DBLE研发工作揭秘

爱可生开源社区 阎虎青

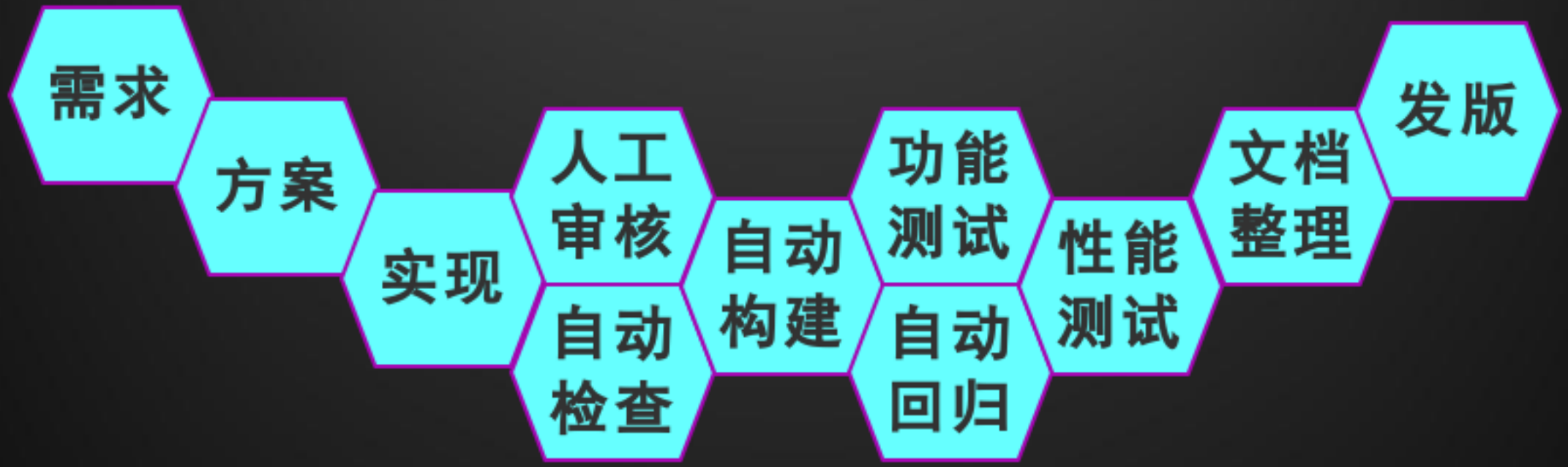
个人介绍

阎虎青

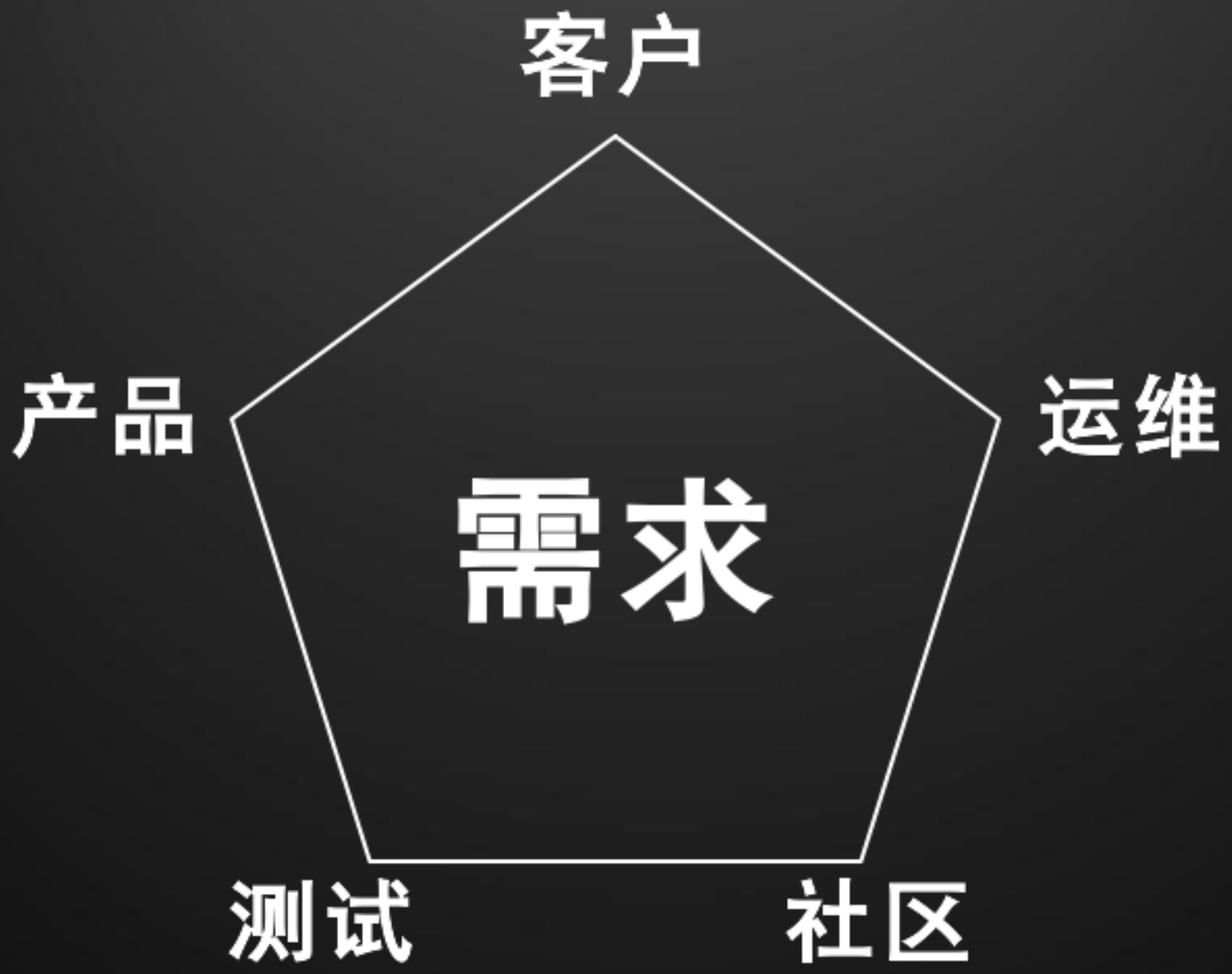
- 玩过读写分离，分库分表，数据备份传输
- 始终在一线从事开发工作
- 目前为爱可生DBLE负责人，兼职段子手

目录
CONTENTS

研发流程



研发流程-需求来源



研发流程-方案

三堂会审

产品:
描述需求
接受反诘

开发:
理解需求
技术风险评审

测试:
理解需求
设计测试点

研发流程-自动化检查

- 单元测试

- 编码规范

- 静态分析

- 可持续集成

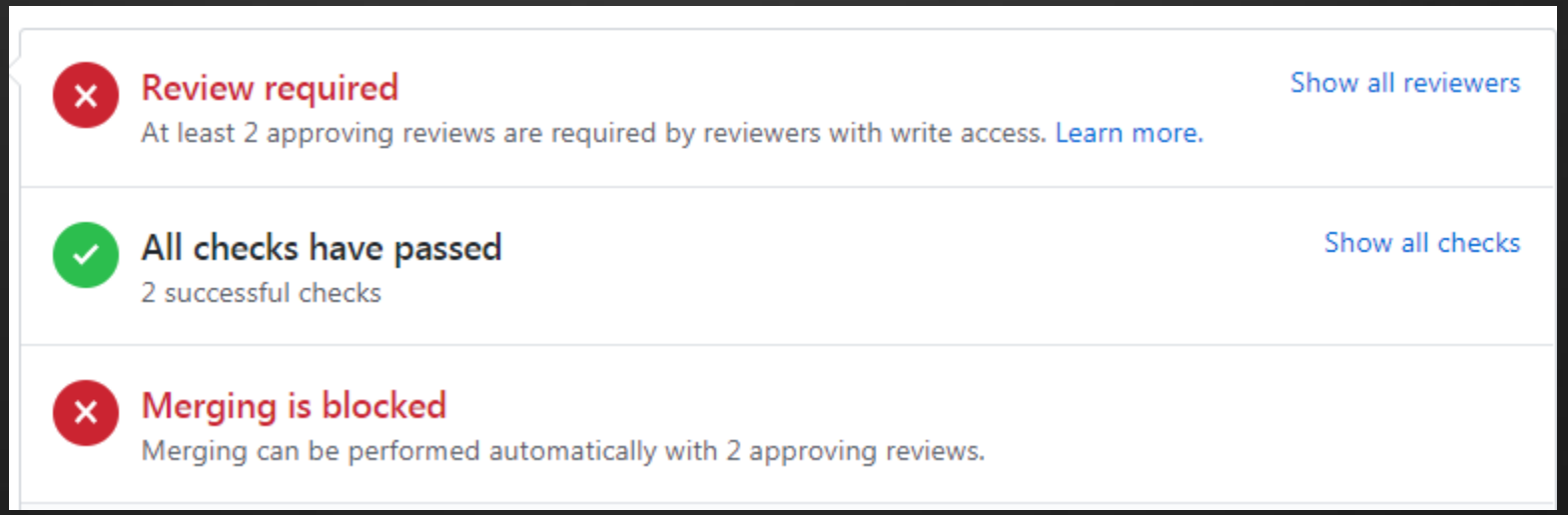
- 安全分析

研发流程-Travis CI

研发流程-Review

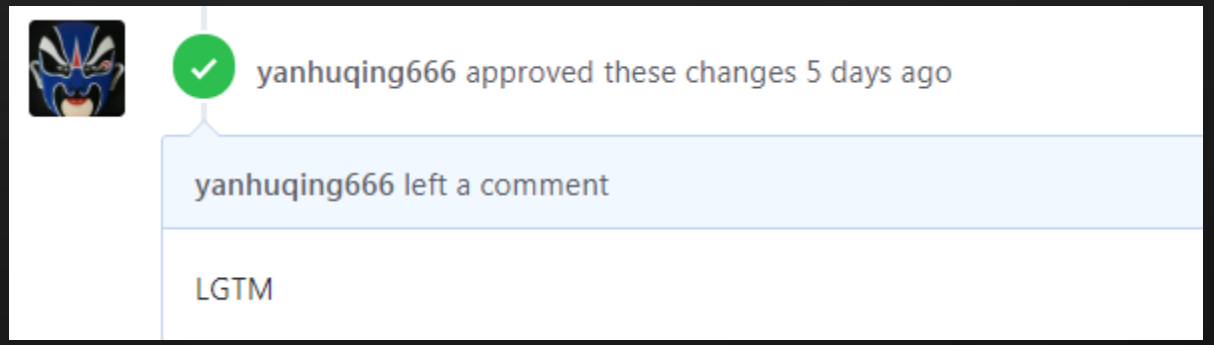
规则:

1.至少2个committer 审核通过
2.自动化检查通过



审核黑话:

- OK
- LGTM(looks good to me)
- ~~AYYY~~



研发流程-测试

- 自动化测试

JACOCO
Java Code Coverage

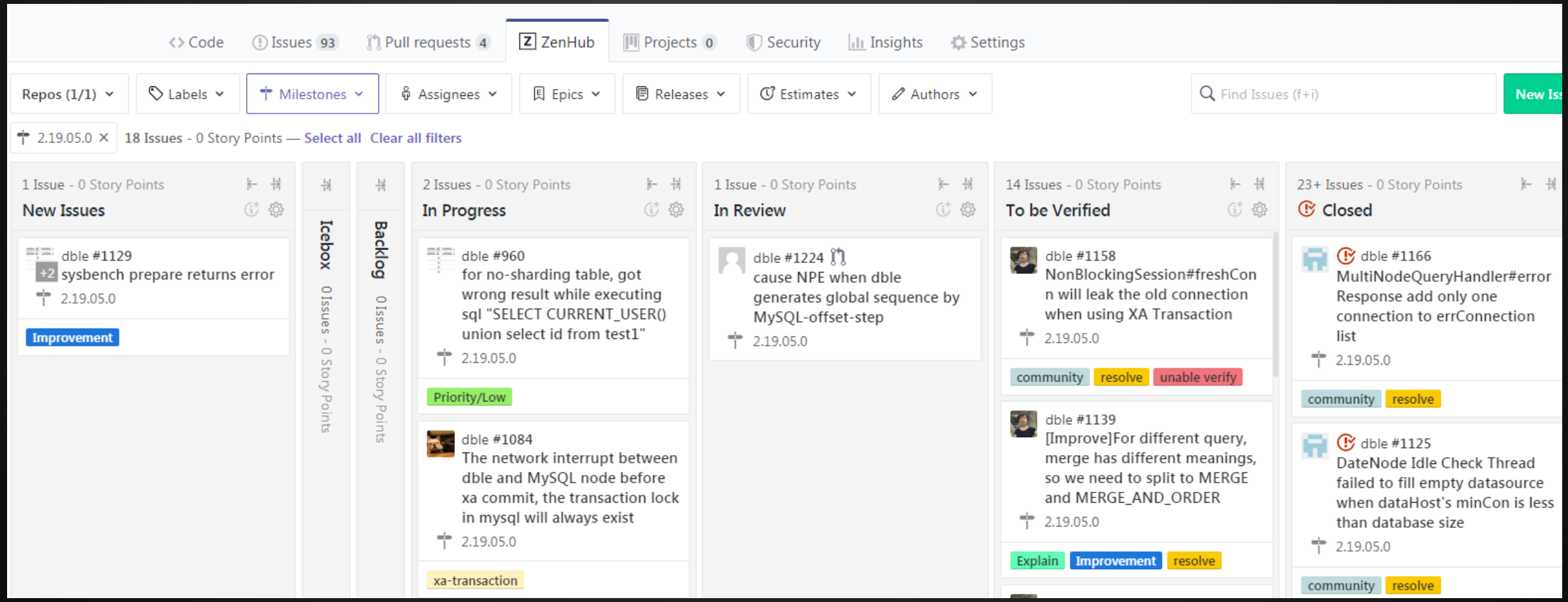
- 用例管理

TestLink

- 功能测试点

ProcessOn 思维导图

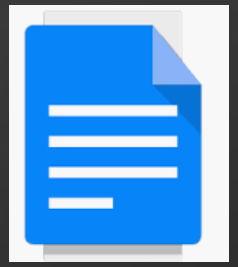
研发流程-流程管理



研发流程-发版

前置项-测试

- 功能测试结束，归档报告
- 自动化测试通过
- 性能回归测试通过



前置项-文档

- 新功能&变更
- 新参数
- 兼容项的缺省值变更
- 整理主项目 Release Notes

前置项-版本相关代码

- show @@help 包含所有新加的功能
- show @@sysparam 包含所有新加的参数
- server/schema/rule 模板文件版本变更

前置项-Docker

- docker镜像构建
- docker & compose

发布

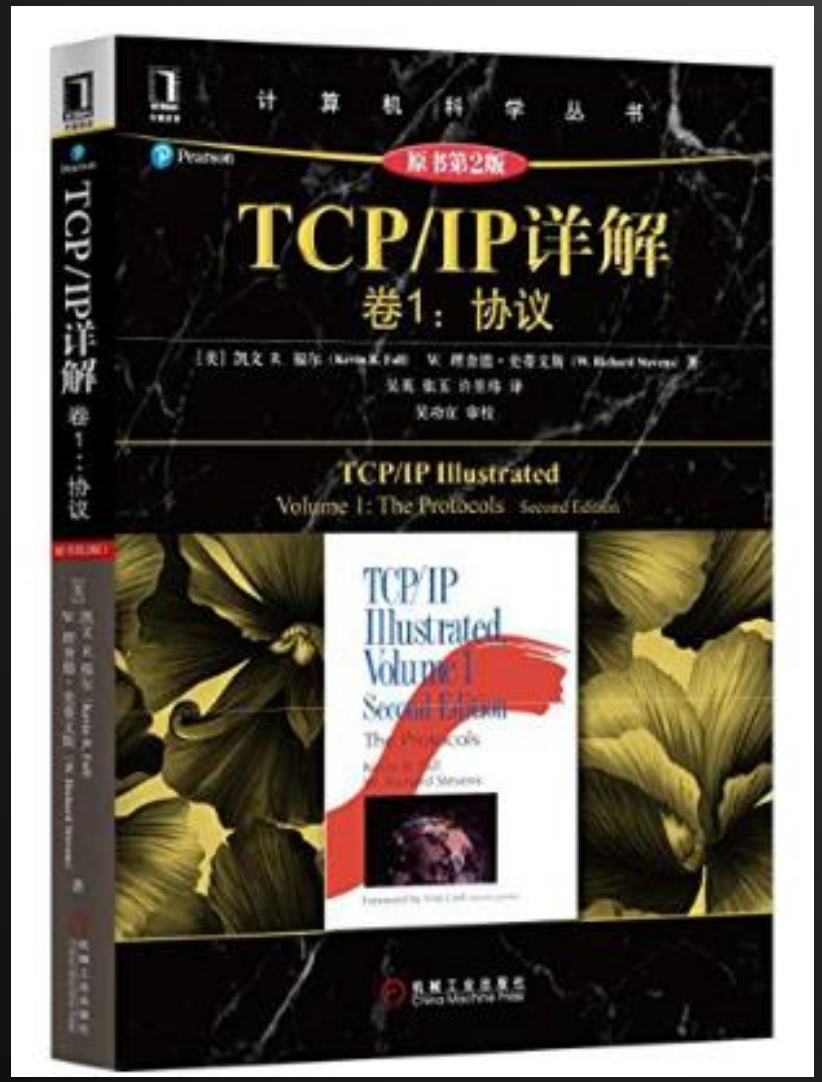
1. 文档 gitbook/pdf
2. 可执行软件包
3. docker镜像

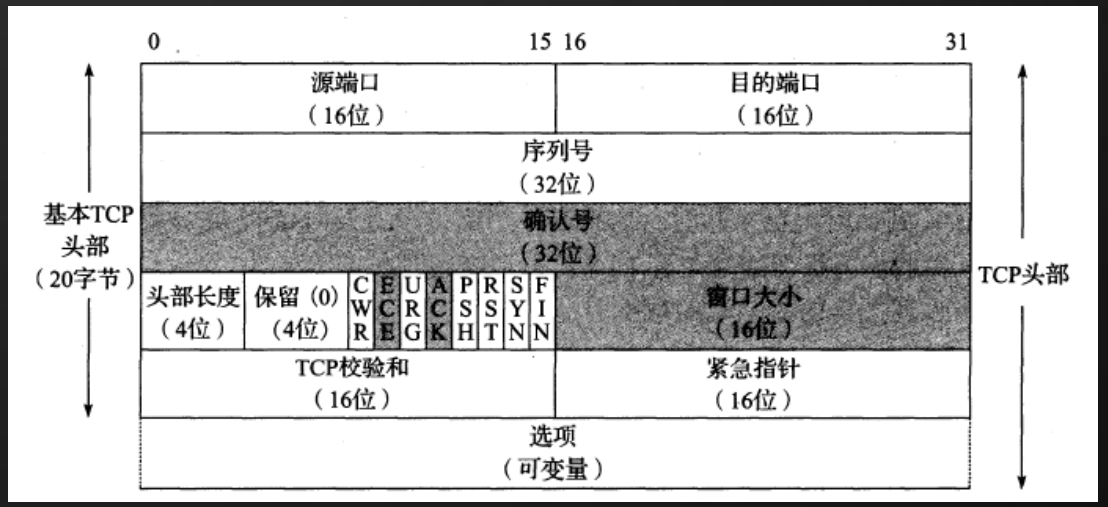
目录
CONTENTS

研发工具-生产力工具

CheckStyle-IDEA

研发工具-协议与抓包





TCP/IP协议： http://www.52im.net/topic-tcpipvol1.html

研发工具-协议与抓包

Chapter 14 MySQL Client/Server Protocol

Table of Contents

Client/Server Protocol

Overview

The MySQL protocol is used between MySQL Clients and a MySQL Server. It is implemented by:

- Connectors (Connector/C, Connector/J, and so forth)
- MySQL Proxy
- Communication between master and slave replication servers

The protocol supports these features:

- Transparent encryption using SSL
- Transparent compression
- A **Connection Phase** where capabilities and authentication data are exchanged
- A **Command Phase** which accepts commands from the client and executes them

Further reading:

- **Protocol Basics**
- **Connection Lifecycle**

MySQL协议：

https://dev.mysql.com/doc/dev/mysql-server/latest/PAGE_PROTOCOL.html
https://dev.mysql.com/doc/internals/en/client-server-protocol.html

研发工具-协议与抓包

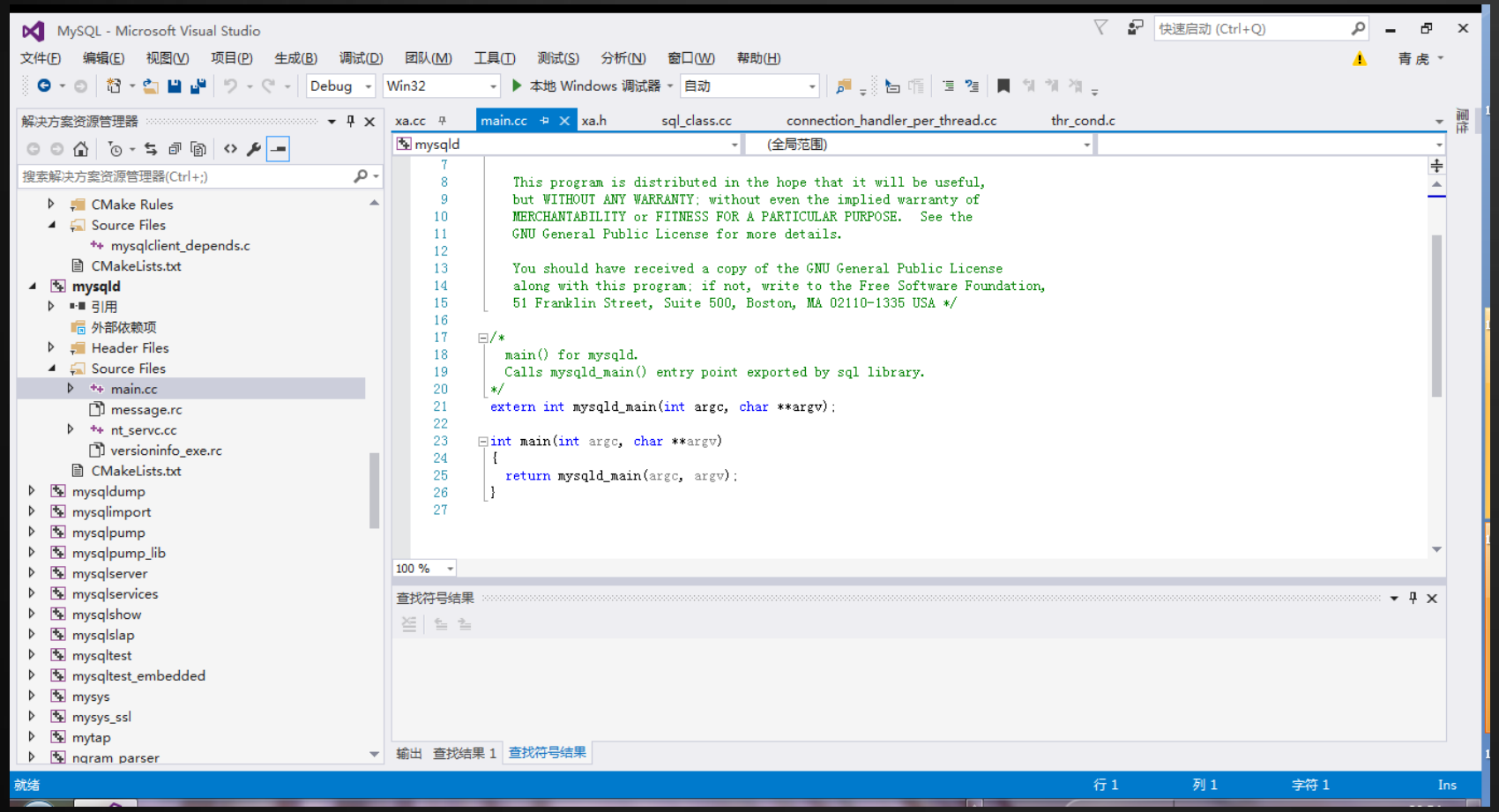
```
root@ubuntu:~# tcpdump  -i eth0 -nn port 3306 and host  192.168.2.206
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
06:21:46.287529 IP 192.168.2.206.52193 > 10.186.61.2.3306: Flags [S], seq 4151688470, win 8192, options [mss 1460,nop,wscale 2,nop,nop,
sackOK], length 0
06:21:46.287655 IP 10.186.61.2.3306 > 192.168.2.206.52193: Flags [S.], seq 4227748850, ack 4151688471, win 29200, options [mss 1460,nop
,nop,sackOK,nop,wscale 7], length 0
06:21:46.289884 IP 192.168.2.206.52193 > 10.186.61.2.3306: Flags [.], ack 1, win 4380, length 0
06:21:46.291163 IP 10.186.61.2.3306 > 192.168.2.206.52193: Flags [P.], seq 1:83, ack 1, win 229, length 82
06:21:46.293591 IP 192.168.2.206.52193 > 10.186.61.2.3306: Flags [P.], seq 1:203, ack 83, win 4359, length 202
06:21:46.293623 IP 10.186.61.2.3306 > 192.168.2.206.52193: Flags [.], ack 203, win 237, length 0
06:21:46.293961 IP 10.186.61.2.3306 > 192.168.2.206.52193: Flags [P.], seq 83:94, ack 203, win 237, length 11
06:21:46.310263 IP 192.168.2.206.52193 > 10.186.61.2.3306: Flags [P.], seq 203:240, ack 94, win 4356, length 37
06:21:46.311125 IP 10.186.61.2.3306 > 192.168.2.206.52193: Flags [P.], seq 94:186, ack 240, win 237, length 92
06:21:46.516610 IP 192.168.2.206.52193 > 10.186.61.2.3306: Flags [.], ack 186, win 4333, length 0
06:21:51.094173 IP 192.168.2.206.52193 > 10.186.61.2.3306: Flags [P.], seq 240:259, ack 186, win 4333, length 19
06:21:51.095800 IP 10.186.61.2.3306 > 192.168.2.206.52193: Flags [P.], seq 186:496, ack 259, win 237, length 310
06:21:51.303328 IP 192.168.2.206.52193 > 10.186.61.2.3306: Flags [.], ack 496, win 4256, length 0
```

TCPDUMP

No.	Time	Source	Destination	Protocol	Length	Info
102	2019-06-11 14:21:45.484029	192.168.2.206	10.186.61.2	TCP	66	52193 → 3306 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
103	2019-06-11 14:21:45.486957	10.186.61.2	192.168.2.206	TCP	68	3306 → 52193 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128
104	2019-06-11 14:21:45.487069	192.168.2.206	10.186.61.2	TCP	54	52193 → 3306 [ACK] Seq=1 Ack=1 Win=17520 Len=0
109	2019-06-11 14:21:45.490198	10.186.61.2	192.168.2.206	MySQL	136	Server Greeting proto=10 version=5.7.21-log
112	2019-06-11 14:21:45.490803	192.168.2.206	10.186.61.2	MySQL	256	Login Request user=action
113	2019-06-11 14:21:45.503627	10.186.61.2	192.168.2.206	TCP	60	3306 → 52193 [ACK] Seq=83 Ack=203 Win=30336 Len=0
114	2019-06-11 14:21:45.503634	10.186.61.2	192.168.2.206	MySQL	68	Response OK
115	2019-06-11 14:21:45.504622	192.168.2.206	10.186.61.2	MySQL	91	Request Query { select @@version_comment limit 1 }
117	2019-06-11 14:21:45.510360	10.186.61.2	192.168.2.206	MySQL	146	Response
122	2019-06-11 14:21:45.713045	192.168.2.206	10.186.61.2	TCP	54	52193 → 3306 [ACK] Seq=240 Ack=186 Win=17332 Len=0
202	2019-06-11 14:21:50.289799	192.168.2.206	10.186.61.2	MySQL	73	Request Query { show databases }
203	2019-06-11 14:21:50.294828	10.186.61.2	192.168.2.206	MySQL	364	Response
213	2019-06-11 14:21:50.499338	192.168.2.206	10.186.61.2	TCP	54	52193 → 3306 [ACK] Seq=259 Ack=496 Win=17024 Len=0

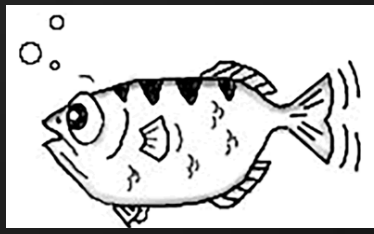
WIRESHARK

研发工具-MySQL源码阅读/调试



Visual Studio

Community 2015



GDB

The GNU Project Debugger

目录
CONTENTS

故障诊断-方法

日志:

- wrapper.log
- dble.log
- gc.log
- 慢查询日志（与故障无关）
- 分布式事务日志（普通实现方式）
- 分布式事务状态日志（XA实现方式）

参考公开课第九节《故障分析》

故障诊断-工具

JDK家族

- jstack
Java的堆栈跟踪工具。

- jstat
JVM统计监控工具。

-jmap
Java内存映射工具。

- jhat
Java堆分析工具 。

- jconsole
监控Java虚拟机的图形工具

Linux命令

- top

  ps -ef | grep java

  top -H -p pid

- lsof /netstat/ss

  查看连接状态

- Dmesg

  系统日志

故障诊断-工具 btrace

- BTrace **是什么**

  Java平台的动态追踪工具

- **用来做什么**

- **生产环境**：在不重启应用的情况下，获取程序执行过程中的信息用诊断
- **测试环境**：非安全方式重现一些难以重现的bug

故障诊断-工具 btrace

怎么使用?

- 安装JDK
- 设置环境变量
- 编写trace脚本
- 运行

典型用法:

<btrace>/bin/btrace <PID> <trace_script>

Thank You