摘要

随着大规模电子商务行业中数据传输量的增加，管理如此大量的请求成为一个关键问题。通过在分布式服务器系统上应用任务调度来处理请求是一种有效的方法。但是，当服务器节点在短时间内处理极大量的请求时，过高负载量是不可避免的。没有有效的高负载监视方法，很难高效的对后端服务器集群进行监控和管理。据我们所知，监控高负载异常节点的现有方法既不灵活也不直观，并且不能检测服务器节点异常，例如定位客户端发送的不合理请求。在本文中，我们提出了一种基于真实数据集的可视化分析的作业调度监控方法，该方法允许监控人员了解该区域中运行节点的状态，并通过各种视图组件观察导致高负载服务器的可疑请求。这种思路为服务器端任务调度集群监测提供了一种全新的方法，并且经过测试显示是可行且有效的。

关键词：数据可视化 任务调度 异常检测

Abstract

With the increasing amount of data transmission in large-scale e-commerce industry, managing such an enormous amount of requests simultaneously becomes a key problem. It is an effective method to handle the requests by applying task scheduling on the distributed server system. However, high load on the servers is inevitable in scheduling when requests of a server node process extreme large amount of tasks in a short period of time. It is difficult to maintain load balance without an effective high-load surveillance approach. To best of our knowledge, existing methods of monitoring high-load abnormal nodes are neither flexible or intuitive and are not capable of detecting server node anomalies such as positioning unreasonable requests sent by the clients. In this paper, we propose a job scheduling monitoring method based on visual analysis from a real dataset, which allows the monitoring personnel to know the status of running nodes in the area and observe the suspected requests causing a high load of servers via various view components we inject into the system.

Keywords: Task scheduling, visual analysis, abnormal detection

第一章　绪 论

1.1　研究工作的背景与意义

现如今网络电子商务项目正在蓬勃发展，人们在享受着足不出户购物和信息浏览，但是往往当使用者的数目变得足够多时，安全隐患往往随之到来。尽管各大互联网企业采取了一些应对措施，比如将用户节点置于类比虚拟机的容器中，或者使用算法对用户请求进行调度处理，使指令被分散在不同的服务器上进行处理，但如果控制这台用户节点的服务器因为某些原因出现超负荷运转，甚至宕机的话，目前整个整个相关领域是没有很好的监视系统来全程监视负责处理用户请求的服务器的运行状态的。比如全球先进的超级计算机研究所——美国德克萨斯州高级计算中心 (TACC) 现在存在检测正在运行服务器的监视器，但是功能和用户操作流程十分不人性化：他们必须通过具有丰富经验、工作年限很长的后台管理者，通过报错信息得到运行异常的服务器编号，再去监测系统手动输入，通过查看后端返回的一系列运行参数，通过经验判断这台服务器的出错原因从而采取补救措施。类似这样的做法，缺点十分明显，比如在整个操作的过程中，操作流程十分僵硬，充斥着很多人为感知和人为判断，使得操作效率的低下；或者在某一时刻，如果因为在某一地区的用户普遍都出现了系统故障导致在短时间内出现了大面积的服务器运行异常，在如此庞大的工作量之下，工作人员的补救效率在不先进的系统下是很迟钝的，这种中间环节的纰漏就会导致解决过程的滞后，从而引发连带问题。

1.1.1　title

1.1.1.1　title

1.1.1.1.1　title