MySQL 如何实现安全连接？

原创 姚嵩 爱可生开源社区 2025年04月10日 18:30 上海

作者：姚嵩，不知道是地球人还是外星人，知道的可以留言告诉小编...

爱可生开源社区出品，原创内容未经授权不得随意使用，转载请联系小编并注明来源。

本文约 2200 字，预计阅读需要 8 分钟。

1. 引言

在搭建某些环境的时候，常常会遇到需要输入 SSL/TLS 相关文件的情况，例如：**ssl-ca**、**ssl-cert**、**ssl-key** 等，在不了解其内在逻辑的情况下，会对这些参数比较陌生。

本文以 MySQL 为例，了解创建安全连接的方式，以点入面了解这些安全方面的知识。

2. 连接 MySQL 将面临的安全问题

- **两端被冒充**：客户端、服务端被冒充
- **密码泄露**：密码明文传输、暴力破解、碰撞攻击等
- **通讯信息泄露**：信息明文传输
- **通讯信息被篡改**

2.1 两端被冒充的解决方法

两端被冒充的解决方法是：签名。

验证 Client 端

Client 连接时提供证书和私钥，Server 端使用 CA 证书对 client 端提供的证书进行验证，确认 Client 端的真实性。Client 端对消息进行 hash 后，由私钥对 hash 值进行签名（加密），将消息和签名一起发送给 Server 端。Server 端使用公钥进行解签名（解密），并对消息进行 hash，将解签名得到的 hash 值和对随机串进行 hash 得到的 hash 值进行对比，以确认 Client 身份。

签名可以确认消息的完整性和验证发送方身份。

验证 Server 端

Server 端会向 Client 端发送其数字证书，这个证书包含了服务器的相关信息以及由证书颁发机构（CA）签发的签名，用于证明服务器的身份。

2.2 密码泄露的解决方法

mysql_native_password 密码插件（不推荐）

- 8.0.24 开始弃用，8.4 中默认禁用，9.0.0 中删除。
- 使用 SHA1 进行 2 轮哈希后，存入 `mysql.user` 表中。
- 插件支持质询-响应机制，Client 端不直接传送明文密码，而是和 Server 端提供的随机数进行计算，Server 端也进行相同的计算，Client 将计算后的结果发送给 Server 端，若 Client 端计算结果和 Server 端计算结果相同，则密码正确，非常快速且不需要加密连接。
- 解决了明文传输问题，但 SHA1 存在理论上的弱点，尤其是在碰撞攻击方面，SHA1 被认为不再安全。
- 因为哈希计算时未加盐的缘故，导致两个相同的密码进行 2 次哈希后得到的计算结果相同，进而存在密码泄密风险。

sha256_password 密码插件（不推荐）

- 使用 SHA256 对加盐的密码进行多次哈希，确保哈希转换更安全，将转换后的密码存入 `mysql.user` 表中。
- 即使密码相同，经过加盐转换后，存入 `mysql.user` 表中的结果也不同。

caching_sha2_password 密码插件（推荐）

- 与 sha256_password 类似，但对加盐密码进行了 5000 次 SHA256 转换，转换结果存储于 `mysql.user` 表中。
- 对加盐密码的哈希次数由参数 `caching_sha2_password_digest_rounds` 控制。
- 插件会缓存密码哈希，用于加速连接。Client 请求连接时：
  - 若 Server 端找到缓存的密码哈希，则直接对比 Client 端发送的密码哈希和 Server 端存储的密码哈希，若相同，则验证成功。

    - 若 Server 端未找到缓存的密码哈希，则 Client 端需要请求 Server 端的公钥（RSA 密钥对中的公钥），并使用公钥对密码进行加密并发送给 Server 端，Server 端收到加密后的密码后，使用私钥进行解密，将解密后的密码和盐进行 5000 次 SHA256 转换，并于存储在 `mysql.user` 表中的密码哈希进行对比，一致则密码正确，登录成功。

- 清理 Server 端缓存的密码哈希的方式：

    - 变更用户密码时，该用户缓存的密码哈希将被删除。可以通过 ALTER USER/SET PASSWORD/GRANT 变更密码。
    - 使用 RENAME USER 重命名用户时，将从内存中删除其密码哈希缓存条目。
    - 执行 FLUSH PRIVILEGES 时，将删除所有缓存的密码哈希值。

- 相关变量：

    - caching_sha2_password_private_key_path：只读变量，指定保存私钥的路径。
    - caching_sha2_password_public_key_path：只读变量，指定保存公钥的路径。
    - caching_sha2_password_auto_generate_rsa_keys：只读变量，是否自动生成 RSA 密钥对文件（RSA key-pair files）。
    - caching_sha2_password_digest_rounds：只读变量，caching_sha2_password 在密码生成过程中使用的哈希轮数。

- 相关状态：

    - caching_sha2_password_rsa_public_key：展示 caching_sha2_password 身份验证插件使用的 RSA 公钥的值。

- SSL/TLS 连接方式：

    - 当使用 SSL/TLS 时，连接是安全的，则 RSA 密钥对是不必要的，密码将以明文方式发送，但由于连接是安全的，因此无法窥探。

2.3 通讯信息泄漏的解决方法

通讯信息泄漏的解决方法是：SSL 安全连接。

SSL/TLS 为了生成会话密钥用于安全连接，客户端生成一个随机字符串作为预主密钥，并使用服务器的公钥对其进行加密，客户端将加密后的预主密钥发送给服务器。

服务器使用私钥解密接收到的信息，得到预主密钥。然后，客户端和服务器根据相同的算法和预主密钥独立计算出主密钥。双方根据之前交互的信息协商确定一个具体的加密套件(含对称加密算法、哈希算法等)，并使用主密钥衍生出的密钥进行后续通信。

一旦建立了安全连接，所有传输的数据都将使用协商好的对称加密算法和密钥进行加密和解密。

3. MySQL 操作

3.1 MySQL 防止密码泄露

我们只讨论 `caching_sha2_password` 密码插件生成的用户。

在非安全连接（非 SSL）时，使用 RSA 密钥对进行密码交换。

```
-- 确认账户不是强制使用 SSL
select user,host,plugin,ssl_type,ssl_cipher,x509_issuer,x509_subject
from mysql.user where user='mgr_user';
```

ssl_type 字段为空，表示用户不强制使用 SSL

当使用非 SSL 连接时，则需要使用 RSA 密钥进行密码交换，此时需要请求/指定 Server 端的公钥，使用公钥对密码进行加密。

非 SSL 连接

3.2 MySQL 创建安全连接

MySQL 5.7 开始，默认编译已经包含了对 SSL 的支持，我们查看下如何使用 SSL 创建安全连接。

Server 端是否支持 SSL？

```
-- 8.0.26 之前使用，YES 表示 server 提供 SSL 支持，DISABLED 表示不支持 SSL
show variables where variable_name in('have_openssl','have_ssl');
-- 8.0.26 及之后的版本，mysql_main 通道为 Yes 即可
select * from performance_schema.tls_channel_status where PROPERTY='Enabled';
```

查看数据库是否强制 SSL？

- `ssl_type` 字段值为 **空** 时表示不强制 SSL；
- `ssl_type` 字段值为 **SSL** 时表示强制 SSL；
- `ssl_type` 字段值为 **X509** 时表示强制 SSL，且要进行客户端身份验证；

```
select user,host,plugin,ssl_type,ssl_cipher,x509_issuer,x509_subject from mysql.user wher
```

使用安全连接

当 `\s` 得到的结果中 SSL 不是 `Not in use` 时，则表示 SSL 连接。

```
mysql -uys -pxxxx -h10.186.65.6 -P8038 --ssl-mode=REQUIRED
```

SSL 连接

3.3 MySQL 不支持 SSL 的场景

1. 配置文件中配置了 `skip_ssl` ，禁用了 SSL；
2. `ssl_ca` / `ssl_cert` / `ssl_key` 异常时（文件不存在、权限异常、内容不准确等），MySQL 开启支持 SSL 会失败（报错会打印到 MySQL 错误日志中）；

3.4 MySQL 校验客户端身份

需要校验客户端身份时，需要--ssl-cert 指定客户端证书，--ssl-key 指定客户端私钥:

```
mysql -uys -pxxxx -h10.186.65.6 -P8038 \
--ssl-cert=~/client_ssl/client-cert.pem \
--ssl-key=~/client_ssl/client-key.pem
```

3.5 强制用户使用身份验证

MySQL 设置强制用户使用 SSL 安全连接、X509 进行身份验证。

```
-- 强制用户使用 SSL 安全连接(会校验服务器身份)
alter user xxx require SSL ;

-- 强制用户使用 SSL 安全连接的同时校验客户端身份
alter user xxx require X509 ;

-- 不强制使用 SSL
alter user xxx require none ;
```

3.6 替换证书

如果证书过期，可以直接替换证书，并重载 TLS 相关配置即可：

```
alter instance reload tls ;
```

4. 总结

通过 **签名** 可以校验通讯双方的身份，以及防止信息被篡改。

通过 **RSA 密钥对** 或者 **SSL 安全链路** 的方式实现密码的安全传递。

SSL 还可以用于创建安全连接。

本文关键字：#MySQL# #安全连接# #身份验证# #数字签名#

故障分析 | MySQL 8.0 中多字段虚拟列引发的宕机
故障分析 | 如何解决由触发器导致 MySQL 内存溢出？
故障分析 | 查询 ps.data_locks 导致 MySQL hang 住
故障分析 | TCP 缓存超负荷导致的 MySQL 连接中断

✨ Github：https://github.com/actiontech/sqle

📚 文档：https://actiontech.github.io/sqle-docs/

💻 官网：https://opensource.actionsky.com/sqle/

👥 微信群：请添加小助手加入 ActionOpenSource

🔗 商业支持：https://www.actionsky.com/sqle

MySQL 231 安全连接 1 身份验证 1 数字签名 1

MySQL · 目录

上一篇
第 54 期：使用 JSON 格式的执行计划优化 SQL

下一篇
事务持续执行之谜：怎样找出对行记录上锁的 SQL？