

文章编号: 1674-9057(2017)01-0203-05

doi:10.3969/j.issn.1674-9057.2017.01.031

数据库连接池上的终端用户追踪与审计

谢统义^{1,2}, 马岩¹, 黄保华¹

(1. 广西大学 计算机与电子信息学院, 南宁 530004; 2. 广西教育学院 图书馆, 南宁 530023)

摘要: 在广泛使用的多层结构数据库系统中, 应用系统独立维护终端用户信息, 致使终端用户身份不能传递到数据库管理系统, 从而带来诸多安全问题。为解决这一问题, 提出了数据库连接池上的终端用户追踪算法, 通过分析 Web 日志来识别终端用户身份, 根据 Web 请求和提交数据库执行的 SQL 语句间的对应关系建立 SQL 语句与终端用户的映射, 并以此为基础进行终端用户追踪。基于所提出算法设计和实现了数据库安全审计系统, 表明数据库连接池上的终端用户追踪算法可行有效。

关键词: 数据库; 连接池; 终端用户追踪; 安全审计

中图分类号: TP309.7

文献标志码: A

数据库应用系统是目前计算机应用系统最主要的形式之一, 因为绝大多数系统都需要数据库管理系统 DBMS(database management system)的支持来进行结构化数据的存储和处理。在各种数据库应用系统中, 多层结构的 B/S(Browser/Server)模式 Web 应用系统又是最主要的存在形式。近年来为适应云计算^[1]和大数据处理的要求出现了大量 NoSQL 数据库系统^[2], 但基于 SQL 的关系数据库系统目前仍然是最主流和最广泛使用的数据库管理系统。在多层结构数据库应用系统中, 为应对数据库连接建立过程中计算、通信和内存开销过大而使性能不佳的问题, 数据库连接池(database connection pool)^[3]技术被广泛应用。连接池技术的应用改善了数据库应用系统整体的性能, 但却带来了比较严重的安全问题, 比如 SQL 注入^[4]等。

目前针对数据库安全的研究较少^[5], 主要集中在同态加密算法与加密数据库查询^[6-8]、数据库审计^[9]、外包数据库与云数据库安全^[10-11]等方面, 数据库连接池安全性方面的研究更是非常少。本文先介绍数据库连接池技术的基本原理并分析连接池所带来的安全问题, 然后提出一种在连接池中对终端

用户进行追踪的方法, 最后设计基于数据库连接池的安全审计系统以验证算法的可行性和有效性。

1 数据库连接池的原理及安全问题

1.1 数据库连接池基本原理

终端用户登录到 Web 应用系统访问和操作数据库内容时, Web 应用系统会向数据库提出连接申请, 操作完毕后释放此连接。若用户频繁发起数据库相关操作, Web 应用程序则需频繁地向数据库管理系统申请连接, 那么数据库管理系统会忙于创建、释放连接等操作^[12], 严重占用系统资源, 影响数据库管理系统的运行效率。应用数据库连接池技术可解决上述问题。数据库连接池是介于应用系统与数据库之间的“缓冲池”, 它保存了一定数目的数据库连接。当应用系统需要使用数据库连接时, 直接向连接池提出申请使用连接, 而不需要向数据库申请建立连接。数据库连接池的原理如图 1 所示。

数据库连接池的工作内容是建立连接、管理连接和释放连接^[13]。它首先创建好连接池最小连接数 MinConn 定义的数据库连接, 并存放在一个

收稿日期: 2016-03-08

基金项目: 国家自然科学基金项目(61262072); 广西中青年骨干教师基础能力提升项目(KY2016YB578); 南宁市科学研究与技术开发计划项目(20125258; 20131052)

作者简介: 谢统义(1976—), 男, 硕士, 讲师, 研究方向: 网络信息安全, 28922111@qq.com。

通讯作者: 黄保华, 博士, 副教授, bhhuang66@gxu.edu.cn。

引文格式: 谢统义, 马岩, 黄保华. 数据库连接池上的终端用户追踪与审计[J]. 桂林理工大学学报, 2017, 37(1): 203-207.

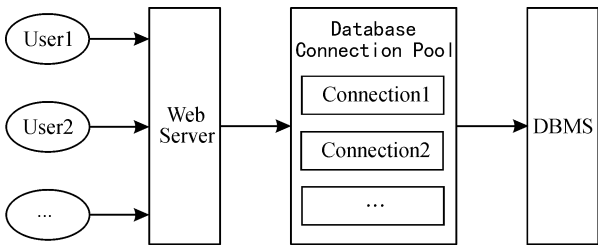


图 1 数据库连接池原理
Fig. 1 Database connection pool principle

叫“空闲池”的虚拟容器中。当应用系统申请使用连接时，连接池查看“空闲池”中是否有剩余的连接，若有剩余连接，则直接将此连接分配给用户；若没有剩余连接，则检查目前总连接数是否小于连接池所设定的最大连接数 MaxConn，若小于，则创建一个新的数据库连接放入连接池并分配给应用系统；否则将应用系统的连接请求放入“等待序列”中。等待是有时限的，在时限内，若有连接释放出来，则连接池将此连接分配给应用系统；若在等待时间限内无连接被释放，则连接池抛出异常并返回“申请失败”的信息给应用系统^[14]。在连接池工作过程中，被释放而未被立即使用的连接会被放入“空闲池”中以备后续分配使用。

1.2 数据库连接池带来的安全问题

- (1) 数据库管理系统无法识别终端用户信息。连接池收到的来自 Web 应用系统的参数中并不包含终端用户的信息，而只有用于连接到数据库管理系统的数据库账户信息及需要执行的数据库操作。这样数据库连接池就无法识别终端用户信息，数据库管理系统也无法获得终端用户的身份信息。
 - (2) 数据库管理系统无法实施访问控制。由于连接池对终端用户身份信息的隔离，使数据库管理系统针对终端用户的访问控制功能无法实现，因为访问控制的主体、客体和操作三要素中，数据库可识别的主体并不是实际的数据访问者。
 - (3) 在数据库上的安全审计难以进行。从审计的角度看，在使用数据库连接池的环境中由于数据库管理系统无法获得终端用户的身份信息，因此也无法记录其所执行的数据库操作所对应的实际请求者，也就没有了安全审计的可能性。
- 数据库连接池带来的上述问题可导致数据库管理系统的身份认证、访问控制和安全审计等基本安全功能失效。如果其上的 Web 应用系统安全

措施不全面，就可能导致非常严重的安全问题，比如 SQL 注入。攻击者通过 SQL 注入不但可以窃取数据库内容，而且可以执行扩展命令控制数据库管理系统所在的整个计算机。

2 终端用户追踪

数据库连接池安全问题的根源在于其并不接收和检查终端用户的身份信息，只要是应用系统的连接申请均予以回应并分配连接。要求应用系统向数据库连接池传递终端用户信息是一种可能的解决安全问题的方法，但在工程实践中不可行，因为这破坏了数据库连接池的标准性和透明性，需要对众多应用系统进行源代码级的大量修改。因此，在连接池上对终端用户身份进行追踪，即绑定终端用户及其所执行的数据库操作是解决数据库连接池安全问题的基础。

2.1 问题定义

Web 应用环境中数据库连接池输入侧有终端用户 $U = \{0, u_1, u_2, \dots, u_{nu}\}$ 发来的 Web 请求序列 $R = \langle r_1, r_2, \dots, r_{nr} \rangle$ ，输出侧有通过连接池提交给数据库管理系统执行的 SQL 语句序列 $S = \langle s_1, s_2, \dots, s_{ns} \rangle$ 。上述表达式中， $0 \in U$ ，指 0 用户，表示还没有完成身份认证的用户； nu 表示用户数量， nr 表示 Web 请求的数量， ns 表示 SQL 语句的数量。终端用户追踪 *trace* 定义如下：

$$trace(s) = \begin{cases} 0: s \text{ 是身份认证请求执行的语句,} \\ u: s \text{ 是终端用户 } u \text{ 发出的请求执行的语句.} \end{cases}$$

通过终端用户追踪 *trace*，任何提交给数据库管理系统执行的 SQL 语句 s ，都能够找到提交 Web 请求导致其执行的终端用户 u 。对正在执行身份认证的 Web 请求导致执行的 SQL 语句， $u = 0$ ，因为这时终端用户身份尚未确定。

从用户使用应用系统的角度看，当用户执行某个应用系统功能 f 时，触发一序列 Web 请求 $R_f = \langle rf_1, rf_2, \dots, rf_{nr_f} \rangle$ 并导致数据库管理系统执行一序列 SQL 语句 $S_f = \langle sf_1, sf_2, \dots, sf_{ns_f} \rangle$ 。 nr_f 是应用系统功能 f 触发的 Web 请求数量， ns_f 是功能 f 导致执行的 SQL 语句数量。设应用系统的功能集合 $F = \{0, f_1, f_2, \dots, f_{nf}\}$ ， $0 \in F$ 表示身份认证功能， nf 表示应用系统中除身份认证功能外的功能个数。

以功能为中介，只要找到功能 f 与 Web 请求序列 R_f 和提交数据库执行的 SQL 语句序列 S_f 的对应

关系，就可从应用系统日志和数据库日志中识别和追踪终端用户。这个对应关系的寻找可以通过学习过程来实现。

2.2 学习过程

学习过程就是串行顺序执行功能集 F 中的各个功能来获得功能与 Web 请求和 SQL 语句间的对应关系的过程。

Alg_1:学习算法

```
输入:无;输出:RSM
i = 0
For each f ∈ F
  Ri = ∅
  Si = ∅
  Execute f
  Ri = < Web requests >
  Si = SQL statements >
  Add( Ri, Sj )into RSM
  i = i + 1
End for
```

通过学习过程可以获得 Web 应用系统的请求-语句模型

$$RSM = \{ (R_i, S_i) | 0 \leq i \leq n \}$$

2.3 追踪过程

(1) 终端用户集 U 的生成。进行终端用户追踪，首先要有终端用户集合 U 。在 Web 应用系统中，终端用户及其身份信息是 Web 应用系统自己维护的，因此终端用户集合 U 可依据应用系统提供的用户信息手工构造或从应用系统中导出使用。

(2) Web 请求串行化。Web 服务器被多用户并发访问，各终端用户的 Web 请求被并行执行。每个终端用户对 Web 应用的访问会被 Web 服务器组织为一个会话，并可在 Web 日志中记录会话号 sid ，因此可依据会话号对 Web 请求进行串行化。

Alg_2:Web 请求串行化算法

```
输入:R;输出:RSS
RSS = ∅
For each r ∈ R
  Get sid from r
  Add r to RSsid
  If not RSsid ∈ RSS then
    Add RSsid into RSS
  End if
End for
```

(3) Web 会话身份识别。通过 Web 日志中的会话号和请求 (Get 或 Post) 参数，可确定会话对应的身份信息，即可根据 Web 应用的具体情况构

造身份识别函数 $identify(sid) = u$ 。

虽然 $identify$ 函数的构造是应用系统相关的，但其构造并不复杂，因为该函数只需要在功能 0 所执行的 Web 请求中读取一个请求参数即可。

(4) RSM 匹配。RSM 匹配实现用户追踪的核心功能——Web 请求序列与 SQL 语句序列间的映射，该映射通过 RSM 匹配算法实现。

Alg_3:RSM 匹配算法

```
输入:数据库日志记录的 SQL 语句序列 S, RSS
输出:被标记了终端用户的 S
For each RSsid ∈ RSS
  u = identify( sid )
  p = 1
  For each ( Ri, Si ) ∈ RSM
    ntmp = | Ri |
    If | RSsid | ≥ ntmp then
      Rtmp = left( RSsid, ntmp )
      If Rtmp = Ri then
        For each s ∈ Si
          Find sj = s( sj ∈ S, j ≥ p )
          Tag sj with u
          p = j
        End for
      RSsid = RSsid - Rtmp
    End if
  End if
End for
```

在 Alg_3 中， $left(RS_{sid}, ntmp)$ 表示取序列 RS_{sid} 的前 $ntmp$ 个元素构成新的序列； $| \cdot |$ 表示集合或序列中元素的个数。执行 Alg_3 后，数据库日志记录的 SQL 语句序列 S 中的语句被标记了相应的终端用户 u ，实现了终端用户追踪的功能。

3 审计系统设计与实现

本节设计数据库连接池上的安全审计系统 DCPAS (database connection pool audit system)，以追踪执行数据库操作的终端用户的身份信息，实现数据库操作的可审查性。

3.1 DCPAS 功能

数据库连接池上的安全审计系统 DCPAS 的功能主要包括：(1) 基于 Web 日志获取终端用户的 Web 请求。通过日志提取模块提取 Web 应用系统中的日志信息，然后交由日志处理模块进行处理以获得具体的 Web 访问请求；(2) 基于数据库日志获取 SQL 语句。通过数据库日志获取模块提取数据库执行的 SQL 语句形成 SQL 语句序列；(3)

SQL 语句对应的终端用户的标记。对数据库执行的 SQL 语句进行用户标记,使数据库操作和终端用户信息进行绑定,为数据库安全审计提供依据。

3.2 DCPAS 结构

根据 DCPAS 的功能需求,可设计出其总体结构。DCPAS 结构如图 2 所示。

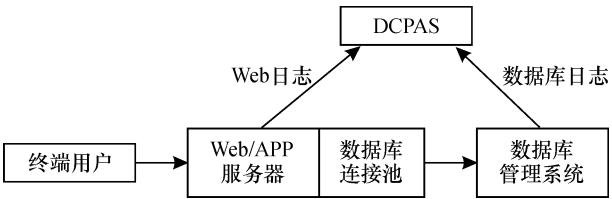


图 2 DCPAS 的结构
Fig. 2 Architecture of DCPAS

DCPAS 分为四大功能模块: Web 日志提取与处理模块 Weblogger、数据库日志提取与处理模块 DBLogger、学习模块 Leaner 和匹配模块 Matcher。Weblogger 根据 Web 服务器的配置情况读取 Web 服务器的日志信息,从中抽取 URL,以及 Get 和 Post 参数。DBLogger 按照数据库管理系统的配置读取数据库日志,并从中抽取 SQL 语句。Leaner 从 Weblogger 获取 Web 请求信息,从 DBLogger 获取 SQL 语句系列,执行 Alg_1 学习算法生成 RSM 模型并记录下来,供以后使用。Matcher 从 Weblogger 和 DBLogger 得到 Web 请求和 SQL 语句序列作为输入,将 Web 请求串行化后,依据 Leaner 学习得到的 RSM 模型对 SQL 语句序列中的语句进行终端用户标记,从而实现 SQL 语句的终端用户追踪。

3.3 DCPAS 的实现与测试

(1) 实现环境及关键实现细节

为检验终端用户追踪算法的有效性,按照本节的设计对 DCPAS 进行了实现。本实现基于 Windows Server 2012、Tomcat 1.8、Oracle Database 11g 和 Java SE Development Kit 8 进行。

DCPAS 系统中实现的 Weblogger 从 Tomcat 日志文件 localhost_access_log.[当前日期].txt 读取 Web 访问请求,并从请求中分离出请求参数。实现的 DBLogger 借助自己的 JDBC Driver^[15] 获取 SQL 语句,并调用该驱动中的 SQL 语句解析功能对语句进行分析以实现参数的泛化。

前文中已经提到,identify 函数的实现是应用系统相关的,因此本次系统的实现针对一个简单的图

书借阅系统。通过该系统 Tomcat 服务器日志发现,终端用户通过发送形如 http://192.168.0.100/login.jspu = user1&p = 123456 的 Web 请求来进行身份认证,因此 identify(sid)函数可直接读取 sid 对应的会话中 login.jsp 请求提交的参数 u 的值。

(2) 测试环境及测试结果

测试环境包括服务器 1 台和客户机 3 台。服务器硬件配置为 Intel i7 CPU/16 GB, RAM/1 TB HD, 运行 Windows Server 2012/Tomcat 1.8/Oracle Database 11g/Java SE Development Kit 8;客户机硬件配置为 Intel i5 CPU/4 GB, RAM/500 GB HD, 运行 Windows 8.1/Internet Explorer 10。

测试的学习阶段使用一个客户端以 user1 身份登录串行执行各功能。学习完成后,在 3 台客户机上同时运行自动化测试工具 WAPT 以模拟 3 个用户 user1、user2 和 user3 并发访问 Web 的应用情况,生成 Web 日志和 SQL 语句。从 DCPAS 的实现情况和表 1 中的测试结果可以看出,本文提出的终端用户追踪算法是可实现的,在多用户并发访问 Web 服务器的情况下也是有效的。

表 1 测试结果
Table 1 Test result

序号	名称	Web 请求数	SQL 语句数	结果
1	学习登录功能	6	2	每个测试 RSM 中都增加一个记录,包括相应的 Web 请求序列和 SQL 语句序列
2	学习用户信息修改功能	2	1	
3	学习图书借出功能	2	3	
4	学习图书归还功能	2	3	
5	使用 WAPT 进行所有功能并行访问测试	3 018	3 506	3 506 条 SQL 语句都成功标记上终端用户,标记用户 0、user1、user2、user3 的语句条数分别为 6、1 600、1 200、700

4 结 论

目前有关数据库安全方面的研究主要集中在新兴数据库形式,如外包数据库、云数据库、No-SQL 数据库的安全上,而广泛使用的数据库系统存在的安全问题,近年来很少有进行深入研究的相关工作。本文针对 Web 应用系统中数据库连接池阻断终端用户身份向数据库管理系统传递这一安

全问题,提出了数据库连接池上的终端用户追踪算法。该算法通过学习来建立 Web 请求和应用系统提交数据库执行的 SQL 语句间的映射关系以实现 SQL 语句到终端用户的追踪。基于该算法设计实现的数据库安全审计系统能够有效标记提交数据库执行的 SQL 语句对应的终端用户,实现了面向终端用户的数据库访问的可审查性,对保证数据库系统的安全性有重要意义。

本文提出的终端用户追踪算法目前只能在应用系统的功能被完整执行后进行,不能适应实时追踪审计的要求,具有一定局限性。在下一步工作中将改进和扩展该算法以适应实时追踪审计的要求。

参考文献:

- [1] 孙赵旭,谢晓兰,周国清,等. 基于Hadoop的Apriori算法与实现[J]. 桂林理工大学学报, 2014, 34(3): 584-588.
- [2] 张滨,陈吉荣,乐嘉锦. 大数据管理技术研究综述[J]. 计算机应用与软件, 2014, 31(11): 1-5.
- [3] 孙朝云,张羽. 基于B/S结构网上评教系统设计与实现[J]. 计算机应用与软件, 2012, 29(3): 183-186.
- [4] Huang B H, Xie T Y, Ma Y. Anti SQL injection with statements sequence digest [C] //The Spring World Congress on Engineering and Technology 2012 (SCET2012). New York: IEEE eXpress Conference Publishing, 2012: 563-566.
- [5] 吴开均,汤殿华,曾兵. 基于数据库外层加密的洋葱式加密模型探讨[J]. 通信技术, 2013, 46(12): 70-73.
- [6] Tople S, Shinde S, Chen Z F, et al. AUTOCRYPT: Enabling homomorphic computation on servers to protect sensitive web content [C] //The 2013 ACM SIGSAC Conference on Computer and Communications Security (CCS2013). New York: Association for Computing Machinery, 2013: 1297-1309.
- [7] Arasu A, Eguro K, Kaushik R, et al. Querying encrypted data [C] //The 2014 ACM SIGMOD/PODS Conference. New York: Association for Computing Machinery, 2014: 1259-1261.
- [8] Popa R A, Redfield C M S, Zeldovich N, et al. CryptDB: protecting confidentiality with encrypted query processing [C] //The 23rd ACM Symposium on Operating Systems Principles. New York: Association for Computing Machinery, 2011: 85-100.
- [9] Lu W T, Miklau G, Immerman N. Auditing a database under retention policies[J]. VLDB Journal, 2013, 22(2): 203-228.
- [10] Arasu A, Blanas S, Eguro K, et al. Secure database-as-a-service with cipherbase [C] //The ACM SIGMOD International Conference on Management of Data. New York: Association for Computing Machinery, 2013: 1033-1036.
- [11] Li T, Ye X J, Wang J M. Protecting data confidentiality in cloud systems [C] //The 4th Asia-Pacific Symposium on Internetwork. New York: Association for Computing Machinery, 2012: 1-12.
- [12] 王秀义. 基于JDBC的数据库连接池及实现[J]. 计算机系统应用, 2005(4): 36-39.
- [13] 商杰,朱成立. 数据库连接池技术研究与应用[J]. 现代电子技术, 2007(5): 95-97.
- [14] 鲍宏图,刘实,张德修,等. JSP软件开发中数据库连接池技术的探讨与应用[J]. 内蒙古大学学报(自然科学版), 2007, 38(3): 337-341.
- [15] Huang B H, Wang T J, Ma Y, et al. Schema of enhancing user authentication and encrypting Lob data in JDBC driver for database [C] //International Conference on Computer Science and Network Security. Pennsylvania: DEStech Publications, Inc., 2014: 374-378.

Terminal user tracing and audit in database connection pool

XIE Tong-yi^{1,2}, Ma Yan¹, HUANG Bao-hua¹

(1. School of Computer and Electronic Information, Guangxi University, Nanning 530004, China; 2. Library, Guangxi College of Education, Nanning 530023, China)

Abstract: In the widely used multi-layer database system, application system can maintain terminal users information by itself. This prevents user transporting identity from terminal user to database management system, and brings a lot of security problems of database. In order to solve this problem, the algorithm of tracing terminal user in database connection pool is proposed. This algorithm identifies the identity of terminal user by analyzing Web log of application server. Through matching the relationship between Web request and SQL statements submitting to execute in database, the algorithm can map SQL statements to terminal user. The audit system designing based on the algorithm and its implementation shows that the terminal user tracing algorithm in database connection pool is feasible and valid.

Key words: database; connection pool; terminal user tracing; security audit