MySQL数据库安全解决方案Linux认证考试 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/644/2021_2022_MySQL_E6_95_B0_E6_8D_c103_644887.htm 随着网络的普及，基于网络的应用也越来越多。网络数据库就是其中之一。通过一台或几台服务器可以为很多客户提供服务，这种方式给人们带来了很多方便，但也给不法分子造成了可乘之机。由于数据都是通过网络传输的，这就可以在传输的过程中被截获，或者通过非常手段进入数据库。由于以上原因，数据库安全就显得十分重要。因此，本文就以上问题讨论了MySQL数据库在网络安全方面的一些功能。 帐户安全 帐户是MySQL最简单的安全措施。每一帐户都由用户名、密码以及位置（一般由服务器名、IP或通配符）组成。如用户john从server1进行登录可能和john从server2登录的权限不同。 MySQL的用户结构是用户名/密码/位置。这其中并不包括数据库名。下面的两条命令为database1和database2设置了SELECT用户权限。 GRANT SELECT ON database1.* to abc@server1 IDENTIFIED BY password1. GRANT SELECT ON database2.* to abc@server1 IDENTIFIED BY password2. 第一条命令设置了用户abc在连接数据库database1时使用password1。第二条命令设置了用户abc在连接数据库database2时使用password2。因此，用户abc在连接数据库database1和database2的密码是不一样的。 上面的设置是非常有用的。如果你只想让用户对一个数据库进行有限的访问，而对其它数据库不能访问，这样可以对同一个用户设置不同的密码。如果不这样做，当用户发现这个用户名可以访问其它数据库时，那将会造成麻烦。 MySQL使用了很多

授权表来跟踪用户和这些用户的不同权限。这些表就是在mysql数据库中的MyISAM表。将这些安全信息保存在MySQL中是非常有意义的。因此，我们可以使用标准的SQL来设置不同的权限。 一般在MySQL数据库中可以使用3种不同类型的安全检查： 登录验证 也就是最常用的用户名和密码验证。一但你输入了正确的用户名和密码，这个验证就可通过。 授权 在登录成功后，就要求对这个用户设置它的具体权限。如是否可以删除数据库中的表等。 访问控制 这个安全类型更具体。它涉及到这个用户可以对数据表进行什么样的操作，如是否可以编辑数据库，是否可以查询数据等等。访问控制由一些特权组成，这些特权涉及到所何使用和操作MySQL中的数据。它们都是布尔型，即要么允许，要么不允许。下面是这些特权的列表： SELECT SELECT是设定用户是否可以使用SELECT来查询数据。如果用户没有这个特权，那么就只能执行一些简单的SELECT命令，如计算表达式（SELECT 1 2）,或是日期转换（SELECT Unix_TIMESTAMP(NOW( ))等。 INSERT UPDATE INDEX INDEX决定用户是否可以对表的索引进行设置。如果用户没有这个权限，那么将无法设置表中的索引。 ALTER CREATE GRANT 如果一个用户拥有这个GRANT权限，那么他就可以将自己的权限授给别的用户。也就是说，这个用户可以和其它用户共享自己的权限。 REFERENCES 有了REFERENCES权限，用户就可以将其它表的一个字段作为某一个表的外键约束。 除了以上的权限外，MySQL还有一些权限可以对整个MySQL进行操作。 Reload 这个权限可以使用户有权执行各种FLUSH命令，如FLUSH TABLES, FLUSH STATUS等。

Shutdown 这个权限允许用户关闭MySQL Process 通过这个权限，用户可以执行SHOW PROCESSLIST和KILL命令。这些命令可以查看MySQL的处理进程，可以通过这种方式查看SQL执行的细节。 File 这个权限决定用户是否可以执行LOAD DATA INFILE命令。给用户这个权限要慎重，因为有这个权限的用户可以将任意的文件装载到表中，这样对MySQL是十分危险的。 Super 这个权限允许用户终止任何查询（这些查询可能并不是这个用户执行的）。 以上几种权限是非常危险的，在给用户授权限时要非常谨慎。 MySQL中的SSL 以上的帐户安全只是以普通的Socket进行数据传输的，这样非常不安全。因此，MySQL在4.1版以后提供了对SSL（Secure Scokets Layer）的支持。MySQL使用的是免费的OpenSSL库。 由于MySQL的Linux版本一般都是随Linux本身一起发布，因此，它们默认时都不使用SSL进行传输数据。如果要打开SSL功能，需要对hava_openssl变量进行设置： MySQL的Windows版本已经将OpenSSL加入了。也面的命令是查看你的MySQL是否打开了SSL功能。 SHOW VARIABLES LIKE have_openssl. ---------- ------- | Variable_name | Value | ---------- ------- | have_openssl | NO | ---------- ------- 1 row in set (0.00 sec) 如果返回的是NO，那么说明你需要将OpenSSL编译进自己的MySQL 在有时你可能需要将用户名和密码进行加密传输。在这时可以使用下面GRANT命令： GRANT ALL PRIVILEGES ON ssl_only_db.* to abc@% IDENTIFIED BY "password!" REQUIRE SSL. 还可以通过 REQUIRE x509 选项进行SSL传输: GRANT ALL PRIVILEGES ON ssl_only_db.* to abc@% IDENTIFIED BY "password!" REQUIRE x509. 你还可以

使用REQUIRE SUBJECT来指定一个特定的客户端证书来访问数据库。 GRANT ALL PRIVILEGES ON ssl_only_db.* to abc@% IDENTIFIED BY "password!" REQUIRE SUBJECT "/C=US/ST=New York/L=Albany/O=Widgets Inc./CN=client-ray.example.com/emailAddress=raymond@example.com". 也许你并不关心使用的是什么客户许可，而仅仅关心的是你的证书。那么你可以使用REQUIRE ISSUER来实现： GRANT ALL PRIVILEGES ON ssl_only_db.* to abc@% IDENTIFIED BY "password!" REQUIRE ISSUER "/C=US/ST=New 20York/L=Albany/O=Widgets Inc./CN=cacert.example.com/emailAddress=admin@example.com". SSL还可以直接通过密码进行加密。可以使用REQUIRE CIPHER设置密码。 GRANT ALL PRIVILEGES ON ssl_only_db.* to abc@% IDENTIFIED BY "password!" REQUIRE CIPHER "EDH-RSA-DES-CBC3-SHA". 上面使用了GRANT命令对用户权限进行设置。而这些信息都是保存在授权表中，这些表是安全系统的心脏。在这些表中保存了每一个用户和客户机所具有的权限。如果正确地操作这些表，将会对数据库的安全起到积极的作用，而如果使用不慎，将是非常危险的。 下面让我们来看看MySQL中的最要的5个授权表。 user 用户表保存了用户的权限和被加密的密码。这个表负责确定哪些用户和客户机可以连接到服务器上。 host 这个表为每一个客户机分配权限，它并不考虑用户的权限。MySQL在确定是否接收还是拒绝一个连接时，首先考虑的是user表。而使用GRANT或REVOKE命令并不影响host表，我们可以通过手工方式修改这个表中的内容。 db db表保存了数据库层的权限信息。

tables_priv 这个表存储了表的权限信息。 columns_priv 这个表保存了单独列的权限信息。通过这个表，可以将操作某一列的权限授予一个用户。 哈希加密 如果数据库保存了敏感的数据，如银行卡密码，客户信息等，你可能想将这些数据以加密的形式保存在数据库中。这样即使有人进入了你的数据库，并看到了这些数据，也很难获得其中的真实信息。 在应用程序的大量信息中，也许你只想交很小的一部分进行加密，如用户的密码等。这些密码不应该以明文的形式保存，它们应该以加密的形式保存在数据库中。一般情况下，大多数系统，这其中包括MySQL本身都是使用哈希算法对敏感数据进行加密的。 哈希加密是单向加密，也就是说，被加密的字符串是无法得到原字符串的。这种方法使用很有限，一般只使用在密码验证或其它需要验证的地方。在比较时并不是将加密字符串进行解密，而是将输入的字符串也使用同样的方法进行加密，再和数据库中的加密字符串进行比较。这样即使知道了算法并得到了加密字符串，也无法还原最初的字符串。银行卡密码就是采用的这种方式进行加密。 MySQL提供了4个函数用于哈希加密：PASSWORD, ENCRYPT, SHA1和MD5。下面让我们试一试这4个函数，看看会得到什么结果。我们以加密字符串"pa55word"为例进行说明： 让我们先来看看MD5函数 SELECT MD5(pa55word).
---------- | MD5(pa55word) |
---------- |
a17a41337551d6542fd005e18b43afd4 |
---------- 1 row in set (0.13 sec) 下面是PASSWORD函数 SELECT PASSWORD(pa55word).

---------- | PASSWORD(pa55word) | ---------- | 1d35c6556b8cab45 | ---------- 1 row in set (0.00 sec) 下面是ENCRYPT函数 SELECT ENCRYPT(pa55word). ---------- | ENCRYPT(pa55word) | ---------- | up2Ecb0Hdj25A | ---------- 1 row in set (0.17 sec) 上面的每个函数都返回了一个加密后的字符串。为了区分加密字符串的大小写，最好在使用ENCRYPT生成加密字符串时，将这个字段定义成CHAR BINARY类型。 上面列举了3种加密的方法，但我认为使用MD5加密是最好的。这是因为这样做可以将明文密码显示在处理列表中或是查询日志中，这样便于跟踪。如下面的INSERT语句使用插入了一条记录，其中的密码使用了MD5进行加密： INSERT INTO table1 (user, pw) VALUE (user1, MD5(password1)) 可以通过如下的语句进行密码验证： SELECT * FROM table1 WHERE user = user1 AND pw = MD5(password1) 哈希加密方法可以很好地对密码进行加密，使用了这种方法加密，密码将无法 恢复成明文。 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com