一个 XML 数据库强制访问控制模型

冯学斌　洪　帆
（华中科技大学计算机学院　武汉 430074）

摘　要　目前关于 XML 数据库安全性的研究大多是基于自主访问控制或者基于角色的访问控制，然而高安全等级的系统要求支持强制访问控制策略。本文建立了一个 XML 对象的分解与合成规则，在此基础上提出了一个支持多实例的多级安全 XML 数据库模型，给出了其体系结构和安全策略，并且对其安全性进行了分析。本文的研究结果可以为信息敏感部门处理半结构化信息提供理论模型上的支持。

关键词　XML 数据库，多级安全，多实例，安全模型

A Mandatory Access Control Model for XML Database

FENG Xue-Bin　HONG Fan
(College of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074)

Abstract　XML database are playing increasingly important role in Web applications. Researchers of XML database security have paid more attention on discretional access control and role-based access control, rather than mandatory access control which are required by the top security applications. A mandatory access control model for XML database which supports polyinstantiation by using resolving and composing algorithm in syntax level is proposed in paper. The architecture and the security are also discussed in the paper.

Keywords　XML database, Multilevel security, Polyinstantiation, Security model

1　背景

1988 年，W3C(World Wide Web Consortium)发布了扩展标记语言 XML。由于 XML 设计得十分灵活，容易创建和解析，并且和具体的平台、厂商无关，因此迅速成为 Web 服务平台中数据表示的基本语言。目前 XML 在许多领域得到应用，结合这些应用领域的需求，提出了许多具体的数据表示标准，例如数学标记语言(MathML)、化学标记语言(CML)等。

随着 XML 应用范围的不断扩展，越来越多的数据采用 XML 技术来管理，XML 数据库本身的安全性也日益重要。目前，关于 XML 数据库的安全性，国内外进行了许多研究。例如文[1]提出了一个细粒度的访问控制模型，该模型支持不同的访问控制粒度，也支持来自于祖先节点、DTD(XML Scheme)的权限传播，另外还支持 Soft/Hard 策略。文[2]将授权和加密相结合，实现了一个基于层次的密钥管理模式，其密钥的生成和访问策略相关，能够生成最少数量的密钥，而且实现了分布式协作更新。文[5]提出了一个 XML 数据库上的安全视图方法。

但是，目前关于 XML 数据库访问控制技术的研究中，大部分都集中在自主访问控制和基于角色的访问控制上，对于强制访问控制的研究则很少。虽然文[4]提出了一个强制访问控制策略，但是对于多级安全 XML 数据库上的多实例现象并没有进行进一步的研究。

由于多级安全数据库在敏感信息的处理中具有十分重要的意义，因此，本文将对支持多实例的 XML 数据库强制访问控制模型进行研究。本文将在第 2 节首先提出一个 XML 数据的合成与分解规则，在第 3 节将其扩展为多级安全 XML 模型，在第 4 节探讨该模型的框架结构，在第 5 节给出了安全策略的实施过程，在第 6 节对模型的安全性进行了分析。

2　基本 XML 数据模型

这一节，我们首先介绍一个 XML 的基本模型[7]，然后在该模型上定义分解与合成规则。由于该模型简单直观，在结构上和 XML 直接对应，因此很容易实现 XML 数据与模型的相互转换。

在下面的定义中，C 表示所有常量的集合，包括所有的字符串、数字等基本元素(语法符号除外)。Null 表示空元素，并且 $\text{Null}\notin C$。

定义 1　一个 XML 对象是一个由三个元素(名字、属性、值)组成的三元组 (name, attrib, value)。其中，

(1) 名字 name 是一个常量，$\text{name}\in C$；

(2) 属性 attrib 是一个由 $m(m\geqslant 0)$ 个二元组组成的列表 $((an_1,av_1),(an_2,av_2),\cdots,(an_m,av_m))$，其中 $an_i(1\leqslant i\leqslant m)$ 是一个常量，称为属性名，$av_i(1\leqslant i\leqslant m)$ 是一个常量，称为属性值；

(3) 值 value 是一个空元素 Null、常量或者 $n(n\geqslant 0)$ 个 XML 对象组成的集合 $\{o_1,o_2,\cdots,o_n\}$。

定义 2　如果 XML 对象 o 的值是一个 XML 对象集合 $\{o_1,o_2,\cdots,o_n\}$，那么称 $o_i(1\leqslant i\leqslant n)$ 为 o 的儿子，称 o 为 o_1 的父亲，$o_1,o_2,\cdots,o_n$ 中任意两个 o_i 和 $o_j(1\leqslant i,j\leqslant n)$ 互为兄弟。对于任意 XML 对象 $o_1,o_2,\cdots,o_n$，如果 $o_i+1(1\leqslant i<n)$ 是 o_i 的儿子，那么称 $o_2,\cdots,o_n$ 是 o_1 的后代，而且称 o_1 是 $o_2,\cdots,o_n$ 的祖先。

规则 1　对于一个 XML 对象 (n,a,v)，如果值域 v 是一个 XML 对象集合 $\{v_1,v_2,\cdots,v_n\}$，那么它可以按照分解规则

冯学斌　讲师，博士生，研究方向为数据库安全；洪　帆　教授，博士导师，研究方向为信息安全。

被改写为 XML 对象的集合{(n,a,v_1), (n,a,v_2),…,(n,a,v_n)}。

规则 2 如果{(n,a,v_1), (n,a,v_2),…,(n,a,v_n)}是一个 XML 对象集合,那么它可以按照合成规则被合成为一个 XML 对象$(n,a,\{v_1,v_2,\cdots,v_n\})$。

定义 3 对于一个 XML 对象(n,a,v),可以反复使用分解规则,直到所有成员的值域是一个常量为止,那么该 XML 对象最终可以被分解为多个形如$(n_1,a_1,(n_2,a_2,(\cdots,(n_m,a_m,v)\cdots)))$的元素组成的集合(其中 $v\in C(\{\text{Null}\})$。我们称这样的每个元素$(n_1,a_1,(n_2,a_2,(\cdots,(n_m,a_m,v)\cdots)))$为范式。

对于一个 XML 对象,可以反复使用分解规则,将其最终被分解为范式集合。反之,对于一个范式集合,可以反复使用合成规则,最终合成为一个 *XML* 对象。因此有以下结论成立:

定理 1 任一 XML 对象都等价于某个范式集合。

3 多级 XML 数据模型

我们进一步对基本 XML 模型进行扩充,以便处理安全标签。

定义 4 安全标签集合 LC 是常量 C 的一个子集,并且 LC 的成员满足偏序关系。

定义 5 一个多级安全 XML 对象是一个具有安全标签属性的 XML 对象,即一个多级安全 XML 对象是一个由三个元素(名字、属性、值)组成的三元组 (name, attrib, value)。其中,

(1) 名字 name 是一个常量,name$\in C$;

(2) 属性 attrib 是一个由 $m(m\geqslant 1)$个二元组组成的列表$((an_1,av_1),(an_2,av_2),\cdots,(an_m,av_m))$,其中 an_1 是安全标签属性,$av_1\in$LC 是安全标签值,$an_i(2\leqslant i\leqslant m)$是属性名,$av_i(2\leqslant i\leqslant m)$是属性值;

(3) 值 value 是一个空元素 Null、常量或者 $n(n\geqslant 1)$个多级安全 XML 对象组成的集合$\{o_1,o_2,\cdots,o_n\}$。

定理 2(多级安全完整性约束) 对于任意 XML 对象 o,o_1,如果 o_1 是 o 的儿子,那么 o_1 安全标签支配 o 的安全标签。

推论 2 对于多级安全 XML 对象 o_1,o_2,如果 o_2 是 o_1 的后代,那么有 o_2 的安全标签支配 o_1 的安全标签。

推论 3 对于多级安全 XML 对象 $o,o_1,o_2,\cdots,o_n$,如果 $o_1,o_2,\cdots,o_n$ 是 o 的后代,那么 $o_1,o_2,\cdots,o_n$ 安全标签的下界支配 o 的安全标签。

定义 7 如果两个多级安全 XML 对象 $o_1=(n,a_1,v_1)$,$o_2=(n,a_2,v_2)$互为兄弟,其中 $a_1=((an_{11},av_{11}),(an_{12},av_{12}),\cdots,(an_{1m},av_{1m}))$,$a_2=((an_{21},av_{21}),(an_{22},av_{22}),\cdots,(an_{2m},av_{2m}))$,并且 $av_{11}\neq av_{21}$,$an_{1i1}=an_{2i}$,$av_{1i1}=av_{2i}$(对于所有 $2\leqslant i\leqslant m$),则 o_1,o_2 互为不同安全标签下的实例。

规则 3 对于一个多级安全 XML 对象,我们可以按照多级安全 XML 对象分解规则进行分解。

(1)对于多级安全 XML 对象 o,重复使用分解规则将其改写为范式集合 $V=\{(n_{11},a_{11},(n_{12},a_{12},(\cdots,(n_{1m_1},a_{1m_1},v_1)\cdots))),(n_{21},a_{21},(n_{22},a_{22},(\cdots,(n_{2m_2},a_{2m_2},v_2)\cdots)))\cdots,(n_{y1},a_{y1},(n_{y2},a_{y2},(\cdots,(n_{ym_y},a_{ym_y},v_y)\cdots)))\}$;

(2)对于 V 中的每个元素$(n_{i1},a_{i1},(n_{i2},a_{i2},(\cdots,(n_{im_i},a_{im_i},v_i)\cdots)))$,根据 a_{im_i} 中安全标签元素值的不同,将范式集合划分为不同子集;

(3) 对于每个范式子集,重复使用合成规则,分别合成为 XML 对象,分别提交相应的数据库管理。

规则 4 对于多个 XML 对象,我们可以按照多级安全 XML 对象合成规则对其合成

(1) 将分别管理的各个 XML 对象分解为范式列表;

(2) 将上一步得到的范式列表合并;

(3) 反复使用合成规则,将第 2 步的结果合成为 XML 对象。

4 系统结构

系统的访问控制由 TCB 来实施。基于多级安全 XML 对象的分解算法,我们采用一个简单的 TCB 结构来实施访问控制策略,从而便于安全模型的验证,比较容易达到较高的安全等级。系统体系结构如图 1 所示。在图中混合采用了 Native XML 数据库和 XML-Enabled 关系数据库,有利于原有数据库系统的迁移和集成。

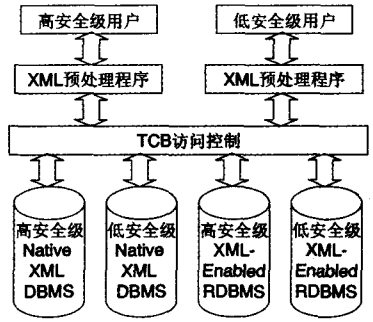

图 1 系统体系结构

系统为每个用户分配一个 XML 预处理进程,该进程的安全标签等于用户的安全标签。系统使用安全标签不同的数据库分别存储 XML 数据。TCB 根据进程的安全标签和用户的安全标签实施强制访问控制策略。由于 TCB 不参与 XML 数据库的其他管理事务,可以简化 TCB 的功能设计,从而避免了复杂系统功能本身可能带来的缺陷,也便于进行形式化验证,从而使得系统能够满足高安全等级的要求。

5 访问控制策略实施过程

关系数据库中数据的操纵通常使用 SQL 语言。和关系数据库不同,XML 数据库上的数据操作通常通过 XQuery 和 XPath 语言完成。我们将 XML 数据库上的操作分为 4 类:Query、Insert(包括 Append)、Update、Remove。下面分别讨论这四类操作:

1. Query 操作的主要过程

(1) 用户向预处理进程发出查询请求;

(2) 预处理进程向安全标签被用户安全标签支配的数据库发出查询请求;

(3) TCB 根据用户安全标签和数据库安全标签过滤查询请求,如果用户安全标签支配数据库安全标签,就许可查询访问,否则禁止;

(4) 预处理进程将各数据库返回的 XML 对象分解为范式集合,然后将所有范式集合合并并且重新合成为 XML 对象,根据查询请求,并向用户返回符合查询条件的 XML 对象。

2. Insert(Append)操作的主要过程

(1) 用户向预处理进程发出插入节点 o 请求;

(2) 预处理进程向安全标签被用户安全标签支配的数据库发出查询请求；

(3) TCB根据用户安全标签和数据库安全标签过滤查询请求，如果用户安全标签支配数据库安全标签，就许可查询访问，否则禁止；

(4) 预处理进程将各数据库返回的XML对象分解为范式集合，然后将所有范式集合合并并且重新合成为XML对象；根据Select属性指定的条件，确定插入位置的父节点 p，那么节点 o 的范式为 $(n_1,a_1,(n_2,a_2,(\cdots,(n_p,a_p,(n_o,a_o,v_o))\cdots)))$；在属性列表 a_o 中加入安全标签，标签值等于用户安全标签，新的属性列表为 a_v'；

(5) 预处理进程向安全标签等于用户安全标签的数据库发出插入节点请求，请求插入节点 $(n_1,a_1,(n_2,a_2,(\cdots,(n_p,a_p,(n_o,a_o',v))\cdots)))$；

(6) TCB根据用户安全标签和数据库安全标签过滤查询请求，如果用户安全标签等于数据库安全标签，就许可插入访问，否则禁止；

(7) 预处理进程将数据库返回的结果提交用户。

3. Update操作的主要过程

(1) 用户向预处理进程发出请求，请求更新指定节点的值为 v；

(2) 预处理进程向安全标签被用户安全标签支配的数据库发出查询请求；

(3) TCB根据用户安全标签和数据库安全标签过滤查询请求，如果用户安全标签支配数据库安全标签，就许可查询访问，否则禁止访问；

(4) 预处理进程将各数据库返回的XML对象分解为范式集合，然后将所有范式集合合并并且重新合成为XML对象；根据Select属性指定的条件，确定需要更新的节点 o，假设节点 o 的范式为 $(n_1,a_1,(n_2,a_2,(\cdots,(n_o,a_o,v_o)\cdots)))$；

(5) 预处理进程向安全标签等于用户安全标签的数据库发出请求，查询节点 $(n_1,a_1,(n_2,a_2,(\cdots,(n_o,a_o',*)\cdots)))$ 是否存在(其中 a_o' 由属性列表 a_o 中的安全标签值更新为用户安全标签得到，$*$ 代表任何值)，如果存在则更新其值为 v，否则请求插入节点 $(n_1,a_1,(n_2,a_2,(\cdots,(n_o,a_o',v)\cdots)))$；

(6) TCB根据用户安全标签和数据库安全标签过滤查询请求，如果用户安全标签等于数据库安全标签，就许可更新(或插入)访问，否则禁止；

(7) 预处理进程将数据库返回的结果提交用户。

4. Remove操作的主要过程

(1) 用户向预处理进程发出删除节点的请求；

(2) 预处理进程向安全标签被用户安全标签支配的数据库发出查询请求；

(3) TCB根据用户安全标签和数据库安全标签过滤查询请求，如果用户安全标签支配数据库安全标签，就许可查询访问，否则禁止；

(4) 预处理进程将各数据库返回的XML对象分解为范式集合，然后将所有范式集合合并并且重新合成为XML对象；根据Select属性指定的条件，确定所要删除的节点 o，假设节点 o 的范式为 $(n_1,a_1,(n_2,a_2,(\cdots,(n_p,a_p,(n_o,a_o,v_o))\cdots)))$；

(5) 预处理进程检查 a_o 中安全标签值是否等于用户安全标签，如果相等，则向该数据库发出删除节点请求；否则返回错误；

(6) TCB根据用户安全标签和数据库安全标签过滤查询请求，如果用户安全标签等于数据库安全标签，就许可删除访问，否则禁止；

(7) 预处理进程将数据库返回的结果提交用户。

6 安全性分析

多级安全的思想来自于BLP模型[8,9]。BLP模型是一个状态机模型，该模型指出，如果一个的初始状态是安全的，并且任一状态转换规则是安全的，那么系统自身是安全的。

定理3 对于任意一个XML数据库T，存在一个初始状态 Z_0，该状态是安全的。

证明：令 $Z_0=(\varphi,\varphi,F)$，在此状态下，任一主体所访问的客体XML对象为空，并且访问控制矩阵为空，因此状态 Z_0 是安全的。

定理4 操作Query、Insert、Update、Delete符合基本安全属性。

证明：因为在操作Query、Insert、Update、Delete的过程中，TCB对数据库读操作进行监控(每个操作的第3步)，从而保证用户安全标签支配数据库的安全标签，并且各数据库的安全标签等于操作对象节点的安全标签，所以每个用户的安全标签支配其访问对象的安全标签。因此对于任一 $b=(s,o,\underline{x})\in S\times O\times A$，其中 $\underline{x}(\{\text{Query, Insert, Update, Delete}\})$，有 $f(s)\geqslant f(o)$ 成立。也就是说对于任一安全状态 $v=(b,M,f)\in V$，有操作Query、Insert、Update、Delete符合基本安全属性。

定理5 操作Query、Insert、Update、Delete满足 $*$ 属性。

证明：因为在操作Insert、Update、Delete的过程中，TCB对数据库写操作进行监控(每个作的第6步)，从而保证用户安全标签等于数据库的安全标签，并且各数据库的安全标签等于操作对象节点的安全标签，所以每个用户的安全标签等于其访问对象的安全标签。因此对于任意 $b_1,b_2\in S\times O\times A$，其中 $b_1=(s,o_1,\underline{x}_1)$，$b_2=(s,o_2,\underline{x}_2)$，$\underline{x}_1\in\{\text{Insert,Update,Delete}\}$，$\underline{x}_2\in\{\text{Query,Insert,Update,Delete}\}$，有 $f(o_1)\geqslant f(o_2)$ 成立。也就是说对于任一安全状态 $v=(b,M,f)\in V$，有操作Query、Insert、Update、Delete满足 $*$ 属性。

因此，根据BLP模型公理及推论，可以得出以下结论：

推论4 本文所定义的模型是安全的。

结论 传统的多级安全数据库研究都是基于关系数据库的，由于关系数据库本身的限制，半结构化数据很难在关系数据库中有效地进行管理。XML数据库为我们提供了半结构化数据的管理方法，然而由于XML数据库是一种树状信息结构，不同于关系数据库的二维表格结构，因此，无法将关系数据库的研究成果直接应用于XML数据库的管理中。本文建立了一个支持多实例的多级安全XML数据库模型，通过对XML对象进行分解，使得XML对象能够根据安全等级分别存储，从而能够达到高安全等级的需求，并且对该模型进行了安全性分析，从而为使用XML数据库处理敏感信息提供有效的理论支持。在今后的工作中，我们将对多种访问策略(例如自主访问控制、基于角色的访问控制)的结合进行进一步的研究，也将进一步探讨XML安全模型的实现机制。

参考文献

1 Damiani E, di Vimercati SDC, Paraboschi S, Samarati P. A fine-grained access control system for XML documents. ACM. Transactions on Information and System Security (TISSEC), 2002, 5:

169～202
2 Beritino E, Castano S, Ferrai E. Securing XML documents with Author-X. IEEE Internet Computing, May/June 2001. 21～31
3 Bertino E, Sandhu R. Database Security-Concepts, Approaches, and Challenges. IEEE Transactions on Dependable and Secure Computing, 2005, 2(1): 2～19
4 LI Lan, HE Yong-Zhong, FENG Deng-Guo. A fine-grained mandatory access control model for XML documents. Journal of Software, 2004, 15(10): 1528～1537
5 Kuper G, Massacci F, Rassadko N. Generalized XML Security Views. SACMAT 2005. In: 10th ACM Symposium on Access Control Models and Technologies, Stockholm, Sweden, June 2005. 77～84
6 Kudo M, Hada S. XML Document Security based on Provisional Authorization. In: Proceedings of the 7th ACM Conference on Computer and Communication Security, Athens, Greece, 2000. 87～96
7 Liu Mengchi. A logical foundation for XML. In: Proceedings of the 14th Int'l Conf. on Advanced Information Systems Engineering, Toronto, Canada, 2002. 568～583
8 LaPadula L J, Bell D E. Secure computer systems: A mathematical model: [Technical Report 2547 (Volume II)]. The MITRE Corporation, Bedford, Massachussetts, May 1973
9 Bell D E, LaPadula L J. Secure Computer Systems: Mathematical Foundations and Model: [Technical Report M74-244]. The MITRE Corporation, Bedford, Massachussetts, May 1973

(上接第 145 页)

图 5 给出了不同缓存命中率下 postmark 每秒完成的事务数量。从图 5 可以看出，随着缓存命中率的提高，请求吞吐率得到提高。当命中率达到 95%后，缓存不命中带来的请求处理时延对系统的事务吞吐率的影响比较弱，表现出事务吞吐率逐渐平稳的趋势。

同样，我们还获得各个命中率下请求比率，其计算公式是 ratio = MS 向 BS 发起的请求数量 / MS 收到的客户段的请求数量。请求比率趋势图如图 6 所示。每个计算中包含创建测试文件集所需要的 10000 次通信。从图 6 可以看出，缓存命中率将明显影响 BS 的负载压力。

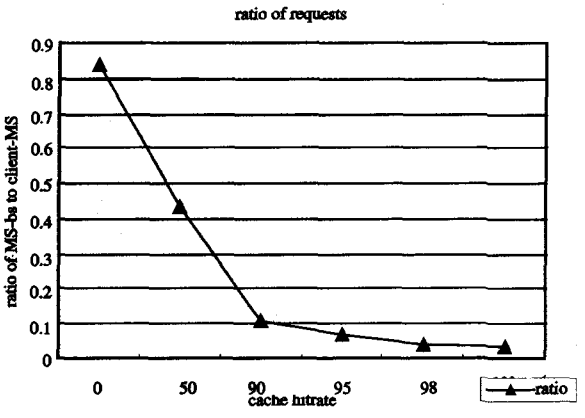

图 6 BS 请求数量随缓存命中率变化图

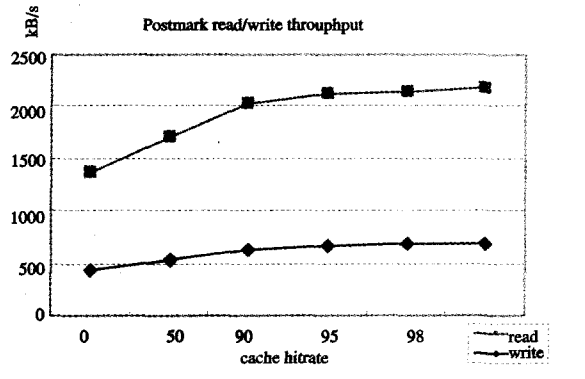

图 7 数据读写吞吐率随缓存命中率变化图

图 7 是同一测试获得的各种缓存命中率下客户端的读写吞吐率。尽管 BWFS 通过文件数据访问的控制流和数据流分离机制给用户提供并发的数据读写，但由于文件数据较小，元数据请求对小文件数据读写吞吐率的影响要比对大文件的数据读写吞吐率的影响明显得多。反应到图 7 上，元数据请求过程的时延对系统读写吞吐率的影响非常明显。同样，当缓存命中率达到 95%后，由于缓存不命中带来的元数据处理时延对整个读写的影响比较弱，系统读写吞吐率表现出平稳的趋势。

5 相关研究

由于网络存储应用的多样化，分布式文件系统元数据服务研究成为热点，其关键问题是如何分布元数据请求负载。已有研究成果可以归纳为两类。一类是文件系统目录子树分区法[4]，目录及其所有子节点的请求由一个元数据服务器完成处理。这种策略人为地限制了用户的请求不能跨服务器完成，并且元数据服务不能根据文件系统目录结构进行深度扩展。第二类是以哈希的方式管理分布[5～7]。哈希法可以加快元数据到服务器的映射过程。但由于哈希法主要以静态因素为函数参数，它不能很好地适应系统动态的规模变化，哈希参数变化的波及面很广，对系统的扩展能力具有很大的限制作用。

结束语 本文描述了 BWMMS 的层次元数据管理机制，动态灵活的元数据分布策略。同时，由于各个 MS 的元数据分布信息缓存管理的有效性对元数据请求的处理延迟和元数据请求的并发同步有着至关重要的作用，我们着重描述各个元数据服务器对分布信息缓存的管理，并通过实验评测缓存管理主要因素对元数据请求处理延迟的影响。

在后面的工作中，我们将对系统进行更详尽的评测，将针对多种典型应用进一步验证 BWMMS 动态灵活的元数据分布管理机制的适用性，将进行元数据分布策略的自学习能力研究。

参 考 文 献

1 Ousterhout J K, Costa H D, Harrison D, et al. A trace-driven analysis of the Unix 4.2 BSD file system. In: Proceeding of the 10th ACM Symposium on Operating Systems Principles (SOSP '85), Dec. 1985. 15～24
2 杨德志，黄华，张建刚，许鲁. 大容量、高性能、高扩展能力的蓝鲸分布式文件系统. 计算机研究与发展，2005，42(6)
3 postmark. http://www.netapp.com/tech_library/3022.html
4 Levy E, Silberschatz A. Distributed file systems: Concepts and examples. ACM Computing Surveys, 1990, 22(4)
5 Corbett P F, Feitelso D G. The Vesta parallel file system. ACM Transactions on Computer Systems, 1996, 14(3): 225～264
6 Brandt S A, Miller E L, Long D D E, et al. Efficient Metadata Management in Large Distributed Storage System. In: Proc. of the 20th IEEE/11th NASA Goddard Conference on Mass Storage Systems and Technologies, 2003
7 Yan J, Zhou Y L, Xiong H, et al. A Design of Metadata Server Cluster in Large Distributed Object-based Storage. In: Proc. of the 21th IEEE/12th NASA Goddard Conference on Mass Storage Systems and Technologies, 2004