分布式系统可用性分析和提升

- 可靠性
  - 可靠性定义
  - 硬件可靠性
    - 磁盘
    - 电源
    - 网络
  - 提升可靠性
    - 提升硬件可靠性
- 可用性
  - 可用性定义
- 分布式系统可用性
  - 可用性的观测指标
  - 可用性提升
    - 提升系统MTBF
    - 降低系统MTTR

可靠性

可靠性定义

$$R = exp(-\frac{t}{t_{MTBF}})$$

Where ↓

R为可靠性; ↓

t为系统运行时间; ↓

t_{MTBF}为平均故障间隔时间;

根据上面的公式可知：当t_{MTBF}确定时，随着时间t的增大，$R \to 0$；

当时间t确定时，随着t_{MTBF}的增大，$R \to 1$；

当$t = t_{MTBF}$时，$R = 38.6\%$。说明一个系统运行t_{MTBF}之后，不发生故障的概率为$p = 38.6\%$，发生故障的概率为$q = 1 - p = 61.4\%$ 。

工程中，使用一阶近似：即认为$R_{t=t_{MTBF}} = 50\%$和$R_{t=2*t_{MTBF}} = 0$。

硬件可靠性

磁盘

根据硬盘厂商提供的$t_{MTBF}=140$万小时，那么运行70w小时发生故障的概率为50%，运行140w小时发生故障的概率为100%。硬盘的年故障率：

$$1-R_{disk}=1-exp(-\frac{365*24}{140*10^4})=0.6\%$$

实际磁盘的年故障率：$3\%-8\%$。

电源

电源的$t_{MTBF}=10$万小时，那么电源的年故障率为：

$$1-R_{power}=1-exp(-\frac{365*24}{10*10^4})=8.4\%$$

总结：网络、主板 和内存也有上面的类似年故障率。

提升可靠性

串联系统的可靠性：

$$p=\prod_{i=1}^{N}p_i$$

并联系统的可靠性：

$$p=1-\prod_{1}^{N}(1-p_i)$$

结论：串联系统降低系统可靠性，并联系统提高系统的可靠性。

提升硬件可靠性

单机的可靠性提升：

磁盘：可以多备份(磁盘阵列)。

电源：双电源。

网络：双网接入+专线+双交换机。

分布式系统(多机)：

并联多台机器进行数据存储(GFS)。

可用性

可用性定义

$$A = \frac{t_{MTBF}}{t_{MTBF}+t_{MTTR}}$$

Where

A为系统的可用性

t_{MTBF}为系统的平均故障间隔时间，为系统平均可用时间

t_{MMTR}为系统的平均故障恢复时间，为系统平均不可用时间

结论：提升系统的可用性，增加t_{MTBF}和降低t_{MTTR}

分布式系统可用性

分布式系统各个系统相互依赖，这些系统可以分为**可降级的系统**和**不可降级的系统**。可降级的系统我们可以不考虑它对整个系统可用性的影响；而对于不可降级系统构造了系统的核心路径，其中任何一个系统的不可用将导致整个系统的不可用。**对于无法缩短的核心链路，我们可以通过提高这些不可降级模块的可用性。**

可用性的观察指标

可以以年为单位，统计这一年内的**可用时间**$t_{可用}$和一年内的**不可用时间**$t_{不可用}$，通过上面的公式进行计算和统计。同样可以减小粒度以月为单位进行统计和计算。

可用性提升

可用行的提升可以归纳为两点：**降低故障发生概率**和**快速恢复故障，降低故障发生概率**就是在增大**可用时间，故障快速恢复**就是在减小**不可用时间。**

影响系统可用性的几个主体：

1、人

2、硬件平台

3、软件系统

人

1、硬件平台搭建和运维人员

2、系统架构人员

3、软件系统的开发人员

4、软件系统的测试人员

5、软件系统的部署和运维人员

6、软件系统的值班人员

硬件平台

1、

软件系统

1、存储系统

2、中间件系统

3、RPC系统

4、业务系统

方法论：

0、系统要有合理的层级架构、隔离耦合、扇入扇出、容灾部署。

1、合理设计系统的核心链路模块和可降级模块。

2、合理解决系统中的单点问题。

3、合理的超时、拒绝和防雪崩。

4、完善的系统监控。

5、日志追踪定位系统。

6、完善的硬件监控和报警系统。

7、自动化的版本管理、部署、发布和回退系统。

8、完善的单元测试、回归测试、流程测试、预发布灰度和线上灰度。

结论

可靠性作为系统故障概率的衡量，而可用性是系统可用性的衡量。两者之间虽然不存在直接的关系，但是更高的可靠性必然带来更高的可用性。可靠性可以通过多备份的方法来增加系统的可靠性，系统的可用性却是一个综合性的治理工程。此外可用性的治理不像可靠性通过合理的备份就有一个可衡量的提升，而是一个不断迭代、反馈和优化的过程。

作为一个系统的设计者在设计一个新的系统的时候，在初期的时候可能更多的考虑的是系统能否正常满足需求和工作，但是当系统成熟和稳定后更多的精力是投入在可用性的治理上。

对于可用性的治理：

1、单点治理

2、网络治理

3、容灾治理

4、架构治理

- 4.1、层级治理
- 4.2、降级治理
- 4.3、解耦治理
- 4.4、扇入扇出治理
- 4.5、隔离治理

5、业务治理

- 5.1、业务模型重构治理
- 5.2、代码重构治理

6、监控治理

7、定位治理

8、告警治理

9、版本治理

10、灰度治理