大规模数据库监控的zabbix玩法

ZABBIX 2019 Conference

王伟

数据库架构师，京东商城

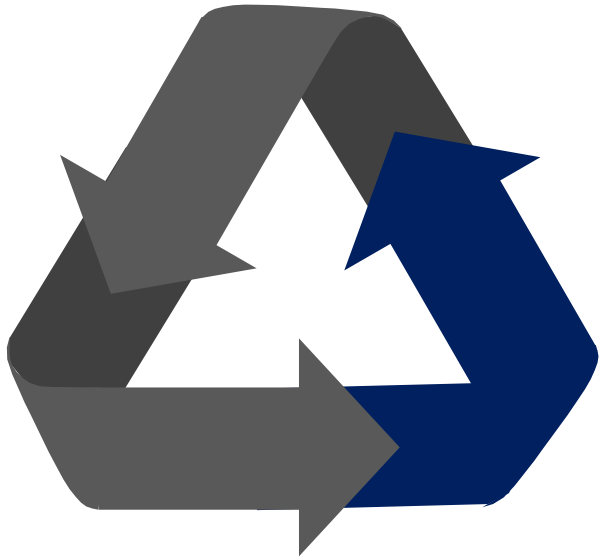
个人简介

王伟，

现任京东商城资深数据库架构师，Oracle ACE for MySQL，ACMUG秘书长。10多年MySQL大规模数据库运维经验，致力于MySQL数据库的性能优化、监控和智能运维工作，努力实现数据库管理的规范化、流程化、自助化、智能化以及可视化。

运维三把斧

监控



高可用

备份

监控特点

监控对象

服务器监控

主要监控服务器如：CPU 负载、内存使用率、磁盘使用率、登陆用户数、进程状态、网卡状态、服务器硬件等。

应用程序监控

主要监控该应用程序的服务状态，吞吐量和响应时间，因为不同应用需要监控的对象不同，这里不一一列举。

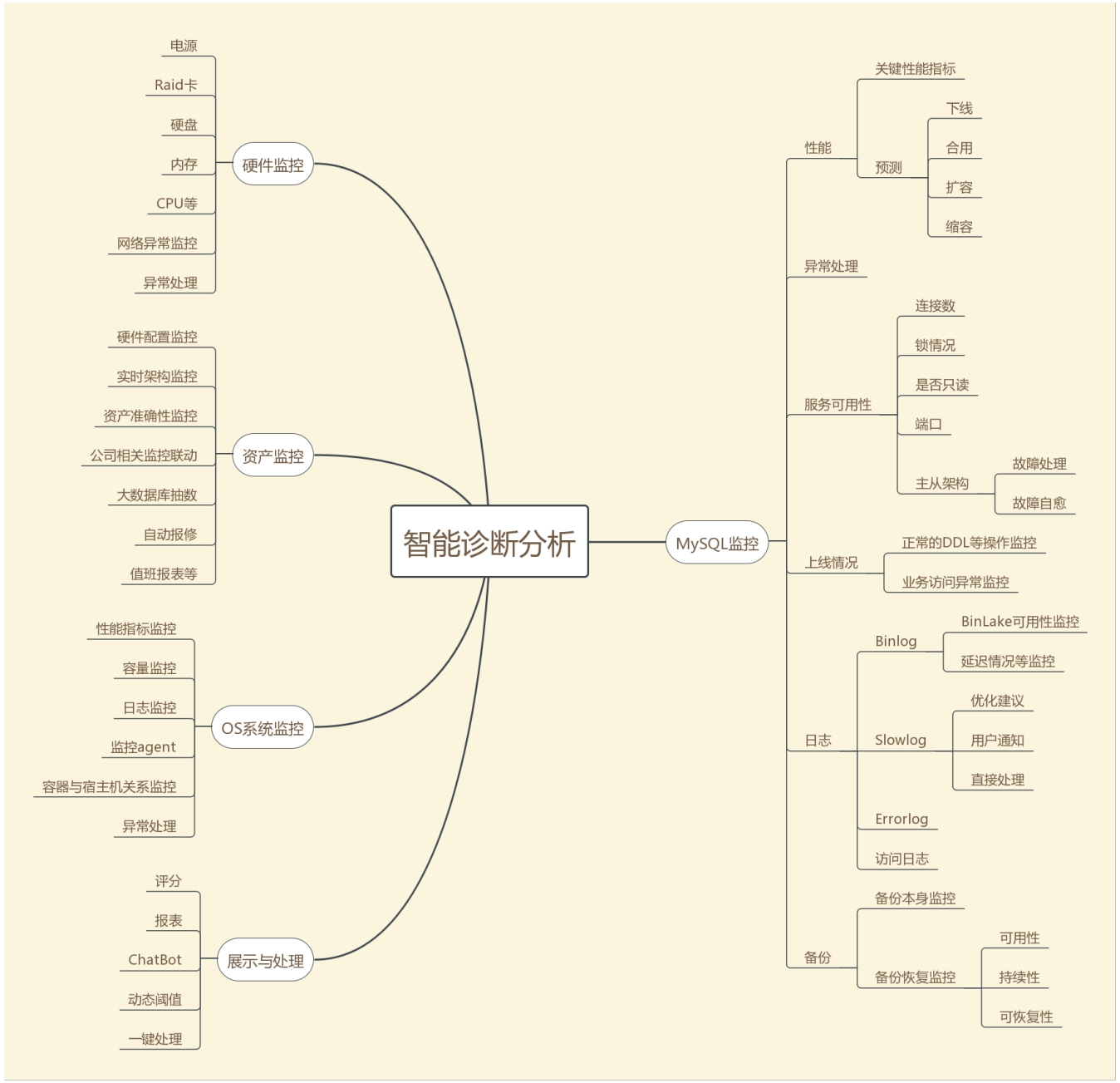
数据库监控

一般监控数据库状态，连接情况，数据库表或者表空间的使用情况，是否有死锁，错误日志，性能信息等等。

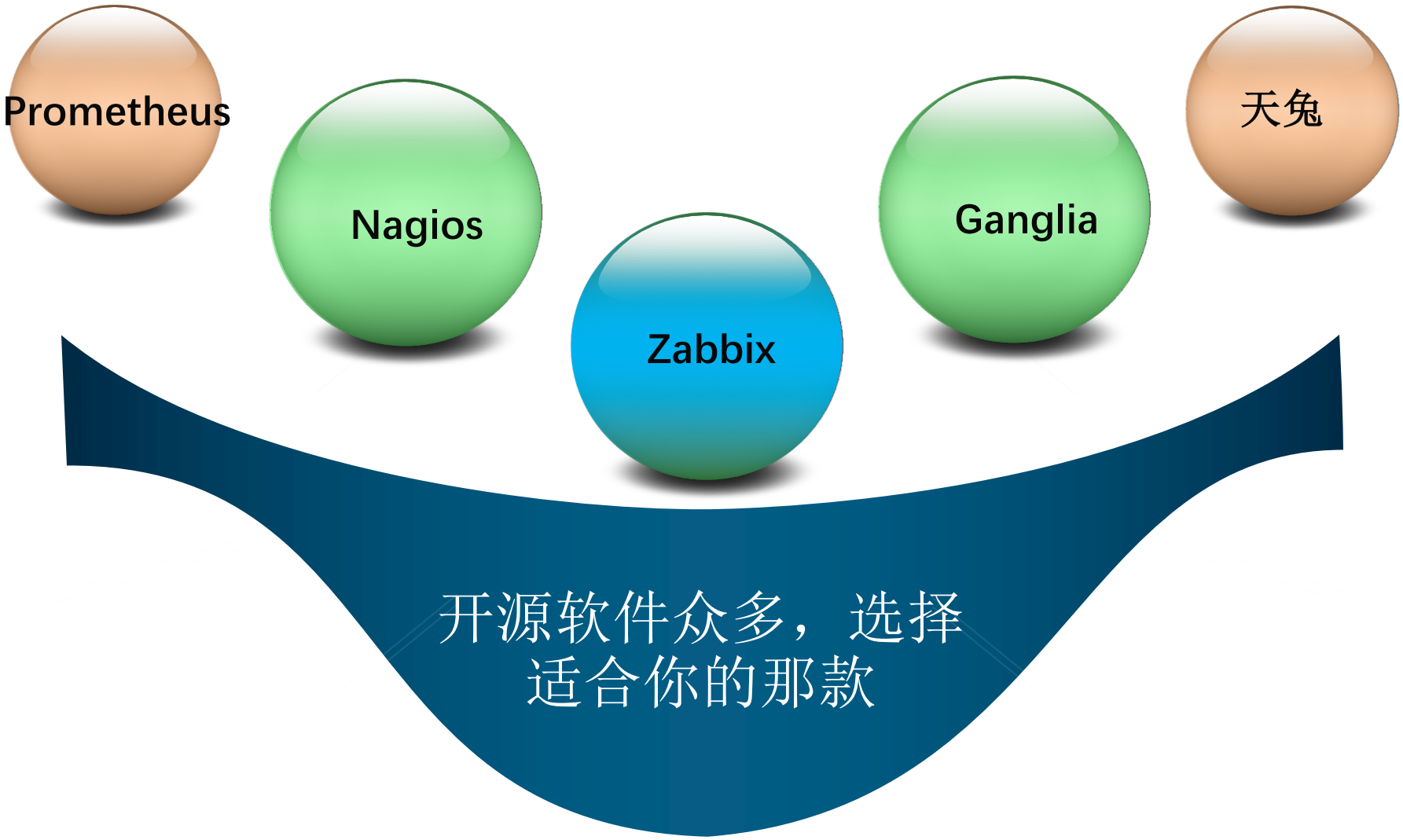
网络监控

主要监控当前的网络设备运行情况、网络状况，网络流量等。

数据库常用监控项


智能诊断分析
硬件监控
电源
Raid卡
硬盘
内存
CPU等
网络异常监控
异常处理
资产监控
硬件配置监控
实时架构监控
资产准确性监控
公司相关监控联动
大数据库抽数
自动报修
值班报表等
OS系统监控
性能指标监控
容量监控
日志监控
监控agent
容器与宿主机关系监控
异常处理
展示与处理
评分
报表
ChatBot
动态阈值
一键处理
MySQL监控
性能
关键性能指标
预测
下线
合用
扩容
缩容
异常处理
服务可用性
连接数
锁情况
是否只读
端口
主从架构
故障处理
故障自愈
上线情况
正常的DDL等操作监控
业务访问异常监控
日志
Binlog
BinLake可用性监控
延迟情况等监控
Slowlog
优化建议
用户通知
直接处理
Errorlog
访问日志
备份
备份本身监控
备份恢复监控
可用性
持续性
可恢复性

监控软件选择



Zabbix优点

1

自动发现服务器和网络设备

这功能有点鸡肋，在多种应用、多种设备混合场景下不实用，给zabbix整体运维管理带来不便。这在实际使用中也存在各种问题，特别是在设备种类繁多、数量较大的情况下，不建议使用这个功能，zabbix监控添加设备结合CMDB来完成，这样定位设备用途和添加模板将会更加准确；

2

底层自动发现

这点很方便实用，比如自带的自动发现系统分区、自动发现多网卡等。这个功能也是可以自定义的，比如监控一台主机上的MySQL多实例服务，通过这个功能可以轻松搞定；

3

分布式的监控体系和集中式的web管理

zabbix支持主动监控和被动监控模式(模式是相对于客户端来说的，主动推监控数据给服务器端或是服务器端来拉取监控数据。建议使用主动模式，以便减轻服务器端压力)， 并且可以实现秒级监控，这点是一些监控软件达不到的，但对重要业务来说，这点很重要；

Zabbix优点

4

支持范围广

支持监控多种设备以及目前市面常见的各种OS、服务、日志等，可以使用自带的agent监控，也有无agent监控等多种监控方法，如SNMP；

5

灵活的监控项设置

zabbix本身已经支持很多常见的监控项，用户也可以自己写脚本来灵活自定义监控项，可以灵活组合多项报警阈值来准确报警，如监控硬盘的报警阈值可以设置为达到硬盘空间80%并且剩余空间低于50G时报警；

6

高水平的业务视图监控资源，监控情况展示方面可以垂直、水平对比展示。

比如一套数据库的分片，可以把所有主库的某个性能指标做在一个Graphs中，可以方便对比各个主库的负载情况是否均衡。也可以将多个Graphs做成一个Screens，然后在一个Screens中可以看到多种性能指标的各种情况，方便直观的进行对比；

Zabbix优点

7

灵活的用户权限设置

支持自定义事件和邮件发送，也支持报警升级及日志审计；

8

基于zabbix报警的故障自愈。

Zabbix具有规范化的故障处理流程，对报警进行分级、分类，可以自动处理一些低级别、固化处理方法的故障，以达到快速恢复故障的效果。这点很重要，做的好可以最大限度的保证业务的可用性稳定性，降低人为操作失误风险以及人员成本。

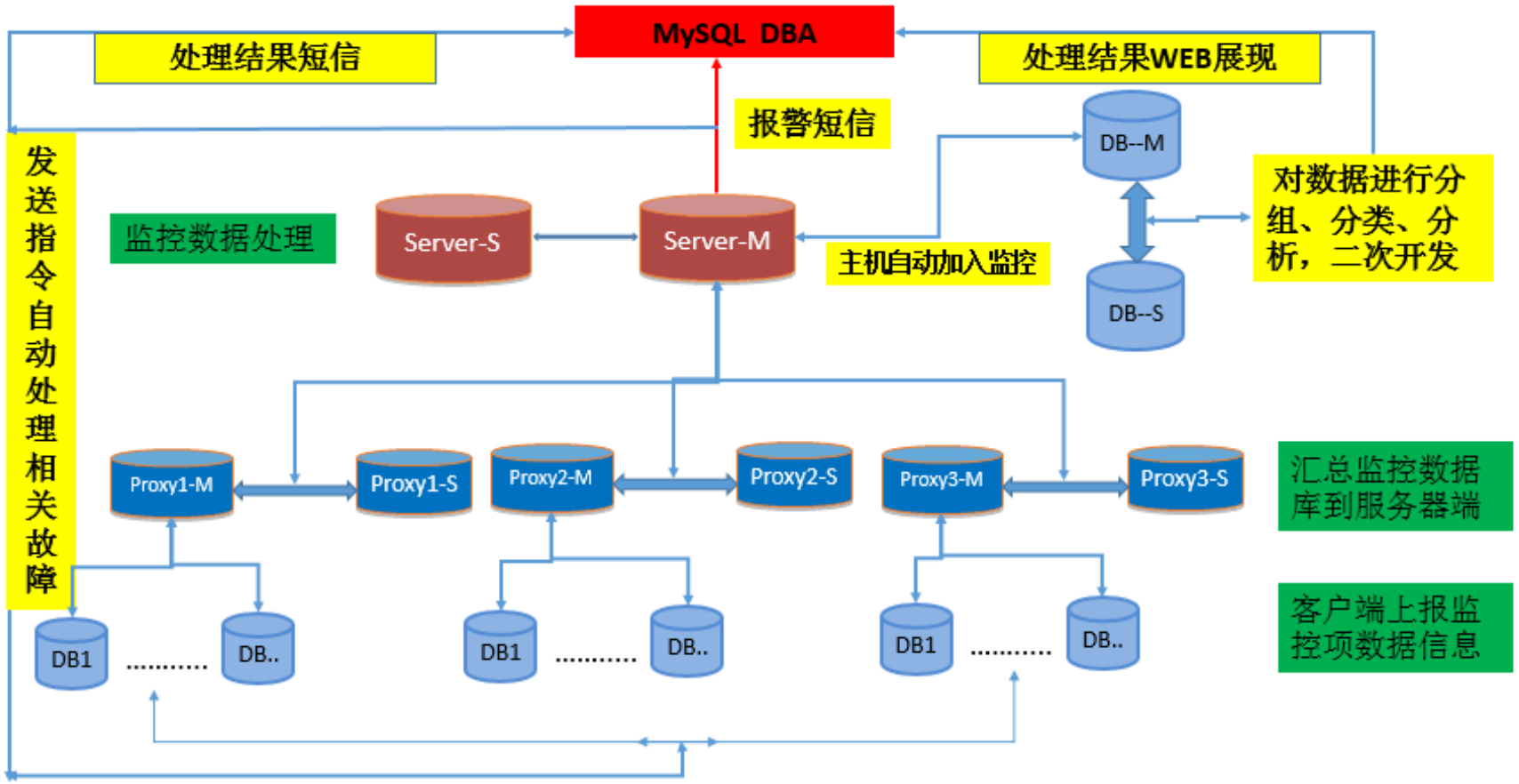
9

强悍的内置API。

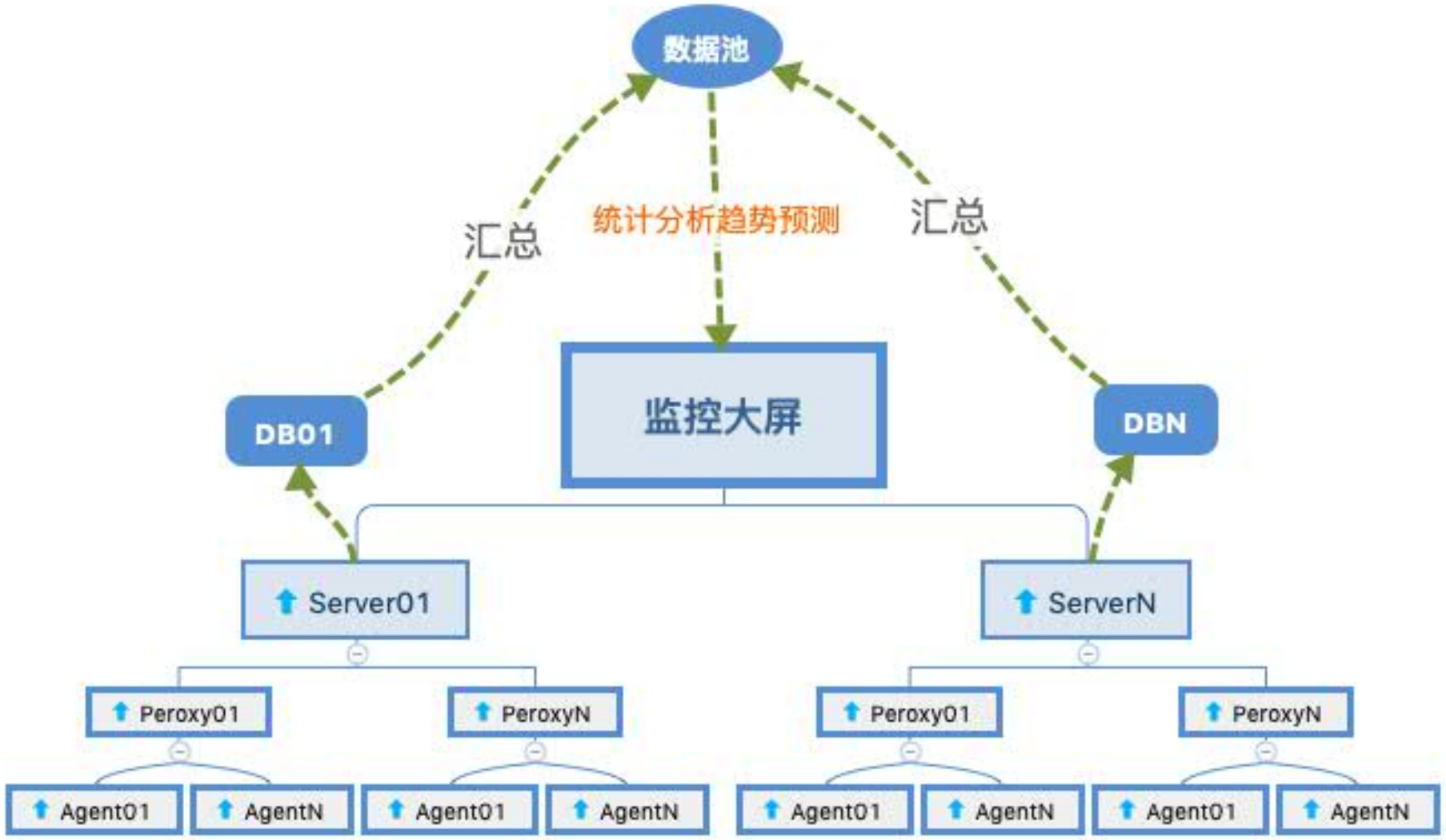
几乎所有的zabbix服务器端web页面配置操作，都可以通过他自身的API来完成，用户可以非常方便地对它进行二次开发，以满足自己的自动化运维需求。

Zabbix单Server 架构

基于zabbix的智能监控系统流程图：



Zabbix多Server 架构



监控规范

1. 定义告警优先级策略
2. 定义告警信息内容标准
3. 本地监控脚本的规范化
4. 故障业务自我修复功能
5. 监控项和主机名规范化
6. 监控数据汇总分析
7. 监控业务系统进行分级
8. 监控范围及目标

Zabbix优化

配置文件优化

主要对进程数量、缓存大小、超时时间参数进行调整，禁用掉如VMware、Java等方面不使用的监控方式。

监控项优化

精简监控项，只监控必要的监控项，监控项的类型最好使用数字，尽量避免使用字符，数据采集频率设置得当。

触发器优化

在使用过程中，尽量选择速度较快的函数。配置Trigger时，也应注意使用正确的逻辑，错误的逻辑可能导致数据库查询较慢的现象。

数据库优化

对监控数据表进行分区，根据自身的情况设置合适的分区删除策略，同时关闭自身的历史数据删除策略。

Zabbix触发器

Trigger中，正则表达式函数last(),nodata()的速度最快，min()、max()、avg()的速度最慢。在使用过程中，尽量选择速度较快的函数。配置Trigger时，也应注意使用正确的逻辑，错误的逻辑可能导致数据库查询较慢的现象。

- 表达式 1 :
  {www.zabbix.com:system.cpu.load[all,avg1].last(0)}>5
  load1> 5

- 表达式2:
  {www.zabbix.com:system.cpu.load[all,avg1].last(0)}>5|{www.zabbix.com:system.cpu.load[all,avg1].min(600)}>2
  load1>5或load连续10分钟大于 2

- 表达式 3 :
  {smtp1.zabbix.com:net.tcp.service[smtp].last(0)}=0&{smtp2.zabbix.com:net.tcp.service[smtp].last(0)}=0
  两台器的SMTP同时故障

表达式4:
{zabbix.zabbix.com:tick.nodata(180)}=1
三分钟没有收集到数据

表达式5:
{zabbix:system.cpu.load[all,avg1].min(300)}>2&{zabbix:system.cpu.load[all,avg1].time(0)}>000000&{zabbix:system.cpu.load[all,avg1].time(0)}<060000
00:00-06:00之间load1连续五分钟大于 2

表达式6：
{public monitor:system.cpu.load[,avg1].abschange(0)}<9
最后一次的load 1 值减前一次的load 1 值，共结果的绝对值小于 9

表达式7:
{Zabbix server:tcp,21.delta(300)}>0
五分钟内最大值和最小值之差大于 0

表达式8:
{public.monitor:system.cpu.load[,avg1].avg(600)}
>
{publicmonitor:system.cpu.load[,avg1].avg(600,86400)}+10
当前 1 小时的平均load1>一天以前1小时的平均load1加10,avg时间默认为S，也可以为次数,如avg（#600）代表600次的平均值

表达式9
{public monitor:system.cpu.load[,avg1].count(600,12,” gt”)}>20
最后600秒locd大于12的次数大于20

表达式10
{public monitor:system.cpu.load[,avg1].count(#10,12,” gt” ,86400)}>20
24小时以前最后10次load大于12的个数大于20

趋势预测
{host:vfs.fs.size[/,free].forecast(1h,,10h)}<=0
{host:vfs.fs.size[/,free].timeleft(1h,,0)}<=10h
基于1小时的数据，预测10小时可用空间小于等于0报警

监控数据分析

容量管理

- 性能指标监控
- 集群数据汇总
- 性能瓶颈分析

性能指标

- 数据库及OS性能指标分析
- 延迟及响应时间监控

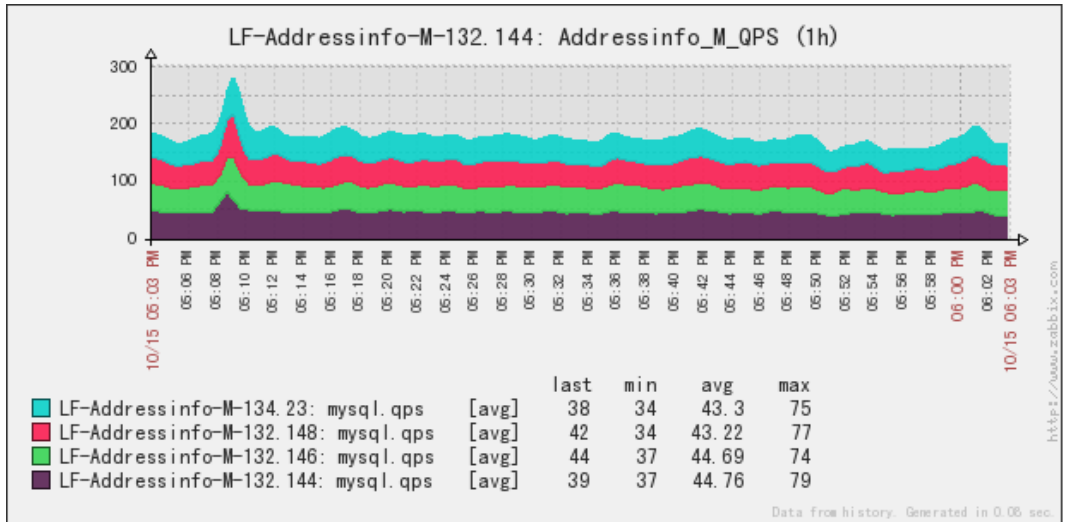
日志监控

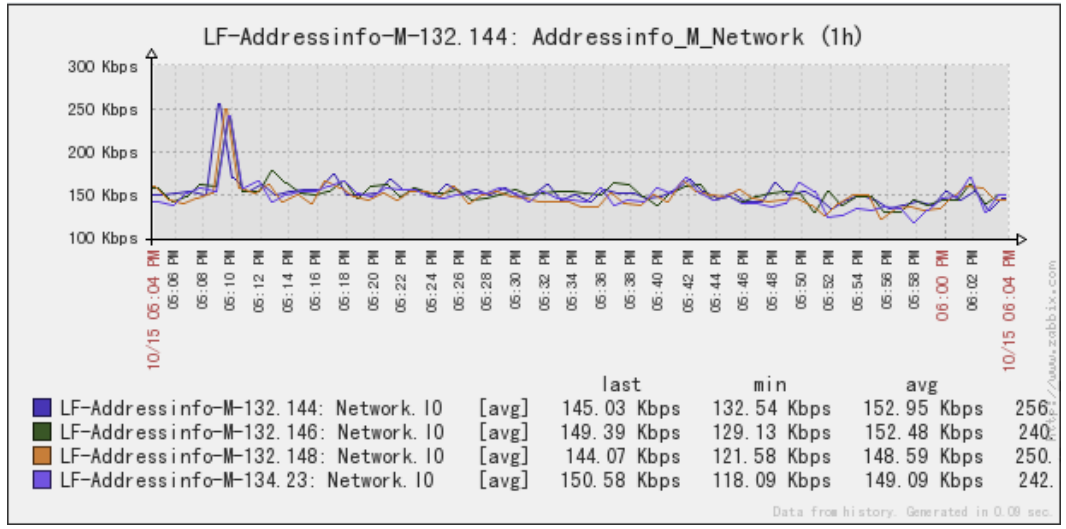
- 数据库日志监控
- 慢查询日志
- 文件监控

数据分析

- 同比环比数据
- TOP数据分析
- 汇总数据统计

集群数据汇总对比


LF-Addressinfo-M-132.144: Addressinfo_M_QPS (1h)
300
200
100
0
10/15 05:03 PM
10/15 06:03 PM
last min avg max
LF-Addressinfo-M-134.23: mysql.qps [avg] 38 34 43.3 75
LF-Addressinfo-M-132.148: mysql.qps [avg] 42 34 43.22 77
LF-Addressinfo-M-132.146: mysql.qps [avg] 44 37 44.69 74
LF-Addressinfo-M-132.144: mysql.qps [avg] 39 37 44.76 79
Data from history. Generated in 0.05 sec.


LF-Addressinfo-M-132.144: Addressinfo_M_Network (1h)
300 Kbps
250 Kbps
200 Kbps
150 Kbps
100 Kbps
10/15 05:04 PM
10/15 06:04 PM
last min avg
LF-Addressinfo-M-132.144: Network.IO [avg] 145.03 Kbps 132.54 Kbps 152.95 Kbps 256.
LF-Addressinfo-M-132.146: Network.IO [avg] 149.39 Kbps 129.13 Kbps 152.48 Kbps 240.
LF-Addressinfo-M-132.148: Network.IO [avg] 144.07 Kbps 121.58 Kbps 148.59 Kbps 250.
LF-Addressinfo-M-134.23: Network.IO [avg] 150.58 Kbps 118.09 Kbps 149.09 Kbps 242.
Data from history. Generated in 0.09 sec.

故障自愈

磁盘空间调整

Bin-log、备份文件、
MySQL日志等清理

CPU过高处理

业务允许下kill超长SQL
并发送给研发

MySQL参数调整

连接数、刷硬盘参数的
自动调整

宕机处理

自动切换与尝试重启等
操作

基于微信的交互式分类故障处理，让运维移动端快速处理线上复杂情况的故障，让故障自愈更加人性化、智能化。

对Zabbix一些期待

1：zabbix主机注册时间没有固定的字段保存，只能在审计日志中查历史，但日志是不可能永久保留的。

2：分布式监控大屏：不大利于大规模监控，大规模的监控不只是代理的分布式，还有server的分布式，满足跨机房汇总多个server监控信息的功能，类似早期的node那种分布式，但只需要做汇总报警等只读报表展示即可。

3：配置下发到代理的进程只有一个，代理较多的时候这个进程负载较高，可以修改成在配置文件中自定义进程数量的类型。

4：运维自动化、智能化的精髓在无人值守，复杂的东西简单化，目前有些功能配置比较复杂，比如接收Prometheus的监控数据的配置，太繁琐了。

5：SQL问题，zabbix中很多拼接出来的SQL语句，数据量大的时候效率低下，甚至全表扫描；在页面更新模板时效率极其低下（包含更新监控项和触发器），如果是大量实例的模板，更新是很恐怖的事情；一般的单个值的更新，其实是很简单的一条SQL就可以搞定：

update zabbix.items set delay=5 where key_='icmpping';

update zabbix.items t,zabbix.functions f set f.parameter='30' where f.itemid=t.itemid and t.key_='agent.ping’ ;

6、报警收敛：实例的多种报警合并，减少报警个数


让监控不止于监控，
让运维不止于运维。

加入组织

扫码入群

关注公众号

关注微博

联系我们


021-6978-6188


china@zabbix.com


www.zabbix.com/cn
www.grandage.cn


Zabbix开源社区


Thank You!
ZABBIX 2019
Conference