链滴

通过 JDBC 访问强制 SSL 通讯配置的 SQLServer 数据库

作者：laolee

原文链接：https://ld246.com/article/1514435225855

来源网站：链滴

突然有一天，发现原来run得好好的数据库通讯程序不work了，查看错误日志是因为SQLServer服器端，开启了TCP客户端强制SSL认证。

说到SSL认证，先借用Oracle官方文档普及一下证书及SSL。

关于数字证书

数字证书（或简称证书）是在 Internet 上唯一地标识人员和资源的电子文件。证书使两个实体之间够进行安全、保密的通信。

证书有很多种类型，例如个人证书（由个人使用）和服务器证书（用于通过安全套接字层 [SSL] 技术服务器和客户机之间建立安全会话）。有关 SSL 的更多信息，请参见关于安全套接字层。

证书是基于**公钥加密**的，公钥加密使用数字**密钥**对（很长的数字）对信息进行**加密**或编码，从而使信只能被目标收件人读取。然后，收件人对信息进行**解密**（解码）即可读取该信息。

一个密钥对包含一个公钥和一个私钥。拥有者对公钥进行分发并使任何人都可以使用该公钥。但是拥者永远不会分发私钥；私钥始终是保密的。由于密钥与数学相关，因此使用了密钥对中的一个密钥进加密的数据只能通过密钥对中的另一个密钥进行解密。

证书就好像一本护照：它可以标识持有者并提供其他重要信息。证书由称为**证书授权机构** (Certificati n Authority, CA) 的可信赖第三方发布。CA 类似于护照申领办公室：它将验证证书持有者的身份并证书进行签名，以使他人无法伪造或篡改证书。CA 对证书进行签名之后，持有者可以提供该证书作身份证明并建立经过加密的保密通信。

最重要的是，证书会将拥有者的公钥绑定到拥有者的标识。与护照将照片绑定到其持有者的个人信息似，证书将公钥绑定到有关其拥有者的信息。

除了公钥以外，证书通常还包括以下信息：

- 持有者的姓名和其他标识，例如使用证书的 Web 服务器的 URL 或个人的电子邮件地址。
- 发布证书的 CA 的名称。
- 失效日期。

数字证书受 X.509 格式的技术规范约束。要检验 certificate 领域中某个用户的身份，验证服务将使用 X.509 证书的通用名称字段作为主体名称来检验 X.509 证书。

关于证书链

Web 浏览器已预先配置了一组浏览器自动信任的**根** CA 证书。来自其他证书授权机构的所有证书都须附带**证书链**，以检验这些证书的有效性。证书链是由一系列 CA 证书发出的证书序列，最终以根 CA 证书结束。

证书最初生成时是一个**自签名**证书。自签名证书是其签发者（签名者）与主题（其公钥由该证书进行证的实体）相同的证书。如果拥有者向 CA 发送证书签名请求 (CSR)，然后输入响应，自签名证书将证书链替换。链的底部是由 CA 发布的、用于验证主题的公钥的证书（回复）。链中的下一个证书是证 CA 的公钥的证书。通常，这是一个自签名证书（即，来自 CA、用于验证其自身的公钥的证书）且是链中的最后一个证书。

在其他情况下，CA 可能会返回一个证书链。在此情况下，链的底部证书是相同的（由 CA 签发的证，用于验证密钥条目的公钥），但是链中的第二个证书是由其他 CA 签发的证书，用于验证您向其发了 CSR 的 CA 的公钥。然后，链中的下一个证书是用于验证第二个 CA 的密钥的证书，依此类推，

至到达自签名的**根**证书。因此，链中的每个证书（第一个证书之后的证书）都需要验证链中前一个证的签名者的公钥。

关于安全套接字层

安全套接字层 (Secure Socket Layer, SSL) 是用来确保 Internet 通信和事务安全的最常见的标准。Web 应用程序使用 HTTPS（基于 SSL 的 HTTP），HTTPS 使用数字证书来确保在服务器和客户机之间行安全、保密的通信。在 SSL 连接中，客户机和服务器在发送数据之前都要对数据进行加密，然后由件人对其进行解密。

当 Web 浏览器（客户机）需要与某个安全站点建立连接时，则会发生 **SSL 握手**：

- 浏览器将通过网络发送请求安全会话的消息（通常请求以 https 而非 http 开头的 URL）。
- 服务器通过发送其证书（包括公钥）进行响应。
- 浏览器将检验服务器的证书是否有效，并检验该证书是否是由其证书位于浏览器的数据库中的（并是可信的）CA 所签发的。它还将检验 CA 证书是否已过期。
- 如果证书有效，浏览器将生成一个一次性的、唯一的 **会话****密钥**，并使用服务器的公钥对该会密钥进行加密。然后，浏览器将把加密的会话密钥发送给服务器，这样服务器和浏览器都有一份会话钥。
- 服务器可以使用其专用密钥对消息进行解密，然后恢复会话密钥。

握手之后，即表示客户机已检验了 Web 站点的身份，并且只有该客户机和 Web 服务器拥有会话密副本。从现在开始，客户机和服务器便可以使用该会话密钥对彼此间的所有通信进行加密。这样就确了客户机和服务器之间的通信的安全性。

SSL 标准的最新版本称为 TLS（传输层安全性）。Application Server 支持安全套接字层 (Secure Socket Layer, SSL) 3.0 和传输层安全性 (Transport Layer Security, TLS) 1.0 加密协议。

要使用 SSL，Application Server 必须拥有接受安全连接的每个外部接口或 IP 地址的证书。只有安了数字证书之后，大多数 Web 服务器的 HTTPS 服务才能够运行。使用使用 keytool 实用程序生成书中介绍的过程来设置您的 Web 服务器可用于 SSL 的数字证书。

程序如何修改

好了，说了那么多，java客户端程序该怎么改呢？主要有两个选择如下：

1. 选择验证服务器端证书有效性。具体操作如下：

- 将服务器端签发的证书导入JDK所在客户机的truststore信任存储库里；
- 程序中connection连接串参数增加如下内容：encrypt=true;trustServerCertificate=false;

encrypt=true代表启用SSL加密通讯，trustServerCertificate=false代表不信任服务器证书，需要客端验证证书有效性。

- 在执行命令中增加两个应用参数：

-Djavax.net.ssl.trustStore=D:\security\KeyStore.jks -Djavax.net.ssl.trustStorePassword=Password

具体文件路径及密码以实际为准。

2. 选择信任服务器端证书，客户端不做证书有效性验证。具体操作如下：

程序中connection连接串增加如下参数，encrypt=true;trustServerCertificate=true;

trustServerCertificate=true即代表客户端信任服务器端证书，不再另行验证。

测试样例代码

下面附上采用方式二最简单方式的测试样例程序。

```
package cn.com.jblog.dbutil;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.logging.Logger;
/**
 * @author lee
 *
 */
public class SQLServerSslDbUtil {
   static Logger log = Logger.getLogger("SQLServerSslDbUtil.class");
    public static void main(String[] args){
        log.info("get connection");
        String connectionString = "jdbc:sqlserver://hostname:1433;encrypt=true;trustServerCert
ficate=true;";
        Connection sourceConnection = null;
        Statement cmkStatement = null;
        ResultSet rs = null;
        try
        {
            log.info("Class.forName");
            Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
            log.info("DriverManager.getConnection");
            sourceConnection = DriverManager.getConnection(connectionString,"username","pa
sword");
            log.info("after getConnection");
            cmkStatement = sourceConnection.createStatement();
            log.info("after createStatement");
            rs = cmkStatement.executeQuery("select count(1) from DBName.dbo.TABLE_NAME");
            if(rs.next()){
                System.out.println("The row count is:"+rs.getString(1));
            }
        }catch (SQLException e1){
            // Handle any errors that may have occurred.
            //TODO
            e1.printStackTrace();
            log.severe(e1.getMessage());
        }catch (Exception e2){
            // Handle any errors that may have occurred.
            //TODO
            e2.printStackTrace();
            log.severe(e2.getMessage());
        }finally{
```

```
            try{
                if(rs!=null&&!rs.isClosed()){
                    rs.close();
                }
               if(cmkStatement!=null&&!cmkStatement.isClosed()){
                    cmkStatement.close();
                }
                if(sourceConnection!=null&&!sourceConnection.isClosed()){
                    sourceConnection.close();
                }
            }catch(Exception ex){
                //TODO
                ex.printStackTrace();
            }
        }
    }
}
```