x项目非功能测试报告

# 性能测试

铺底数据：x万

## 单交易并发

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **X项目单交易并发测试结果-200个并发、执行5分钟** | | | | | | | |
| **测试交易** | **整个链路** | | **区块链底层** | | **区块链底层响应时间占比** | **资源使用详情** | **资源附件** |
| **TPS** | **响应时间(s)** | **TPS** | **响应时间(s)** | **底层时间/链路时间(%)** |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

有无丢数：

## 混合场景

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **X项目混合交易测试结果-200个并发、执行5分钟** | | | | | | | | |
| **测试交易** | **并发数** | **整个链路** | | **区块链底层** | | **区块链底层响应时间占比** | **资源使用详情** | **资源附件** |
| **TPS** | **响应时间(s)** | **TPS** | **响应时间(s)** | **底层时间/链路时间** |
|  | 150 |  |  |  |  |  |  |  |
|  | 50 |  |  |  |  |  |
| 汇总 | 200 |  |  |  |  |  |

有无丢数：

## 稳定性测试

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **应用稳定性测试结果-200个并发、执行12小时** | | | | | | | | |
| **测试交易** | **并发数** | **整个链路** | | **区块链底层** | | **区块链底层响应时间占比** | **资源使用详情** | **资源附件** |
| **TPS** | **响应时间(s)** | **TPS** | **响应时间(s)** | **底层时间/链路时间** |
|  | 150 |  |  |  |  |  |  |  |
|  | 50 |  |  |  |  |  |
| 汇总 | 200 |  |  |  |  |  |

# 高可用测试

## 1.测试结论

例：

1.高可用的结构下，停掉一个应用，流量能够结转，重新启动应用后，流量又能被分担。

2.kill -19挂起一个应用，流量能够结转，kill -18恢复应用后，流量又能被分担。

## 2.测试场景

场景1.两个x应用同时运行，nginx采用轮询的方式负载均衡，构成高可用的结构，带载的情况下，停掉一个应用，检查请求是否都发送到另一台正常的应用上，异步请求有没有失败，端到端有没有丢数。

测试结果（例）：

1. 在一个应用停掉的情况下，后续的请求都发送到正常的应用上，异步请求的成功率为100%，端到端没有丢数。
2. 停掉一个应用后，由于所有请求都发送到一个应用上，所以响应时间会提高一倍，TPS会降一半，这是正常的，因为压力都集中在一台应用上了。

场景2.带载下，启动停掉的应用，请求能够重新被分担，没有丢数。

测试结果（例）：

1. 请求能够重新被分担，异步请求的成功率为100%，没有丢数。
2. 重新启动停掉的应用后，由于请求被负载均衡了，所以响应时间会降一半，TPS会提高一倍，这是正常的，因为压力被分担了。

场景3.带载下，使用kill -19挂起一个应用的进程，检查流量是否会切换，使用kill -18恢复进程后，检查是否能恢复负载均衡。

测试结果（例）：

1. 挂起进程时，流量能够切换，异步请求都成功，响应时间增长一倍，TPS下降一半，因为负载全在一台应用上。
2. 恢复进程后，响应时间和TPS又恢复到原来的水平，恢复负载均衡。

# 安全测试

## 1.主要问题

## 2.详细描述

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 项目 | BugNo | 状态 | 主题 | 描述 |
| 项目名称 |  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |