



اگر برنامه‌ای بنویسی،
بالاخره یک روز خراب می‌شود.

سؤال این نیست که «خراب می‌شود یا
نه»، سؤال این است که **چطور سریع**
بفهمی چه اتفاقی افتاده است.



Monitoring



فقط می‌گوید:
«مشکلی وجود دارد.»
(هشدار می‌دهد)

Observability



جواب می‌دهد:
«چرا این مشکل به وجود آمده؟»
(دلیل را پیدا می‌کند)

اصلاً Observability یعنی چه؟

یعنی بتوانی فقط با اطلاعاتی که سیستم تولید می‌کند، بفهمی داخل آن چه خبر است. بدون اینکه وارد سرور شوی یا حدس بزنی.

پیدا کردن خطا → پیدا کردن علت → فهمیدن رفتار سیستم



سه ابزار اصلی



ترکیب این سه، **Observability** را می‌سازد.



Logging چیست؟

(لاگ‌ها تاریخچه اتفاقات هستند)

لاگ یعنی برنامه اتفاقات مهم را
داخل فایل یا سیستم ثبت کند.

- [INFO] کاربر وارد شد.
- [INFO] سفارش ثبت شد.
- [WARNING] پرداخت ناموفق بود.
- [ERROR] خطای دیتابیس رخ داد.

«لاگ برای کامپیوتر نیست؛ برای انسان است.»



۱. زمان (Timestamp)

۲. سطح اهمیت (Level)

۳. پیام واضح (Message)

۴. شناسه کاربر یا درخواست (ID)

۳. پیام واضح (Message)

۵. جزئیات مرتبط (Context)

```
[2023-10-27T14:35:22.105+03:30] [INFO] [REQ: c9d8e7b6a5f4]
User 'ali.jafari' successfully processed payment for order
#7890. Status: OK.
Context: {paymentId: 'PAY-123456', gateway: 'ZarinPal',
amount: 50000, currency: 'IRR'}.
(Logger: com.example.payment.processor.PaymentService) [TID:
1234]
```




اشتباه رایج: هر چیزی را لاگ نکن.

- حجم زیاد
- هزینه ذخیره سازی
- مصرف CPU/RAM
- سخت شدن پیدا کردن خطا

فقط اطلاعات مفید را ثبت کن.

DEBUG	→	مخصوص توسعه
INFO	→	اطلاعات عادی
WARNING	→	اتفاق غیرعادی
ERROR	→	خطا
FATAL	→	توقف برنامه

چیست؟ Metrics



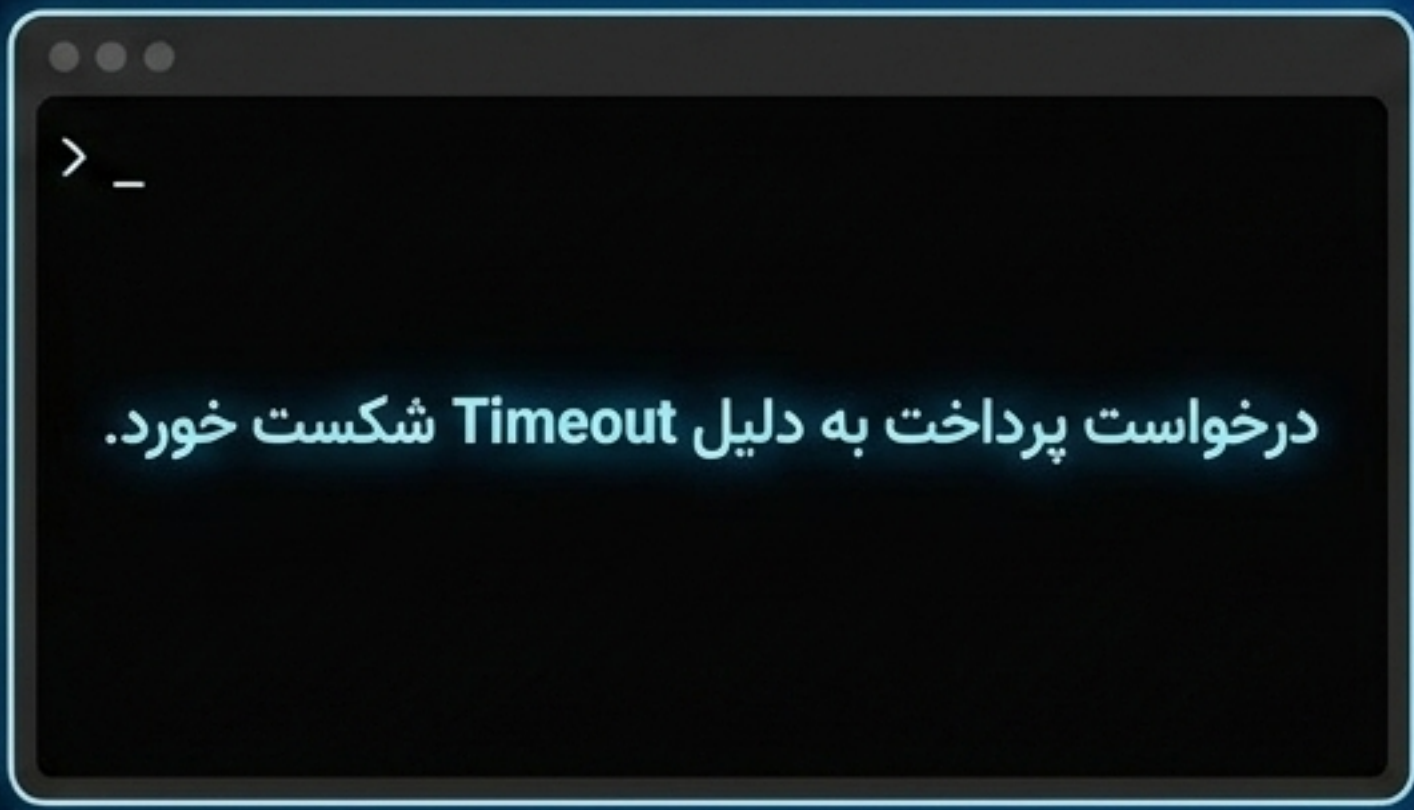
متریک یعنی یک عدد که وضعیت سیستم را نشان می‌دهد.





Metric: فقط عدد را نشان می‌دهد.

Response Time = 850ms



Log: داستان را تعریف می‌کند.

Tracing چیست؟

مسیر کامل یک درخواست را ثبت می‌کند.



چرا Tracing مهم است؟

در سیستم‌های بزرگ، پیدا کردن محل خطا بدون Tracing غیرممکن است.

- درخواست از کجا شروع شد.
- از چه سرویس‌هایی عبور کرد.
- کدام قسمت بیشترین زمان را گرفت.
- دقیقاً کجا شکست خورد.



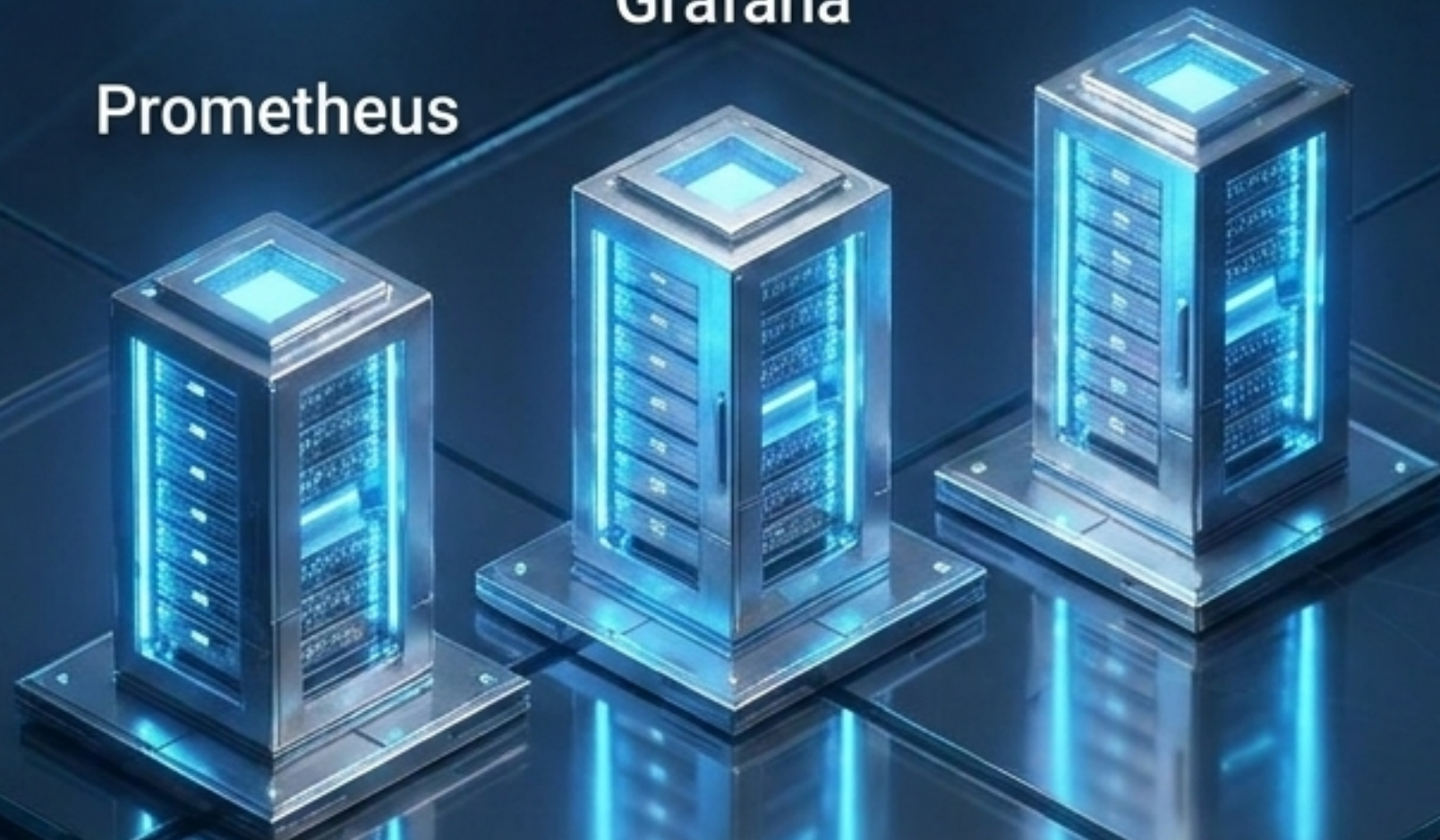
ابزارهای مهندسی (The DevOps Arsenal)

پیاده‌سازی Observability با استاندارد دنیای Cloud Native.

OpenTelemetry

Grafana

Prometheus



Prometheus (The Collector)

محبوب‌ترین ابزار جمع‌آوری Metrics. تقریباً استاندارد دنیای Cloud Native محسوب می‌شود.

- جمع‌آوری داده‌ها
- ذخیره‌سازی
- اجرای Query
- ارسال Alert

Grafana (The Lens)

داده تولید نمی‌کند.
کارش نمایش داده‌هاست.

- داشبورد ساختن
- دیدن نمودارها
- بررسی وضعیت سرورها
- تعریف Alert

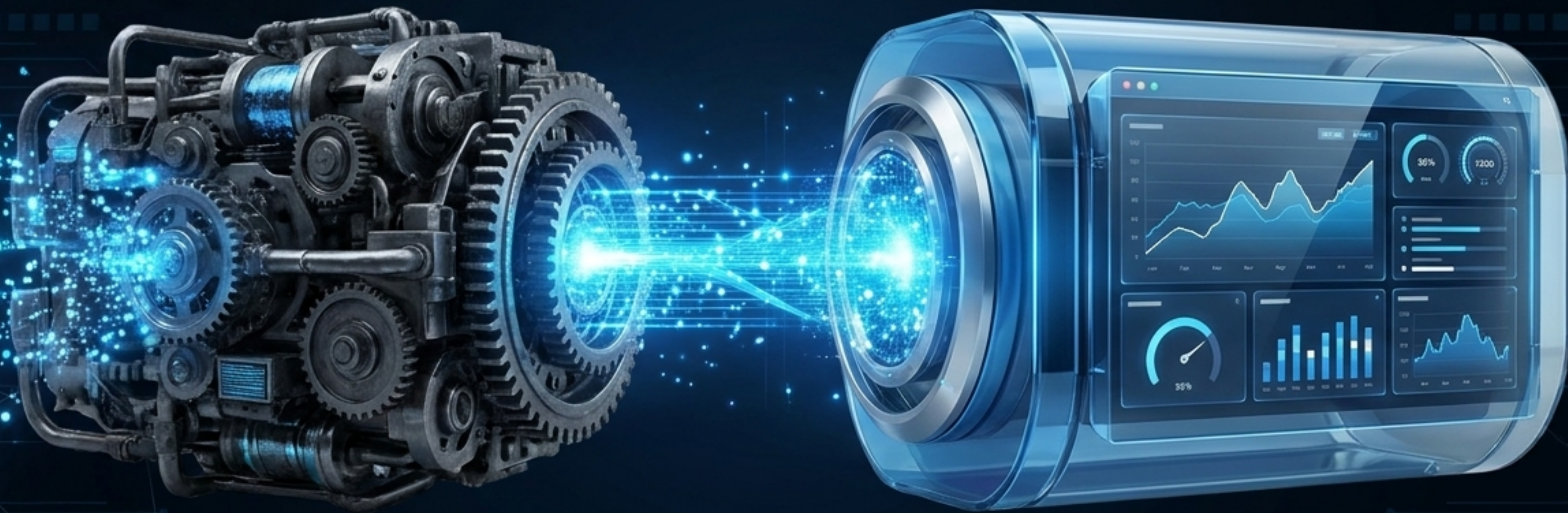




تقسیم وظایف ساده (Symbiosis)

داده جمع می‌کند. Prometheus

Grafana داده را زیبا نمایش می‌دهد.



«مثل رابطه دیتابیس و پنل مدیریت.»



OpenTelemetry (استاندارد جامع)

یک استاندارد واحد برای
جمع‌آوری اطلاعات
سیستم.



همه را یکپارچه کرد.
یک بار برنامه را Instrument
می‌کنی، بعد هر Backend که
خواستی استفاده می‌کنی
(بدون وابستگی).

Alert & Best Practices

Monitoring فقط نمایش نیست. باید وقتی مشکلی پیش آمد، خبر بدهد.

Alert: CPU ۹۰% بالای

نگاه برنامه‌نویس‌های حرفه‌ای

- لاگ واضح بنویس.
- اطلاعات حساس را لاگ نکن.
- فقط Errorها را بررسی نکن؛ Warningها مهم‌اند.
- Dashboard ساده بساز و قبل از وقوع مشکل Alert تعریف کن.



«اگر سیستم قابل مشاهده نباشد، عیب‌یابی
بیشتر شبیه حدس زدن است تا مهندسی.»

