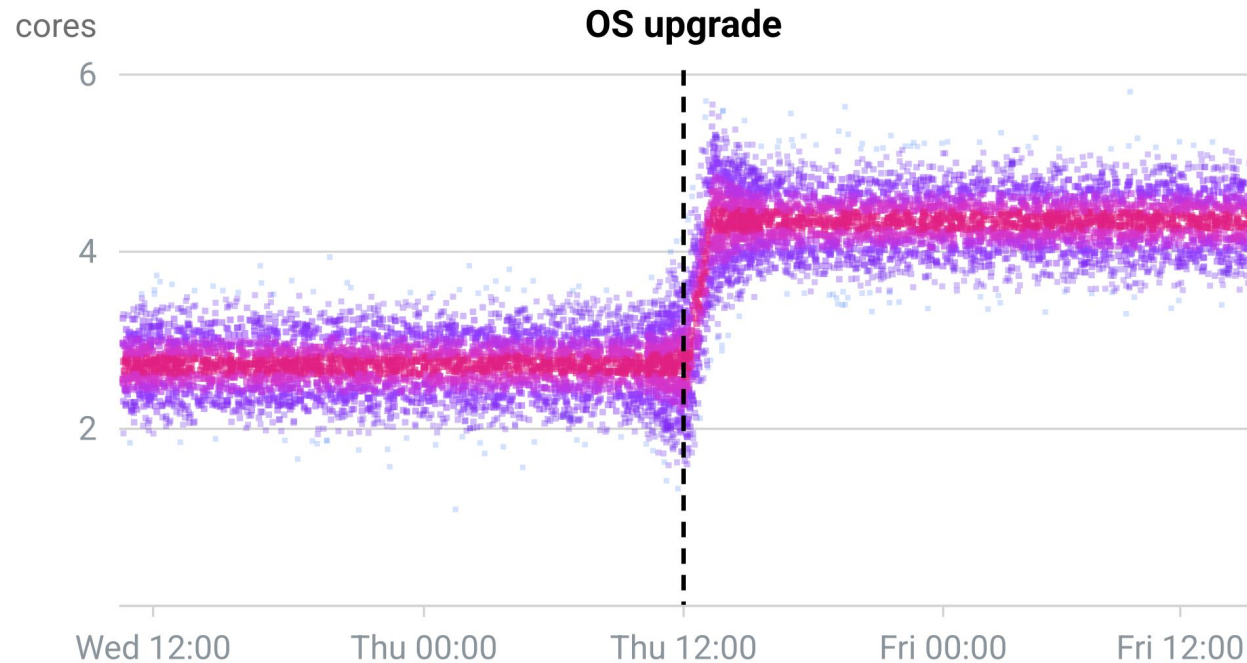


**DATADOG**

# A routine OS upgrade

CPU usage per pod

 service:web-api



**+70%**

**MORE CPU**

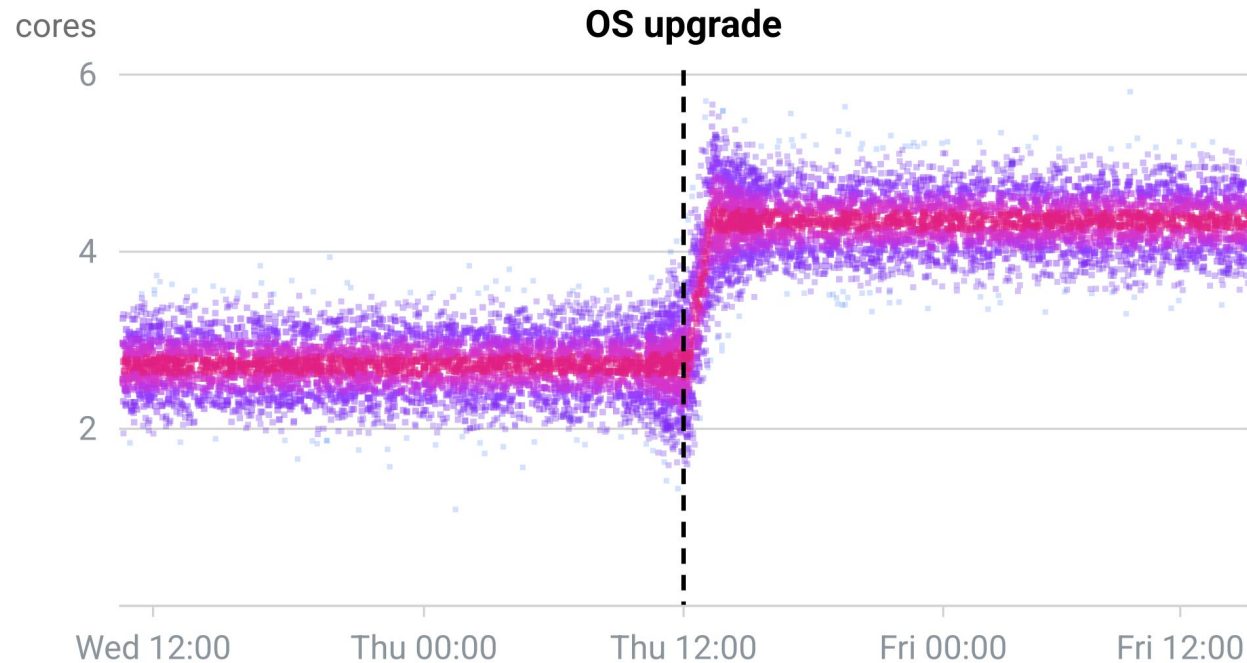
Same code

Same traffic

# A routine OS upgrade

CPU usage per pod

 service:web-api



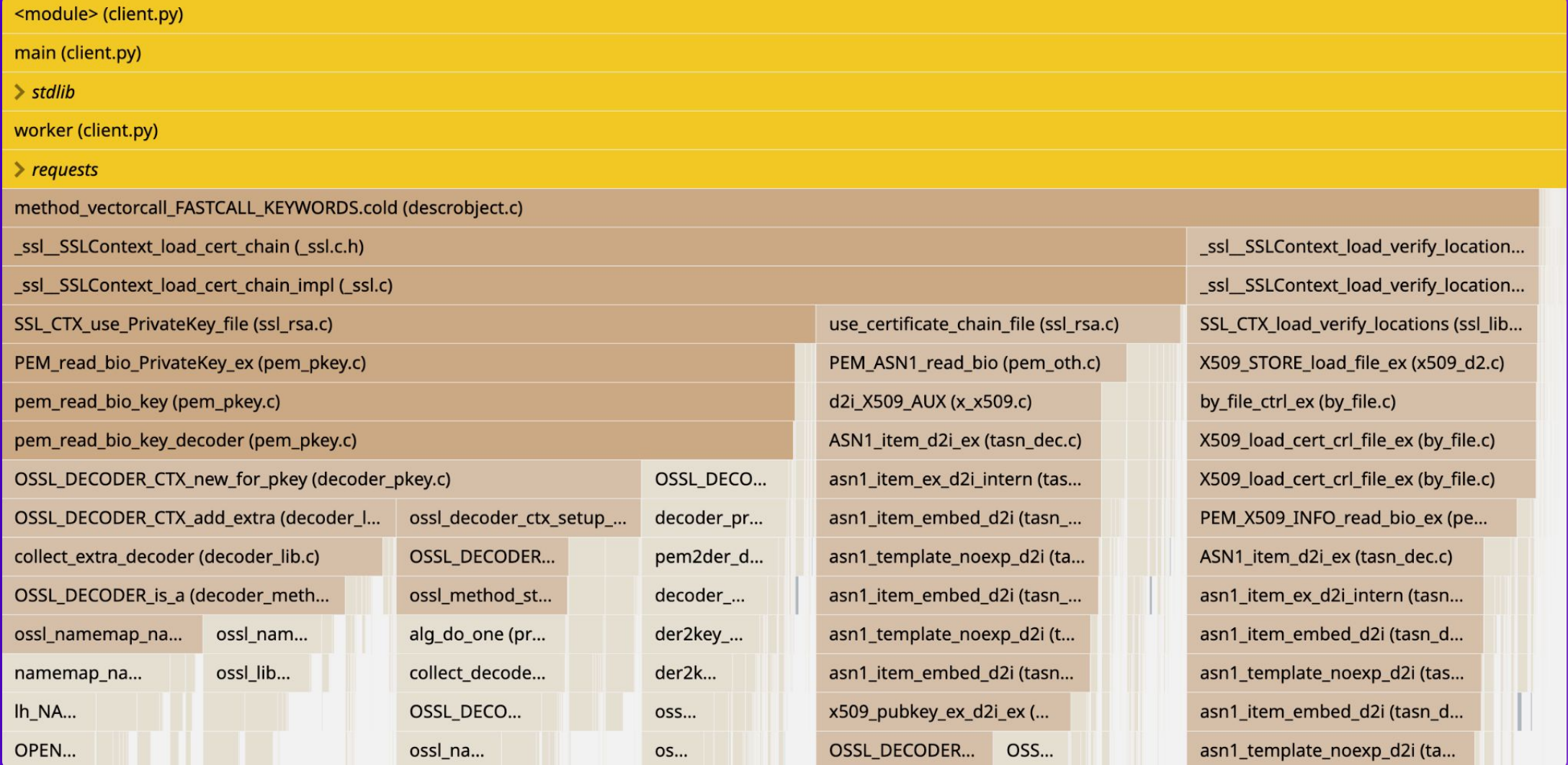
**+70%**  
MORE CPU

Same code  
Same traffic

**+\$3M**  
YEARLY



# A Profiling Flame Graph



# A Profiling Flame Graph

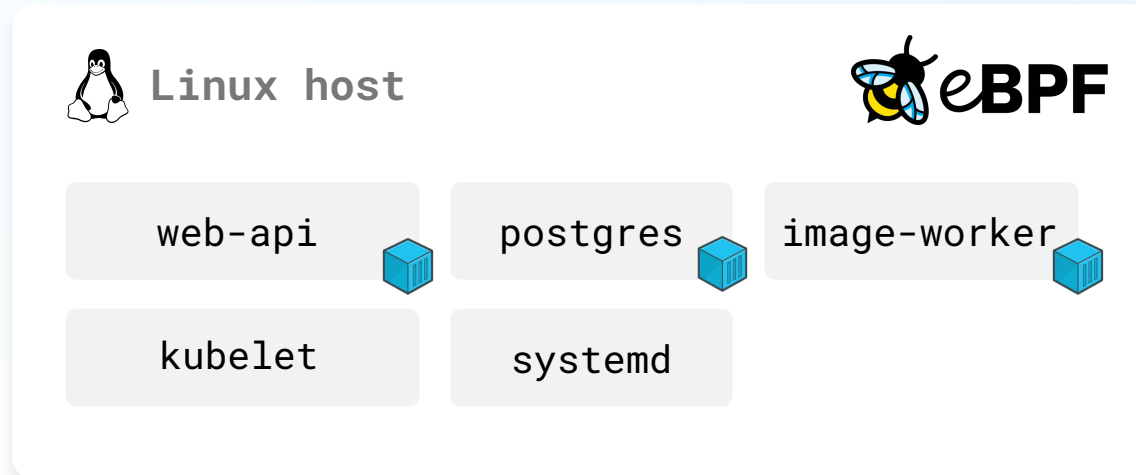


# The OpenTelemetry eBPF Profiler

## A COLLECTOR DISTRIBUTION

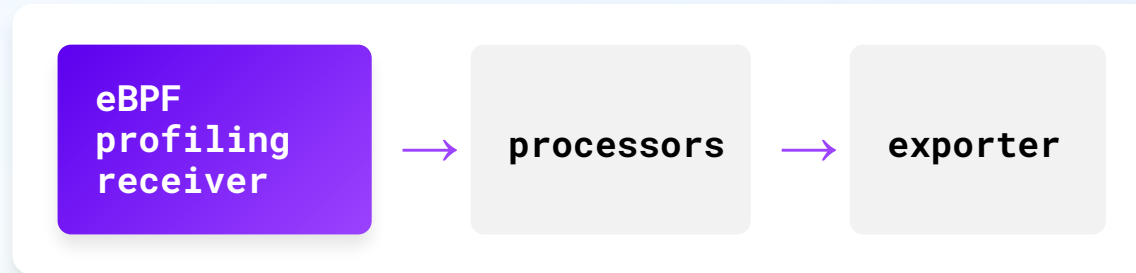


## RUNS ON

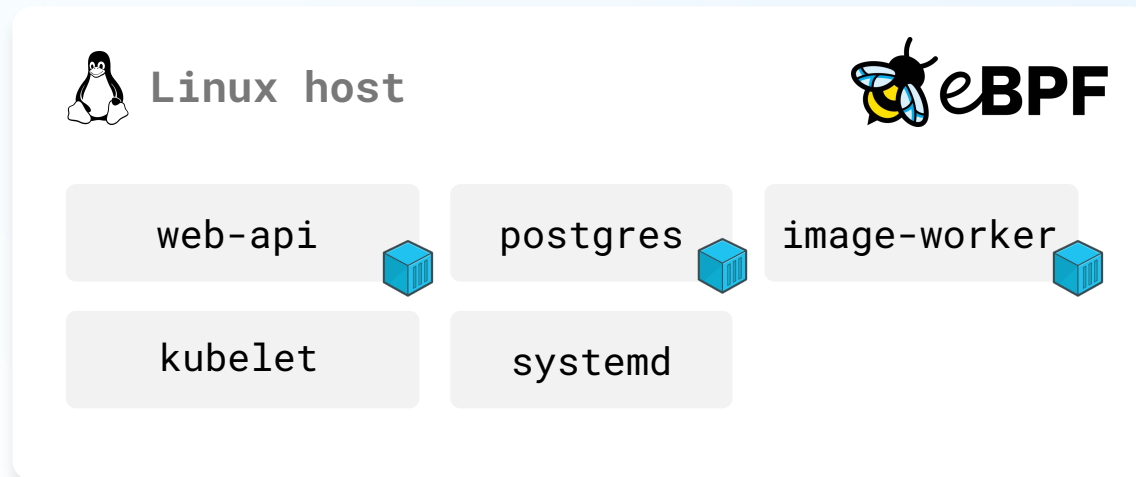


# The OpenTelemetry eBPF Profiler

## A COLLECTOR DISTRIBUTION

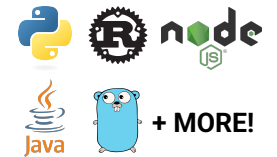


## RUNS ON



**Whole-system visibility**

**Multiple languages**

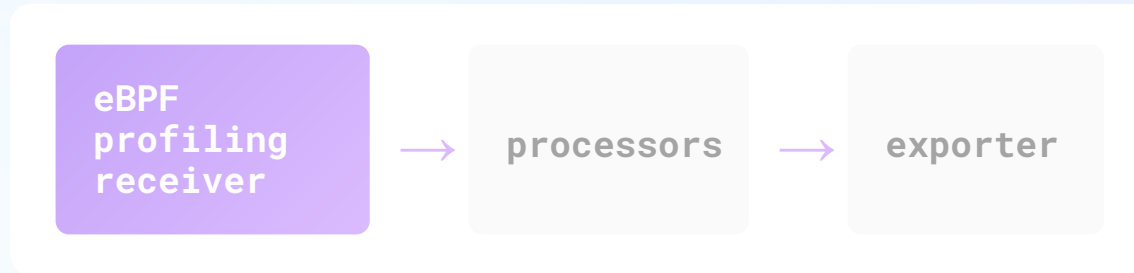


**Zero-instrumentation**

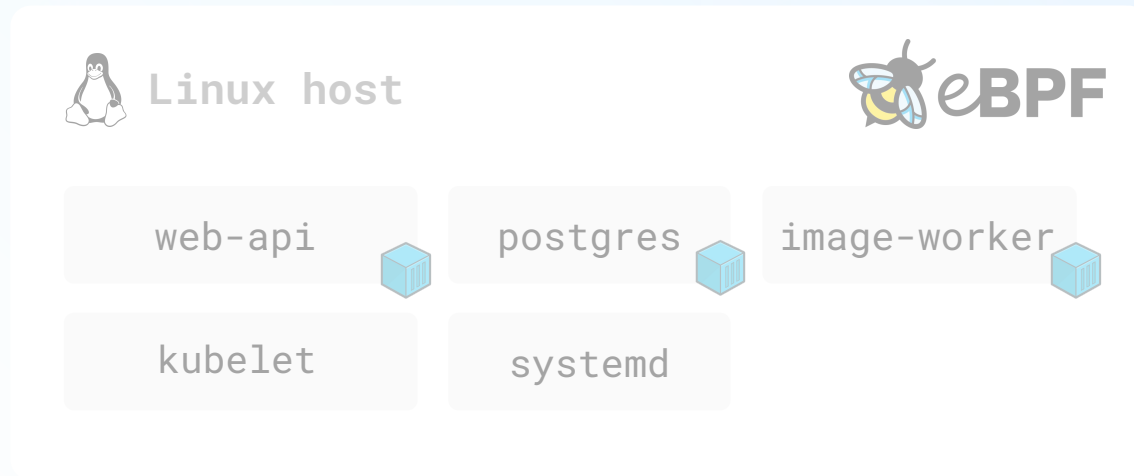
**Low overhead (<1% CPU)**

# The OpenTelemetry Profiles Signal

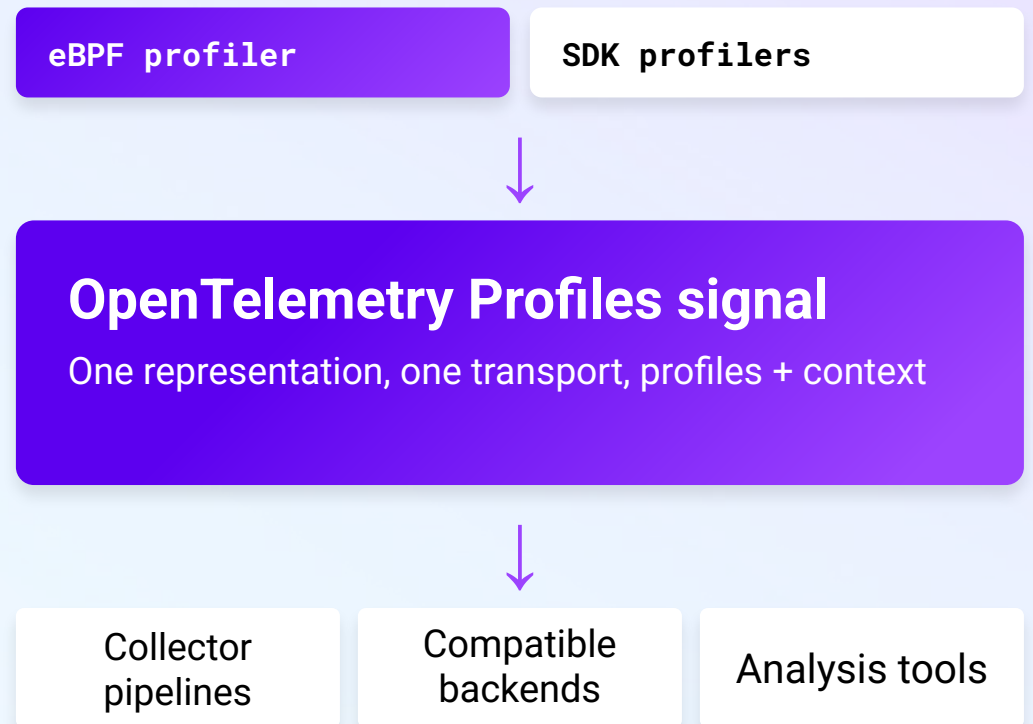
## A COLLECTOR DISTRIBUTION



## RUNS ON

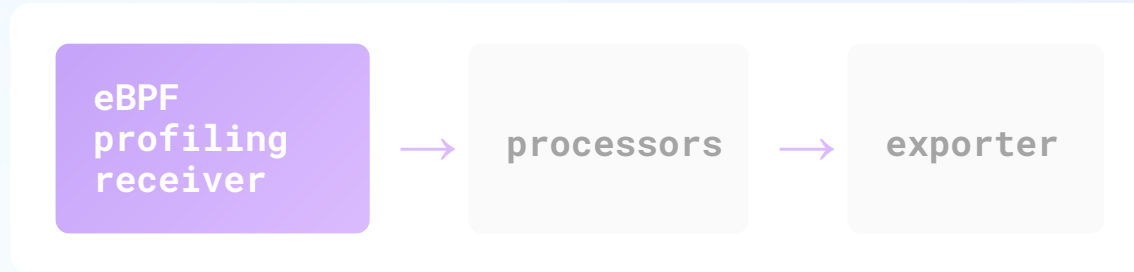


## ONE SIGNAL, MANY CONSUMERS

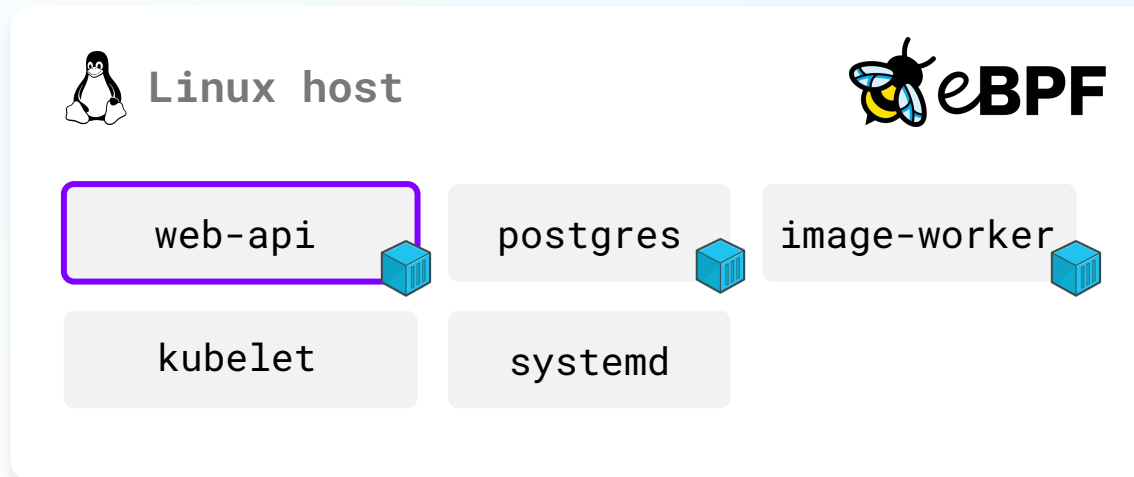


# The OpenTelemetry Profiles Signal

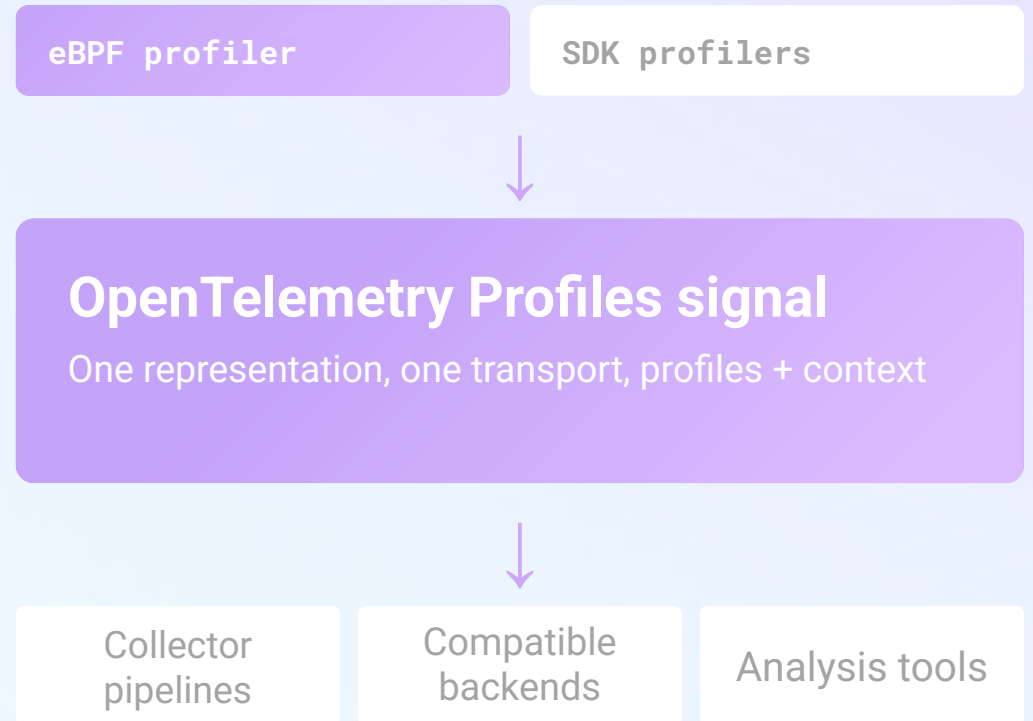
## A COLLECTOR DISTRIBUTION



## RUNS ON

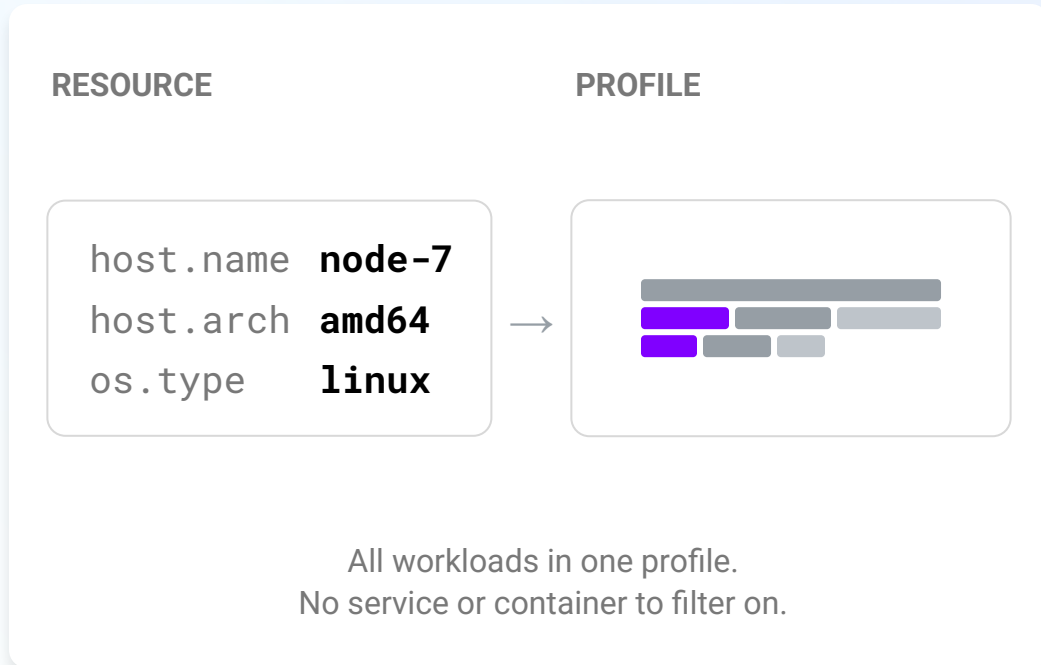


## ONE SIGNAL, MANY CONSUMERS



# One resource for the whole host...

## ONE RESOURCE PER HOST



# ... To one resource per workload

## ONE RESOURCE PER HOST

### RESOURCE

host.name **node-7**  
host.arch **amd64**  
os.type **linux**

### PROFILE



All workloads in one profile.  
No service or container to filter on.

## ONE RESOURCE PER WORKLOAD

### RESOURCE

service.name **web-api**  
container.id **a3f91e7c...**  
k8s.pod.name **web-api-...**



### PROFILE



postgres



image-worker



# But the data got bigger!

## ONE RESOURCE PER HOST

### RESOURCE

host.name **node-7**  
host.arch **amd64**  
os.type **linux**

### PROFILE



All workloads in one profile.  
No service or container to filter on.

## ONE RESOURCE PER WORKLOAD

### RESOURCE

service.name **web-api**  
container.id **a3f91e7c...**  
k8s.pod.name **web-api-...**



postgres



image-worker



### PROFILE



**~2x**  
**TOTAL  
PROFILE DATA**

# Same information, repeated

## web-api



container.id a3f91e7c...  
host.name node-7  
os.type linux

### FUNCTION NAMES

\_\_libc\_start\_main  
memcpy  
...

## postgres



container.id 7b21c9f4...  
host.name node-7  
os.type linux

### FUNCTION NAMES

\_\_libc\_start\_main  
memcpy  
...

## image-worker

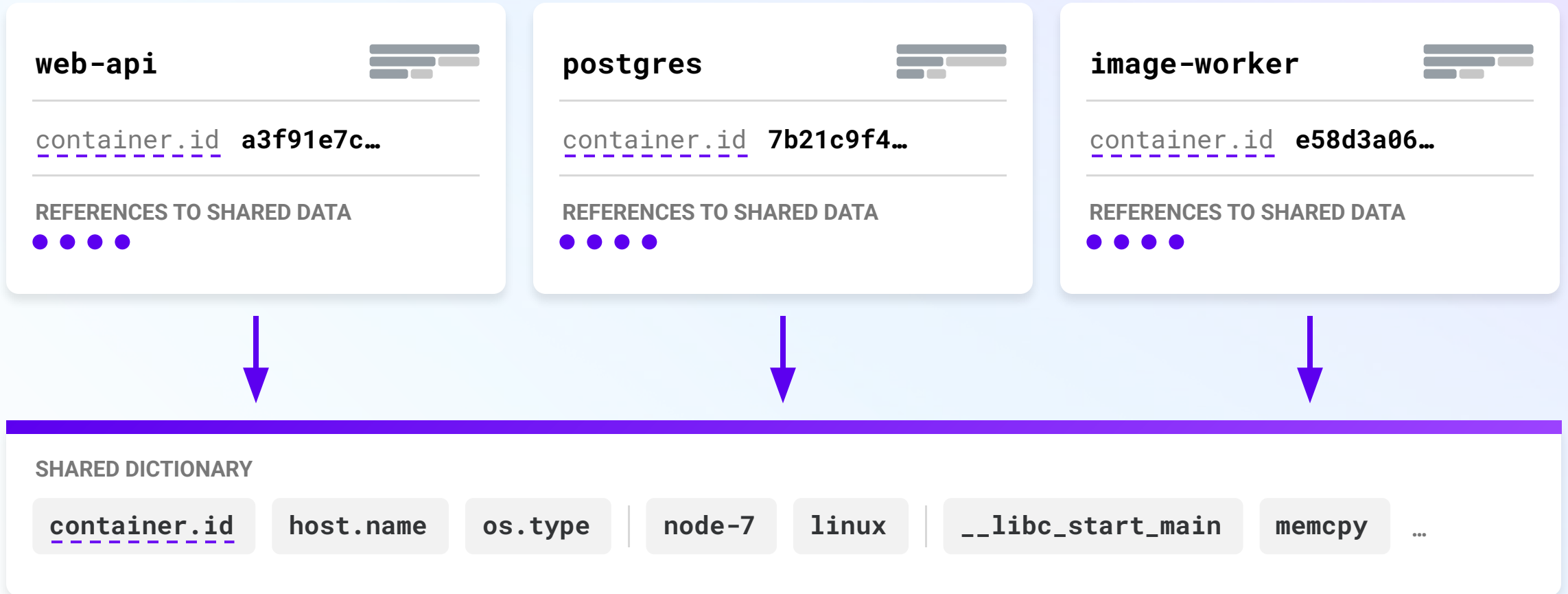


container.id e58d3a06...  
host.name node-7  
os.type linux

### FUNCTION NAMES

\_\_libc\_start\_main  
memcpy  
...

# Shared once, referenced everywhere



# Shared once, referenced everywhere

**~2x**  
**TOTAL** 🤖  
**PROFILE DATA**

web-api



container.id a3f91e7c...

REFERENCES TO SHARED DATA



postgres



container.id 7b21c9f4...

REFERENCES TO SHARED DATA



image-worker



container.id e58d3a06...

REFERENCES TO SHARED DATA



SHARED DICTIONARY

container.id

host.name

os.type

node-7

linux

\_\_libc\_start\_main

memcpy

...

# Shared once, referenced everywhere

~~2x~~  
TOTAL  
PROFILE DATA 🤖

web-api



container.id a3f91e7c...

REFERENCES TO SHARED DATA



postgres



container.id 7b21c9f4...

REFERENCES TO SHARED DATA



image-worker



container.id e58d3a06...

REFERENCES TO SHARED DATA



SHARED DICTIONARY

container.id

host.name

os.type

node-7

linux

\_\_libc\_start\_main

memcpy

...

# Shared once, referenced everywhere

**~1,1x**  
**TOTAL**  
**PROFILE DATA**

web-api



container.id a3f91e7c...

REFERENCES TO SHARED DATA



postgres



container.id 7b21c9f4...

REFERENCES TO SHARED DATA



image-worker



container.id e58d3a06...

REFERENCES TO SHARED DATA



SHARED DICTIONARY

container.id

host.name

os.type

node-7

linux

\_\_libc\_start\_main

memcpy

...

# Comparing Flame Graphs

<module> (client.py)

main (client.py)

> *stdlib*

worker (client.py)

> *requests*

m... method\_vector...

\_ssl\_SSLSocke...

\_ssl\_SSLSocke...

state\_machine...

read\_s... wri...

Before OS upgrade

<module> (client.py)

main (client.py)

> *stdlib*

worker (client.py)

> *requests*

\_P...

method\_vectorcall\_FASTCA...

method\_vectorcall\_NOARG...

cfu...

\_ssl\_SSLContext\_lo...

\_s...

\_ssl\_SSLSocket\_do\_handsh...

tp...

\_ssl\_SSLContext\_lo...

\_s...

\_ssl\_SSLSocket\_do\_handsh...

\_ss...

SSL\_CTX\_us...

us...

SS...

state\_machine (statem.c)

\_ss...

PEM\_read\_...

P...

X...

read\_state\_machi...

writ...

S...

pem\_read\_...

d...

b...

tls\_pr...

pem\_read\_...

A...

X...

ASN1\_...

OSSL\_D...

a...

X...

asn1\_i...

OS...

a...

P...

asn1\_i...

col...

a...

A...

asn1\_...

O...

a...

a...

asn1\_i...

a...

a...

asn1\_...

a...

a...

asn1\_...

a...

a...

x509...

a...

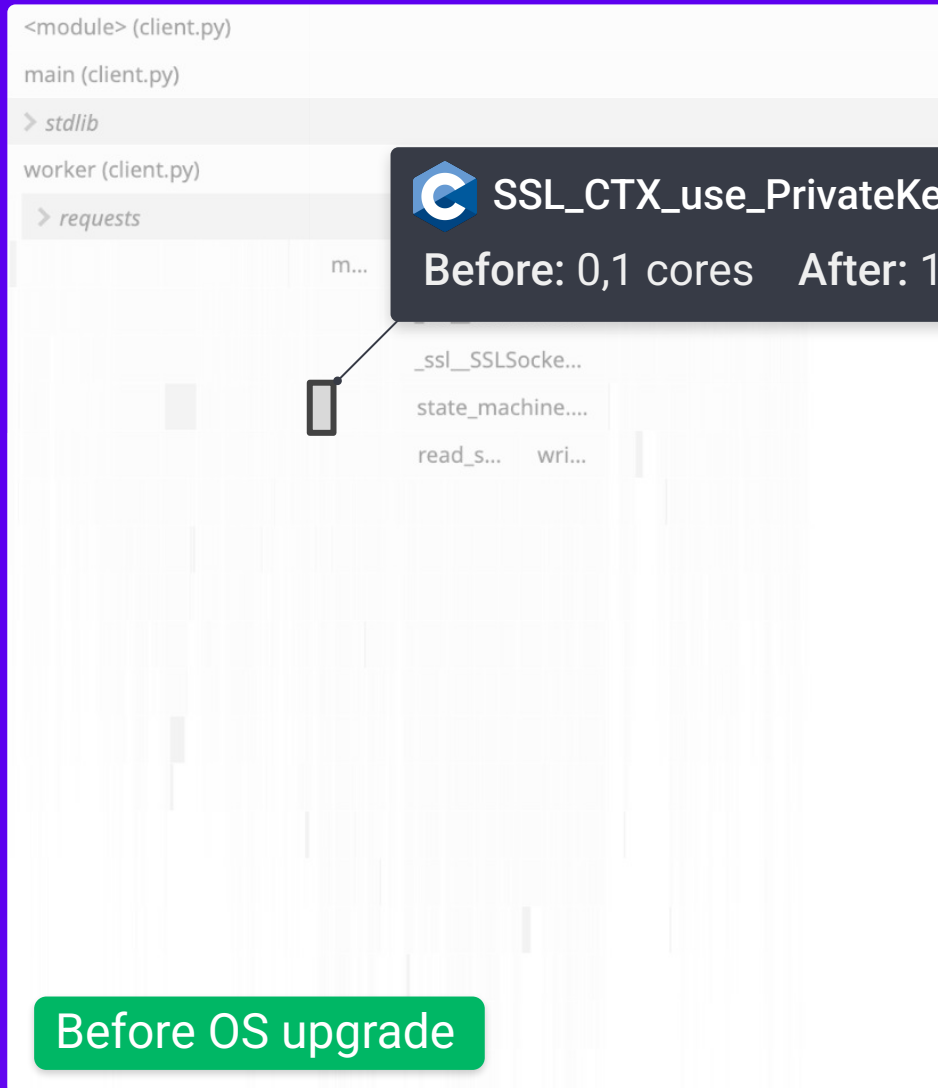
a...

O...

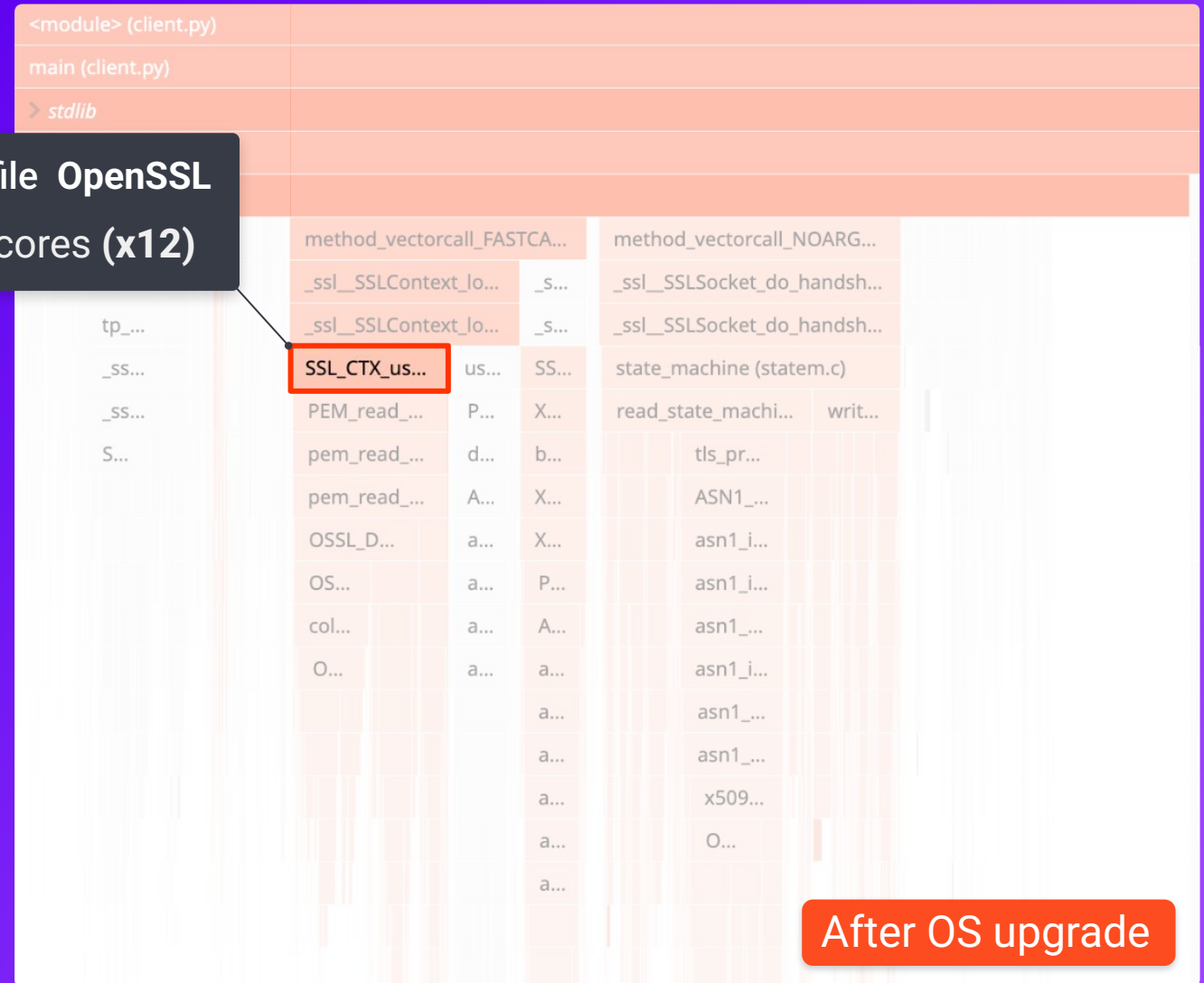
a...

After OS upgrade

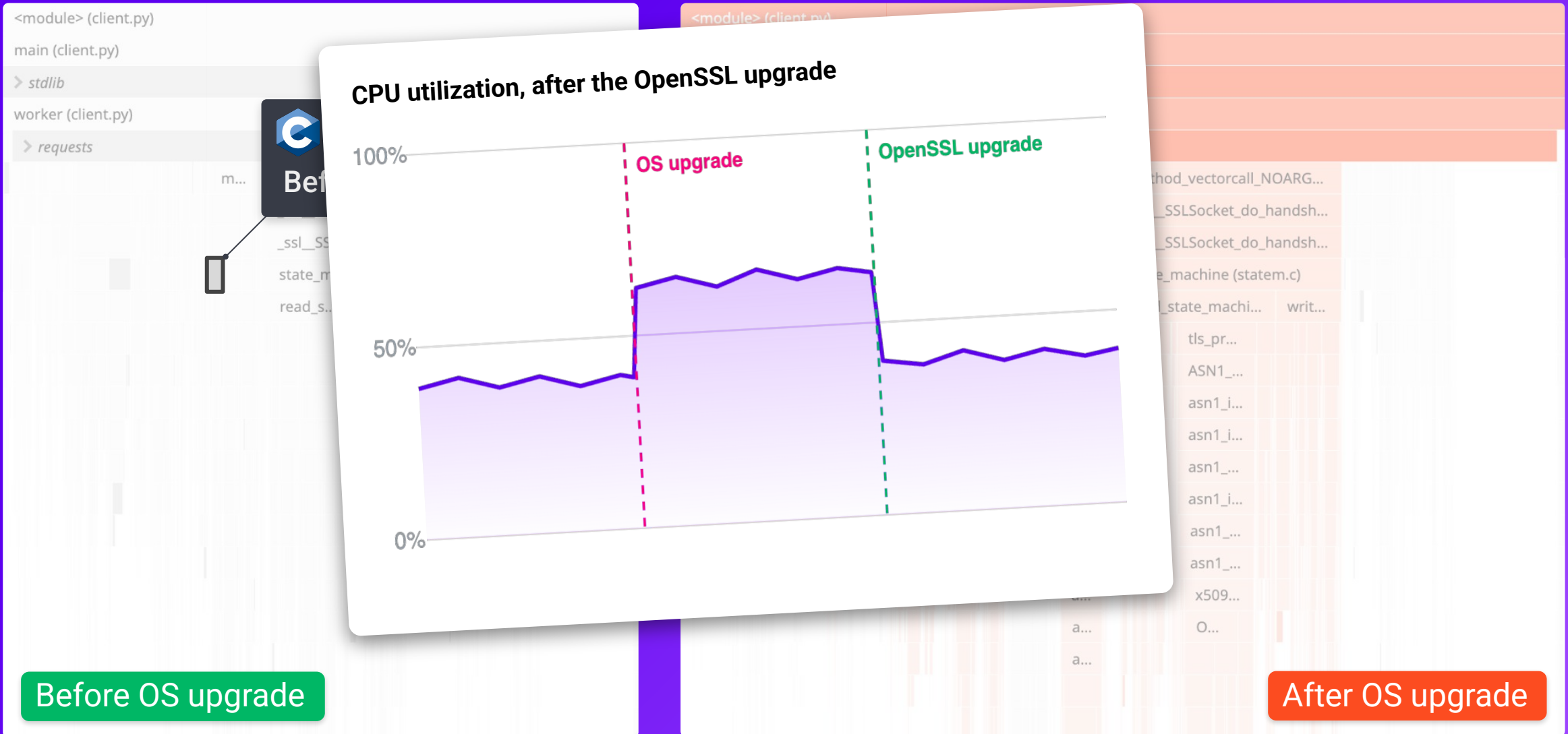
# Comparing Flame Graphs



 **SSL\_CTX\_use\_PrivateKey\_file OpenSSL**  
Before: 0,1 cores    After: 1,2 cores (x12)



# Comparing Flame Graphs



# Try it now!

## Run the OTel demo

See the OpenTelemetry eBPF profiler in action.

## Pick a service

Open the flame graph and see where the CPU time goes.

## Try it on your systems!

Start with one Linux host.

Do you know what your CPU is doing?

[opentelemetry.io/docs/demo](https://opentelemetry.io/docs/demo)

# Thank you!

Feedback about the Profiles Signal or the eBPF profiler?

Reach out in:   #otel-profiles

To learn more:

- [OpenTelemetry eBPF Profiler](#)
- [OpenTelemetry Profiles Signal](#)



↑ Talk feedback



DATADOG