Node.js中低成本实现错误告警

Jan 27, 2016

相比其他语言（特指PHP）而言，Node.js应用更需要关注出错信息，因为一旦处理不慎，就会导致应用crash。

一种偷懒的方法是使用PM2之类的进程管理软件来启动Node.js进程，从而达到出错crash后自动重新启动应用的目的。

当然更好的办法则是手工捕获错误，然后进行适当的处理，防止应用产生未被接住的错误导致crash。

在捕获到Node.js产生的错误后，下一步自然是记录到错误日志中，以便日后可以进行分析，并针对性地排查修改。本文要说的，即是对错误日志的处理方式之一——告警。

告警是运维工作中非常重要的一个环节，它能让开发者（维护者）及时获知应用出错状态和详情，及早介入处理，将线上故障的影响降低到最低。而要实现告警功能，则需要从两方面入手，一方面是对错误信息进行集中处理（分类、分级、合并、限流等），另一方面需要将这些错误信息及时推送出去。

推送

推送渠道可以有很多种，常见的包括邮件、短信、微信等，Geek一点的还可以考虑用slack、GTalk(死了吧)、Telegram机器人推送什么的。

为了降低开发成本，这里选用了Server Chan作为推送服务，它的使用极其简单，只要登录之后就会获得一个key，然后访问带key的URL `http://sc.ftqq.com/{KEY}.send` 即可完成消息推送。而推送的渠道则有两种，一种是手机客户端，另一种是微信。想要哪种就使用哪种，在网站绑定即可，推送时是不分渠道的。

日志

接下来是应用的错误日志收集，在打听了很多方案之后先用了bunyan这个模块作为日志记录工具。bunyan的优势在于：

- 结构化日志数据，方便后续整理分析
- 完善的错误分级 `fatal` / `error` / `warn` / `info` / `debug` / `trace` 一应俱全
- 多种错误处理方式：文件、控制台、流
- 可扩展：可以通过扩展流的方式自定义错误处理逻辑
- 日志文件自动滚动

这里我们主要用到bunyan的扩展性，自定义一个流来获取错误，然后在自定义的逻辑中调用推送逻辑完成告警。

大概的代码：

```
var ServerChan = require('bunyan-serverchan');
var logger = bunyan.createLogger({
        name: 'myapp',
        streams: [{
                level: 'error',
                stream: new ServerChan({key:'MY_KEY'})
        }]
});
```

首先我们定义了一个 `logger` 用来记录日志，记录到的 `error` 级别以上的日志会送给 `ServerChan` 的实例（一个“stream”）。关于 `ServerChan` ，稍后解释。

接下来，在出错的地方调用 `logger` 记录错误：

```
xxx.on('error',function(err){
        logger.error(err, 'Something went wrong:%s',err.message);
});
```

此时错误的记录部分就算完成了，bunyan会负责将错误信息传递给 `ServerChan` 的实例。

ServerChan模块

在上面的代码中，我们通过 `require('bunyan-serverchan')` 引入了 `ServerChan` ，这个模块负责接受错误信息，并调用Server Chan的URL完成推送。

那这个模块到底是什么呢?

没错，是我写的，欢迎到https://github.com/TooBug/bunyan-serverchan围观。

事实上这个模块的逻辑极其简单，调用构造函数之后会生成一个对象，只要保证这个对向的 `write` 方法是存在的，即可以用于bunyan的自定义stream。也就是说，虽然bunyan的概念中是一个自定义stream，但我们并不需要真的实现一个stream，只需要一个有 `write` 方向的对象即可。

`write` 方向负责接受错误信息，并完成自定义逻辑（推送）。

结

Over，就这么简单。