一个开放环境中 Client/Server 结构的日志审计系统

吴涛　　丁伟　　高毓航

（东南大学计算机科学与工程系，南京 210096）

【摘要】本文介绍了一个分布式日志审计系统的设计和实现，该系统基于 Client/Server 的体系结构，实现了开放环境中用户行为的无否认审计。该系统收集远程 Client 端的日志及配置信息，汇总、整理后存入 Server 端的数据库；向用户提供各种查询报表，同时向管理员提供灵活方便的系统管理功能。

【关键词】日志；审计；数据库；无否认

中图分类号：TP311

A Client/Server Log Audit System in Open Environment

Wu Tao　Ding Wei　　Gao Yuhang

(Southeast University, Computer Science Dept., 210096 Nanjing, P.R.China)

【Abstract】This paper introduces the design and realization of a distributive log audit system. Based on Client/Server structure, the system realized no-deny audit to user behavior in open environment. It gathers log and configuration information from distance clients and puts them into database after some processing work. A variety of report forms as well as flexible system management is available to users and administrators.

【Key words】log; audit; database; no-deny

计算机系统的防护机制一旦建立以后，就需要对他们进行监控。这种监控系统行为的过程叫做审计（auditing）。日志(log)文件是安全系统的一个重要组成部分，它们形成了计算机系统发生情况的

作者简介：吴涛，硕士研究生，主要研究方向为网络应用。

丁伟，工学博士，东南大学计算机系副教授、硕导，主要研究方向包括网络管理、网络安全、网络体系结构、开放分布式处理等。

高毓航，硕士研究生，主要研究方向为网络安全。

定稿日期：2000-08－01

历史记录（或称审计跟踪）。随着 Internet 的逐渐推广，网上行为的审计跟踪变得越来越重要。

传统的日志审计工具针对单个服务器进行，分析对象一般为 Web 日志文件，通过对文件内容的分类和汇总，获得服务器的访问概况。系统的功能比较标准，能满足一般商业站点的普通审计要求。但大部分审计工具只能处理使用“标准（常用日志格式）”的日志文件，灵活性与适应性较差，难以实现不同日志之间相关信息的综合处。此外，当被审计的日志数量增加到一定程度时，传统工具采用文件系统方式管理数据，其速度和性能将会急剧下降。更重要的是，这种方式的审计各子系统间彼此独立，无法在一个有关的范围内（如拥有多台服务器的内部网系统）对单个用户的整体行为进行审计。

本文将介绍一个分布式的日志审计系统，它采用 client/server 结构，可用于在开放环境下对多个不同类型服务器的日志进行整体审计。其主要的设计和实现思路是用“中间日志格式”屏蔽不同原始日志中语法和语义的差异，使其一方面不仅可用于 WEB 服务器，也可用于对其他各种类型服务器的审计，另一方面还可以方便地扩展到各种类型的 OS 平台。系统以配置管理的方式使其控制的覆盖范围可以任意变化；采用中央数据库系统管理数据来提高系统的性能和安全性；利用数字签名技术，提供无否认条件下对用户行为的审计，是本系统的另一个重要特征，它保证了审计结果的可靠性。本文还讨论了系统实现过程中遇到的一些问题，如日志读取周期的选择；时钟的统一；以及验证未通过时采取何种反应策略等。

1.关于日志格式

在 INTERNET 环境中，一般的服务器日志均包括以下一些条目：

- Host：标识请求主机。大部分服务器都能记录 IP 地址或主机名。
- Authuser：该域记录访问的用户名。
- Date-time：服务器在每次完成请求后记录它自己的当前日期和时间。
- Request：这个域记录指定的请求名。包括请求的方法、请求的文件名和协议名称。
- Status：返回的状态代码。
- Bytes：显示本次请求所传输的总的字节数。

除此之外，一些服务器还记录扩展日志和错误日志等。扩展日志包括引用日志和用户代理。引用日志主要回答两个问题：用户是从哪里来的以及用户是如何来的？用户代理记录每次请求所使用的浏览器；错误日志记录服务器事件，包括启动和关闭的消息，也为每次不成功的访问记录扩展跟踪信息。

2.系统总体框架

2.1 系统功能

本系统的主要功能是向应用系统提供基于 INTERNET 环境的规范的审计服务平台，同时采用数字签名提供的数据完整性保护和无否认功能来保证这种开放环境中审计结果的真实和可靠。具体的基本功能包括：

- 客户端代理收集不同服务器上的原始日志，将其转换为统一的“中间日志格式”字符流并加上数字签名，安全传送给远程的审计服务器；
- 审计服务器接收客户端传送来的字符流，进行签名验证，最后将数字签名及转换格式后的字符流存入审计信息数据库；
- 用户可以查询各 Server 的总体情况，如访问总次数、资源数及相关细节；
- 用户可以查询特定用户、特定资源的访问情况，还可以查询特定时间段内特定用户（类）对特定资源（类）的具体访问细节；
- 管理员可以调整系统的覆盖范围，控制各类运行参数(包括 Client 端和 Server 端)；
- 管理员可对审计信息数据库中的信息进行签名验证。

2.2 系统体系结构

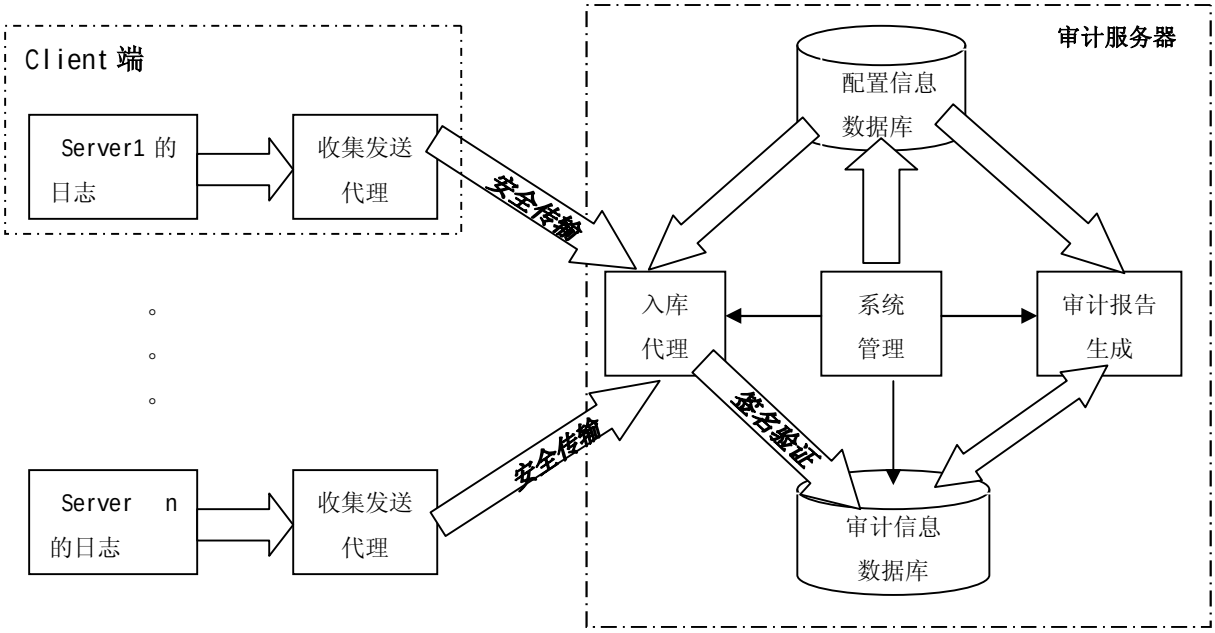


图 1 系统总体框架

系统的基本运行方式采用的 Client/Server 结构，总体框架如图 1 所示。系统中的客户端（Client）指的是各需要审计的远程 Server。安装在客户端（Client）的发送代理收集原始日志、将其转换并加上签名后发送给服务器端。服务器端（Server）则由三部分组成：入库代理用于接收来自客户端的数据，验证后存入审计数据库；系统管理部分协调其它各部分的工作并接受管理员的配置调整信息；报告生成部分接收用户的查询请求，生成报表返回给用户。

2.3 中间数据格式的结构

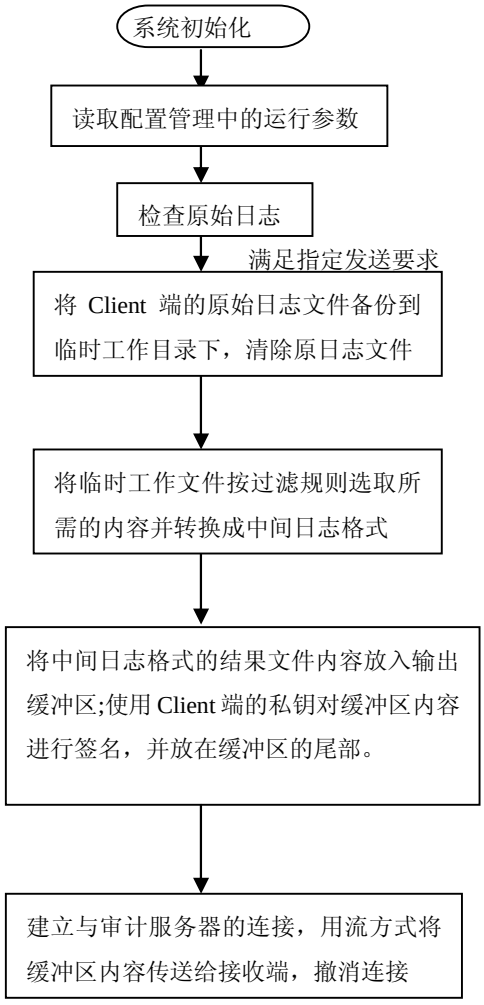
原始日志的语法和语义各不相同，如果针对每种日志开发专门的处理程序，这将会带来大量的冗余工作量，程序的灵活性与适应性都较差，且难以实现不同格式日志间相关信息的综合处理。所以在系统中，我们采用“中间日志格式”来屏蔽不同日志间的差异，提高处理效率。其结构如下：

char	*user_name	用户编号
char	*user_host	用户主机名
char	*ip_address	用户 IP 地址
char	*access_time	访问时间
char	*url	用户所访问文件的 URL
char	*status	访问状态码(标准：RFC2068)
int	size	如果访问成功，表示所访问的文件大小
char	*domain	源域名称
。。。		

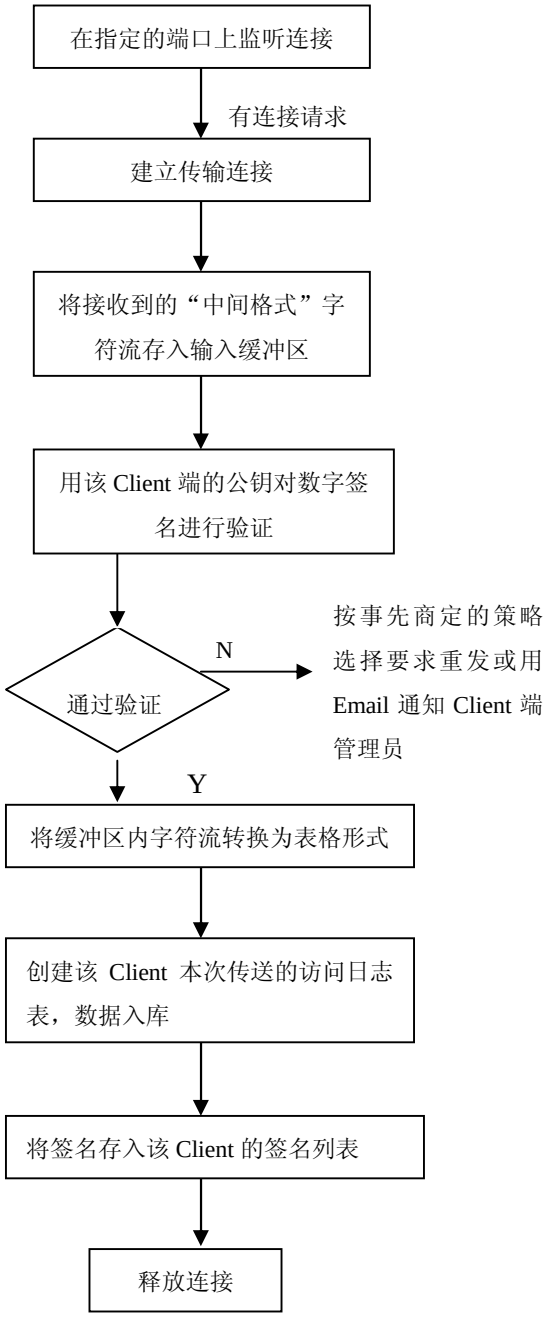
3.系统的实现

3.1 工作流程

3.1.1Client 端

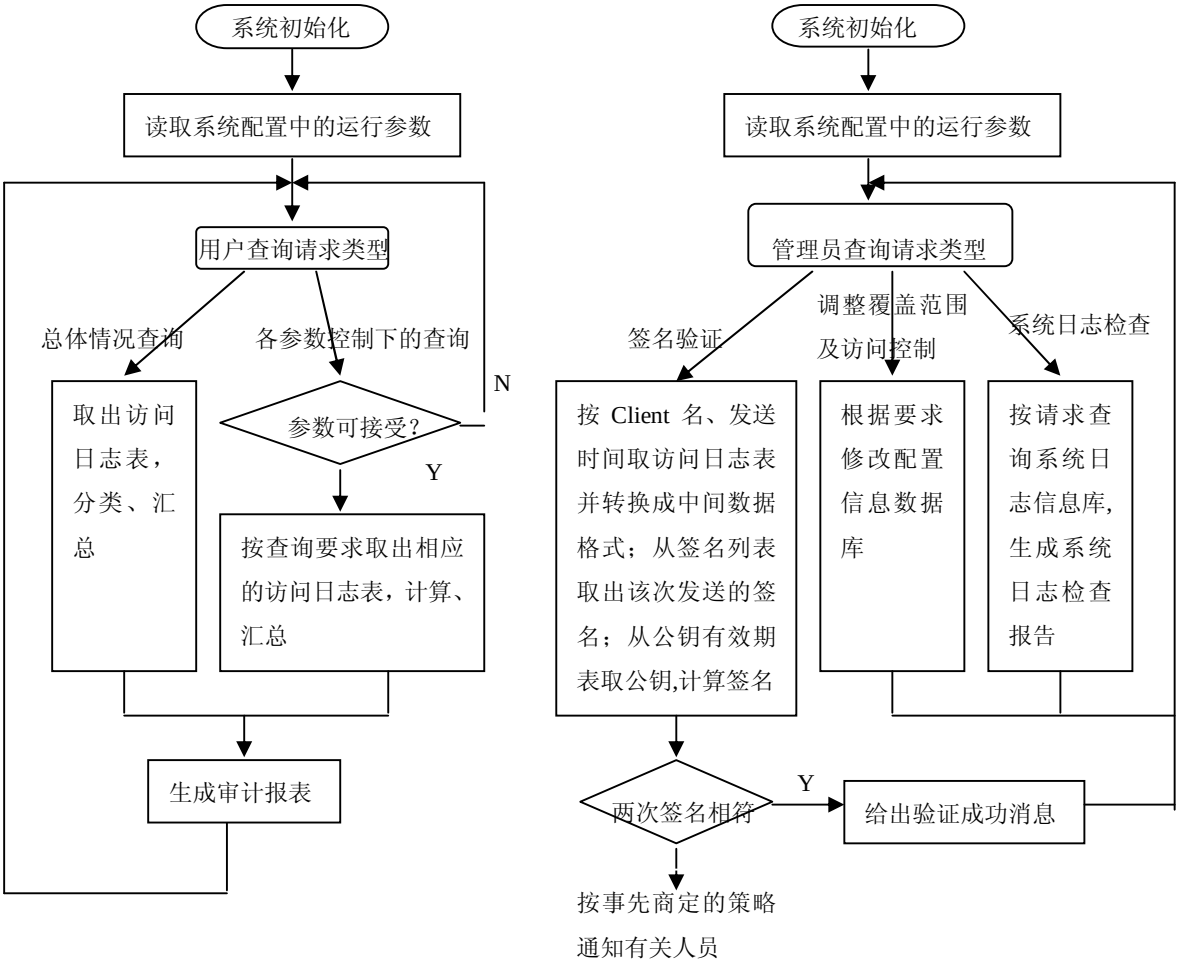


客户端代理程序的工作流程



入库代理程序的工作流程

3.1.2Server 端



普通用户及管理员查询程序工作流程

3.2 数据结构(数据库)

在审计信息数据库内，我们主要设计了四类表格：访问日志表、签名列表、公钥有效期表和覆盖域表。访问日志表记录每次传送的日志信息；签名列表存放日志发送的时间及其对应的签名；公钥有效期表存放每个公钥的起始、终止时间；覆盖域表记录系统内各个域的信息。以下是四类表格的 C 语言描述。

3.2.1 访问日志表的 C 语言描述

```
typedef struct Access_DomainName_SendTime
{
    DBINT    idtentity      id;                 /*序列号*/
    DBCHAR          *user_code;                 /*访问者编号*/
    DBCHAR          *user_host;                 /*访问者主机名*/
    DBCHAR          *user_IP;                   /*访问者 IP 地址*/
    DBDATETIME      access_time;                /*访问时间*/
    DBCHAR          *access_URL;                /*访问页面的 URL*/
    DBCHAR          *status_code;               /*返回的状态码*/
    DBINT           byte_number;                /*传输的字节数*/
```

```
}ACCESS_DN_ST
```

3.2.2 签名列表的 C 语言描述

```
typedef struct Sign_DomainName
{
    DBINT    idtentity      id;
    DBDATETIME              send_time;          /*日志发送时间*/
    DBTEXT          *sign;                  /*数字签名*/
}SIGN_DN
```

3.2.3 公钥有效期表的 C 语言描述

```
typedef struct Keys_DomainName
{
    DBINT    idtentity      id;
    DBDATETIME              begin_time;         /*公钥有效期的起始时间*/
    DBDATETIME              end_time;           /*公钥有效期的终止时间*/
    DBBINARY                public_key;         /*公钥*/
}KEYS_DN
```

3.2.4 覆盖域表的 C 语言描述

```
typedef struct Cover_Domain
{
    DBINT    idtentity      id;
    DBINT                   *domain_code;       /*区域编号*/
    DBCHAR                  *domain_name;       /*区域名*/
    DBCHAR                  *describe;          /*其它描述信息*/
}COVER_DN
```

4.其它有关问题和系统功能的改进

4.1 设计和实现中需考虑的有关问题

- 日志发送策略，即日志读取周期的选取。可以按指定时间发送或在日志文件到达指定长度时发送，也可以是这两者的结合。本系统旨在有关的应用系统提供一个安全的审计服务平台，而日志在 Client 端存在不安全隐患，但频繁的发送显然也是不可取的，所以合理的发送策略是保证数据安全和系统稳定的关键。

- 时钟不一致。每个 Client 端和审计服务器都有自己的时钟，访问日志中记录的是各 Client 端的时钟；日志发送时 Client 端和审计服务器分别记录自己的时钟，这显然会带来一些问题。可以考虑两种解决方案：一种是使用审计服务器的时钟，每隔一段规定的时间 Client 端校对并调整自己的时钟；另一种方案是 Client 端保留各自的时钟，在日志发送时由审计服务器记录两者的时间差。

- 数据入库时，验证未通过时的处理。入库代理在每次入库前使用该 Client 端的公钥验证数据的完整性和可靠性，传输错误、数据被篡改以及 Client 端或审计服务器端的计算错误都会导致验证失败，这时可以考虑要求 Client 端重发或用 Email 通知 Client 端的管理员，因此 Client 端的日志文件需要保留一段时间。

- 服务器端缓冲区的设置。在审计服务器的入库代理处有可能存在请求入库队列。一旦系统覆盖范围扩大，Client 增多，日志发送的频率加大，各 Client 的入库请求就有可能在 Server 端形成队列。

可以要求各 Client 在指定的时间发送以减低形成队列的可能性，选择适当的缓冲区个数和服务策略对于提高系统性能也很重要。

l 系统自身日志检查。系统日志记录了普通查询用户和管理员的操作轨迹，是发现安全事件、评判系统安全程度、进而加强系统安全管理的重要手段。系统日志本身的安全是日后检查的基础，而且系统日志检查需考虑由第三方来完成。

4.2 下一步希望对系统的功能做以下扩展：

l 支持用户计费审计：日志信息记录了用户对资源的访问情况，如果资源所有者有收费请求，系统可以根据汇总后的用户审计报表，计算用户的访问费用。

l 支持用户会话审计："会话"被定义为某段时间内来自指定 IP 地址的一系列请求，也称为"唯一访问(Unique visit)"。确定了一个会话后，审计程序就能给出这个会话的额外信息，包括每段时间内某访问者的会话次数；总的会话持续时间的平均值；每次会话期间的平均访问页面数。解决这个问题的关键在于会话边界的确定。

l 支持会话路径审计：如果能识别指定的用户会话，那么就能够跟踪该用户的路径。分析程序跟踪每一个会话，找出最常用的路径，可以根据这些数据来确定最受欢迎的入口页、出口页以及最常用的会话路径。

l 进一步提高审计精确度：内容缓存是影响审计精确度的主要因素，包括本地缓存和代理缓存。大部分 ISP 和镜像站点为它们的成员访问 Web 提供了缓存代理，这些访问请求都不能被服务器发现，会影响审计的精确度。

5.结束语

随着 Internet 的逐渐推广，网上行为的审计跟踪在某些方面非常重要。是远程教育、电子商务等应用领域必不可少的支持工具。本系统的主要特点在于不仅能完成常规的日志审计，还建立了一个开放环境下，对用户的行为提供无否认审计的框架模型。可以实现多个远程 Server 的日志收集和审计，并且采用数据库维护日志及配置信息，在当网络规模较大，日志数量很多时查询等操作速度要优于采用文件系统管理数据。经进一步完善，相信该系统对于分析较大规模网络中用户的行为，对用户的行为提供无否认的审计仲裁等方面会有更多的探索。

参考文献

1. Robin Burk, David B.Horvath 《UNIX 技术大全－－Internet 卷》北京 机械工业出版社 1999
2. http://www.netstore.de/stats/index.html
3. http://www.analog.cx/
4. http://www.microsoft.com/siteserver/site/
5. 陶浦洲 李强 《Sybase 数据库技术大全》北京 科学出版社 1997