基于 iSCSI 的 IP 存储广域网可用性分析*)

刘 军[1,2] 王 刚[1] 刘 璟[1]

(南开大学信息技术科学学院 天津 300071)[1] (天津财经大学信息系 天津 300222)[2]

摘 要 随着远程容灾、内容分发、数据网格等不断发展,要求数据存储的范围从局域网拓展到广域网。本文提出了基于 iSCSI 协议的 IP 存储广域网系统模型,并利用一种新的可用性度量标准—任务完成概率,分析了 IP 存储广域网的可用性。分析结果表明:当任务数量达到百万时,系统完成概率在 99.7%左右。

关键词 IP 存储广域网,网络存储,可用性,任务完成概率

Availability Analysis of iSCSI-Based IP Storage Wide Area Network

LIU Jun[1,2] WANG Gang[1] LIU Jing[1]

(College of Information Technical Science, Nankai University, Tianjin 300071)[1]

(Department of Information, Tianjin University of Finance and Economics, Tianjin 300222)[2]

Abstract With the increasingly development of remote disaster-tolerance, content distribution and data grid, there is a crucial need to extend the scope of data storage from LAN to WAN. The IP-SWAN system model based on iSCSI protocol is proposed in the paper. The availability analysis of IP-SWAN is made with a novel metric for availability: the probability of task completion. The analysis results show that the probability of the whole system task completion is approximately 99.7% when the number of tasks reaches a million.

Keywords IP-SWAN, Networked storage, Availability, Probability of task completion

1 引言

随着计算机和网络技术的飞速发展,用户对数据存储的需求由原来的以主机为中心、以存储设备为中心,逐渐过渡到以网络为中心。由于光纤传输距离的限制,通过 SAN 方式连接的主机和存储系统之间的连接距离有限。随着远程容灾[1]、内容分发、数据网格[2]等不断应用,要求数据存储的范围从局域网拓展到广域网。

IP 存储广域网(SWAN, Storage Wide Area Network)使跨越广域网范围的数据存储成为可能,它是一种利用 IP 网络存储技术实现全球范围内块级数据访问的存储网络结构。它可以在现有广域网环境中,使用户利用 IP 存储协议实现对全球范围内网络存储环境的访问。

在 IP-SWAN 存储网络结构中,服务器与存储设备之间通过 IP 数据网络连接。IP 数据网络以 IP 和以太网为骨干,块级数据封装在 FCIP、iFCP 和 iSCSI 等 IP 网络存储协议中,通过 IP 和以太网传输介质在服务器和存储设备之间传播。

IP-SWAN 主要用于数据的远程存储,如异地间的数据交换、异地间的数据备份、远程数据容灾、内容分发等,它可以提供比现有技术更及时和有效的方法和手段。通过 IP-SWAN 对数据进行远程备份和镜像,可以使用户减少因灾难及意外事件带来更大的损失。因此,IP 存储广域网的可靠性与可用性显得极为重要。本文针对传统的可靠性分析的局限性,拟从用户任务完成概率角度,来分析 IP-SWAN 系统的可用性。

2 IP-SWAN 存储结构

构建 IP-SWAN 主要有两种方式:一种是基于光纤通道的 SAN 在广域网范围的扩展;另一种是广域网内的主机和存储设备之间完全基于 IP 网络存储协议进行访问的存储网络,支持此功能的 IP 网络存储协议包括 iSCSI、HyperSCSI、ENBD 等。由于 iSCSI 协议将 TCP/IP 协议和 SCSI 很好地结合起来,所以基于 iSCSI 协议建立的 IP 存储广域网具有开放性好、易于扩展、建设成本低等特点。

我们基于 iSCSI 协议构建了 IP 存储广域网,其系统结构如图 1 所示。每个 Target 节点作为远端存储节点,连接在存储节点上的存储设备通过 iSCSI 协议映射到 iSCSI Initiator 端的主节点上,形成一个具有统一地址空间的虚拟存储设备。然后,在 iSCSI Initiator 端利用 LINUX 软 RAID 设备驱动程序,将虚拟存储设备构建为不同级别的 RAID 冗余结构,从而将远程的存储设备组织成为本地的大容量、高可靠性的网络虚拟存储空间。

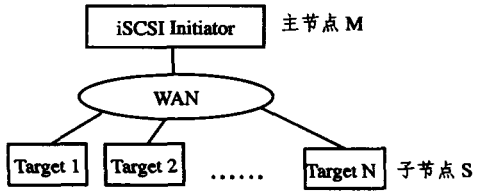

图 1 IP-SWAN 系统结构

*)本课题得到国家自然科学基金(60273031)、高校博士点科研基金(20020055021)和天津市高等学校科技发展基金(20041603)项目资助。刘 军 副教授,博士研究生,研究方向为网络存储、并行与分布式系统等。王 刚 副教授,博士,研究方向为数据布局、并行与分布式系统。刘 璟 教授,博士生导师,研究方向为并行与分布式系统、海量存储、并行算法。

对于 IP-SWAN 系统结构，可抽象为一个主节点 M 带动若干个子节点，子节点提供存储资源，主节点负责数据的处理。在这种模型结构中，所有的数据都流经主节点 M 才能到达各个存储子节点 S。子节点 S 需要处理从 M 节点传送下来的数据信息，并将结果返回给主节点 M，由主节点 M 返回给用户。

3 IP-SWAN 可用性分析

3.1 可用性的度量

分析网络存储系统的可靠性和可用性，一般用平均数据丢失时间(MTTDL)来度量[3]。通过存储设备的平均无故障时间(MTTF)和平均修复时间(MTTR)以及存储系统的结构，可以计算出系统的平均数据丢失时间，但这种分析方法仅适用于传统的串、并联结构。为了分析更通用的存储系统，我们从用户的角度出发，来研究评价系统的可用性的度量方法。

对于任何结构的网络存储系统，都可以抽象为一些数据流在上面流动。由于 IP-SWAN 系统具有一些分布式的特点，不同的数据流可能流向不同的存储节点，所以会出现如下现象：

· 如果某个存储节点出现故障，只要主节点能正常工作，则其它存储节点上的应用不受任何影响。

· 对于损坏的节点，其上的任务将会受到影响，如果损坏的是主节点，则整个系统的所有任务都被破坏。

从上述分析可知，存储系统不能只用简单的“好”与“坏”来表述。实际上，对用户来说，其最关心的是提交给存储系统的任务是否能成功完成。因此，我们提出用“任务完成概率”来评价存储系统的可用性。

所谓任务完成概率是指一个任务提交给存储系统，该存储系统成功地完成了该任务，并将结果返回给用户的概率[4]。

3.2 IP-SWAN 系统任务完成概率分析

在存储系统中，一个任务通常指用户在一次使用存储系统的过程中所使用的所有数据，包括 P 和 Q 两部分。P 部分是数据信息，包含了用户对存储系统的请求数据等信息。Q 部分是验证信息，包含了需要通过存储系统身份验证、序列验证等信息。对于 IP-SWAN，其抽象的任务流模型如图 2 所示。M 代表主节点，S 代表存储节点。显然，一个任务完成的条件是：在整个任务流持续时间内，M 节点和 S 节点必须同时完好。

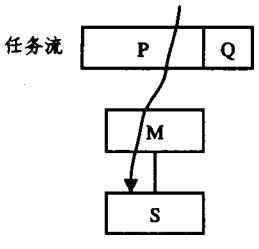

图 2 IP-SWAN 的任务流结构模型

在 IP-SWAN 系统中，一个用户任务典型的执行过程为：用户将任务提交给 iSCSI Initiator 即主节点 M，在主节点进行安全认证，认证通过后，通过主节点访问远端的 Target 存储节点即节点 S，完成真正的用户操作，最后将结果返回给用户。

假设一个任务流 K，在 t_0 时刻提交给系统，对于 Q 部分的数据，在 M 节点完成安全认证所需时间为 t_1，存储节点完成 P 部分数据所需时间为 t_2，用 t_3 表示传输 P 部分数据所需时间，则主节点 M 故障导致任务失败的概率为：

$$\alpha=\int_{t_0}^{t_0+t_1+t_2+t_3}M(t)\,dt \tag{1}$$

其中，$M(t)$ 表示主节点 M 的故障概率密度。

类似地，在任务完成时间内，存储节点 S 发生故障导致任务失败的概率为：

$$\beta=\int_{t_0+t_1}^{t_0+t_1+t_2}S(t)\,dt \tag{2}$$

其中，$S(t)$ 表示 S 节点的故障概率密度。

显然，对整个 IP 存储广域网系统中主节点 M 和子节点 S 之一发生故障都会导致一个任务失败。因此，一个任务流成功完成的概率为：

$$\Delta=(1-\alpha)(1-\beta) \tag{3}$$

按照可靠性理论[5]，对于一般计算机系统，其故障事件是相互独立的随机变量序列，服从指数分布。根据泊松过程的定义可推导出系统在时刻 t 前发生故障的概率为：

$$P=1-e^{-t/MTTF} \tag{4}$$

对于存储设备来说，$MTTF$ 都非常大，新的 SCSI 高档硬盘可以达到上百万小时。如果使用容错技术，则可以将整个磁盘阵列的 $MTTF$ 提高到上万年。对于一般的任务流，其运行时间远远小于 $MTTF$，公式(4)可以近似化简为：

$$P=t/MTTF \tag{5}$$

利用公式(5)，对公式(1)进行化简：

$$\alpha=\int_0^{t_0+t_1+t_2+t_3}M(t)\,dt-\int_0^{t_0}M(t)\,dt$$

$$\alpha=\frac{t_0+t_1+t_2+t_3}{MTTF_m}-\frac{t_0}{MTTF_m}$$

$$\alpha=\frac{t_1+t_2+t_3}{MTTF_m} \tag{6}$$

其中，$MTTF_m$ 是主节点 M 的平均无故障时间。

同样，由公式(2)可以推导出：

$$\beta=\frac{t_2}{MTTF_s} \tag{7}$$

其中，$MTTF_s$ 是子节点 S 的平均无故障时间。将公式(6)和(7)代入(3)，得到：

$$\Delta=1-\frac{t_2}{MTTF_s}-\frac{t_1+t_2+t_3}{MTTF_m}+\frac{t_2}{MTTF_s}\cdot\frac{t_1+t_2+t_3}{MTTF_m} \tag{8}$$

在公式(8)中，$MTTF_s$ 和 $MTTF_m$ 都非常大，远远大于其它项。例如：假设 $t_1=0.3s$，$t_2=2s$，$t_3=3s$，$MTTF_m=40$ 万小时，$MTTF_s=100$ 万小时，则可以计算出单个任务的 $\Delta=99.9999996875\%$。

从公式(8)容易看出，提高管理节点(iSCSI Initiator)的可靠性 $MTTF_m$ 和存储节点(iSCSI Target)的可靠性 $MTTF_s$，可以有效地提高系统的可用性。

3.3 总体可用性分析

对实际应用系统来说，用户对系统的访问请求非常频繁，从而造成对存储系统的访问请求也相当频繁，每天可达上百万次访问。因此，最终应该研究极大数量任务的总体成功概率。

假定有 n 个存储节点，一个任务流通过 M 节点到达第 i 个存储节点的概率为 γ_i，则对于给定的 S 个任务流，其全部完好的概率为：

$$P=\sum_{i=1}^{n}\prod_{j=1}^{S\cdot\gamma_i}\Delta_j \tag{9}$$

其中 Δ_j 表示到达第 j 个节点的任务流完好的概率。当任务

数量达到百万时，系统完成概率在99.688%左右。

总结 本文在分析基于iSCSI的IP存储广域网可用性中，从用户的角度出发，提出了基于任务完成概率的新的可用性评价标准，并在理论上对IP-SWAN系统进行了可用性分析，分析结果表明IP-SWAN系统具有良好的可用性。

未来应该使用任务完成概率对更多的网络存储系统进行可用性研究，以验证这种分析方法的有效性。另外，可以考虑是否还有其它的可用性度量标准，能更准确、更全面地评价存储系统的可用性。

参考文献

1 Chang F, Ji M, Leung S-T, et al. Myriad: cost-effective disaster tolerance. In: Conf. on File and Storage Technologies, Monterey, CA, Jan. 2002. 103～106

2 Cancio G, Fisher S M, Folkes T, et al. The DataGrid Architecture: [Data Grid TechReport DataGrid-ATF-01]. 2001

3 Oggerino C. High Availability Network Fundamentals, Cisco Press, May 2001

4 熊伟. 基于网络的虚拟存储服务及其相关问题的研究: [南开大学信息技术科学学院博士学位论文]. 2004

5 Siewiorek D, Swarz R. The Theory and Practice of Reliable System Design, Digital Press, 1982

（上接第40页）

余网络 N_i 上求得最大流，转Step7。

Step6 ①计算 Δ-剩余网络 N_i 上的流 $F_\Delta(x,y)=f(x,y)\cdot\Delta$，并将 $F_\Delta(x,y)$ 叠加到 $F(x,y)$ 上，则有 $F(x,y)=F(x,y)+F_\Delta(x,y)-F_\Delta(x,y)\cdot 01add(F(y,x))-F_\Delta(y,x)\cdot 01add(F(x,y))$；

②构造 N_i 的剩余网络 $N'_i(R_i(x,y)+F(y,x),s(x),t(y))$，其中 $R_i(x,y)=A_i(x,y)-(F_\Delta(x,y)-F_\Delta(x,y)\cdot 01add(F(y,x)))$；令 $A_i(x,y)=R_i(x,y)$；

③求 N'_i 的 Δ-剩余网络 $N''_i(A'_i(x,y)+F(y,x),s(x),t(y))$，有

$$A'_i(x,y)=\begin{cases}R_i(x,y), R_i(x,y)\geqslant\Delta;\\ 0, \text{否则}\end{cases}$$

④求 Δ-剩余网络 N''_i 的单位容量网络 $N_{01}(E_{01}(x,y),s(x),t(y))$，其中 $E_{01}(x,y)=01add(A'_i(x,y)+F(y,x))$，转Step5。

Step7 $\Delta=\Delta/2$，转Step2。

4 实验结果及分析

为了检验基于增广路径的符号ADD算法的性能，本文利用Colorado大学的CUDD软件包[9]，在运行平台Windows2000，P4 1500MHz CPU，128MB RAM上，通过随机产生的一些随机图，将本文算法与Dinic算法及Karzanov算法进行了大量的实验对比。

在实验中，我们随机产生不同大小、密度及边容量的有向加权图。一般来说，对于随机图 $N(E(x,y),s(x),t(y))$，$E(x,y)$ 的ADD中的共享结点是非常少的，此可近似认为是符号算法的最差情况[4]。基于这些随机图，本文算法与Dinic算法及Karzanov算法的比较结果如表1所示。当有向加权图较大时，在Dinic算法和Karzanov算法产生溢出错时，而本文算法仍能够在较短的时间内计算出最大流。此外，对具有相同大小、密度的有向加权图，本文算法的执行时间随边容量的变小而减少，这主要是因为在相同大小及密度的条件下，随着有向加权图的边容量的减小，那么 $E(x,y)$ 的ADD的终结点数会随之减少，共享结点就会越多，相应的ADD总结点数就会随之减少，为此算法的性能就越好。

以上实验结果表明，本文算法的空间复杂度较低，可处理更大规模的问题。

表1 符号ADD算法与Dinic算法的执行时间比较表

名称	节点数	边数	容量	最大流	符号ADD算法		Dinic算法		Karzanov算法	
					相数	时间	相数	时间	相数	时间
r1. max	200	5.0×10^2	$[0,10^4]$	4.042×10^3	14	1.031	4	0.01	7	0.015
r2. max	500	2.0×10^5	$[0,10^3]$	1.88288×10^5	10	18.703	4	17.344	4	9.391
r3. max	448	2.0×10^5	$[0,10^3]$	2.19194×10^5	10	17.468	5	18.266	4	7.547
r4. max	800	6.392×10^5	$[0,10^5]$	1.246914×10^7	15	82.672	4	219.844	4	235.813
r5. max	10^3	4.995×10^5	[0,1]	9.99×10^2	1	0.015	溢出错		溢出错	
r6. max	10^3	4.995×10^5	$[0,10^3]$	2.46855×10^5	10	69.594	溢出错		4	5171.594
r7. max	10^3	9.99×10^5	$[0,10^3]$	4.82324×10^6	10	89.031	溢出错		4	4227.078
r8. max	10^3	4.995×10^5	$[0,10^5]$	7.88754×10^6	15	180.55	溢出错		溢出错	

参考文献

1 Dinic E A. Algorithm for solution of a problem of maximum flow in networks with power estimation. Soviet Math Dokl, 1970, 11 (8): 1277～1280

2 Karzanov A V. Determining the maximum flow in a network by the method of preflows. Soviet Math Dokl, 1974, 15(3): 434～437

3 Akers S B. Binary decision diagrams. IEEE Transaction on Computer, 1978, 27(6): 509～516

4 Hachtel G D, Somenzi F. A symbolic algorithm for maximum flow in 0-1 networks. Formal Methods in System Design, 1997, 10 (2-3): 207～219

5 Bachar R I, Frohm E A, et al. Algebraic decision diagrams and their applications. Formal Methods in Systems Design, 1997, 10 (2-3): 171～206

6 Bachar R I, Hachtel G D, et al. An ADD-based algorithm for shortest path back-tracing of large graphs. In: Proc. Great Lakes Symposium on VLSI, Notre Dame, 1994. 248～251

7 Bachar R I, Cho H, et al. Timing analysis of combinational circuits using ADD's. In: Proc. of the European Conf. on Design Automation, Paris, France, 1994. 625～629

8 Gabow H N. Scaling algorithms for network problems. Journal of Computer and System Sciences, 1985, 31(2): 148～168

9 Somenzi F. CUDD: CU decision diagram package release 2. 3. 1. http://vlsi. Colorado. edu/～fabio/CUDD/cuddIntro. html, 2001