Node.js 日志监控方案和实践

- 一个 Node 初学者的服务稳定性和性能探索

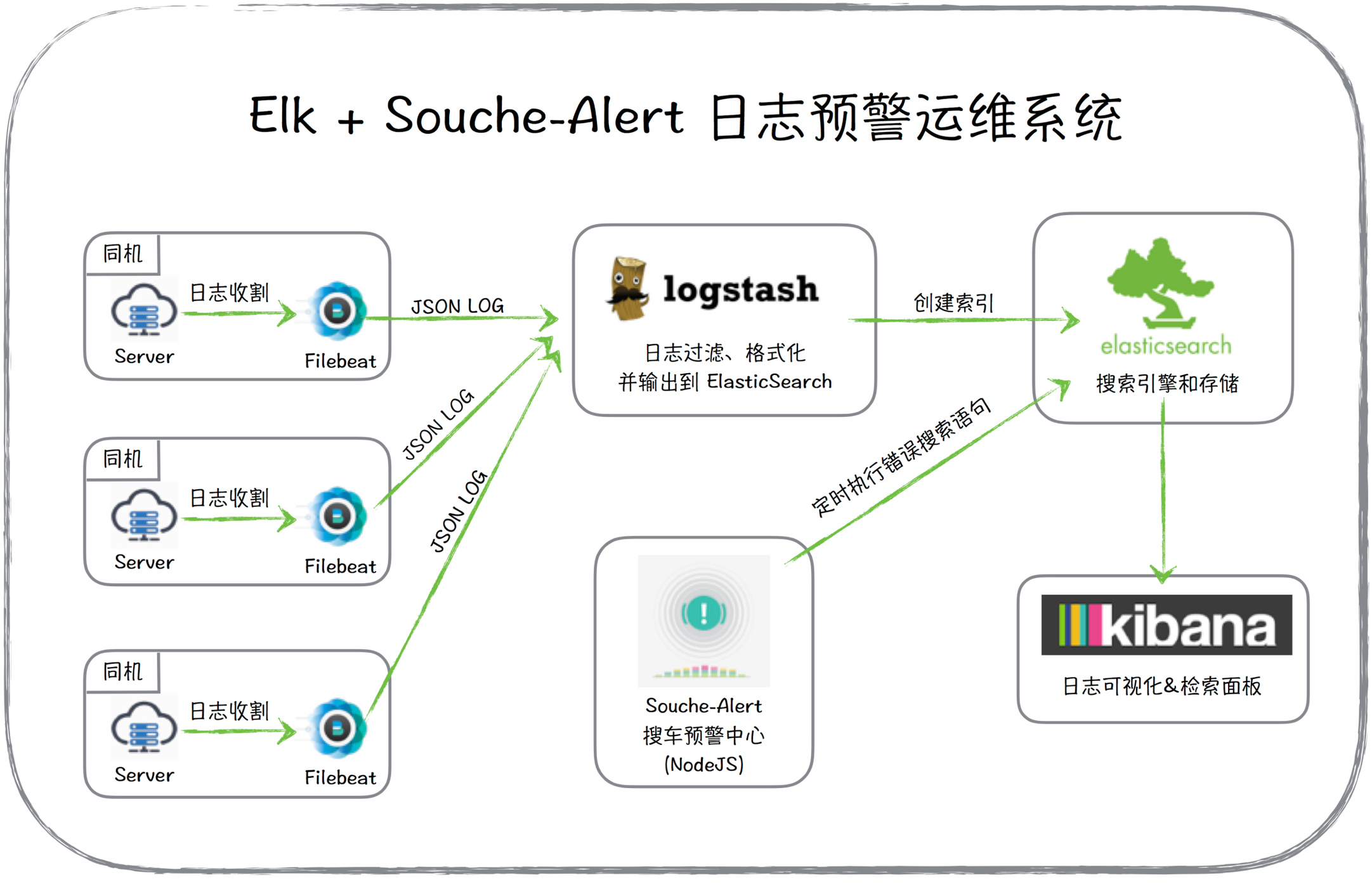
面对的困境

- 线上集群日志未聚合、导致追溯困难
- 日志爆炸、导致问题定位犹如大海寻针
- 业务逻辑级别的监控预警缺失
- OneAPM 对 API 响应时间的监控灵敏度过低
- OneAPM hack 过度了~~

ELK + Souche-Alert 日志预警运维系统

Grafana + Telegraf + InfluxDB API 响应时间监测

ELK + Souche-Alert 日志预警运维系统



日志生成 - SDK与分级

- 基于 node-bunyan 的二次封装
- log 分级输出
  - fatal(60)：会使整个服务不可用
  - error(50)：会使某个请求不可用
  - warn(40)：服务可用，可能需要关注
  - info(30)：一般用来记录请求信息
  - debug(20)：调试用
  - trace(10): 调用第三方服务/包时的输出信息

日志生成 - RequestID

- requestID 跟踪
  - 如果请求 header 部分未指定 X-Request-Id，则系统自动填充格式如 w{weeks}{day of week}_uuid 值
  - 默认记录每次 API 请求的完整参数、头部信息，并在 Express 的 req.x_request_id 上绑定本次请求的 X-Request-Id
  - 在 API 响应体 header 部分返回 X-Request-Id 字段
  - 测试环境记录返回体。
  - 可以根据用户 token 模拟抓包

日志生成 - 报错行捕获

```
function getCallerInfo(limit) {
    if (this === undefined) {
        // Cannot access caller info in 'strict' mode.
        return;
    }
    var obj = {};
    var saveLimit = Error.stackTraceLimit;
    var savePrepare = Error.prepareStackTrace;
    Error.stackTraceLimit = limit;

    Error.captureStackTrace(this, getCallerInfo);

    Error.prepareStackTrace = function (_, stack) {
            var caller = stack[limit - 1];
            if (sourceMapSupport) {
                caller = sourceMapSupport.wrapCallSite(caller);
            }
            obj.file = caller.getFileName();
            obj.line = caller.getLineNumber();
            var func = caller.getFunctionName();
            if (func)
                obj.func = func;
    };

    this.stack;

    Error.stackTraceLimit = saveLimit;
    Error.prepareStackTrace = savePrepare;
    return obj;
}
```

日志生成 - 通用埋点

- 未捕获错误

```
process.on('uncaughtException', processErrorHandler);
process.on('unhandledRejection', processErrorHandler);
```

- express error handling

```
app.use(function(err, req, res, next){
    // ....
    byLog.error(err, null, {
        req_id: req.x_request_id
    });
});
```

日志生成 - 通用埋点

- 测试环境 res.send Hack

```
res.$send = res.send;
res.send = function (result) {
    if(typeof result === 'object' && isDev) {
      byLog.error('INFO SEND', result);
    }
    // code here .....
    res.$send.apply(this, arguments);
}
```

日志生成 - 格式优化

- JSON LOG 优化
  - 保持单层解构
  - Value 部分用 util.inspect 至多做5层转换
- elasticsearch-template 优化
  - 确定字段关闭分词或索引

日志预警 - Souche-Alert

- 为什么不用 watcher
  - 调试困难 & 我们的需求没那么复杂
  - 稳定性欠佳，ElasticSearch 的插件形式，耦合过重。
  - 需要商业授权

日志预警 - Souche-Alert

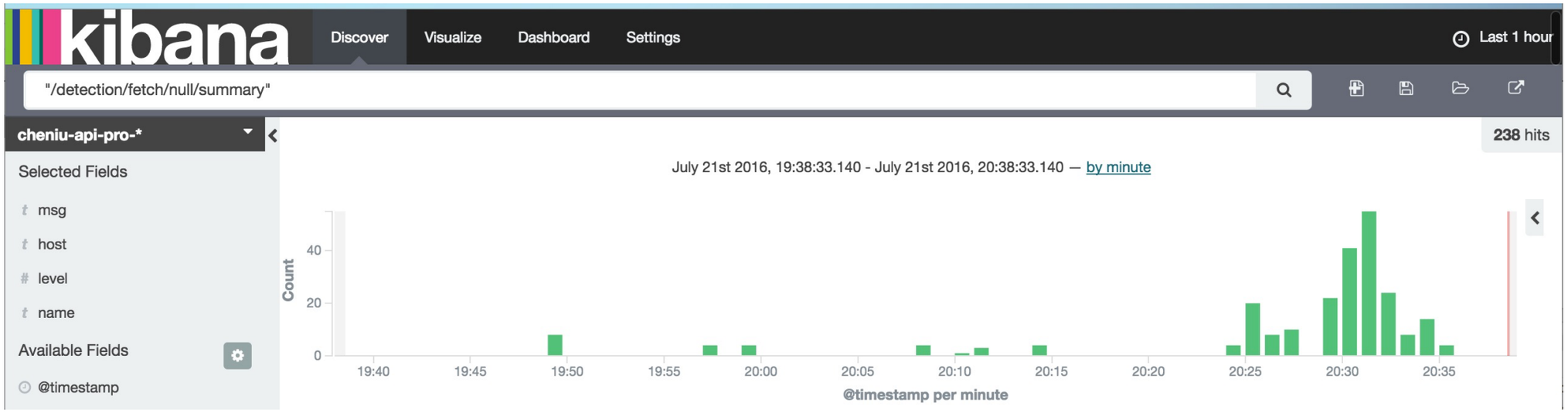
- 如何实现最基础的监控
  - 定时调用 ElasticSearch 的 URL Search 轮询 level >=50 的错误日志
  - 生成 Kibana 错误链接，并调用 kibana 的 shorten 接口
  - 利用 nodemailer 通过企业邮箱发送告警信息

[Level] ErrorId link: Summary Message

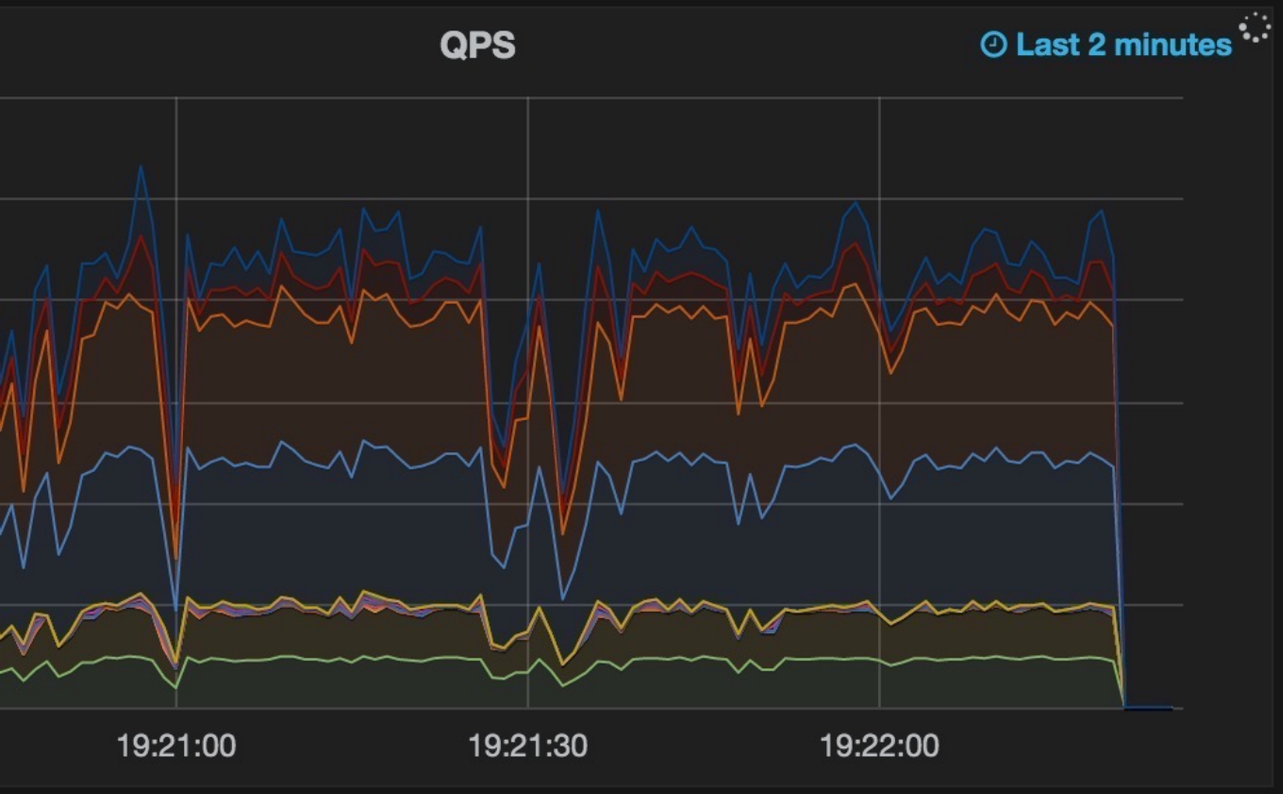
....

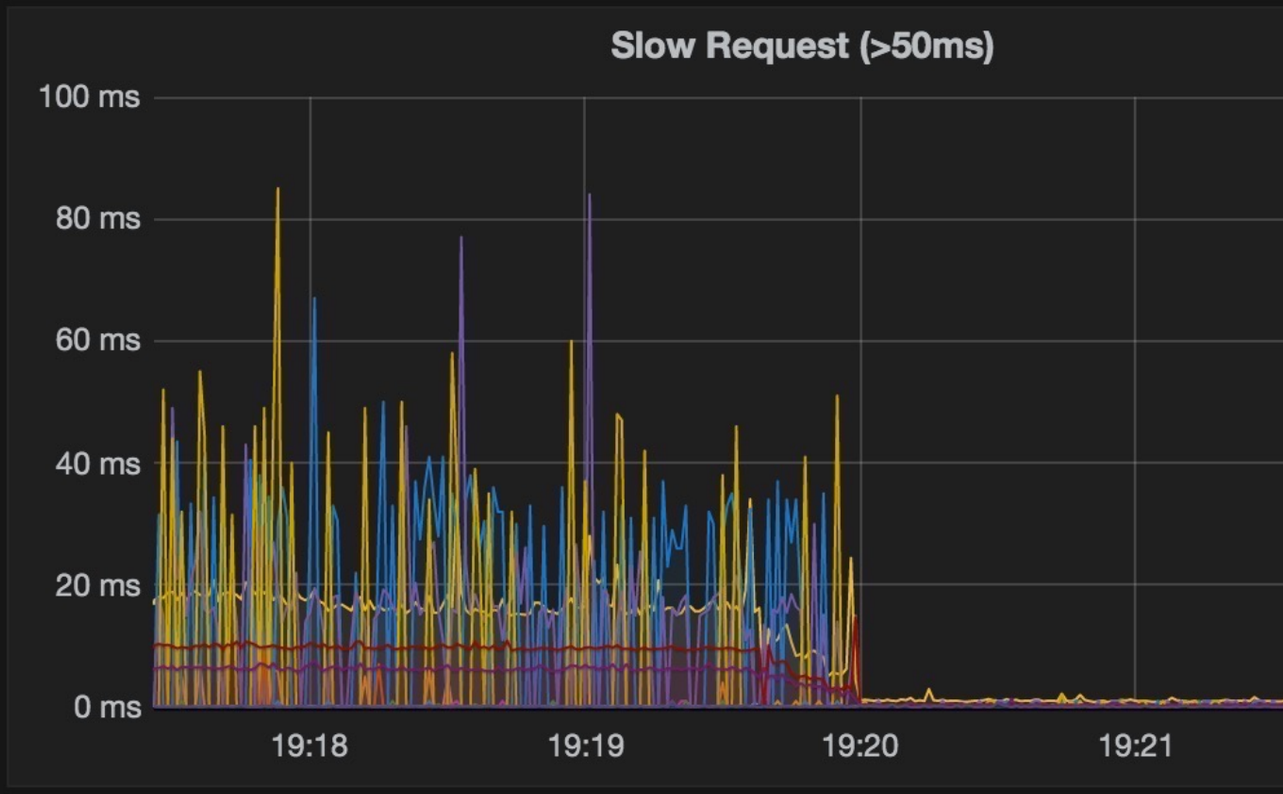
一些收效

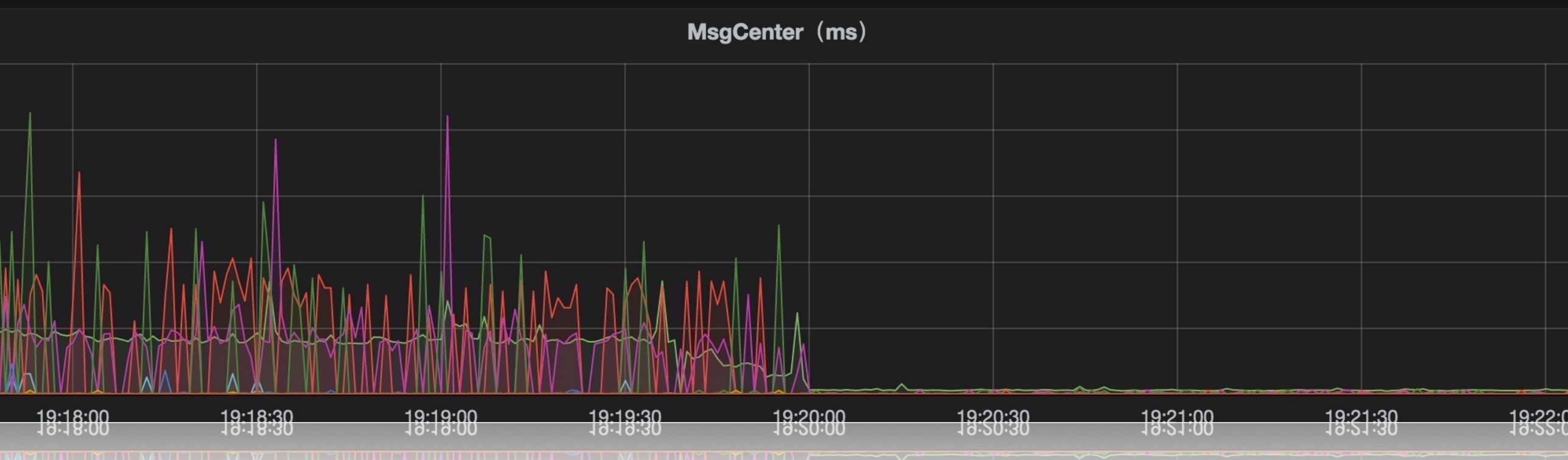
- 服务脏数据捕获，填补测试盲区



- 我们几乎都能比用户先发现问题。
- 用户请求聚合，快速恢复事故现场，开发效率的提升







API响应时间监测

记一次有趣的发现

TODO

- 实现函数与特定请求的绑定
- 日志的清洗和静默规则
- 网络 I/O 的监控
- 更加多样化的告警渠道，如微信、短信

“拍砖时刻”

– *plusman.cn*