ALTINITY

Where's the "Auto" in Auto-Instrumentation?

Josh Lee
josh@altinity.com

ALTINITY

Questions about Auto-Instrumentation

Do I need to modify my application code?

Do I need to use specific libraries?

Will it work without Kubernetes?

Josh Lee
josh@altinity.com

It depends…

01 — What are we trying to do with instrumentation?
What is “auto-instrumentation”?
Hint: there is more than one right answer

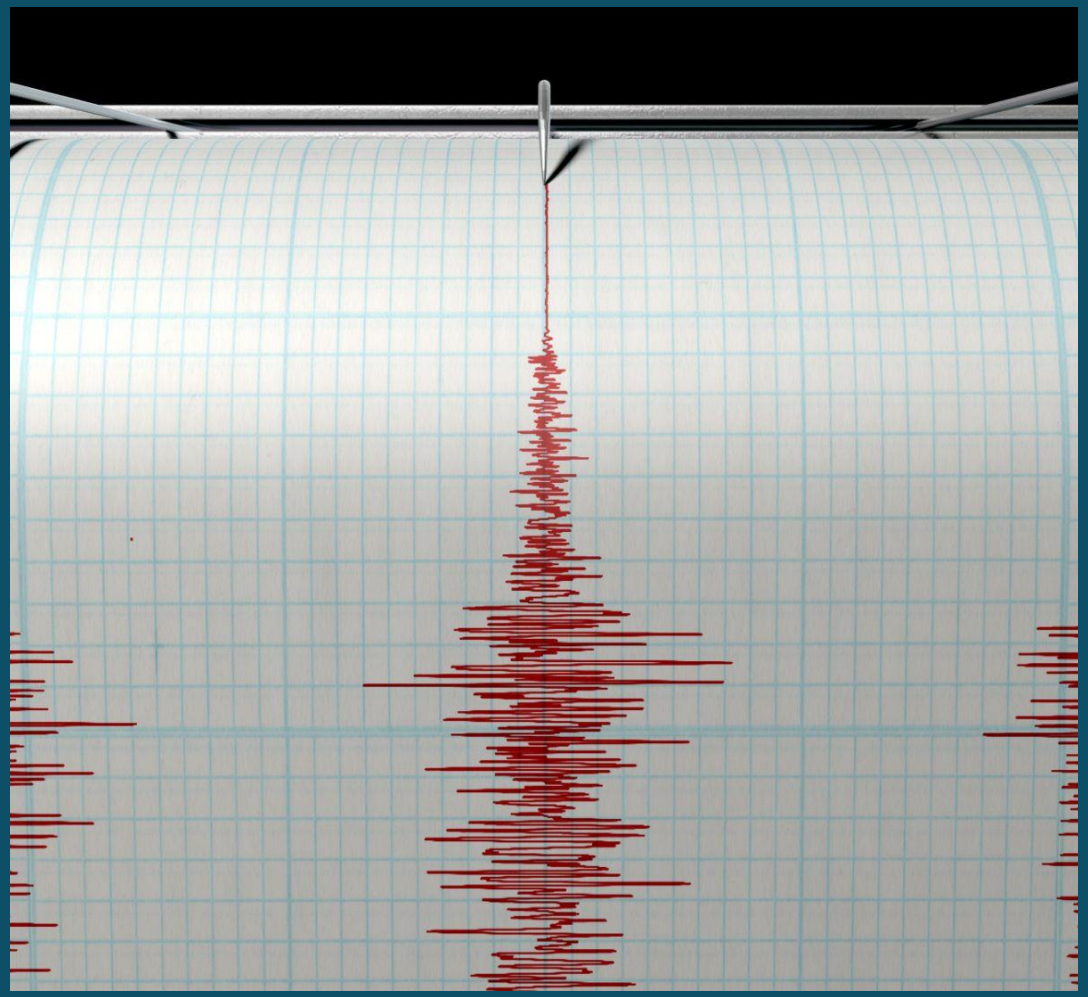
02 — Different kinds of auto-instrumentation
The Kubernetes Operator

03 — When is Auto-Instrumentation not enough?

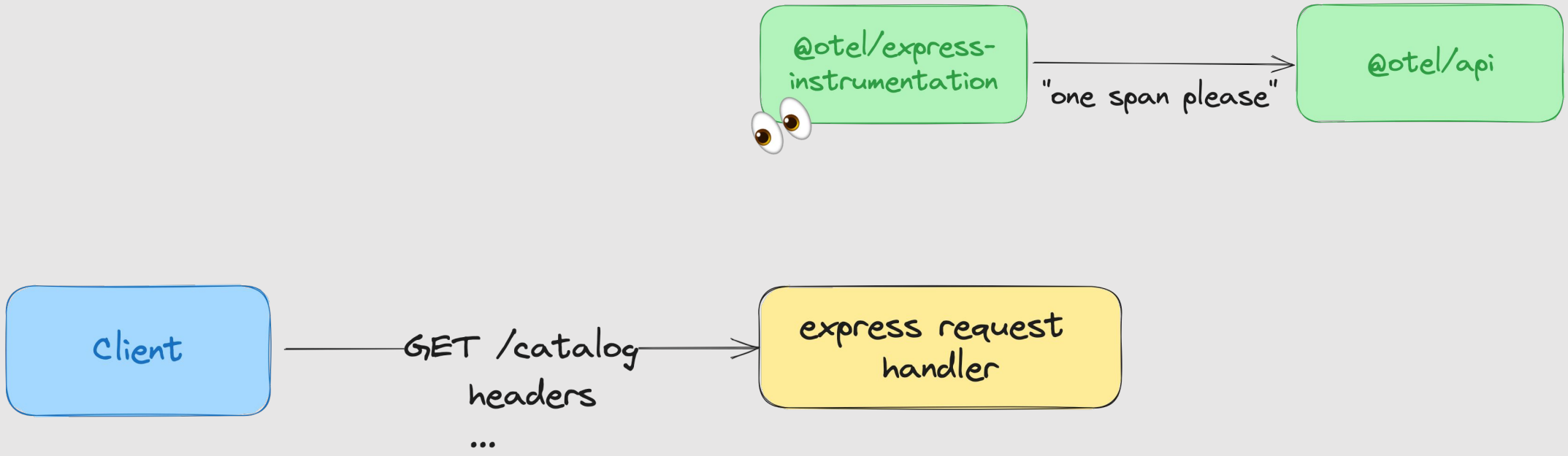
Josh Lee
josh@altinity.com

ALTINITY

Instrumentation is the process of translating *interesting things* into telemetry signals



Josh Lee
josh@altinity.com



Josh Lee
josh@altinity.com

ALTINITY

Interesting events

Requests,
Queries,
& Messages

Errors,
Exceptions,
& Events

Function Calls
(with arguments)

Josh Lee
josh@altinity.com

Contextualizing metadata

User Context

Who are they and what are they trying to do?

Infrastructure Context

What is the state of the resources?

Organizational Context

Who is responsible?

Josh Lee
josh@altinity.com

ALTINITY

```
// Copyright The OpenTelemetry Authors
// SPDX-License-Identifier: Apache-2.0

const opentelemetry = require('@opentelemetry/sdk-node');
const {getNodeAutoInstrumentations} = require('@opentelemetry/auto-instrumentations-node');
const {OTLPTraceExporter} = require('@opentelemetry/exporter-trace-otlp-grpc');
const {OTLPMetricExporter} = require('@opentelemetry/exporter-metrics-otlp-grpc');
const {PeriodicExportingMetricReader} = require('@opentelemetry/sdk-metrics');
...

const sdk = new opentelemetry.NodeSDK({
  traceExporter: new OTLPTraceExporter(),
  instrumentations: [
    getNodeAutoInstrumentations(...)
  ],
  metricReader: new PeriodicExportingMetricReader({
    exporter: new OTLPMetricExporter(),
  }),
  resourceDetectors: [
    ...
  ],
});

sdk.start();
```

Instrumentation in OpenTelemetry

- OpenTelemetry SDK
- OpenTelemetry API
- Instrumentation Libraries

Josh Lee
josh@altinity.com

ALTINITY

```
// Copyright The OpenTelemetry Authors
// SPDX-License-Identifier: Apache-2.0

const opentelemetry = require('@opentelemetry/sdk-node');
const {getNodeAutoInstrumentations} = require('@opentelemetry/auto-instrumentations-node');
const {OTLPTraceExporter} = require('@opentelemetry/exporter-trace-otlp-grpc');
const {OTLPMetricExporter} = require('@opentelemetry/exporter-metrics-otlp-grpc');
const {PeriodicExportingMetricReader} = require('@opentelemetry/sdk-metrics');
...

const sdk = new opentelemetry.NodeSDK({
  traceExporter: new OTLPTraceExporter(),
  instrumentations: [
    getNodeAutoInstrumentations(...)
  ],
  metricReader: new PeriodicExportingMetricReader({
    exporter: new OTLPMetricExporter(),
  }),
  resourceDetectors: [
    ...
  ],
});

sdk.start();
```

Josh Lee
josh@altinity.com

ALTINITY

… Isn't that Auto-Instrumentation?

Josh Lee
josh@altinity.com

ALTINITY

Instrumentation Libraries *target specific libraries* and modify them (or observe them) to call the OTel APIs, creating telemetry

opentelemetry-instrumentation-bunyan	chore: release main (#2497)
opentelemetry-instrumentation-cassandra	chore: release main (#2497)
opentelemetry-instrumentation-connect	chore: release main (#2497)
opentelemetry-instrumentation-dns	chore: release main (#2497)
opentelemetry-instrumentation-express	chore: release main (#2497)
opentelemetry-instrumentation-fastify	chore: release main (#2497)
opentelemetry-instrumentation-generic-p...	chore: release main (#2497)
opentelemetry-instrumentation-graphql	chore: release main (#2497)
opentelemetry-instrumentation-hapi	chore: release main (#2497)
opentelemetry-instrumentation-ioredis	chore: release main (#2497)
opentelemetry-instrumentation-knex	chore: release main (#2497)
opentelemetry-instrumentation-koa	chore: release main (#2497)
opentelemetry-instrumentation-memcached	chore: release main (#2497)
opentelemetry-instrumentation-mongodb	chore: release main (#2497)
opentelemetry-instrumentation-mysql	chore: release main (#2497)
opentelemetry-instrumentation-mysql2	chore: release main (#2497)

Josh Lee
josh@altinity.com

ALTINITY

What is Auto-Instrumentation?

"Auto-Instrumentation" Meta Packages

Packages that can be configured to automatically include relevant instrumentation libraries based on the presence of other libraries.

No-Code Instrumentation Agents + Extensions

Mechanisms for adding instrumentation to an application package after it has already been compiled or bundled.

Instrumentation-Injection (w/ Kubernetes Operator)

The Kubernetes Operator can automatically inject no-code instrumentation into matching workloads.

Josh Lee
josh@altinity.com

Mechanisms

Code-based Library-Instrumentation

Observe other modules in the same process without directly modifying their source, and emit telemetry*

No-Code Instrumentation

Modify or observe another program (binary, bytecode, etc.) and emit telemetry*

Code-based Library-instrumentation
Monkey Patching
Function Wrapping
Middleware Injection
Event Observation

No-Code Instrumentation Mechanisms
Monkey Patching (JS, Python)
Runtime Agent (Java, .NET)
Interpreter Extension (PHP)
eBPF (Go)

Josh Lee
josh@altinity.com

ALTINITY

Monkey Patching

```
new InstrumentationNodeModuleDefinition(
    'express',
    ['>=4.0.0 <5'],
    moduleExports => {
      const routerProto = moduleExports.Router as unknown as express.Router;
      // patch express.Router.route
      if (isWrapped(routerProto.route)) {
       this._unwrap(routerProto, 'route');
      }
      this._wrap(routerProto, 'route', this._getRoutePatch());
      ...
      return moduleExports;
    },
    moduleExports => {
      if (moduleExports === undefined) return;
      const routerProto = moduleExports.Router as unknown as express.Router;
      this._unwrap(routerProto, 'route');
      this._unwrap(routerProto, 'use');
      this._unwrap(moduleExports.application, 'use');
    }
  ),
];
```

Josh Lee
josh@altinity.com

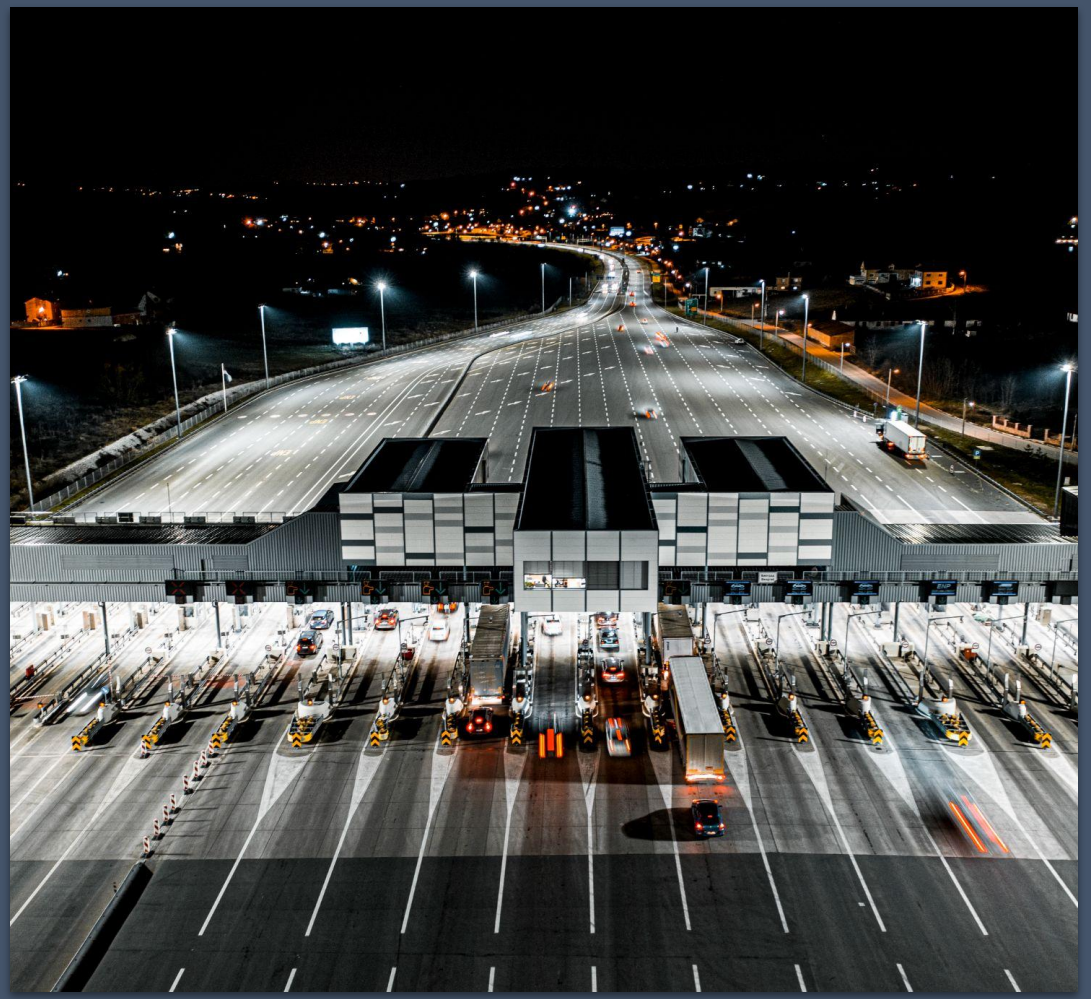
Function Wrapping

```php
private static function _hook(..., ?string $class, string $function, string $name, int $spanKind = SpanKind::KIND_SERVER): void
{
    hook(
        class: $class,
        function: $function,
        pre: static function ($object, ?array $params, ?string $class, ?string $function, ?string $filename, ?int $lineno) use ($instrumentation, $name, $spanKind) {
            $span = self::builder($instrumentation, $name, $function, $class, $filename, $lineno)
                ->setSpanKind($spanKind)
                ->startSpan();
            Context::storage()->attach($span->storeInContext(Context::getCurrent()));
        },
        post: static function ($object, ?array $params, mixed $return, ?Throwable $exception) {
            self::end($exception);
        }
    );
}
```

Josh Lee
josh@altinity.com

ALTINITY

Middleware Injection



Josh Lee
josh@altinity.com

ALTINITY

Event Observation

```elixir
@doc false
def attach_router_start_handler(_opts) do
  :telemetry.attach(
    {__MODULE__, :router_dispatch_start},
    [:phoenix, :router_dispatch, :start],
    &__MODULE__.handle_router_dispatch_start/4,
    %{}
  )
end
```

Josh Lee
josh@altinity.com

ALTINITY

No-Code Instrumentation Mechanisms

Josh Lee
josh@altinity.com

ALTINITY

	Instrumentation Libraries	No-Code Instrumentation	Mechanism
Java	✅	✅	Java Agent + Bytecode Injection
Python	✅	✅	Python Agent + Monkey patching
JavaScript	✅	✅	Monkey patching
.NET	✅	✅	.NET Profiler + Bytecode Injection
Go	✅	✅*	eBPF
PHP	✅	✅	Interpreter Extension + Autoloading
Erlang / Elixir	✅		
Ruby	✅		

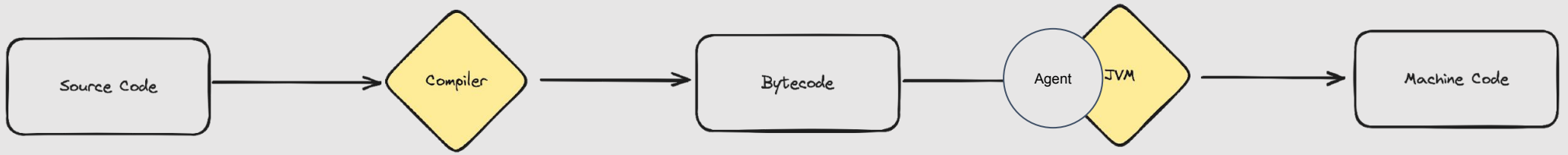
Josh Lee
josh@altinity.com

Bytecode Injection

Runtime agents can inject OpenTelemetry SDKs and library instrumentation at the bytecode level.

Functions can also be wrapped and/or monkey patched.

Josh Lee
josh@altinity.com



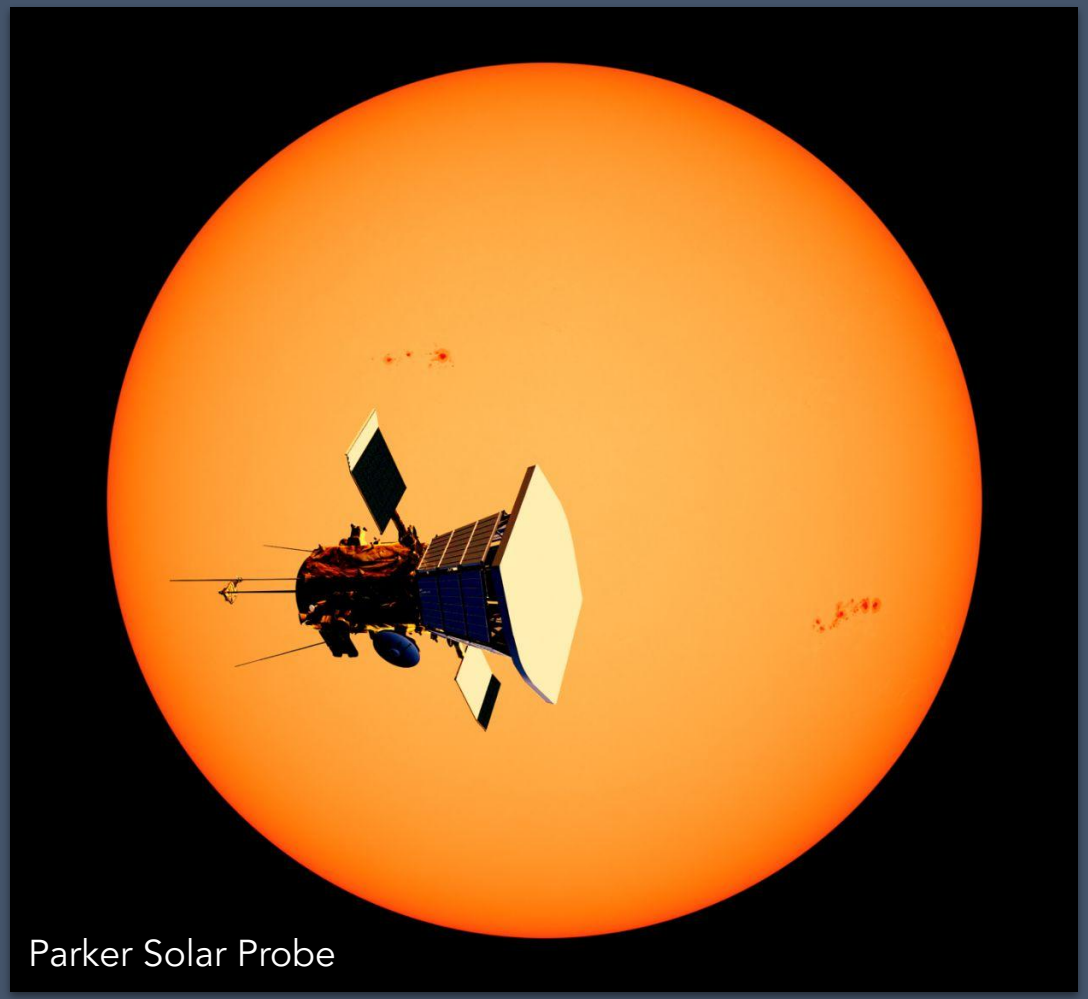
Josh Lee
josh@altinity.com

ALTINITY

eBPF

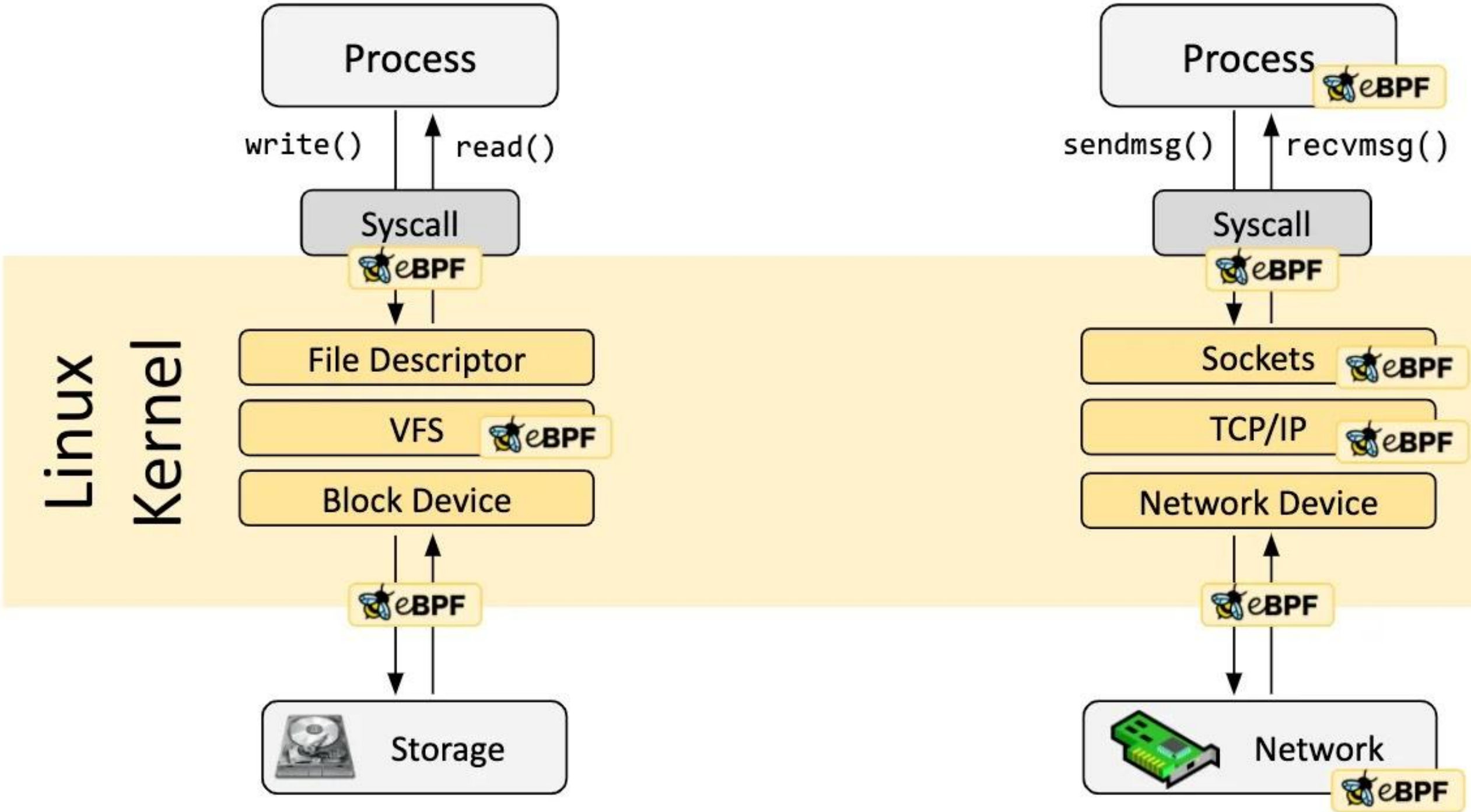
Compiled binaries can't be modified with instrumentation.

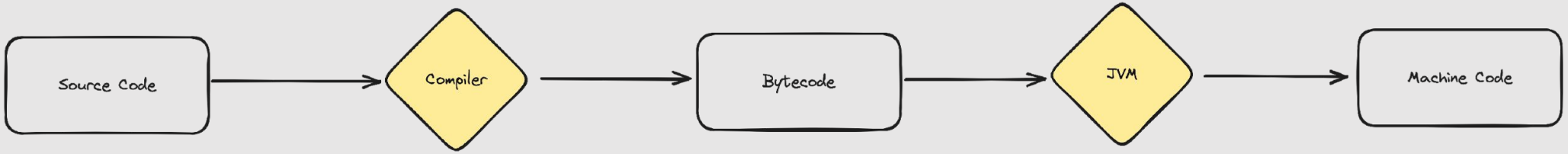
eBPF + uProbes offer visibility.



Parker Solar Probe

Josh Lee
josh@altinity.com


Process
write()
read()
Syscall
eBPF
File Descriptor
VFS
eBPF
Block Device
eBPF
Storage
Linux
Kernel
Process
eBPF
sendmsg()
recvmsg()
Syscall
eBPF
Sockets
eBPF
TCP/IP
eBPF
Network Device
eBPF
Network
eBPF



Josh Lee
josh@altinity.com

ALTINITY

Injecting Instrumentation

Josh Lee
josh@altinity.com

ALTINITY

Kubernetes Operator

```
kubectl apply -f - <<EOF
apiVersion: opentelemetry.io/v1alpha1
kind: Instrumentation
metadata:
  name: demo-instrumentation
spec:
  exporter:
    endpoint: http://demo-collector:4318
  propagators:
    - tracecontext
    - baggage
  sampler:
    type: parentbased_traceidratio
    argument: "1"
EOF
```

Josh Lee
josh@altinity.com

ALTINITY

What have we learned?

Josh Lee
josh@altinity.com

ALTINITY

Questions about Auto-Instrumentation

Do I need to modify my application code?

Do I need to use specific libraries?

Will it work without Kubernetes?

Josh Lee
josh@altinity.com

ALTINITY

When is Auto-instrumentation *not enough*

Josh Lee
josh@altinity.com

ALTINITY

Essential Custom Spans

"Stack" trace information

Incompatible frameworks or libraries

Especially for monolithic services

Josh Lee
josh@altinity.com

Essential Manual Annotation

User Experience & Client Details

Details about the user (who they are or what they are trying to do) from the Application or Client contexts.

Business Metrics & Dimensions

Runtime details about business operations (e.g. regions, departments, revenue, cost)

Team / Business Unit Ownership

Especially if it isn't defined by infrastructure context.

Extra useful in monolith-like services.

Incompatible or Bespoke System Architecture

Heavy use of non-HTTP messaging, IPC, RPC *

Compiled languages (besides Go)

Josh Lee
josh@altinity.com

ALTINITY

When is Auto-Instrumentation *awesome*

Josh Lee
josh@altinity.com

When is Auto-Instrumentation *awesome*

Combined with Manual Annotation

Reduce toil with a consistent starting point for customization

Fill Gaps in E2E Distributed Tracing

Legacy services

Skill-gap on central observability team

Quickly Understand System Topology

Complete tracing can reveal architecture details

Cannot modify original source code

Auto-instrumentation is the only option here

Off-the-shelf OSS components can be deployed as-is

Josh Lee
josh@altinity.com

ALTINITY

Include Auto-Instrumentation *early**

* especially for http-based apps

Josh Lee
josh@altinity.com

ALTINITY

Thank You

Josh Lee
josh@altinity.com

OTel Observatory

ALTINITY

Connect with me

https://www.linkedin.com/in/joshuamlee/

Josh Lee
josh@altinity.com