第 34 卷 第 5 期 计 算 机 工 程 2008 年 3 月
Vol.34 No.5 Computer Engineering March 2008
·安全技术· 文章编号：1000—3428(2008)05—0134—02 文献标识码：A 中图分类号：TP309

基于 TDS 协议的安全性测试技术

余 静，鲁云萍
(江南计算技术研究所，无锡 214083)

摘 要：在分析 SQL Server 数据库通信框架和 TDS 协议结构的基础上，编写了 Fuzzer 工具——TDS_fuzzer。该测试工具针对 TDS 协议设计特殊数据包，实现了数据转变、字符串、字段组合这 3 种测试方法。通过测试 MS SQL Server 的 2 个重要漏洞，验证了其有效性。
关键词：表格格式数据流协议；MS SQL Server 数据库；基于块的协议分析；模糊化处理

Test Technique of Security Based on TDS Protocol

YU Jing, LU Yun-ping
(Jiangnan Institute of Computing Technology, Wuxi 214083)

【Abstract】 This paper introduces TDS protocol and communication between SQL Server database systems. Fuzzer for TDS protocol is done. Special packets are designed and three test methods of data mutation, string and field combination are implemented. The validity of the tool is proved by testing two known vulnerabilities of MS SQL Server.
【Key words】 TDS protocol; MS SQL Server; block-based protocol analysis; fuzzing

1 概述

表格格式数据流协议(TDS 协议)是微软公司的 SQL Server 数据库服务端与客户端所遵循的通信协议。根据 TDS 协议构造出的各种畸形数据包可能造成服务器遭受拒绝服务、缓冲区溢出等攻击。因此，对 TDS 协议进行分析、测试，了解各种漏洞的成因，能及时发现数据库软件中可能存在的问题，从而更好地保护 SQL Server 数据库系统的安全。

模糊化处理(fuzzing)是安全测试人员查找协议处理程序和应用程序中存在缺陷的一种常用技术。fuzzing的基本思想是编写一个测试处理程序，该程序能实现协议的大部分功能，提供正确格式的访问请求，向协议的各种字段输入大量随机数。Dave Aitel在 2002 年提出了一种基于块的协议分析方法[1]，利用网络协议中的已知元素，排除未知元素的影响，智能缩减程序的潜在空间，能有效地发现协议相关应用程序的安全漏洞，及时弥补测试目标设计或实现上的不足。

2 TDS 协议

2.1 TDS 协议通信构架

TDS是SQL Server服务端与客户端之间通信的应用程序级协议[2]。它建立在网络传输层协议(TCP)之上，定义了传输信息的类型和顺序，负责全部数据的传输细节。在客户端，SQL查询语句封装成TDS数据包，由客户端Net-Library接收并生成网络协议数据包发送出去。在服务器上，与客户端相匹配的服务端Net-Library接收客户端发送的网络协议数据包，析取出TDS数据包之后，将TDS数据包中的SQL查询语句传递给关系数据库，完成对数据库的操作。SQL Server通信构架如图 1 所示。

SQL Server缺省情况下使用的网络协议是TCP/IP，传输层采用面向连接的TCP协议。因此，客户端和数据库服务端通过 3 次握手在传输层建立TCP连接后，SQL Server使用TDS协议进行通信，具体步骤如下[3]：

(1)客户端先向服务端发送一个预登录请求，通过数据端的响应获取 MS SQL Server 的一些设置值，比如 SQL Server 版本，是否支持加密等信息，这是客户端在构造 TDS 包的一个依据；

(2)客户端先向服务端发动一个包含认证、缓冲区容量等信息的登录请求，服务端返回确认登录的响应包，这样数据库和客户端的会话就建立起来了；

(3)客户端发送查询请求，等待数据库服务器的回应。服务器执行查询并将查询结果(对列的描述、数据、完成信息等)返回给客户端。

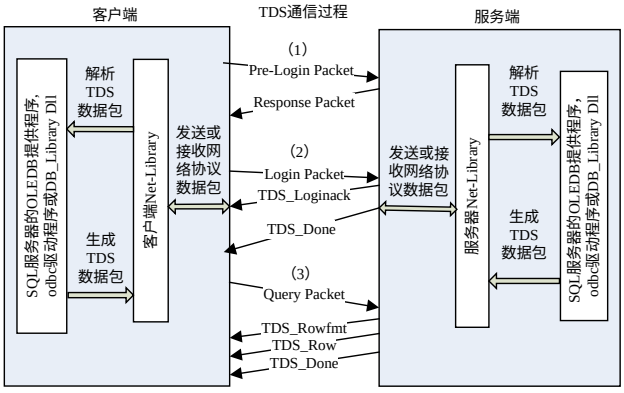

图 1 SQL Server 通信构架

2.2 TDS 协议结构

如果客户端的请求或是服务端的回应过长就会被分割成多个协议数据单元(PDU)，数据单元的大小通常在会话建立

基金项目：国家“863”计划基金资助项目(2003AA146010)
作者简介：余 静(1982－)，女，硕士，主研方向：网络安全；鲁云萍，工程师
收稿日期：2007-04-06 **E-mail：**jxyujing@gmail.com

时由双方协商好(TDS 数据包大小默认为 512 B)，每个 TDS 数据包都由一个 8 B 的头部和数据部分组成，其中，TOKEN 字段域 1 B，表示 TDS 操作请求种类；LENGTH 字段域 2 B，表示 TDS 包的总长度，包括 TDS 包头的长度。TDS 是一个以 TOKEN 为基础的协议，一个 TOKEN 用一个字节表示，表明接下来的数据和值的格式。TDS DATA 部分由 TDS 操作请求种类(TDS HEADER 中的 TOKEN)决定，一般说来，TDS 包大致分为查询、登录、响应、取消这 4 种操作。

3 TDS 协议的安全性测试分析

3.1 测试方法

为对基于TDS通信协议的SQL Server数据库进行黑箱测试，本文在Windows平台下采用基于块的协议分析(block-based protocol analysis)方法编写了Fuzzer工具TDS_fuzzer。TDS_fuzzer构建了包含块大小信息和字节队列的基于块的数据结构；构造了一些特殊的数字或字符串作为程序的输入，检查程序是否能够处理这些异常数据[4]。为提高TDS_fuzzer程序测试的效率，本文针对TDS协议作了以下设计：

(1)输入的测试数据中有一部分是由Transact-SQL语句中的“@”,“;”,“*”或 select, insert, delete 等常用符号、字符串组成的；

(2)提供了数据转变、字符串、字段组合等多种有效方法来确定测试过程是否出现异常[5]；

(3)TDS_fuzzer 测试尽可能覆盖整个 TDS 协议，从验证前到验证后的语句，包括预登录请求、登录请求、通过认证后服务端/客户端的查询和 RPC 消息；

(4)使用 OllyDbg, File Monitor, Process Explorer, Register Monitor 等多种调试工具，监视 SQL Server 服务器端的内存使用、网络活动、文件系统活动、注册文件访问的情况，关注异常，并对其进行分析产生测试结果报告。

测试框架如图 2 所示。

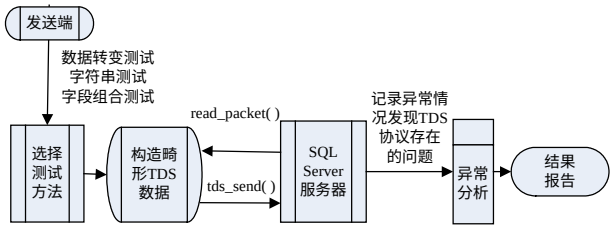

图 2 测试框架

3.2 测试实现

根据 TDS 协议的结构，本文实现了 3 种测试方法：数据转变，字符串，字段组合。其中，数据转变测试使用协议中未定义的数值去填充某些协议字段触发脆弱点；字符串测试采用超长的字符串去填充某些协议结果触发脆弱点；组合字段测试破坏协议字段的某些关联结构的关联关系触发脆弱点。本文结合这 3 种测试方法，对 MS SQL Server 的 2 个重要漏洞进行了分析，重现漏洞的发现过程，验证了测试工具的有效性。

3.2.1 字符串测试

字符串处理不当是引发软件漏洞的主要原因之一，因此，对协议中字符串部分来说 fuzzing 是非常有效的测试方法。对于测试的字符串字段，TDS_fuzzer 除了用一些超长的字符、空字符串去填充外，还使用了二进制数据、格式化字符串。

Microsoft SQL Server 在预验证时存在缓冲区溢出漏洞。在 PRELOGIN PACKET 包的字段指示头结构中存在一个 SINSTNAME 字段域，它填充的是客户端要求使用的实例名，协议并没有对字段域的长度作限制，对 SINSTNAME 字段域进行 fuzzing，用超长字符填充后的 TDS 数据包，会造成数据库服务端的缓冲区溢出。

本文用函数 my_tds_pre( )构造预登录请求包的头部 TDS HEADER，type 字节填入 0x12 表示这是一个预登录包。s_block_start("TDSData")标示块的开始，搜索标示为“TDSData”的块监听器，同时更新其内部“start”指针；函数 s_binary() 将二进制数据填充到结构中；s_binary_block_size_halfword_bigendian("TDSData")将 2 个空字节推入块数据的队列，并分配一个名为“TDSData”的块监听器，将监听器的内状态设为一个 halfword_bigendian 字。

```
int my_tds_pre( )
{
block_start("TDSData");
s_binary("12");/*type*/
s_binary("01"); /* status */
s_binary_block_size_halfword_bigendian("TDSData");/*Length */
s_binary("00 00");   /* signed num*/
s_binary("00");   /*packet num*/
   s_binary("00");    /*window*/
return 1;
}
```

预登录请求包的数据部分，除了 SINSTNAME 字段域，字段指示头和信息的其他字段均填写合法数据。函数 s_string_variable() 将 s_fuzzstring[] 的超长的字符串填充到 SINSTNAME 字段域，结尾处 s_block_end("TDSData")标示块的结束，同时确定监听器的大小。

3.2.2 数据转变测试

针对数据包中关键字段或功能标志字段，本文使用协议中未定义的同类型值或字符串去填充、改变。比如 TDS HEADER 中一个字节的 TOKEN 字段，决定了 TDS 数据包的类型：0x01 是请求包，0x02 是登录包……，在这种测试模式下，TDS_fuzzer 用 0x00-0xff 这 255 个字符去填充该字段，其中包括 0x00, 0xff 等协议中未作定义的数据值。

正常情况下，向 SQL Server Resolution 的监听端口 1434 发送一个值为 0x20 的单字节 UDP 包，SQL Server 会返回服务器的主机名、版本、网络库、服务器监听端口等敏感信息。也就是说数据库服务器接收到这个 UDP 包后，根据这一个字节的内容进行处理并作相应的回应。TDS_fuzzer 数据转变测试将 UDP 的这一字节作为 Fuzzing 的数据块，用 0x00-0xff 字符和以这些字符为首字节的较长字符串填充。向目标机的 1434 端口发送构造好的畸形数据包，通过 Ethereal 抓包工具分析以及目标服务器上调试工具跟踪调试，可以观察到：当服务器端接收到第 1 个字节设为 0x04 的 UDP 包时，SQL 监视线程会获取 UDP 包中的数据并使用它来尝试打开一个注册键，比如：发送 0x04+800 个字符 A 这样的 UDP 数据包，SQL 服务程序会打开注册表 HKLM\Software\Microsoft\Microsoft SQL Server\AAAA…AAAA\MSSQLServer\CurrentVersion，其中 800 个字符 A 超长的字符串，造成栈的缓存区溢出，并重写在栈上保存的返回地址。

3.2.3 组合字段测试

把 2 个相关字段中的一个字段作为测试的数据块，用各种字符串、随机值填充，造成字段信息不一致。TDS_fuzzer

（下转第 138 页）