文章编号:2095-476X(2015)3/4-0069-06

基于数据库安全保障的审计系统的设计与实现

张伟伟, 郑峰弓, 张秋闻

(郑州轻工业学院 计算机与通信工程学院,河南 郑州 450001)

摘要:针对目前大多数据库内部操作不透明,易造成机密信息泄漏、资源滥用等问题,设计了一套数据库审计系统.该系统采取旁路监听的方式,对同一局域网中服务器的指定端口进行流量抓取,获得监听网络数据,通过分析抓取到的网络数据包,将审计信息及时备份到安全的数据库中,以备查找与分析.系统通过监控外界用户对数据库的访问操作,记录操作行为,并及时反馈给审计人员,从而使审计人员能够实时掌握数据库系统的安全状态,有效保障数据库的安全.测试结果表明,系统的有效性和实时性良好.

关键词:数据库安全;审计系统;MySQL;SQL Server

中图分类号:TP392 **文献标志码**:A **DOI**:10.3969/j.issn.2095-476X.2015.3/4.015

Design and implementation of audit system based on database security control

ZHANG Wei-wei, ZHENG Feng-gong, ZHANG Qiu-wen

(*College of Computer and Communication Engineering, Zhengzhou University of Light Industry, Zhengzhou* 450001)

Abstract: In view of the drawbacks that internal operations of present database are mostly opaque, which likely causes confidential information leakage and resource abuse, a set of database audit system was proposed. The bypass to monitor was adopted in the system, through traffic grab to the specified port number on the server in the same LAN, network packets could be obtained. And then these captured network packets were analyzed, finally the analyzed audit information would backup to the secure database in a timely manner for search and analysis. The system could effectively ensure the security of database by monitoring the external users access to the database operation, recording operation behavior, and timely feeding back to the auditor controlling the database system security status in real time. The test results showed that the system had good character of effectiveness and real-time.

Key words: database security; audit system; MySQL; SQL Server

0 引言

目前,国内外大多数据库系统没有对外界用户提供访问接口,无法从数据库系统内部对系统的安全性采取加强措施,因此只有利用数据库安全加固技术来提高系统安全性能.现有数据库系统的安全维护主要通过数据库日志文件完成.这样的方式存在很大的弊端,例如:数据库审计的开启会影响数

收稿日期:2014-11-05

基金项目:国家自然科学基金项目(61403349,61302118,61402422);河南省教育厅科技攻关重点项目(14B520066,15A520033);郑州轻工业学院博士基金项目(2013BSJJ044)

作者简介:张伟伟(1986—),女,河南省驻马店市人,郑州轻工业学院讲师,博士,主要研究方向为智能计算.

据库本身的性能,日志文件的安全管理必须依赖数据库系统,日志记录的复杂性也增加了查找和分析的难度,因此有效的数据库安全防护技术是必不可少的. 但如果通过重新设计数据库的存储方式来增强安全性,必将带来很多不必要的开销. 而数据库安全加强技术则可以轻松化解以上难题,既不需要改变原数据库,还可以提供更有效的安全保障[1-4],数据库审计系统便是由此而来的. 数据库审计分为对数据库事件进行记录和对记录信息进行分析两部分. 国内外学者在数据库审计方面做了很多研究,并取得了丰富的研究成果. M. Bishop[5] 认为审计应该包含日志和审计分析两部分. 针对不同的技术和侧重点,研究人员也分别提出了不同的数据库审计系统,比如基于网络侦听,基于 XML 和 Web Service,基于入侵检测,基于误用检测等[2,6-8]的数据库审计系统. 但这些数据库审计系统大多存在效率低、功能不定和安全保障弱等问题. 鉴于此,本文针对 MySQL 和 SQL Server 两大主流数据库设计一种数据库审计系统,以期在不改变已有数据库的网络结构、不影响其他服务器性能的前提下,达到外界用户对数据库操作行为进行监控的目的.

1 审计系统模块搭建

数据库审计系统的工作目标是实时监视并记录外界用户对数据库的操作行为[9-12]. 当事件发生时,数据库审计系统自动记录操作者、操作时间、操作对象和操作行为,并实时存储,以实现审计信息的查询、分析等功能. 审计系统分为审计引擎核心块、事件收集器、审计信息入库控制块和审计中心四个模块,审计系统工作原理如图 1 所示.

1.1 审计引擎核心块

审计引擎核心块对事件收集器收集的数据包顺序进行以太网协议解析、数据库通信协议解析,将分析到的审计信息直接通过 ZeroMQ 转送给审计信息入库控制块,不负责审计信息的入库操作,这就减轻了审计核心部分的运行负荷. 审计引擎核心块工作在 Linux 系统上,主要利用 Libpcap 分析数据包技术对不同数据库进行审计. 此模块采用C ++ 的设计模式——工厂模式:只需根据配置文件的内容,就可以对相应数据库协议进行分析,该模式使得系统具有较强的可扩展性,有利于扩充后续各种数据库.

1.1.1 Libpcap 网络抓包及分析 网络安全维护中最常用的方法是对网络数据包的分析,这是以捕获网络数据包为前提的. 网络数据包的捕获速度和抓获能力直接影响到数据分析的效率和准确率[13-14].

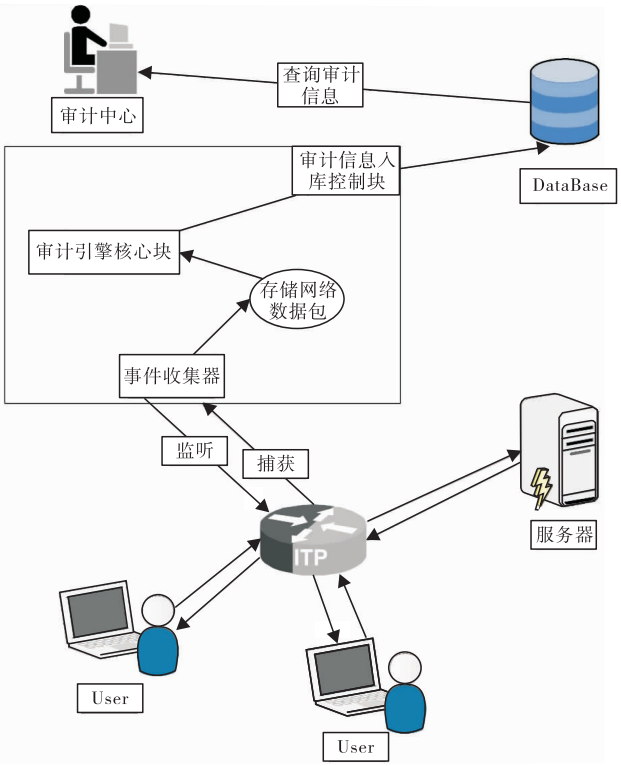

图 1 审计系统工作原理图

本系统主要利用 Libpcap 进行网络抓包和网络数据包的分析. 在审计引擎的事件收集器中,Libpcap 的主要工作是抓取指定服务器的流量数据,在事件收集器中通过配置文件设置过滤规则,从而获取更有效的数据流量. 在审计引擎核心块中,Libpcap 主要负责网络数据包的解析. 它可以将以太网的数据、有效的载荷数据解析出来,为下一步的数据信息解析奠定基础.

1.1.2 ProtocolBuffer 数据序列化 ProtocolBuffer 结构化序列数据的方法是由谷歌公司提出的,常用作通信协议的编写、数据保持方式等. 本系统采用 ProtocolBuffer 结构化序列数据编写通信协议,主要解决审计引擎核心块和审计信息入库控制块之间的数据传输问题. 其操作步骤如下:首先在核心模块序列化审计信息,然后通过 ZeroMQ 通信把完成序列化的数据发送到审计信息入库控制块,审计信息入库控制块接收到数据后,再将数据反序列化,最后根据格式将数据及时地保存到数据库中.

1.1.3 ZeroMQ 通信技术 ZeroMQ 有多种通信模式,常见的有请求—应答模式、发布—订阅模式、管

道共享模式. 根据需求分析,本系统采用了具有快速传递大量数据优势的发布—订阅模式. 系统设定审计引擎核心块为信息的发布端,审计信息入库控制块为订阅端,实时检查并接收发布的消息,而 ZeroMQ 通信主要工作在这两者之间.

1.2　事件收集器

系统中的事件收集器主要用来监听、捕获网络中对数据库数据操作的行为数据包,将抓到的数据包存到指定文件夹,供审计引擎核心块审计. 事件收集器同样工作在 Linux 机器上,随时待命,当有事件发生时及时捕获,没有网络流量时就“休息”,可以大大减轻系统的负载. 根据系统设计,每抓获 20 个Cap 数据包就保存起来,数据包的文件名以文件的创建时间命名.

1.3　审计信息入库控制块

审计信息入库控制块采用 MySQL ++ 数据库存储技术,避免了大量 SQL 句柄的书写,可以工作在任意一台机器上,及时接收审计引擎核心块发布的审计信息,并快速存储到数据库中.

1.4　审计中心

审计中心在 Windows 平台上工作,其任务是实现数据库审计信息系统的查询. 本模块在 JavaWeb 上开发,并设定了分角色登录的机制. 角色分为管理员、普通用户两种. 管理员用户拥有最高权限,主要负责用户信息和审计信息的管理,而普通用户只能进行相应的审计信息查询,比如可以根据事件发生的时间、操作者的 IP 地址等进行相关信息的查询.

2　审计引擎的设计

2.1　协议解析

事件收集器捕获的网络数据包主要由以太网头部、IP 首部、TCP 首部和有效的数据载荷组成. 审计引擎的功能是把需要的有效数据从网络数据包中解析出来,其过程主要涉及 TCP/IP 层协议的解析、应用层协议的解析. 在 TCP/IP 层的数据解析的基础上,还需获得应用层的有效数据,而这些数据是以 MySQL,TDS 协议结构为基础的.

2.1.1　MySQL 协议的解析　1)报文结构. 报文分为消息头和消息体两部分,其中,消息头占用固定的 4 个字节,消息体长度由消息头中的长度字段决定,报文结构如图 2 所示.

Packet length 表示报文数据的实际长度;Packet number 表示当前请求消息的编号,在一次完整的请求/响应交互过程中,用于保证消息顺序的正确性,每次客户端发起请求时,序号值都会从 0 开始计算. 消息体用于存放请求的内容及响应的数据.

2) Server Greeting Pocket 信息格式. Server Greeting Pocket 信息格式如图 3 所示.

Protocol 为服务协议版本号;Version 为服务版本信息;Server capabilities 为服务器权能标志,用于与客户端协商通信方式;Server status 为服务器状态.

3) Login Packet 信息格式. Login Packet 信息格式如图 4 所示.

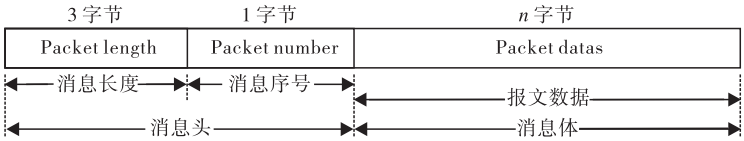

图 2　报文结构

1 字节	n 字节	4 字节	9 字节	2 字节	1 字节	2 字节	13 字节	13 字节
Protocol	Version(end with'0')	Thread id	Salt	Server capabilities	Chara-cter set	Server status	Unused	Salt

图 3　Server Greeting Pocket 信息格式

2 字节	2 字节	4 字节	1 字节	23 字节	n 字节	1 字节	20 字节	n 字节
Client capabilities	Extended Client capabilities	Max packet	Character set	Unknown	Username (end with '0')	SHA1	Password	Initialdatabase (end with'0')

图 4　Login Packet 信息格式

Client capabilities 为客户端权能标志,表示与客户端的协商通信方式;Max packet 为最大消息长度,表示客户端发送请求报文时所支持的最大消息长度值;Character set 为字符编码,表示通信过程中要用到的字符编码;Username 为用户名,表示客户端登陆用户的用户名称;Password 为用户登录的密码,一般是加密过的数据.

4)Commands Packet 信息格式. Commands Packet 信息格式如图 5 所示.

Command 表示请求的类型,Parameter 表示客户端发送的请求命令.

5)Error 信息格式. Error 信息是执行状态的一种,客户端协议的 Error 信息格式如图 6 所示.

255 是一个标志位,表示为 Error 信息;Errorno 是 MySQL 对应的错误号;"#"用于将错误号与错误信息分开;SQL_state 表示 SQL 执行的状态;Error msg 表示错误信息.

6)OK 信息格式. OK 信息是 SQL 执行的成功状态信息,客户端 OK 信息格式如图 7 所示.

0 是一个标志位,表示 OK 信息;Affected rows 表示 SQL 执行影响的记录行数;Server status 表示 SQL 执行后 Server 的状态值;Message 表示传输的信息. 7)EOF 信息格式. EOF 信息是用于表示传输数据结束的信息,客户端 EOF 信息格式如图 8 所示.

254 是一个标志位,表示 EOF 信息.

上述是 MySQL 协议的一些基本协议格式,接下来是根据协议格式进行数据解析. 通过各个报文的格式特点,对于 MySQL 数据库客户端与服务器之间传输的数据,可以解析出客户端的登录名,客户端对服务器做了哪些操作及服务器返回的一些状态和消息.

2.1.2 TDS 协议的解析 1)报头格式. TDS 报头格式如图 9 所示.

Token 表示 TDS 操作请求种类;Status 表示信息状态:值为 0 时表示还有后续报,值为 1 时表示此包为当前 TDS 会话中的最后一个包;Length 表示 TDS 数据包总长度,其中含 TDS 包头的长度;Signn 表示命名管道信息要用的通道数,此字段通常值为 0;Packetn 表示 TDS 包在当前 TDS 操作请求中的序号;Winnum 表示在确认信息收到前必须发送的框架数目.

2)登录报格式. 登录报格式如图 10 所示.

报文头 Token 值为 0x10,表示当前 TDS 报文为客户端登录报. 登录报内容主要描述报文头的报文数据长度、协议版本和一些标识信息位. 指示字段是以 4 字节为一组存储的数据,其中偏移值占 2 字节,长度占 2 字节. 分别存储客户端主机名称偏移和长度、登录的用户名称偏移和长度、登录的密码偏移和长度、客户端应用程序名称偏移和长度、服务器端主机名称偏移和长度、预留 4 字节、库名称偏移和长度、本地名称偏移和长度、数据库名称偏移和长度. 最后依次存储的是具体的客户端主机名称、登录的用户名称、登录密码、应用程序名称、服务器端主机名称、库名称.

3)SQL 请求报文格式. SQL 请求报文格式如图 11 所示.

客户端 SQL 语言命令包报文格式首先是 8 个

图 5 Commands Packet 信息格式

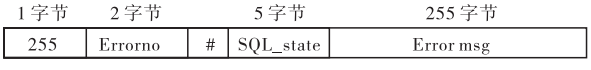

图 6 客户端协议 Error 信息格式

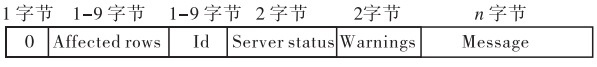

图 7 客户端 OK 信息格式

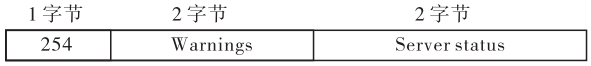

图 8 客户端 EOF 信息格式

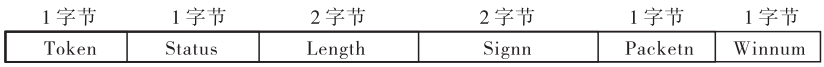

图 9 TDS 报头格式

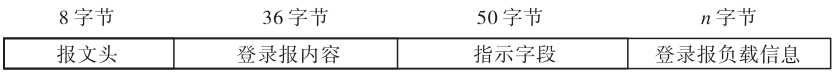

图 10 登录报格式

字节的报文头,紧接着是用 Unicode 编码的客户端 SQL 语言命令,最后是固定的 4 个字节的 SQL 语言结束符,其报文头 Token 值为 0x01,表示当前 TDS 报文为客户端 SQL 语言命令报.

4)响应报格式. 响应报格式如图 12 所示.

响应报的前 8 个字节依然是报文头,紧接着是响应的一些信息标志. 其响应报类型的不同主要根据报文头 Token 值来区别.

2.2　后台审计引擎的运行与配置

1)事件收集器的运行与配置. 图 13 显示的是事件收集器的工作状态,当前抓包的对象是对服务器的 3306 端口(MySQL 数据库的默认端口)进行监听并捕获有效数据. 监听的端口和抓包规则可由 filter. ini 配置文件设定.

2)审计引擎核心块的运行与配置. 图 14 显示的是审计引擎核心块的工作状态,当前是时间收集器收集的外界对 MySQL 数据库的一些操作数据包的解析,具体解析的是哪个数据库协议是由 dbport. ini这个配置文件设定的.

3)审计信息入库控制块的运行. 图 15 显示的是审计信息入库控制块的工作状态. 该模块进入工作前首先要连接本地的数据库:连接成功,会进入接收审计信息的等待状态;连接失败,会显示相应的错误.

3　系统测试

系统测试的主要目的是检测系统能否在大网络流量中正常工作,其关键是检测审计引擎的实时性和有效性.

审计引擎工作环境:64 位 CentOS 6.5 操作系统平台下,硬件配置为 CPU 2.6 GHz,RAM 512 M,网络带宽为 100 Mb/s.

测试环境:由三台 Linux 机、一台 Windows 机、审计引擎、审计中心、数据库服务器构成网络. 数据库服务器装配有 MySQL 和 SQL Server 2000 两种数据库,并将其与审计引擎安置到同一局域网中. 一台 Linux 机和一台 Windows 机负责对数据库服务器中的数据库进行操作,另外两台 Linux 机负责发送大量的包含数据库操作语句的网络数据包,从而模拟现实的网络环境.

测试的数据来源:系统的测试数据源一定要使用实际中外界用户对数据库服务器操作的数据流量. 本次测试的数据源是捕获到的某两家公司财务部对其数据库服务器(两家数据库服务器分别是 MySQL,SQL Server 2000)的操作流量数据包. 具体做法是将两台装有 Wireshark 的 Linux 机安置在与两家财务部数据库服务器相同的局域网中,分别设置两个 Wireshark 的抓包过滤规则,并开启抓包工具,进行 24 h 监听,尽可能多地抓取带有数据库操作语句的流量数据包.

测试工具:Wireshark,Tcpreply.

测试步骤:1)安装并配置带有 Wireshark 抓包工具的 Linux 机;2)获取包含大量数据库操作语句的网络数据包;3)组建网络环境;4)两台 Linux 机运行 Tcpreply 向局域网中发送捕获来的网络数据包,两个操作员分别操作一台 Linux 机和一台 Windows 机,对一台自己配置的数据库服务器进行操作;

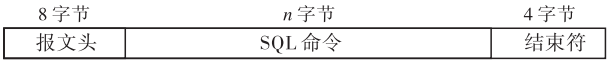

图 11　SQL 请求报文格式

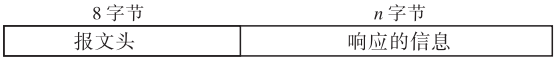

图 12　响应报格式

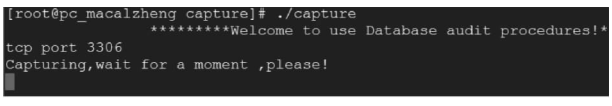

图 13　收集器工作状态

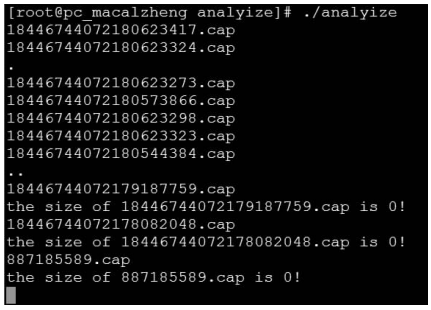

图 14　审计引擎核心块的工作状态

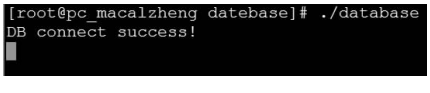

图 15　审计信息入库块的工作状态

5)统计丢包率和误报率.

测试结果:在 100 Mb/s 网络环境下,对 MySQL 数据库服务器审计的丢包率低于 2%,对 SQL Server 2000 数据库服务器审计的丢包率低于 1%,而两者的误报率也均小于 0.5%. 这说明在100 Mb/s网络环境下,数据库审计引擎可以正常工作,且有效性较高. 在测试中可以检测到审计引擎能够 24 h 实时监听外界对数据库服务器的操作,这也证明审计引擎具有良好的实时性.

4 结语

本文设计了一种针对 MySQL,SQL Server 两种数据库的数据库审计系统,实现了审计引擎、审计信息的收集与存储、审计信息的查询三部分功能. 通过以太网协议解析和应用层协议解析环节,获取被审计数据库的操作时间、操作地点 IP、操作对象 DB、操作行为四要素. 本审计系统实现了对日常数据库操作行为的监控和分析,系统扩展性较强,设计实现着重于用户管理和审计信息的查询反馈,对于审计报警、审计报表功能将后续完成.

参考文献:

[1] 索炜. 浅谈 IT 技术在数据库审计领域的应用[N]. 中国审计报,2013 - 01 - 09(8).

[2] 钱正麟,高航,李曙强. 基于网络侦听的数据库审计方法[J]. 计算机系统应用,2014,23(4):97.

[3] 徐国智. 基于 Web 2.0 技术的数据库审计管理系统的设计与实现[D]. 北京:清华大学,2012.

[4] 余本德,陈永义. 对数据库审计的思考[J]. 华南金融电脑,2008(1):21.

[5] Bishop M. Computer Security:Art and Science [M]. New York:Addison Wesley,2002.

[6] 李亿红,徐韧,程祥圣. 基于 XML 和 Web Service 的数据库审计系统[J]. 计算机应用与软件,2010,27(1):198.

[7] 王渊,马骏. 一种基于入侵检测的数据库安全审计[J]. 计算机仿真,2007,24(2):33.

[8] 李晶媛,韩慧莲. 一个基于误用检测的数据库安全审计系统[J]. 计算机与数字工程,2009,37(10):116.

[9] 李丽萍,杨寅春,何守才,等. 数据库安全中审计的设计与实现[J]. 计算机科学,2005,32(S7):83.

[10] Liu P. Architectures for intrusion tolerant database systems[C]//Proceedings of 2002 18th Annual Conference on Computer Security Applications, Piscatway: IEEE, 2003:311.

[11] Stevens W R. TCP-IP 详解卷 1:协议[M]. 范建华,胥光辉,张涛,等译. 北京:机械工业出版社,2000.

[12] 冯玉东,冯明卿,余宁. ASP 常见安全隐患及防范措施[J]. 郑州轻工业学院学报:自然科学版,2005,20(3):102.

[13] 胡滨. 基于 Windows 平台的底层网络数据包捕获技术[J]. 计算机工程与设计,2005,26(11):3037.

[14] 刘斌,代素环. 基于 Libpcap 的数据包捕获机制的实现[J]. 农业网络信息,2008(9):62.