第一章 分布式系统的特征

分布式系统：其组件分布在联网的计算机上，组件之间通过传递消息进行通信和动作协调的系统。

分布式系统的重要特征：组件的并发性、缺乏全局时钟、组件故障的独立性。

构造分布式系统的挑战：处理其组件的异构性、开放性（允许增加或替换组件）、安全性、可伸缩性（用户的负载或数量增加时能正常运行的能力）、故障处理、组件的并发性、透明性和提供的服务质量的问题。

分布式系统：其硬件或软件组件分布在联网的计算机上，组件之间通过传递消息进行通信和动作协调的系统。

分布式系统的显著特征

|  |  |
| --- | --- |
| 并发 | 用户可以在各自的计算机上工作，在必要时共享诸如Web页面或文件之类的资源。 |
| 缺乏全局时钟 | 网络上的计算机与时钟同步所达到的准确性是有限的，即没有一个正确时间的全局概念。 |
| 故障独立性 | 网络故障导致网上互联的计算机的隔离，但这并不意味着它们停止运行，事实上，计算机上的程序不能够检测到网络是出现故障还是网络运行得比通常慢。类似的，计算机的故障或系统中程序的异常终止（崩溃），并不能马上使与它通信的其它组件了解。系统的每个组件会单独地出现故障，而其它组件还在运行。 |

构造和使用分布式系统的主要动力来源于对共享资源的期望。

分布式系统的例子

|  |  |
| --- | --- |
| Web搜索 |  |
| 大型多人在线游戏（Massively Multiplayer Online Game，MMOG） |  |
| 金融交易 | 金融交易系统的重点：对感兴趣数据项的通信和处理。  事件：在分布式系统中，感兴趣数据项。  分布式基于事件的系统  复杂事件处理（Complex Event Processing，CEP）：交易系统必须处理各种各样的事件流，这些事件流高速到达，经常需要实时处理来检测表示交易机会的模式。这在过去曾经是手工处理的，但在竞争压力下变成自动处理。 |

推动分布式系统发展的趋势

|  |  |
| --- | --- |
| 出现了泛在联网技术 | 防火墙的作用：保护企业内部网，防止未授权的消息进出网络。  互联网服务提供商（Internet Service Provider，ISP）：给个体用户和小型组织提供宽带链接和其他类型连接的公司，使他们能获得互联网上任何地方的服务；同时提供诸如电子邮件和Web托管等本地服务。  企业内部网通过主干网实现互相链接。  主干网：具有高传送能力的网络链接，通常采用卫星连接、光缆和其他高带宽线路。 |
| 出现了无处不在计算 | 移动计算：用户在移动或访问某个非常规环境时执行计算任务的性能。  位置感知（上下文感知的计算）：为用户在其移动时提供访问资源（如打印机）或方便的利用附近的销售点的情形在不断增加。  无处不在计算：对在用户的物理环境（包括家庭、办公室和其他自然环境）中存在的多个小型、便宜的计算设备的利用。  无处不在：小型计算设备最终将在不会引人注意的日常物品中普及。也就是说，它们的计算形为将透明地紧密捆绑到这些日常物品的物理功能上。  自发互操作  服务发现：让访问者的设备在访问地的网络上通信，并将设备与合适的本地设备相关联。 |
| 对多媒体设备的需求增加 | 多媒体支持：以集成的方式支持多种媒体类型的能力。  离散类型媒体。例如：图片、正文消息  连续类型媒体。例如：音频、视频  吞吐量的单位：帧/秒  网络播放（webcasting）：在互联网上广播连续媒体（典型的是音频或视频）的能力。 |
| 把分布式系统作为一个设施 | 数据中心：网络化的设施，为用户或机构提供对拥有大量数据的数据仓库的访问。  计算基础设施  操作系统虚拟化  云计算（cloud computing）  云：一组基于互联网的应用，并且足以满足大多数用户需求的存储和计算服务的集合，这使得用户能大部分或全部免除本地数据存储和应用软件的使用。  集群计算机（cluster computer）：互联的计算机集合，它们紧密协作提供单一的、集成的高性能计算能力。  刀片服务器（blade server）：最小的包含处理和（主存）存储能力的计算元素。  刀片系统由包含在一个刀片机架中的大量刀片服务器组成。  集群服务器的总目的：提供一系列的云服务，包括高性能计算能力、大容量存储（例如通过数据中心）、丰富的应用服务（如Web搜索——Google依赖大容量集群计算机体系结构来实现其搜索引擎和其他服务）。  网格计算 |

构造分布式系统的主要动机：资源共享

计算机支持协同工作（Computer Supported Cooperative Working，CSCW）

服务：计算机系统中管理相关资源并提供功能给用户和应用的一个单独的部分。

服务器：在连网的计算机上的一个运行程序（一个进程），这个程序接收来自其他计算机上正在运行的程序的请求，执行一个服务并适当地做出响应。

客户：发出请求的进程。

客户-服务器计算：整个方案。

（客户）调用那个服务器上的操作：；客户发送一个要执行的操作请求。

远程调用：客户和服务器之间的完整交互，即从客户发送一个请求到它接收到服务器的应答。

进程

客户对象

服务器对象

一个客户对象调用了一个服务器对象上的方法：在用面向对象语言实现的分布式系统中，资源被封装成对象，并由客户对象访问。

构造分布式系统的挑战

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 异构性 | 多样性和差别   |  | | --- | | 网络 | | 计算机硬件 | | 操作系统 | | 编程语言 | | 由不同开发者完成的软件实现 |   中间件：软件层，它提供了一个编程抽象，同时屏蔽了底层网络、硬件、操作系统和编程语言的异构性。  公共对象请求代理（Common Object request Broker，CORBA）  Java远程方法调用（Remote Method Invocation，RMI）  除了解决异构性的问题外，中间件为服务器和分布式应用的程序员提供了一致的计算模型。这些模型包括   |  | | --- | | 远程对象调用 | | 远程事件通知 | | 远程SQL访问 | | 分布式事务处理 |   移动代码：能从一台计算机发送到另一台计算机，并在目的计算机上运行的代码。  虚拟机 |
| 开放性 | 分布式系统的开放性主要取决于新的资源共享服务能被增加和供多种客户程序使用的程度。  发布（关键接口）  征求意见文档（Requests For Comments，RFC）  万维网联盟（World Wide Web Consortium，W3C）  开放的分布式系统：按……的方式支持资源共享的系统。 |
| 安全性 | 信息资源的安全性   |  |  | | --- | --- | | 机密性 | 防止泄露给未授权的个人 | | 完整性 | 防止被改变或被破坏 | | 可用性 | 防止对访问资源的手段的干扰 |   没解决的挑战   |  |  | | --- | --- | | 拒绝服务攻击 | 用大量无意义的请求攻击服务，使得重要的用户不能使用它。 | | 移动代码的安全性 |  | |
| 可伸缩性 | （系统是）可伸缩的：资源数量和用户数量激增，系统仍能保持其有效性。  可伸缩分布式系统的设计面临的挑战   |  |  | | --- | --- | | 控制物理资源的开销 | 当对资源的需求增加时，应该可以花费合理的开销扩展系统以满足要求。 | | 控制性能损失 |  | | 防止软件资源用尽 |  | | 避免性能瓶颈 | 算法是分散型  规模问题是分布式系统开发中面临的主要问题。 | |
| 故障处理 | 当硬件或软件发生故障时，程序可能会产生不正确的结果或者在它们完成应该进行的计算之前就停止了。  分布式系统的故障是部分的，也就是说，有些组件出了故障而有些组件运行正常。  处理故障的技术   |  |  | | --- | --- | | 检测故障 | 面临的挑战：如何在有故障出现的情况下进行管理，这些故障不能被检测到但可以被猜到。 | | 掩盖故障 | 有些被检测到的故障能被隐藏起来或降低它的严重程度。 | | 容错 |  | | 故障恢复 |  | | 冗余 | 利用冗余组件，服务可以实现容错。 | |
| 并发性 | 在分布式系统中，代表共享资源的任何一个对象必须负责确保它在并发环境中操作正确，这不仅适用于服务器，也适用于应用中的对象。 |
| 透明性 | 迁移透明性（被移动透明性替换）   |  |  | | --- | --- | | 访问透明性 | 用相同的操作访问本地资源和远程资源。 | | 位置透明性 | 不需要知道资源的物理或网络位置（例如，哪个建筑物或IP地址）就能够访问它们。 | | 并发透明性 | 几个进程能并发地使用共享资源进行操作且互不干扰。 | | 复制透明性 | 使用资源的多个实例提升可靠性和性能，而用户和应用程序员无需知道副本的相关信息。 | | 故障透明性 | 屏蔽错误，不论是硬件组件故障还是软件组件故障，用户和应用程序都能够完成它们的任务。 | | 移动透明性 | 资源和客户能够在系统内移动而不会影响用户或程序的操作。 | | 性能透明性 | 当负载变化时，系统能被重新配置以提高性能。 | | 伸缩透明性 | 系统和应用能够进行扩展而不改变系统结构或应用算法。 |   网络透明性：最重要的两个透明性是访问透明性和位置透明性，它们的有无对分布式资源的利用有很大影响。 |
| 服务质量 | 系统主要的非功能特性，即影响客户和用户体验的服务质量是   |  | | --- | | 可靠性 | | 安全性 | | 性能：满足及时性保证的能力。  源于及时性和计算吞吐量 |   适应性：满足变化的系统配置和资源可用性。  时间关键性数据：要求以固定速度处理或从一个进程传送到另一个进程的数据流。  QoS（Quality of Service，服务质量）：系统满足……的截止时间的能力，它的实现取决于所需要的计算和网络资源在相应时刻的可用性。 |

万维网

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 超文本  链接（超链接）  开放  内容  Web基于的主要的标准技术组件   |  |  |  | | --- | --- | --- | | 超文本标记语言（HyperText Markup Language，HTML） | 页面在Web浏览器上显示时指定其内容和布局的语言。 |  | | 统一资源定位器（Uniform Resource Locator，URL）（统一资源标识符（Uniform Resource Identifier，URI）） | 用于识别文档和保存成Web一部分的其他资源。 | URL结构：   |  | | --- | | 模式:模式特定的标识符 |   HTTP URL结构：   |  | | --- | | http://服务器名[:端口][/路径名][?查询][#片段] |   服务器名表示成一个域名系统（Domain Name System，DNS）名字。  默认端口号是80。  查询成分  片段标识符  发布资源  内容管理系统 | | 具有标准交互规则（超文本传输协议，HTTP）的客户-服务器系统体系结构 |  | 主要特征   |  |  | | --- | --- | | 请求-应答交互 | 方法 | | 内容类型 |  | | 一次请求一个资源 |  | | 简单的访问控制 |  | |   动态页面   |  | | --- | | 表单  （服务器）处理（用户的输入）  公共网关接口（Common Gateway Interface，CGI）程序：Web服务器上运行的为客户生成内容的程序。  下载的代码  异步  AJAX（Asynchronous Javascript And XML） |   Web服务   |  | | --- | | 可扩展标记语言（Extensible Markup Language，XML）是一种以标准的、结构化的、特定于应用的格式表示数据的方式。  自描述的：包含数据元素的名字、类型和结构。  REST（Representational State Transfer）体系结构 |   对Web的讨论   |  | | --- | | 资源描述框架  语义Web：由互连的元数据资源组成的Web。 | |

第二章 系统模型

|  |  |
| --- | --- |
| 物理模型 | 考虑组成系统的计算机和设备的类型以及它们的互连，不涉及特定的技术细节。 |
| 体系结构模型 | 从系统的计算元素执行的计算和通信任务方面来描述系统。  计算元素：或是单个计算机或是通过网络互联的计算机集合。  客户-服务器和对等模型是分布式系统中的两种最常使用的体系结构模型。 |
| 基础模型 | 采用抽象的观点描述大多数分布式系统面临的单个问题的解决方案。   |  |  |  | | --- | --- | --- | | 交互模型 | 处理分布式系统的性能问题并解决在分布式系统中设置时间约束的困难，例如消息传送的时间约束。 | 考虑在系统元素之间通信的结构和顺序。 | | 故障模型 | 试图给出进程和通信通道故障的一个精确的归约。 | 考虑一个系统可能不能正确操作的方式。 | | 安全模型 | 讨论对进程和通信通道的各种可能的威胁。  安全通道 | 考虑如何保护系统使其不受到正确操作的干扰或不被窃取数据。 | |

分布式系统的设计者要面对的问题

|  |  |
| --- | --- |
| 使用模式的多样性 |  |
| 系统环境的多样性 |  |
| 内部问题 | 包括非同步的时钟、冲突的数据更新、多种涉及系统单个组件的软硬件故障模式。 |
| 外部问题 | 包括对数据完整性、保密性的攻击以及服务拒绝攻击。 |

物理模型是从计算机和所用网络技术的特定细节中抽象出来的分布式系统底层硬件元素的表示。

基线物理模型

最小物理模型：一组可扩展的计算机结点，这些结点通过计算机网络相互连接进行所需的消息传递。

三代分布式系统

|  |  |
| --- | --- |
| 早期的分布式系统 | 一般由通过局域网互联的10~100个结点组成，它们与互联网的连接有限并支持少量的服务。 |
| 互联网规模的分布式系统 | 底层物理基础设施由第1章图1-3所示的物理模型组成，即一个可扩展的结点集合，这些结点通过一个网络的网络（互联网）相互连接。  网络的网络：互联网。 |
| 当代的分布式系统 | 在上述系统中，结点通常是台式机，因此是相对   |  |  | | --- | --- | | 静态的 | 在一段时间里停留在一个物理位置 | | 分立的 | 没有嵌入到其他物理实体内。 | | 自洽的 | 就物理基础设施而言，很大程度上独立于其他计算机。 |   系统的分布式系统  超大规模（Ultra-Large-Scale，ULS）的分布式系统。  系统的系统：一个复杂系统，它由一系列子系统组成，这些子系统本身也是系统，它们一起完成一个或多个特定的任务。 |

一个系统的体系结构是用独立指定的组件以及这些组件之间的关系来表示的结构。

|  |
| --- |
| 可靠性 |
| 可管理性 |
| 适应性 |
| 性价比 |

通信范型

放置

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 通信实体 | |  |  |  | | --- | --- | --- | | 面向系统的实体 | 进程 |  | | 结点 |  | |  | 线程 |  | | 面向问题的实体 | 对象 | 面向对象的方法包括面向对象的设计和面向对象的编程语言。  接口定义语言（IDL） | | 组件 |  | | Web服务 | 一个软件应用，通过URI被辨识，它的接口和绑定能作为XML制品被定义、描述和发现。一个Web服务通过在基于互联网的协议上利用基于XML的消息交换支持与其他软件代理的直接交互。 | |
| 通信范型 | |  |  | | --- | --- | | 进程间通信 | 用于分布式系统进程之间通信的相对底层的支持，包括消息传递原语、直接访问由互联网协议提供的API（套接字编程）和对多播通信的支持。 | | 远程调用 | 覆盖一系列分布式系统中通信实体之间基于双向交换的技术，包括调用远程操作、过程或方法。   |  |  |  | | --- | --- | --- | | 请求-应答协议 | 一对消息的交换，消息从客户到服务器，接着从服务器返回客户，第一个消息包含在服务器端执行的操作的编码，然后是保存相关参数的字节数组，第二个消息包含操作的结果，它也被编码成字节数组。 |  | | 远程过程调用（Remote Procedure Call，RPC） | 远程计算机上进程中的过程能被调用，好像它们是在本地地址空间中的过程一样。 | 访问透明性  位置透明性 | | 远程方法调用（Remote Method Invocation，RMI） |  |  | | | 间接通信 | 空间解耦合：发送者不需要知道他们正在发送给谁。  时间解耦合：发送者和接收者不需要同时存在。  间接通信的关键技术：   |  |  |  | | --- | --- | --- | | 组通信 |  |  | | 发布-订阅系统（分布式基于事件的系统） |  |  | | 消息队列 |  |  | | 元组空间 | 元组：结构化数据项。 |  | | 分布式共享内存（Distributed Shared Memory，DSM） | 分布式共享内存系统提供一种抽象，用于支持在不共享物理内存的进程之间共享数据。 | 分布透明性 | | |
| 角色和责任 | |  |  | | --- | --- | | 客户-服务器 | 搜索引擎  Web抓取 | | 对等体系结构 | 对等方 | |
| 放置 | 诸如对象或服务这样的实体是怎样映射到底层的物理分布式基础设施上的，物理分布式基础设施由大量的机器组成，这些机器通过一个任意复杂的网络互联。  放置策略   |  |  | | --- | --- | | 将服务映射到多个服务器 | 服务器可以将服务所基于的对象集分区，然后将这些分区分布到各个服务器上；或者服务器可以在几个主机上维护复制的对象集。  Sun网络信息服务（Network Information Service，NIS） | | 缓存 | 用于存储最近使用的数据对象，这些被存储的数据对象比对象本身更靠近一个客户或特定的一组客户。 | | 移动代码 | applet  推模式：由服务器而不是客户发起交互。 | | 移动代理 | 一个运行的程序（包括代码和数据），它从一台计算机移动到网络上的另一台计算机，代表某人完成诸如信息搜集之类的任务，最后返回结果。  蠕虫程序 | |

体系结构模式创建在上述讨论过的相对原始的体系结构元素之上，提供组合的、重复出现的结构，这些结构在给定的环境中能运行良好。

几个关键的体系结构模型

|  |  |
| --- | --- |
| 分层体系结构（layering architecture） | 在分层方法中，一个复杂的系统被分成若干层，每层利用下层提供的服务。  网络时间协议（Network Time Protocol，NTP）  平台。一个服务于分布式系统和应用的平台由最底层的硬件和软件层组成。这些底层为其上层提供服务，它们在每个计算机中都是独立实现的，提供系统的编程接口，方便进程之间的通信和协调。  中间件。一个软件层，其目的是屏蔽异构性，给应用程序员提供方便的编程模型。中间件表示成一组计算机上的进程或对象，这些进程或对象相互交互，实现分布式应用的通信和资源共享支持。 |
| 层次化体系结构（tiered architecture） | 一项组织给定层功能的技术，它把这个功能放在合适的服务器上，或者作为第二选择放在物理结点上。  对一个给定应用的功能分解：   |  |  | | --- | --- | | 表示逻辑 | 涉及处理用户交互和修改呈现给用户的应用视图。 | | 应用逻辑 | 设计与应用相关的（也称为业务逻辑，虽然这个概念不仅仅限于业务应用）详细的应用特定处理。 | | 数据逻辑 | 涉及应用的持久存储，通常在一个数据库管理系统中。 |   AJAX（Asynchronous Javascript And XML）  图片（tile）：256×256像素图像。 |
| 瘦客户 | 使得能以很少的对客户设备的假设或需求，获得对复杂网络化服务的访问，这些服务可以通过云解决方案提供。  指的是一个软件层，在执行一个应用程序或访问远程计算机上的服务时，由该软件层提供一个基于窗口的本地用户界面。  虚拟网络计算（Virtual Network Computing，VNC）  在IP上传送键盘、视频和鼠标事件（KVM-over-IP） |
| 代理（proxy） | 主要用于支持远程过程调用或远程方法调用的位置透明性。  一个代理在本地地址空间中被创建，用于代表远程对象。这个代理提供与远程对象一样的接口，程序员调用这个代理对象，因此无须了解交互的分布式特性。  代理也被用于封装其他的功能（诸如复制或缓存的放置策略等）。 |
| Web服务中的业务代理（brokerage） | 由服务提供商、服务请求者和服务代理（提供与请求的服务一致的服务）三部分组成。 |
| 反射（reflection） | 内省：系统的动态发现的特性。  从中调停：动态修改结构或行为的能力。 |

中间件的任务：为分布式系统的开发提供一个高层的编程抽象，并且通过分层，对底层基础设施中的异构性提供抽象，从而提升互操作性和可移植性。

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 中间件的类别 | 最早的中间件实例：远程过程调用包和组通信   |  |  |  | | --- | --- | --- | | 根据通信实体和相关通信范型，遵循五个主要的体系结构模型 |  |  | | 分布式对象 |  |  | | 分布式组件 |  |  | | 发布-订阅系统 |  |  | | 消息队列 |  |  | | Web服务 |  |  |   应用服务器提供了结构以支持应用逻辑和数据存储的分离，以及对其他特性（如安全性和可靠性）的支持。  除了编程抽象以外，中间件也能提供分布式系统的基础设施服务，供应用程序或其他服务使用。 |
| 中间件的限制 | 端到端争论：一些与通信相关的功能，可以只依靠通信系统终点（end point）的应用的知识和帮助，即可完整、可靠地实现。因此，将这些功能作为通信系统的特征不总是明智的（虽然由通信系统提供一个不完全版本的功能有时对性能提高是有用的）。 |

所有系统模型共享的设计需求：

|  |
| --- |
| 实现进程及网络的性能和可靠性特征 |
| 确保系统中资源的安全性 |

基础模型的目的：

|  |
| --- |
| 显式地表示有关正在建模的系统的假设。 |
| 给定这些假设，就什么是可能的、什么是不可能的给出结论。 |

基本模型解决的问题

|  |  |
| --- | --- |
| 交互 | 计算在进程中发生，进程通过传递消息交互，并引发进程之间的通信（信息流）和协调（活动的同步和排序）。 |
| 故障 | 只要分布式系统运行的任一计算机上出现故障（包括软件故障）或连接它们的网络出现故障，分布式系统的正确操作就会受到威胁。 |
| 安全 | 分布式系统的模块特性和开放性将其暴露在外部代理和内部代理的攻击下。 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 交互模型 | 算法：采取一系列步骤以执行期望的计算。  分布式算法。定义了组成系统的每个进程所采取的步骤，包括它们之间消息的传递。   |  |  | | --- | --- | | 通信通道的性能 | 延迟（latency）：从一个进程开始发送消息到另一个进程开始接收消息之间的间隔时间。  （计算机网络的）带宽（bandwidth）：在给定时间内网络能传递的信息总量。  抖动（jitter）：传递一系列消息所花费的时间的变化值。 | | 计算机时钟和时序事件 | 偏移  时钟漂移率（clock drift rate）：计算机时钟偏离绝对参考时钟的比率。  全球定位系统（GPS） | | 交互模型的两个变体 | 同步分布式系统。满足下列约束：   |  | | --- | | 进程执行每一步的时间有一个上限和下限。 | | 通过通道传递的每个消息在一个已知的时间范围内接收到。 | | 每个进程有一个本地时钟，它与实际时间的偏移率在一个已知的范围内。 |   异步分布式系统：对下列因素没有限制的系统：   |  | | --- | | 进程执行速度 | | 消息传递延迟 | | 时钟漂移率 | | | 事件排序 | 逻辑时间 | |
| 故障模型 | 定义了故障可能发生的方式，以便理解故障所产生的影响。  进程故障、通信通道故障   |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 遗漏故障 | 遗漏故障类错误：进程或通信通道不能完成它应该做的动作。   |  |  | | --- | --- | | 进程遗漏故障 | 进程主要的遗漏故障是崩溃。  进程崩溃了：进程停止了，将不再执行程序的任何步骤。  超时。超时的使用：进程用一段固定时间等待某个事件的发生。  故障-停止：如果其他进程能确切检测到进程已经崩溃，那么这个进程。 | | 通信遗漏故障 | 如果通信通道不能将消息从p的外发消息缓冲区传递到q的接收消息缓冲区，那么它就产生了遗漏故障。   |  |  | | --- | --- | | 发送遗漏故障 | 在发送进程和外发消息缓冲区之间的消息丢失。 | | 接收遗漏故障 | 在接收消息缓冲区和接收进程之间的消息丢失。 | | 通道遗漏故障 | 在两者之间的消息丢失。 | |   良性故障包括遗漏故障、时序故障和性能故障。 | | 随机故障（拜占庭故障） | 可能出现的最坏的故障，此时可能发生任何类型的错误。   |  |  | | --- | --- | | 进程的随机故障 | 进程随机地省略要做的处理步骤或执行一些不需要的处理步骤。 | |  |  | | | 时序故障 | （适用于同步分布式系统。）   |  |  |  | | --- | --- | --- | | 故障类型 | 影响对象 | 描述 | | 时钟 | 进程 |  | | 性能 | 进程 |  | | 性能 | 通道 |  | | | 故障屏蔽 | 屏蔽。一个服务通过隐藏故障或者将故障转换成一个更能接受的故障类型来屏蔽故障。 | | 一对一通信的可靠性 | 可靠通信   |  |  | | --- | --- | | 有效性 | 外发消息缓冲区中的任何消息最终能传递到接收消息缓冲区。 | | 完整性 | 接收到的消息与发送的消息一致，没有消息被传递两次。 |   对完整性的威胁来自的方面：   |  | | --- | | 任何重发消息但不拒绝到达两次的消息的协议 | | 心怀恶意的用户 | | |
| 安全模型 | 通过保证进程和用于进程交互的通道的安全以及保护所封装的对象免遭未授权访问可实现分布式系统的安全。   |  |  | | --- | --- | | 保护对象 | 访问权限指定了允许谁执行一个对象的操作。  主体（principal）：将每个调用和每个结果均与对应的授权方相关联。这样的一个授权方。 | | 保护进程和它们的交互 |  | | 敌人（（有时）对手） | 假定敌人能给任何进程发送任何消息，并读取或复制一对进程之间的任何消息。  来自一个潜在敌人的威胁包括   |  |  | | --- | --- | | 对进程的威胁 |  | | 对通信通道的威胁 | 信息的私密性、完整性  系统的完整性 | | | 解除安全威胁 | 密码学和共享秘密  密码学：保证消息安全的科学。  加密：将消息编码以隐藏其内容的过程。  认证。共享秘密和加密的使用为消息的认证（证明由发送方提供的身份）奠定了基础。  安全通道：连接一对进程的通信通道，每个进程代表一个主体行事。  安全通道的特性：   |  | | --- | | 每个进程确切知道其他正在执行的进程所代表的主体身份。 | | 安全通道确保在其上传送的数据的私密性和完整性（防止篡改）。 | | 每个消息包括一个物理的或逻辑的时间戳以防消息被重放或重排序。 |   安全通道的实例：虚拟私网（VPN）、安全套接字（SSL）协议 | | 其他可能的来自敌人的威胁 | |  |  | | --- | --- | | 拒绝服务 |  | | 移动代码 |  | | | 安全模型的使用 |  | |