分布式系统中失效检测器研究①

Research on Failure Detector in Distributed Systems

董　辉　雷大军　(湘南学院 物理系 湖南 郴州 423000)

摘　要： 失效检测器是分布式系统容错的重要手段，失效检测器性能直接影响到分布式系统的容错能力。本文针对根据分布式系统中的失效类型，对相应的失效检测器进行了研究，介绍失效检测的定义、设计方法，分析对失效检测器服务质量的要求。最后指出了失效检测器的进一步研究方向。

关键词： 失效 容错 不可靠失效检测器 拜占庭失效检测器

1　引言

在分布式系统中，由于故障的发生，进程失效是不可避免的。一个进程或者是正确的，或者是失效的。正确的进程会按照程序规定执行正确动作，而根据进程失效后的行为，可将失效分为以下两类[1]：

a) 崩溃－停止失效(crash-stop failure)。当进程出现崩溃－停止失效，进程会停止执行，不会再接收或发送任何消息。这种失效一般由主机崩溃或者进程异常中止所导致。

b) 拜占庭失效(Byzantine failure)。当出现拜占庭失效，进程不会停止执行，但也不会按照相应算法接收或发送消息。这种失效一般由逻辑错误或者是恶意攻击所导致。

在分布式系统中，失效检测是故障恢复、动态重启、可靠性通信、集群管理等功能的基础。不同副本分布在不同的主机上，要检测副本是否出现失效，必须通过消息交换的方式。针对副本失效行为，失效检测也可以分为对崩溃－停止失效的检测和对拜占庭失效的检测。

根据被检测对象的行为，失效检测器可以分为不可靠失效检测器(Unreliable Failure Detector)[1]和拜占庭失效检测器(Byzantine Failure Detector)[2]。其中，不可靠失效检测器方法是基于超时机制，而拜占庭失效检测器方法主要是基于密码机制和状态机方法。接下来，本文将对失效检测器的相关概念、分类、设计方法和失效检测服务质量进一步进行研究。

2　不可靠失效检测器

Chandra 和 Toueg 提出有关在异步系统中使用不可靠的失效检测器(Unreliable Failure Detector, UFD)的问题[3]。一个不可靠失效检测器可以产生下列两个值之一：Unsuspected 和 Suspected。这两种结果都只是提示，这种提示可能精确的反映了进程的故障情况，也有可能是不精确的。提示值 Unsuspected 表示检测器最近已经收到表明进程没有故障的证据，例如，最近从某一进程收到一个消息。提示值 Suspected 表示有迹象表明进程可能已经出故障了。例如，在最长失效检测时间内没有收到来自进程的消息。

2.1 失效检测方法

基于超时机制的失效检测技术有两种最基本的检测方法：推(Push)和拉(Pull)方法[4]。

2.1.1 推(Push)方法

失效检测的 Push 方法检测过程如图 1 所示，被检测主机向检测者周期性发送心跳(heartbeat)消息，如果在某个时间间隔内检测者没有收到被检测主机发来的心跳消息，则怀疑被检测主机失效。在异步系统中，失效检测器不可靠，如果检测者再次收到被检测主机的心跳消息后可以取消怀疑。

① 基金项目:湖南省教育厅科研课题(07C721);湖南省教育厅 2008 年优秀青年基金资助项目(08B073)
收稿时间:2008-10-15

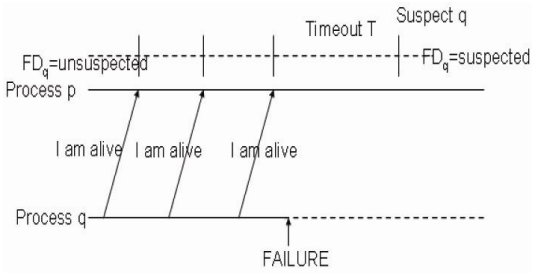

图 1　Push 模式

2.1.2 拉(Pull)方法

失效检测的 Pull 方法检测过程如图 2 所示，检测者主动向被检测主机发送探测消息，例如 ping 被检测主机，被检测者在收到这样的询问消息后，就会向检测者报告自己的存活状态，如果检测者在规定时间内没有收到被检测主机的回复，就怀疑被检测主机失效。

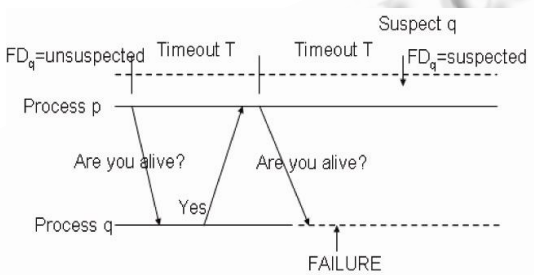

图 2　Pull 模式

上述两种方法中，很明显，Push 方法完成一次检测在网络中只需发送一条消息，而使用 Pull 方法时来回需要两条消息，因此 Push 方法对网络造成的负载要小一些，但 Pull 方法比较灵活，当减少探测消息数量时，Pull 方法的网络负载会远远小于 Push 方法。当然，减少探测消息数量可能会导致检测不到失效。

2.2 失效检测器属性及分类

失效检测器有两种最基本的特性：完整性(Completeness)和准确性(Accuracy)。完整性设定了对失效进程的检测条件，准确性限定了失效检测器所能犯的错误。Chandra 和 Toueg 对这两个特性，又进行了更细致的划分，分为以下几类[10]：

完整性可划分为以下两类：

①强完整性(Strong completeness)：每一个失效的进程最终都会被每个正确的进程怀疑失效。

$$\exists t_0 : \forall t \geq t_0, \forall p \in correct(t), \quad \forall q \in crashed, q \in suspect_p(t) \tag{1}$$

②弱完整性(Weak completeness)：每一个失效的进程最终都会被某些正确的进程怀疑失效。

$$\exists t_0 : \forall t \geq t_0, \forall p \in correct(t), \quad \exists q \in crashed, q \in suspect_p(t) \tag{2}$$

准确性可划分为以下四类：

①强准确性(Strong accuracy)：正确的进程永远不会被怀疑失效。

$$\forall t : \forall p, q \in correct(t), q \notin suspect_p(t) \tag{3}$$

②弱准确性(Weak accuracy)：某些正确的进程永远不会被怀疑失效。

$$\forall t, \exists q \in correct(t), \forall p \in correct(t), q \notin suspect_p(t) \tag{4}$$

③最终强准确性(Eventual strong accuracy)：在某个时间点后，正确的进程都不会被怀疑失效。

$$\exists t_0 : \forall t \geq t_0, \forall p, q \in correct(t), q \notin suspect_p(t) \tag{5}$$

④最终弱准确性(Eventual weak accuracy)：在某个时间点后，某些正确的进程不会被怀疑失效。

$$\exists t_0 : \forall t \geq t_0, \exists q \in correct(t), \forall p \in correct(t), q \notin suspect_p(t) \tag{6}$$

这完整性的两种属性和准确性的四种属性合并成了八类失效检测器，如表 1 所示。

表 1　失效检测器的八种类型

Completeness	Accuracy			
	Strong	Weak	Eventual Strong	Eventual Weak
Strong	P	S	◊P	◊S
Weak	Q	W	◊Q	◊W

从表 1 中可以看出，P 类失效检测器满足强完整性和强准确性，这类失效检测器被称为“完美设计”的失效检测器。由于有 Q≅P，W≅S，◊Q≅ ◊P[5]。因此，主要只研究四类失效检测器，它们的由强到弱的顺序如图 3 所示，其中 S 类失效检测器和◊P 类失效检测器不能进行强弱比较。

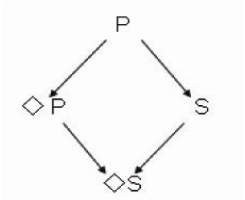

图 3　四类失效检测器的顺序

2.3 失效检测器服务质量

失效检测器最初的研究仅仅只包括异步系统模型中的一组进程，系统规模比较小，因此研究只考虑到

失效检测器的完整性和准确性，以及如何利用不同类型的失效检测器达到共识，忽略了具体应用。判断失效检测器好坏的标准主要是利用完整性和准确性。随着失效检测应用范围不断扩大，人们逐渐注意到分布式环境中网络状况的变化对于失效检测的影响[7-9]。在异步系统中，网络状况是不断变化的。网络负载过重或进程处理速度变化都会影响检测消息的发送和到达时间。因此，研究者提出了自适应网络状况变化的失效检测方法，并进一步扩展了失效检测器的评估标准，提出了失效检测器服务质量(Quality of Service，Qos)的需求[9]。

Chen 等人在文献[7]中研究了失效检测器的服务质量问题，给出了评价失效检测服务质量的几个标准，也为失效检测器的应用提供了更详细的理论指导。首先介绍一些常用状态的符号表示，以符号 T 表示失效检测器认为进程处于正常状态，S 表示失效检测器认为进程处于失效状态，S-transition 表示失效检测器输出由 T 变为 S，T-transition 表示失效检测器的输出 S 变为 T。Chen 给出了几种配置失效检测器服务质量的标准：

①检测时间(Detection time, T_D)：指失效进程 p 从失效发生时间开始，到进程 q 上的失效检测器开始怀疑 p 的时间。

②错误间隔时间(Mistake recurrence time, T_{MR})：指两次连续 S 的间隔时间。即从一个 S-transition 到下一个 S-transition 的时间。

③错误持续时间(Mistake duration, T_M)：指失效检测器从 S 到 T 所需时间。即从 S-transition 到下一个 T-transition 的时间。

④平均错误率(Average mistake rate, λM)：指失效检测器产生怀疑的比率。即在每个时间单元内 S-transition 的平均数。

⑤查询准确概率(Query accuracy probability, P_A)：指的是在随机时间失效检测器输出为 T 的概率。

⑥正确持续时间(Good period duration, T_G)：指失效检测器不发生错误的时间段。

⑦前向正确持续时间(Good period duration, T_{FG})：指从随机时间 t(此时失效检测器是正确的)到失效检测器发生错误的时间。

在这几个标准中，检测时间、错误间隔时间和错误持续时间是最主要的三个标准，其余标准都可用这三个标准来推导。为更有利于理解这三个标准，如图 4 所示。

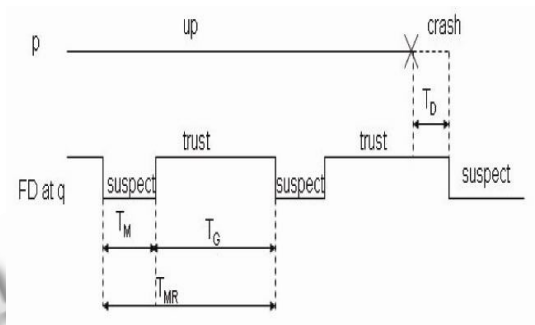

图 4　失效检测服务质量标准

在自适应失效检测算法中，由于没有同步时钟，主要是根据本地心跳消息的接收情况来估计网络状况。图 5 中显示了自适应失效检测服务质量的调整模型。

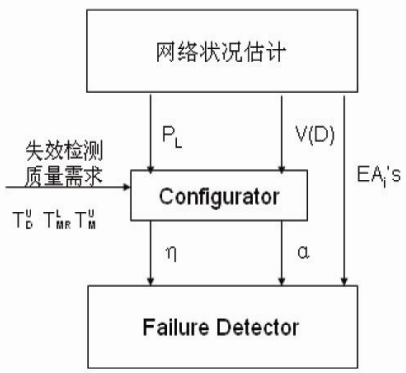

图 5　失效检测服务质量调整模型

其中 T_D^U 是检测时间 T_D 的上界，T_{MR}^L 是平均错误间隔时间的下界，T_M^U 是平均错误持续时间的上界。关系如式(7)所示：

$$T_D \le T_D^U, E(T_{MR}) \ge T_{MR}^L, E(T_M) \le T_M^U \quad (7)$$

P_L和V(D)分别是根据心跳消息计算出的消息丢失概率和消息延迟变量。η是心跳时间间隔，α是修正值。EA_i's 表示理论上预测的第 i 条心跳消息到达时间。

自适应性失效检测方法为应用提供可配置的失效检测服务，可以适应网络环境变化，在一定程度上解

决了网络状况变化对失效检测器性能的影响。同时，不同的应用对失效检测的要求也不同，有些应用要求能尽快检测出进程的失效，以便避免系统阻塞，而有些应用在进行失效切换时，需要进行大量的数据传输，因此要求失效检测器必须具有较高的准确性。

由于自适应失效检测可以适应网络环境变化，并可以配置失效检测器服务质量，让失效检测器具有更好的适应性和扩展性。

3 拜占庭失效检测器

拜占庭失效的行为难以预计，副本通常使用故障屏蔽的方法来处理拜占庭失效。但对于一些可以估计的拜占庭失效行为，还是能够进行相应的失效检测[8-10]。

3.1 可预计拜占庭失效行为

在分布式计算系统中，如果某个进程出现与算法有关的失效，当它与其它进程按照相关协议进行通信时，失效就会被发现。失效检测与收到的协议消息(protocol messages)紧密地联系在一起，即每个进程都要检查收到的消息是否是正确的进程在正确的时间发来的带正确参数的消息。

可预计的拜占庭失效通常包括：变量值出错、忽略或重复了一些消息、伪造消息和表达式赋值错误等等。拜占庭失效检测必须建立在消息不可篡改的假设上。拜占庭失效进程发送的错误消息通常表现为[9]：

①伪造的消息。即进程发送的消息没有正确的签名。

②不一致的消息。即进程给发送给其它进程的消息内容不一样，例如，失效主副本通知副本 p 执行请求 m 的序号是 n，但通知副本 q 执行请求 m 的序号是 n'。

③次序颠倒的消息。通常表现为发送的遗漏和重复。例如前一个请求执行序号是 n，下一个请求的执行序号应该是 n+1。可是失效主副本发送通知要求其它副本下一个请求执行序号是 n+100，或者又将执行序号 n 分配给下一个请求。

3.2 拜占庭失效检测方法

拜占庭失效检测方法通常是基于密码学技术和状态机技术的基础之上。密码学技术是拜占庭容错的基本要求和保障。由于拜占庭失效可能会对消息进行窃听、伪装、篡改、重发或拒绝服务，必须使用密码学用于维持信息的秘密性和完整性，即使信息暴露在潜在的攻击下也应如此。如果没有可靠的信道和密码技术作为支持，存在拜占庭失效的系统将没有办法达到共识。在拜占庭失效模型下，如果使用数字签名实现消息不能伪造，那么如果副本 p 检测到副本 q 在消息交换的过程中发送了不一致的消息 m 和 m'，则 p 的本地失效检测器有证据表明副本 q 是拜占庭失效。

在状态机技术中，副本都能根据有限状态机方式预计其他副本行为。在每个状态，对应一系列允许的接收事件。当副本 p 检测到从副本 q 发来的消息 m 不在允许接收事件范围内，则称 m 为乱序消息(Out-of-order message)，当不允许的事件发生时，则 p 的本地失效检测器有证据表明 q 是拜占庭失效[10]。

4 结束语

失效检测器是发现分布式系统中节点失效的有利工具，但随着分布式系统的不断发展和变化，值得考虑的问题也越来越多。规模不断扩大，需要检测的组件和节点数量也越来越多，在满足一定失效检测服务质量同时，如何相应地降低失效检测器给网络带来的额外负担，是失效检测服务系统需要考虑的重要问题。如今流行的对等网络系统中，节点会动态加入或退出，失效检测器该如何具有良好的可扩张性，以适应系统的变化，值得进一步研究。

参考文献

1 Birman KP. Building Secure and Reliable Network Applications. New York : Manning Publications Co, 1996:6－15.

2 Baldoni R, Helary JM, Raynal M. From Crash Fault-Tolerance to Arbitrary-Fault Tolerance: Towards a Modular Approach. In: IEEE International Conference on Dependable Systems and Network. New Jersey: IEEE, 2000:273－282.

3 Chandra TD, Toueg S. Unreliable failure detectors for reliable distributed system. Journal of the ACM, 1996, 43,(2):225－267.

(下转第 49 页)

(上接第 41 页)

4 Hayashibara N, Cherif A, Katayama T. Failure detectors for large-scale distributed systems. Proceedings 21st IEEE Symposium on Reliable Distributed Systems. New Jersey: IEEE, 2002:404 – 409.

5 Bertier M, Marin O, Sens P. Implementation and performance evaluation of an adaptable failure detector. Proceedings of the International Conference on Dependable Systems and Networks. New Jersey: IEEE, 2002: 354 – 363.

6 陈宁江,魏峻,杨波,等.Web 应用服务器的适应性失效检测.软件学报, 2005,16(11):1929 – 1938.

7 Chen W, Toueg W, Aguilera MK. On the quality of service of failure detectors. IEEE Transactions on Computers, 2002,51(1):561 – 580.

8 Kihlstrom KP, Moser LE, Melliar-Smith PM.The SecureRing Protocols for Securing Group Communication. Proceedings of the Hawaii International Conference on System Sciences. New Jersey: IEEE, 1998: 317 – 326.

9 Baldoni R, Helary JM, Raynal M. From Crash Fault-Tolerance to Arbitrary-Fault Tolerance: Towards a Modular Approach. IEEE International Conference on Dependable Systems and Network. New Jersey: IEEE, 2000:273 – 282.

10 Kihlstrom KP, Moser LE, Melliar-Smith PM. Solving Consensus in a Byzantine Environment Using an Unreliable Fault Detector. Proceedings of the International Conference on Principles of Distributed Systems. New Castle: Hermes Press,1997:61 – 75.