一种文件级连续数据保护系统的实现与生存性量化

吴世忠[1] 熊 琦[1] 刘 晖[1] 刘 林[1] 王丽娜[2]
(中国信息安全测评中心 北京 100085)[1] (武汉大学计算机学院 武汉 430072)[2]

摘 要 针对传统容灾备份技术的不足,设计并实现了一种文件级别连续保护(CDP)系统。首先设计了CDP系统的部署和组成结构;然后描述了CDP数据包、通信协议、数据的逻辑及物理存储以及任务调度等模块的设计思想;在给出总体和详细设计后,给出了任意时间点文件恢复和任意时间点文件视图查询的实现算法;最后使用半马尔可夫过程进行了生存性分析,在指出其脆弱性的前提下给出了生存性增强的可行途径。

关键词 连续数据保护,文件视图,索引日志,数据恢复,半马尔可夫模型
中图法分类号 TP393 **文献标识码** A

Implementation of a File-level Continuous Data Protection System and its Survivability Quantification

WU Shi-zhong[1] XIONG Qi[1] LIU Hui[1] LIU Lin[1] WANG Li-na[2]
(China Information Technology Evaluation Center, Beijing 100085, China)[1] (School of Computer, Wuhan University, Wuhan 430072, China)[2]

Abstract Regarding the defects of traditional disaster tolerant technology, a novel file-level continuous data protection system was proposed. The deposition method and the design of CDP components were described first. Next the structure of CDP package, data transportation protocol, the logical/physical structure for data storing, and task scheduling methodology were designed. The process of data recovery and data view producing algorithm were presented based on the design mentioned above. Then the algorithm for restoring any time point file and querying any time point file view was designed. Finally a model based on Semi-Markov Process was established and survivability metric was calculated, some promising improvement method was given.

Keywords Continuous data protection, File view, Log based on index, Data recovery, Semi-markov process

1 引言

随着信息技术的飞速发展,企业、政府和大型金融机构对重要数据的依赖性越来越高,数据及数据相关的业务甚至直接能够决定企业的生存状况。传统的容灾手段,如备份和快照技术虽然从一定程度上能够满足离散的备份要求,但仍存在可恢复性差、管理复杂的缺点,难以满足保障重要数据信息的要求。

针对这一缺点,全球网络存储工业协会提出了连续数据保护[1](Continuous Data Protection,CDP)的定义。连续数据保护是一种数据容灾方法,可以捕获或跟踪数据的变化,并将其在生产数据之外独立存放,以确保数据可以恢复到过去的任意时间点。

连续数据保护迅速成为数据容灾的新研究点,但目前出现的产品大都来自国外,它们要么成本过高,要么没有自主知识产权,不适合保护涉密数据。本文设计并实现了一个低成本的连续数据保护系统,它能够在网络环境下为重要数据提供连续保护。实验表明,本系统能够实现任意时间点的数据恢复和任意时间点的文件视图查询功能。

本文第1节描述文件级CDP的功能、部署模式以及设计需求;第2节根据第1节的需求进行CDP数据包、通信协议、数据的逻辑和物理存储以及任务调度等模块的设计;第3节在第2节的基础上阐述任意时间点文件恢复和任意时间点文件视图查询的实现流程;第4节使用半马尔可夫过程进行CDP灾难生存系统的生存性分析和增强;最后总结全文并对将来的研究方向进行展望。

2 文件级连续数据保护系统设计

文件级连续数据保护(文件级CDP),其保护粒度为文件,可以针对单个文件提供保护服务。在恢复过程中,可以将该文件恢复到任意时间点的状态上。其特点在于可以跟踪某个特定对象,而不需要考虑主机上的所有文件,具有对象明确、方式灵活的特性。为了实现文件级连续数据保护,必须支持以下3种功能:(1)任意时间点的数据恢复;(2)查询任意时间点的文件视图;(3)连续数据的网络化存储和存储共享机制。

基于上述前提,本文设计的文件CDP系统由两大部分组成:CDP代理,即被保护主机,以及CDP服务器。两者通过网络连接,具体如图1所示。其中CDP代理安装在需要被保护的主机上,提供了数据截获、数据标记、数据缓存传输等功能。

到稿日期:2011-03-10 返修日期:2011-05-26 本文受国家自然科学基金重大研究计划(90818021)资助。
吴世忠(1962-),男,博士,研究员,主要研究方向为网络与信息安全;熊 琦(1983-),博士,助理研究员,主要研究方向为信息安全;刘 晖(1976-),博士,助理研究员,主要研究方向为网络与信息安全;王丽娜(1964-),博士,教授,主要研究方向为网络与信息安全。

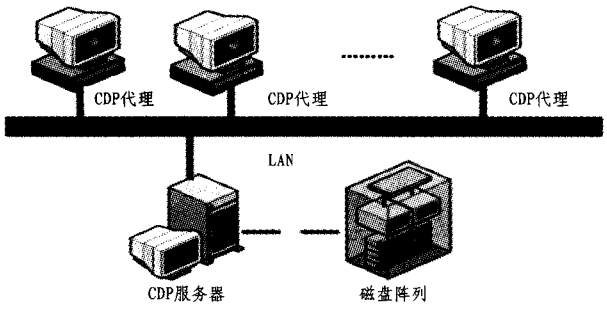


图 1　文件级别连续数据保护系统部署模式

CDP 服务器部署在一台具有足够物理存储空间的主机上，主要对 CDP 代理的监控数据进行存储、组织、管理。CDP 代理包含物理存储系统、日志子系统和管理子系统。其中，物理存储子系统主要管理数据的物理存储；日志子系统用于记录所保护文件的具体操作内容，在需要恢复的时候生成某个具体时间的数据版本；管理子系统主要管理和 CDP 代理的信息交互以及实现备份和恢复的操作流程。

在保护时期，CDP 代理的 I/O 捕获模块将用户对被保护对象的操作捕获下来，将操作信息也记录到 CDP 服务器，CDP 服务器将数据的初始版本和数据的操作存储到服务器本地。在恢复时期，CDP 服务器查找到恢复对象的初始数据版本，以及从初始化到要恢复时间点的日志操作，将这些操作信息在初始数据版本上进行回溯，从而完成任意时间点的数据恢复。

3　文件级 CDP 系统的组成设计

文件级 CDP 系统是一个自上而下的层次结构，如图 2 所示。从上到下分别为 CDP 协议解析器、任务调度器、日志存储子系统、数据存储子系统。CDP 协议解析器的功能是解析 CDP 代理的请求，并调用相应的任务；任务调度器包含 3 个部分：备份任务、恢复任务、文件视图任务。备份任务是将 CDP 代理的备份数据进行日志记录和数据存储；恢复任务是根据 CDP 代理的时间点，组织数据版本，完成数据的组织；视图任务是根据 CDP 代理请求的时间点生成该点的数据视图；日志存储器负责将用户对被保护对象的操作元数据记录下来；数据存储器负责将用户对被保护对象的操作数据记录在存储器中，主要功能是管理数据空间的写入、读取以及回收。

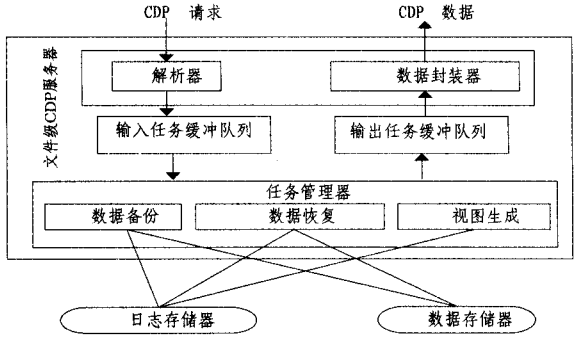


图 2　文件级 CDP 服务器的系统模型

3.1　CDP 协议的定制

虽然文件级 CDP 是基于 TCP 协议的网络备份系统，但进行具体数据交换的时候也需要定制基于 CDP 服务器和 CDP 代理通信的协议。协议在备份、恢复以及视图实现的过程中，都需要通过 CDP 数据报文（见图 3）的格式进行交互。服务器根据数据报文格式（见表 1）进行数据解析，就可以得到 CDP 代理的特定请求，从而调用相应线程接口，完成 CDP 代理所请求的特定任务。

时间 Time	操作 WRITE	文件路径 fullPathName	偏移量 Offset	长度 Length	写内容 Content

图 3　CDP 数据报文格式

表 1　CDP 协议消息功能对照表

网络协议名	消息类型	功能
RA_AUTH_A	认证	CDP 代理认证
RA_ANSAUTH	认证	CDP 服务器认证回复
RA_BACKUP_I	初始化	CDP 代理请求初始化
RA_BACKUP_B	备份	CDP 代理请求备份
RA_BACKUP_R	备份	CDP 服务器备份数据就绪
RA_RESTORE_A	恢复	CDP 代理请求恢复
RA_RESTORE_R	恢复	CDP 服务器恢复数据就绪
RA_VIEWASK_A	视图	CDP 代理请求视图
RA_VIEWANS_R	视图	CDP 服务器视图就绪
RA_FAILED_R	通用	失败
RA_SUCCESS	通用	成功

文件级 CDP 协议主要实现 CDP 代理和 CDP 服务器之间的相互信息交换，CDP 服务器能够针对特定的任务进行特定的处理。CDP 协议通信指令如表 1 所列，分为 4 种类型：认证、备份、恢复和视图。其中备份包含备份初始化和备份两个过程。备份、恢复、视图 3 种指令类型分别对应了 CDP 服务器的 3 种 CDP 任务。

一个完整的 CDP 通信过程如图 4 所示，CDP 代理首先通过 RA_AUTH_A 命令发出认证请求，CDP 服务器通过查找注册信息检查代理的合法性，返回 RA_ANSAUTH；在备份过程中，CDP 代理首先通过 RA_BACKUP_I 或 RA_BACKUP_B 命令请求备份服务，服务器根据现有执行的任务，返回是否准备好的命令 RA_BACKUP_R；在恢复过程中，CDP 代理通过恢复命令 RA_RESTORE_A 请求恢复任务，服务器返回 RA_RESTORE_R，通知 CDP 代理服务器是否准备好恢复。

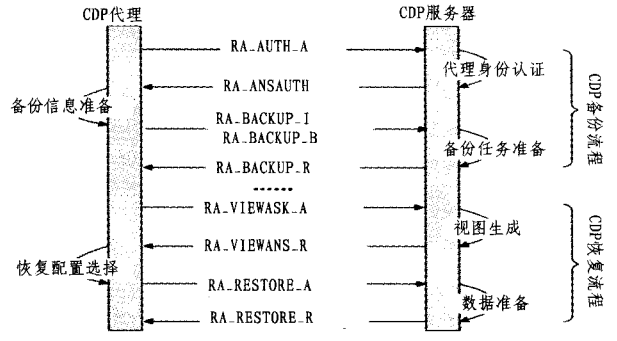


图 4　CDP 通信协议

在备份和恢复过程，网络可能因意外而中断或者发送出的 CDP 数据包无法被接收到。因此在实际实现过程中，系统每传送一个 CDP 数据包时，都会等待该数据包的回复。在备份过程中，CDP 服务器每次收到一个数据包后，立即回复 RA_SUCCESS 的消息，通知 CDP 代理已经收到数据包。在恢复过程中，CDP 服务器在每次发送一个 CDP 数据包后，会等待 CDP 代理的回复。如果在一定时间内没有收到回复信息，CDP 服务器就会认为 CDP 代理已关闭了与 CDP 服务器的通

信链路或通信链路发生中断，然后释放与 CDP 代理相对应的资源。

3.2 任务调度

CDP 服务器认为每个 CDP 代理的服务等级都是一样的，并且将认证、备份、恢复以及视图信息生成作为任务。所以在实现过程中，对 CDP 代理的服务请求采用先来先服务策略(FCFS，First Come First Service)。CDP 服务器创建一个任务缓冲队列，CDP 代理的请求会被暂缓在缓冲队列中。CDP 服务器在处理完一个任务以后就会从图 5 所示的任务队列中取出下一个任务进行处理。

对于 CDP 任务而言，各个任务之间是相互独立的，所以在实现过程中，每个任务对应一个独立的线程来处理这些任务。如图 5 所示，主要的线程包括：备份线程(备份初始化和备份)、恢复线程以及视图获取线程。

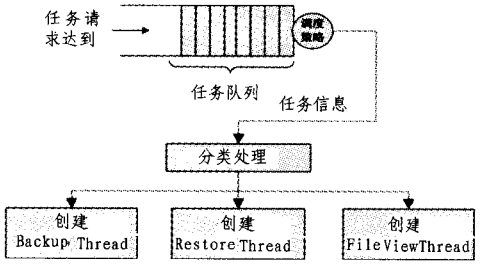


图 5 CDP 任务调度方式

CDP 服务器接收到任务请求，然后通过上文提到的 CDP 协议解析器将任务具体解析成为具体任务的数据结构，并将任务存储在图中对应的任务缓冲的任务队列。

3.3 文件结构的逻辑组织

为了实现查询文件视图的功能，以及在备份和恢复过程中迅速查找到对应对象，CDP 服务器通过文件扩展树来实现。文件扩展树的实现可以迅速地查找到指定对象。在备份过程中，能找到对应节点，进行日志操作的追加；在恢复过程中，通过对文件扩展树的遍历以及日志的查询，根据特定的恢复算法形成在特定时间点的状态(任意时间点的文件视图或者数据版本)。文件扩展树是根据 CDP 代理的目录层次建立的层次数据结构，这个树形结构是常驻 CDP 服务器内存的。直到 CDP 代理对 CDP 服务器进行注销服务的时候，服务器才释放内存。文件节点和目录节点是通过以下的数据结构来实现的。

```
class FileNode
{
  long id;//文件对象的全局标示符
  string name;//该文件在此时的文件名
  bool isExisted;//该文件在 CDP 代理在当前时间是否已被删除
  string attribute;//该文件在当前时间的属性
};
//
class DirectoryNode
{
  long id;//目录对象的全局标示符
  string name;//此目录现在的目录名，方便查找
  bool isExisted;//标记此目录在此时在 CDP 代理是否已被删除
  string attribute;//该目录在当前的属性
  vector<DirectoryNode*> ChildDirectorys;//表示该目录下面包含的目录节点
  vector<FileNode*> ChildFiles;//标识该目录下面包含的文件节点
}
```

文件节点和目录节点的实现中存在一个重要字段“id”，它是一个对象，亦即可以是一个文件或者目录。在 CDP 中，用户对某个对象的名称自动修改是经常发生的，系统为一个对象分配一个全局的 id，就能够优化查询效率，对文件的存储带来方便，存储结构就可以根据一个特定的 id 进行读取和写数据。另一个重要字段就是“isExisted”，表示该节点是否已被删除。树结构中对于 CDP 代理的目录或文件删除只需要将对应的节点标记为不存在，而不是将该节点删除。

另外，后台服务器可能因断电等原因造成停止服务，这就需要将 CDP 服务器的连续数据迁移到其它服务器上。系统在实现的过程中，将上述的树形数据结构保存在 xml 文件中。将此文件迁移到其它服务器上，就可以完成服务器内存状态的迁移，而不至于中断 CDP 服务。

3.4 操作日志的物理存储

为了支持 CDP 代理能够实现在任意时间点的数据恢复，以及在数据保护中需要记录在 CDP 代理上发生所有的操作，CDP 服务器必须有一个高效的日志系统，用以存储 CDP 代理的操作数据的元数据，并且提供高效的日志查找服务。由于大量的操作需要被记录在操作日志文件中，并且各个对象的日志各具独立性，因此日志存储系统应为每个受保护的文件和目录建立操作元数据的日志，采用 N 级索引方式，可以实现日志的动态扩展。每个索引日志在内存中表现为动态申请的内存块，同时同步到磁盘中。

每个索引日志分为两部分：属性修改记录表和内容修改记录表。属性修改记录表中，每个记录为 24 字节，见表 2。

表 2 属性修改记录表的记录块结构

字段名	属性	描述(文件)
Opration-Type	4 字节	{重命名，创建，删除，属性修改}
Modify-length	4 字节	数据的修改长度
CDP-Position	8 字节	修改数据的存储位置
Modify_Time	8 字节	修改时间

内容修改记录表中，每个操作用 32 个字节表示，见表 3。

表 3 内容修改记录表的记录块结构

字段名	属性	描述(文件)
Position-pointer	8 字节	写操作位置
Modify-length	4 字节	数据的修改长度
Operate-Tyep	4 字节	操作类型
CDP-Position	8 字节	修改数据的存储位置
Modify_Time	8 字节	修改时间

索引存储可以对遍历查询的速度进行优化，而将 CDP 代理端截获到的数据采用如上所述索引日志形式进行索引存储，可将每个操作与其目标文件/目录关联起来，进一步优化针对特定对象的操作查询速度，从而为针对特定文件和文件夹的操作记录高效查询和快速恢复操作奠定基础。

3.5 数据物理存储

数据的物理存储是整个 CDP 的基础。将数据存储在 CDP 服务器端，通过恢复算法的组织才能实现任意时间点的数据恢复。物理数据的存储主要是将被保护对象的原始数据以及操作数据存储在 CDP 服务器本地的实现。数据存储是通过聚集存储的思想来实现的，该思想的核心是将每个文件和文件夹的操作数据尽可能连续存储，实现数据访问，从而提

高恢复时的数据存取效率，提高恢复速度。

4 数据恢复功能的实现

4.1 数据恢复的步骤

基于上述模块设计，任意时间点的数据恢复需要经过以下步骤：

1)认证过程。CDP 服务器对 CDP 代理的身份进行认证。

2)备份初始化以及备份过程。备份初始化和备份的过程都是将被保护对象的数据备份到 Data Container 中，将日志记录到 Log Container 中。备份初始化的主要工作是将 CDP 代理的被保护的数据对象在 CDP 服务器端备份一个初始时间点的数据版本；备份过程是将 CDP 代理端的被保护数据的操作作为日志记录在 CDP 服务器。为了将被保护数据的初始化部分与其操作数据区分开来，系统将备份初始化的数据的时间标记为零。

3)获取文件视图。获取文件视图主要是获取在过去时间点的被保护对象的逻辑状态，CDP 服务器进入 View Thread 查询 Log Container，通过视图的组织算法形成在特定时间点的数据逻辑状态。

4)恢复过程。此过程主要是完成 CDP 代理在某个时间点的数据恢复。CDP 服务器调用恢复任务，恢复过程是在 Log Container 中查找到操作日志，并且在 Data Container 中读取到相应的数据，通过数据恢复算法形成在特定时间点的数据版本。

4.2 数据恢复算法

恢复算法是 CDP 系统的核心，也是系统实现的最终目标。CDP 系统在恢复过程中涉及到两个比较重要的计算过程，一个是文件视图获取算法，一个是恢复数据版本获取算法。

(1)文件视图获取算法。需要恢复数据之前，必须预览和选择所需要恢复的文件，所以 CDP 服务器需要根据所给的时间点来生成该时间的文件视图。所需要的参数就是时间和父节点。获取文件视图的过程就是遍历父节点的两个孩子序列的过程，对孩子序列中的每一个节点进行版本运算，得知其在所给时间点时是否存在和得到文件名。对于某一个文件节点的版本运算过程是：

算法 1 任意时间点数据恢复算法

Input：时间点 t，CDP 代理身份，恢复对象

Output：CDP 代理在 t 时刻的视图列表

Step1 Server 检查 CDP 代理身份的有效性。若有效，转 Step2，否则终止算法。

Step2 Server 检查恢复时间点 t 是否是一个有效(备份初始化到最后一次备份之间的时间段)的时间点。有效则转 Step3，否则终止算法。

Step3 Server 检查恢复对象在 Server 本地是否是一个已备份的对象。若是则转 Step4，否则终止。

Step4 根据恢复对象查找到对象对应的索引日志在日志文件的具体位置列表(begin_pos，end_pos)(恢复对象的日志记录的起始位置 t 时刻的最后一次操作的位置)。

Step5 从 Step4 的结果中解析日志记录，忽略对目标对象的写操作，并将记录按序加入日志数组 all_log[n]，并且申请结果集合 result。

Step6 从 Step5 中取一条记录。若为创建操作，将此对象的 id 和名称添加入 result 中；若为删除操作，将 result 中的记录进行删除操作；若为重命名，将对应对象的名称重命名为新名称。

Step7 重复 Step6，直到 all_log[n]遍历完，转 Step8。

Step8 算法结束。

在版本运算的过程中，需要对所给节点参数的所有子节点进行版本运算，以得出所给时间时的存在情况以及名称。在循环中多次使用以上的版本运算算法，可以得到生成某一时间点的文件视图的所有信息。

(2)恢复数据版本获取算法。获取到文件视图之后，用户选择完所希望恢复的文件或者文件夹之后，需要执行的就是具体数据的恢复。

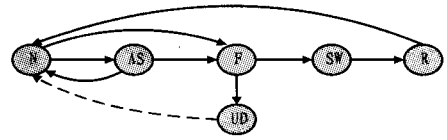
在 CDP 服务器需要恢复某一个文件的时候，进行的是一个 redo 的操作。生成某时间版本的文件的过程，实际上就是由初始化版本文件进行 redo 操作的过程。恢复数据版本的过程中，由初始化版本数据进行 redo 操作，但是其中未涉及到数据的操作实际上都没有被执行。比如某个文件的新建、重命名、删除等操作在文件视图获取之前就已经被计算过了。在执行数据恢复之前，所需要被恢复的文件存在与否、文件名等信息都已经被恢复到所需时间点，只是数据没有被恢复，所以恢复数据版本获取中只有涉及到数据的操作才会被 redo。

恢复的过程实际上分成逻辑上和数据上两个部分，所谓逻辑上的恢复，就是恢复到所需时间点的文件视图，查看当时的文件系统结构，主要的版本运算是计算出文件节点在所给时间点上是否存在，如果存在文件名信息等等；数据上的恢复是将需要恢复的文件重新根据初始化版本 redo 所执行过的数据相关的操作进行恢复，以使该文件恢复到所需时间点时的状态。

5 基于半马尔可夫模型的灾难生存系统生存性量化

为了增强系统的生存性，必须先对其进行生存性量化分析，得出其生存性脆弱点，然后采取必要的生存性增强措施。

假设远程 CDP 服务器发生故障概率极低，根据连续数据保护的工作流程，可以得到基于连续数据保护的灾难生存系统状态变迁模型，如图 6 所示。



N：正常状态。受保护信息系统的更新数据能够及时发送到远程 CDP 服务器。

AS：非同步状态。由于网络或者系统原因而未能维持同步状态，CDP 客户端通过调控手段试图恢复同步。

F：故障状态。受保护信息系统发生故障，影响对外服务。

UD：未检测状态。受保护主机故障未被检测，没有实现远程切换，需要人工处理。

SW：切换状态。远程 CDP 服务器正在定位到正常历史数据，并重组数据索引。

R：修复状态。CDP 服务器通过 iSCSI 协议远程提供数据访问和数据修改服务，同时恢复保护信息系统数据。

图 6 基于 CDP 的灾难生存系统状态变迁模型

图 6 所示的状态变迁模型描述了连续数据保护系统在进行信息系统可生存性增强过程中的动态行为。整个模型包含 $\{N,R\}$ 两个正常服务状态、亚生存状态 $\{AS\}$ 和 $\{F,UD,SW\}$ 3 个不可生存状态。

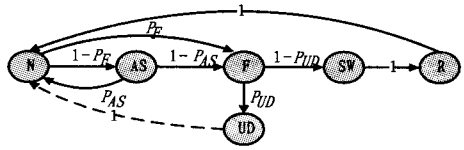
在正常的连续数据保护灾难生存流程中，受保护信息系统和远程 CDP 服务器处于近似同步状态 N，信息系统所产生的更新数据同步传输到远程 CDP 服务器。由于网络带宽或者系统本身的原因，有时同步过程会被干扰，系统会进入 AS 状态。该状态下系统处于亚生存状态，CDP 服务器的数据版本滞后，受保护主机故障后无法切换到最新的完整性状态，但仍然能够通过远程 CDP 服务器提供服务。N 状态的另一种可能是直接停止服务，进入故障状态。F 状态表示受保护信息系统已经发生故障，并影响对外服务的提供。此时有两种可能性：(1)故障被检测到，灾难生存机制发挥作用，后台 CDP 开始重组最新的数据索引，准备通过 iSCSI 提供远程数据访问，保证服务连续提供；(2)故障未被检测到，系统将处于失控报警状态，需要人工介入进行处理，并恢复到 N 状态。当 CDP 服务器成功接管更新数据后，受保护信息系统开始通过远程 CDP 服务器逐步恢复数据，系统处于 R 状态，对外服务仍然正常。

对连续数据保护系统进行动态性建模后，需要对模型进行量化分析，求解出相应的生存性指标，从而精确描述系统的生存性变化态势。

5.1 生存性分析模型的求解

从连续数据保护系统工作的状态变迁模型可知，系统在各状态之间的转移满足马尔可夫特性，即系统将要处于的状态只受当前所处状态影响，与过去状态无关，因而可以采用马尔可夫理论进行分析。马尔可夫模型易于以网络系统状态进行全面有效的描述，精确刻画网络系统随机行为，便于计算各种安全性能指标。为了分析和求解的方便，传统的方法通常假设指数分布来拟合动态行为，但只在一定范围内合理。连续数据保护系统的某些状态变迁的发生具有阶段性，这种过程无法使用时间连续马尔可夫过程进行描述，不适合使用 Kolmogrov 理论来进行求解。

针对这一问题，本文采用基于半马尔可夫过程(SMP)的安全量化分析方法。在 SMP 过程中，状态的驻留时间可以是任意分布，且它们的概率分布函数 PDF 可以依赖于当前状态和下一状态，这符合 CDP 灾难生存系统的状态变迁特征。图 7 所描述的系统 SMP 模型对应于图 6 所描述的嵌入式离散时间马尔可夫链(DTMC)，首先计算 DTMC 过程的稳定状态概率以及每个状态的平均驻留时间，得到 SMP 模型的稳态概率。



P_F：受保护信息系统直接发生故障，停止对外服务的概率；

P_{AS}：受保护信息系统和远程 CDP 服务器恢复同步，回到正常状态的概率；

P_{UD}：受保护信息系统发生故障但未被检测到，处于失控状态，需要进行人工操作的概率。

图 7 基于连续数据保护的灾难生存系统的 DTMC 过程

整个 CDP 灾难生存过程可以看成是离散空间 $S=\{N, AS, F, UD, SW, R\}$ 上的随机过程。定义 Ω 为概率空间，随机变量 $X_n:\Omega\to S$ 和 $T_n:\Omega\to \mathbf{N}$，X_n 表示在第 n 次状态变迁发生时的系统状态，T_n 表示第 n 次变迁发生时的时间($T_0=0$)，设 $V_k=T_{k+1}-T_k$ 表示第 k 次访问状态的驻留时间，则(X,T)过程是一个半马尔可夫过程，或者叫做马尔可夫更新过程(Markov Regenerative Process)。其相关的半马尔可夫核心表示为 Q，其中 $Q(t)=[Q_{ij}(t)]$可以定义为：

$$\begin{aligned}Q_{ij}(t)&=P[X_{n+1}=j, V_k\leqslant t\mid X_0,\cdots,X_n;T_0,\cdots,T_n]\\&=P[X_{n+1}=j, V_k\leqslant t\mid X_n=i]\end{aligned}$$

令 $p_{ij}=\lim\limits_{t\to\infty}Q_{ij}(t)$，则 $P=[p_{ij}]$表示 SMP 的嵌入马尔可夫过程的瞬时变迁矩阵。

为了对 SMP 进行稳态分析，需要知道两类参数：(1)S 中每个状态 i 的平均驻留时间；(2)不同状态 i,j 之间的状态变迁概率 P_{ij}。

SMP 模型的具体求解算法如下所示：

(1)基于图 7 描述的 DTMC 过程，计算所述 DTMC 的稳态概率向量 V。

$$\begin{cases}V=V\circ P\\\sum\limits_i v_i=1\end{cases}\tag{1}$$

其中，

$$P=\begin{array}{c}\\N\\AS\\F\\UD\\SW\\R\end{array}\begin{array}{c}\begin{array}{cccccc}N&AS&F&UD&SW&R\end{array}\\\begin{bmatrix}0&1-P_F&P_F&0&0&0\\P_{AS}&0&1-P_{AS}&0&0&0\\0&0&0&P_{UD}&1-P_{UD}&0\\1&0&0&0&0&0\\0&0&0&0&0&1\\1&0&0&0&0&0\end{bmatrix}\end{array}\tag{2}$$

为图 7 所示 DTMC 的状态变迁概率矩阵，设 $V=[v_N, v_{AS}, v_F, v_{UD}, v_{SW}, v_R]$为 SMP 对应嵌入式马尔可夫过程的稳态概率分布向量，其中 $i\in S$。

将式(2)带入式(1)的第 1 分式，得到 SMP 对应 DTMC 的稳态概率之间的关系：

$$\begin{cases}v_N=v_{AS}P_{AS}+v_{UD}+v_R\\v_{AS}=v_N(1-P_F)\\v_F=v_NP_F+(1-P_{AS})v_{AS}=(1-P_{AS}+P_{AS}P_F)v_N\\v_{UD}=P_{UD}v_F=P_{UD}(1-P_{AS}+P_{AS}P_F)v_N\\v_{SW}=(1-P_{UD})v_F=(1-P_{UD})(1-P_{AS}+P_{AS}P_F)v_N\\v_R=v_{SW}=(1-P_{UD})(1-P_{AS}+P_{AS}P_F)v_N\end{cases}\tag{3}$$

将式(3)带入式(1)的第 2 分式，得到：

$$v_N=\frac{1}{2-P_F+(1-P_{AS}+P_{AS}P_F)(3-P_{UD})}\tag{4}$$

(2)通过半马尔可夫模型计算 SMP 过程的稳态概率向量 Π，必须确定每个状态 i 的平均驻留时间 h_i。针对本节提出的模型，我们设定正常服务状态和亚生存状态的平均驻留时间 h 满足指数分布；对于不可生存状态，其平均驻留时间满足 Weibull 分布。例如图 7 所示的 SW 到 R 状态的转换可能会包含若干次重试，如果最新的连续数据版本不能正常工作，灾难生存系统将采用回滚技术，寻找可用的历史版本数据。

Weibull 分布的分布函数可以表示为 $F(t)=1-e^{-(t/\beta)^\alpha}$，其中 α 决定分布的尺度，β 决定分布的形状。当 $\alpha=1$ 时，变成指数分布，当 $\alpha=3.4$ 时，变成类似正态分布。

定义系统在状态 i 驻留的条件分布函数为 $H_i(t)=P[T_{n+1}-T_n\leqslant t\mid X_n=i]$，则平均驻留时间 $h_i=\int_0^\infty(1-$

$H_i(t))dt$。根据上面对状态驻留时间分布的定义，可以确定各状态的平均驻留时间计算规则。

$$h_N=\frac{1}{\lambda_N},h_R=\frac{1}{\lambda_R};h_{AS}=\frac{1}{\lambda_{AS}}$$

$$h_F=(\frac{1}{\lambda_F})^{1/\alpha_F}\Gamma(1+\frac{1}{\alpha_F}),h_{UD}=(\frac{1}{\lambda_{UD}})^{1/\alpha_{UD}}\Gamma(1+\frac{1}{\alpha_{UD}})$$

$$h_{SW}=(\frac{1}{\lambda_{SW}})^{1/\alpha_{SW}}\Gamma(1+\frac{1}{\alpha_{SW}})$$

式中，$\Gamma(N)=(N-1)!$，λ_N，λ_R，λ_{AS}为相应的变迁速率，λ_F，λ_{UD}，λ_{SW}为 Weibull 分布的形状参数。

令 $H=[h_1,h_2,\cdots,h_n]^T$，通过 DTMC 的稳态概率向量 V 和平均驻留时间，并带入公式 $\pi_i=\frac{1}{V\circ H}diag(V)\circ H=\frac{v_ih_i}{\sum_j v_jh_j}$，$i,j\in X_S$，可以求得半马尔可夫过程的稳态概率 $\Pi=[\pi_i]$，$i\in X_S$。具体如下：

$$\pi_N=h_N/[h_N+(1-P_F)h_{AS}+(1-P_{AS}+P_{AS}P_F)[h_F+P_{UD}h_{UD}+(1-P_{UD})(h_{SW}+h_R)]]$$

$$\pi_{AS}=\frac{(1-P_F)h_{AS}}{h_N}\pi_N$$

$$\pi_F=\frac{(1-P_{AS}+P_{AS}P_F)h_F}{h_N}\pi_N$$

$$\pi_{UD}=\frac{P_{UD}(1-P_{AS}+P_{AS}P_F)h_{UD}}{h_N}\pi_N$$

$$\pi_{SW}=\frac{(1-P_{UD})(1-P_{AS}+P_{AS}P_F)h_{SW}}{h_N}\pi_N$$

$$\pi_R=\frac{(1-P_{UD})(1-P_{AS}+P_{AS}P_F)h_R}{h_N}\pi_N \quad (5)$$

根据前面的定义，系统的生存性权值定为 $W=[1,0.9,0,0,0,1]$。其中正常服务状态为 1，亚生存状态为 0.9，不可生存状态为 0。据此灾难生存系统的生存性为：

$$\begin{aligned}Sur&=W\circ\Pi^T=\pi_N+0.9\pi_{AS}+\pi_R\\&=[h_N+0.9(1-P_F)h_{AS}+(1-P_{UD})(1-P_{AS}+P_{AS}P_F)h_R]/[h_N+(1-P_F)h_{AS}+(1-P_{AS}+P_{AS}P_F)[h_F+P_{UD}h_{UD}+(1-P_{UD})(h_{SW}+h_R)]]\end{aligned} \quad (6)$$

在求得生存性量化公式的前提下，分别从几个方面研究参数对灾难生存系统生存能力的影响。

(1)(h_{AS},P_{AS})表征了系统的同步恢复能力，Sur 同 h_{AS} 和 P_{AS} 之间具有如图 8 所示的函数关系。

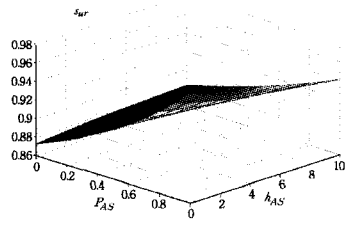


图 8　系统可生存性同 h_{AS} 和 P_{AS} 之间的函数变化图

h_{AS} 和 P_{AS} 对灾难生存系统生存性影响较大，系统的生存能力随着 P_{AS} 的增加和 h_{AS} 的减少而提高，说明提高系统的同步恢复能力能够显著改善 Sur；

(2)h_{SW} 表征了系统的服务切换能力，服务切换的前提条件是受保护服务器的故障状态被成功检测，所以还受系统的故障定位能力漏检率 P_{UD} 的影响。将 P_{UD} 取 5%，10% 和 15%，得出 Sur 同 h_{SW} 之间具有如图 9 所示的函数关系。

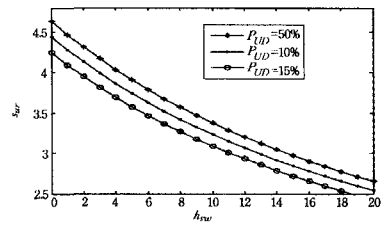


图 9　系统可生存性同 h_{SW} 之间的函数变化图

h_{SW} 对灾难生存系统的生存能力影响较大，说明提高系统的切换相应效率有助于改善生存能力。同时在同样的切换效率下，漏检率较低的系统生存能力更好。

5.2　灾难生存系统生存性增强

根据上述分析结果可以从以下几个方面增强灾难生存系统的生存性：

(1)提高系统的同步恢复能力，可以减少系统处于 AS 状态的驻留时间，同时增加 AS 状态到 N 状态的转化概率。导致 CDP 处于非同步状态的原因，主要是网络的拥塞而导致本地和远程数据的不一致，从而影响恢复能力，因此可以在系统部署的时候增加网络探测机制，对网络状态进行主动化实时探测。当网络状态较差时，使用较粗的传输粒度进行连续数据的更新；当网络状态较好时，使用细粒度进行补偿，保证系统的同步恢复能力。

(2)减少系统在切换状态的驻留时间。由于在最新版本数据不可用的情况下，CDP 服务器可能需要回溯历史数据，找到可用的历史版本对外提供服务。这个过程可能会有反复。可以从两个方面进行优化：①可以通过优化后台索引结构，实现更快便捷的查询检索算法，以提高单位版本数据索引的合成速度；②并非每个时刻的数据版本都具有应用相关的完整性，因此在进行连续数据捕获的同时进行语义标记，以便恢复时查找到有意义的数据版本索引，减少系统的切换驻留时间。故障发现能力也是影响系统生存能力的重要因素，更准确的检测机制能够减少系统处于 UD 状态的概率，并更快地切换到恢复状态 R，继续提供服务。

结束语　本文在文件级别连续数据保护的系统原型上，提出了针对文件级别数据保护的设计和实现方案。CDP 服务器能够提供面向文件的连续数据保护：文件级别 CDP 服务器能够针对被保护主机上的文件进行连续数据保护，保护的时间粒度为零，用户无需干预备份过程。在要求恢复的时候，只需要提供时间点，便可以恢复到任意时刻。文件级别 CDP 服务器能够提供对被保护主机上特定的文件的保护。用户在恢复的时候，可以选择想要恢复的单个或者多个文件进行恢复操作，无需对整个文件进行完整恢复。文件级别 CDP 提供了文件视图功能，能够使用户预览任意时间点上的文件系统结构，从而确定所需要恢复的文件。通过随机模型的生存性分析，给出了生存性增强的若干可能途径。

参 考 文 献

[1] SNIA. Storage Network Industry Association. [EB/OL]. http://www.snia.org/

[2] King R P, Halim N, Garcia-Molina H, et al. Management of Remote Backup Copy for Disaster Recovery [J]. ACM Transactions on Database Systems, 1991, 16(2): 338-368

(下转第 158 页)

取值个数越来越大，ELAs行数与CA(3)行数比越来越小，即因素取值个数越大。用我们提出的方法来构造这种特殊的错误定位表，比用3维覆盖表来作为错误定位表其规模要小得多，定位这个单一的交互错误需要的测试用例数大大减少，从而节约了测试与调试阶段的成本与资源。

结束语 本文在Colbourn和McClary[4]及Mart′lnez等[5,6]提出的错误定位表的基础上研究了一类特殊的错误定位表，即可定位一个交互错误的错误定位表；研究了它们新的构造以及算法生成问题。所得到的错误定位表的覆盖强度在t维覆盖表与$t+1$维覆盖表之间，其行数比$t+1$维覆盖表行数要少得多。然而，对于实际的测试场景，待测系统中有无错误交互或有多少个错误交互都是未知的，这时我们生成的特殊的错误定位表可在加强交互测试充分性的同时用来帮助或减轻未来故障调试或定位阶段的工作。

对未来的工作，我们主要集中在以下几方面。

(1)对于待测系统中有多个错误交互时，研究如何设计错误定位表、这些错误定位表的规模、生成这些错误定位表的高效算法以及设计出的错误定位表定位错误交互的能力与准确性。

(2)对于待测系统中有多个交互错误时，借鉴基于程序谱的方法[16]来对其中可能的交互错误的可能性排序，然后生成附加的测试用例，进行进一步的测试以便精确定位。

(3)对于规模较大的待测系统，错误交互数较多时，生成的错误定位表规模比较大，并且定位的准确性可能不高，这时应研究可定位这些错误的高效的自适应算法，即根据前面的测试用例的运行结果来生成下一条测试用例的算法，来定位与侦测这些交互错误。

参考文献

[1] Kuhn D R, Reilly M J. An investigation of the applicability of design of experiments to software testing[C]// Proceedings of the 27th NASA/IEEE Software Engineering Workshop. NASA Goddard Space Flight Center, 2002

[2] Yilmaz C, Cohen M B, Porter M B. Covering arrays for efficient fault characterization in complex configuration spaces[J]. IEEE Trans. on Software Engineering, 2006, 32(1): 20-34

[3] 徐宝文，聂长海，史亮，等. 一种基于组合测试的软件故障调试方法[J]. 计算机学报，2006，29(1)：132-138

[4] Colbourn C J, McClary D W. Locating and detecting arrays for interaction faults[J]. Journal of Combinatorial Optimization, 2008, 15: 17-48

[5] Mart′lnez C, Moura L, Panario D, et al. Algorithms to locate errors using covering arrays[C]// LATIN 2008. Lecture Notes in Computer Science 4957. Springer, 2008: 504-519

[6] Mart′lnez C, Moura L, Panario D, et al. Locating errors using ELAs, covering arrays and adaptive testing algorithms[J]. SIAM Journal on Discrete Mathematics, 2009, 23: 1776-1799

[7] 严俊，张健. 组合测试：原理和方法[J]. 软件学报，2009，20(6)：1393-1405

[8] Mandl R. Orthogonal latin squares: An application of experimental design to compiler testing[J]. Communications of the ACM, 1985, 28(10): 1054-1058

[9] Brownlie R, Prowse J, Phadke M. Robust testing of AT&T PMX/StarMail using OATS[J]. AT&T Technical Journal, 1992, 71: 41-47

[10] Cohen D M, Dalal S R, Fredman M L, et al. The AETG system: An approach to testing based on combinatorial design[J]. IEEE Trans. on Software Engineering, 1997, 23(7): 437-444

[11] Tung Y W, Aldiwan W S. Automating Test Case Generation for the New Generation Mission Software System[C]// Proc. IEEE Arospace Conf. 2000: 431-437

[12] Bryce R C, Colbourn C J. A Density-based Greedy Algorithm for Higher Strength Covering Arrays [J]. Software Testing, Verification and Reliability, 2009, 19: 37-53

[13] Lei Y, Kacker R, Kuhn D R, et al. IPOG IPOG-D: Efficient Test Generation for Multi-way Combinatorial Testing [J]. Software Testing, Verification and Reliability, 2008, 18(3): 125-148

[14] Cohen M B, Colbouns C J, Gibbons P B, et al. Constructing Test Suites for Interaction Testing [C] // Proceedings of the Intl. Conf. on Software Engineering (ICSE2003). Portland OR, 2003: 38-48

[15] Colbourn C J. Covering array tables [EB/OL]. http://www.public.asu.edu/ccolbou/src/tabby, Dec. 2010

[16] Abreu R, Zoeteweij P, Gemund A. Spectrum-based multiple fault localization[C]// Proc. ASE'09. 2009: 88-99

(上接第114页)

[3] Garcia-Molina H, Polyzois C A. Issues in Disaster Recovery[C]// IEEE Compcon. February 1990: 573-577

[4] Yonghong S, et al. TH-CDP: An Efficient Block Level Continuous Data Protection System[C]// IEEE International Conference on Networking, Architecture, and Storage. 2009

[5] Maohua L, Shibiao L, Tzi-cker C. Efficient Logging and Replication Techniques for Comprehensive Data Protection[C]// 24th IEEE Conference on Mass Storage Systems and Technologies. 2007

[6] Gibson G A. Network attached storage architecture [J]. Communications of the ACM, 2000, 43(11): 59-72

[7] Keeton K, Santos C, Beyer D, et al. Designing for disasters[C]// Proceedings of the 3th USENIX Conference on File and Storage Technologies. San Francisco, CA, USA, 2004: 59-72

[8] 王德军，王丽娜. 容灾技术研究[J]. 计算机工程，2005，31(3)：39-42

[9] Toigo J W. Disaster Recovery Planning: Strategies for Protecting Critical Information Assets (3rd edition) [M]. Prentice Hall PTR, 2002

[10] Stager R K. Method and System for Data Recovery in a Continuous Data Protection System[P]. US 2005/0188256. A1, 2005-08-04

[11] Zhu Ning-ning, Chiueh T-C. Portable and Efficient Continuous Data Protection for Network File Servers [A] // 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'07) [C]. 2007

[12] Wang Li-na, Guo Chi. Building Hot Snapshot Copy Based on Windows File System[J]. Wuhan University Journal of Natural Sciences, 2006, 11(1): 1503-1506

[13] 万继光，吴龙树. 一种网络备份系统的性能分析和优化[J]. 小型微型计算机系统，2008(1)：150-15