MySQL Security

Domas Mituzas, Sun Microsystems

Me

- MySQL Support Security Coordinator (role)
- Did lots of security consulting and systems design work before
- Would prefer not to work on protection. Productivity is so much more fun!

What it is

- Safety of host system
- Safety of MySQL within host system
- Internal MySQL security capabilities

Host system security

- MySQL #1 in shared hosting environments (lots of long-term exposure to attackers)
- Has dangerous features allowing external file access, and possibly - code execution

User-defined functions

- Allows executing external code
- Checks for _init symbol to guard against malicious UDF specifications
- Some system libraries can be used to run arbitrary code this way
- Fix: 5.0.70, plugin_dir

Arbitrary paths

- DATA/INDEX DIRECTORY = /dev/shm
- Allows access outside of datadir
- Problem - can lead to DoS
- Fix - --skip-symbolic-links

FILE privilege

- LOAD DATA & INTO OUTFILE
- Allows access to non-DB files, allows creation of files too
- Especially dangerous with writable ~mysql
- Can be used to craft evil data files/frm/etc
- Fix: --secure-file-priv=/somewhere/outside

YaSSL

- The known major use (as a server) - just inside MySQL
- Security cautious might want to use OpenSSL for SSL needs (more audited)
- Does not cause much harm if disabled

LOAD DATA LOCAL

- Malicious servers can read data from client filesystems
- Every program, every API should have this disabled by default
- Overlooked by many distributions/ software/etc too many times
- No --enable-local-infile builds and MYSQL_OPT_LOCAL_INFILE,0 helps

External libraries

- DNS: --skip-name-resolve
- libc, zlib, openssl

Additional host security

- Better constraining of MySQL is helpful
  - SELinux (support-files/RHEL4-SElinux)
  - AppArmor
- Stack guarding compilers
  - -fstack-protector-all - Ubuntu, etc
- x86_64 NX

AppArmor

```
/usr/sbin/mysqld {
  #include <abstractions/base>
  #include <abstractions/nameservice>
  #include <abstractions/user-tmp>
  #include <abstractions/mysql>

  capability dac_override,
  capability setgid,
  capability setuid,

...
```

```
...

  /etc/mysql/** r,
  /usr/sbin/mysqld mr,
  /usr/share/mysql/** r,
  /var/lib/mysql/ r,
  /var/lib/mysql/** rwk,
  /var/log/mysql/ r,
  /var/log/mysql/* rw,
  /var/run/mysqld/mysqld.pid w,
  /var/run/mysqld/mysqld.sock w,
}
```

(Ubuntu, SuSE)

OOM

- Trivial to send MySQL out-of-memory
- Megabyte query on an empty table can consume gigabyte of RAM
- max_allowed_packet can help
- If system is supposed to do other work, ulimit'ing memory is good practice

Inside the system

- MySQL is as secure as a host it runs on
- All data files are portable
- ACLs can be edited with a simple editor (one can reset root password with ‘vi’)
- Debuggers have lots of power (symbols available, source open, ptrace(), kmem, ...)
- Plaintext data transfer (except for SSL)
- Hash+network snooping enough to log in

Blackbox

- Encrypted file systems alleviate data risks in case of hardware theft
- Stripping debugging symbols makes tracing much more complicated (not impossible for anyone with disassembler)
- --disable-grant-options stop the most easy ACL reset method
- OS allows stripping super-user capabilities

Entering the MySQL

- Users are identified by name+host pair
- Access from unauthorized hosts immediately rejected, before any handshake
- Wildcards can be used for subnets (no CIDR notation though), and subdomains
- Reverse DNS check does bidirectional lookup

Authentication

- 4.0 hash can be used to log in
- 4.1 password hashes can be used to login only in case of intercepted network traffic (challenge + response + storedhash = passcode)
- Possible to trap required hash with debugger/trace
- SSL solves all that, as long as crypto is safe

4.1 PASSWORD()

public_seed=create_random_string()

passphrase=sha1("password")

storedhash=sha1(passphrase)

reply=xor(passphrase, sha1(public_seed,storedhash)

passphrase=xor(reply, sha1(public_seed,storedhash))

sha1(passphrase)==storedhash \o/

Authorization

- ACLs are global, database, table/view, column and at stored-routine level
- They just add up, no exclusions are possible
- ROLEs are not there, 3rd party patches include such functionality (Google v3)
- ACLs stay in memory, so better to keep them lean

Grants

- Some grants are more problematic than others
- Some grants grew their power with features

SUPER saga

- It was fairly limited and safe grant once upon a time
- Allows bypassing max_connections (once)
- KILL, PURGE MASTER LOGS, SET GLOBAL, CHANGE MASTER, DEFINER, BINLOG, triggers, SET LOG-BIN, read_only

SUPER saga is long

- Active SUPER connection will block access to other SUPERs if max_conns run out
- KILLing SUPER-users is fun too! ;-)
- PURGE MASTER LOGS - destroying audit info (one can cripple the index to disable this)
- DEFINER specifications escalate privileges

Very long

- BINLOG command allows changing any data (BUG#31611), mysql.user too
- CHANGE MASTER can point to malicious binlog servers (firewalls help here)
- Triggers can be used to execute dirty work as other users
- Disable audit (binlog) for the session

SUPER must die

- It was not supposed to be ultimate super-user, just few SUPER-like rights
- It was much safer in 4.0 (and even 4.1)
- We're moving away some of actions from SUPER (monitoring used to ask for it)
- Needs reworking of grants system

FILE

- --secure-file-priv is a must
- Data leaks and code executions possible otherwise

PROCESS

- Allows seeing data in processlists
- Got InnoDB status moved over to it (from SUPER...)
- In case of system slowdowns (intentional or not) sensitive data can appear

RELOAD

- Allows FLUSH commands - resetting host error counts, reloading privileges, flushing table data to disk, etc
- Helpful access when attacking a system :)
- = SIGHUP in few cases

REPLICATION SLAVE

- Allows reading binary log information - all statement data, etc
- “SHOW BINLOG EVENTS” can be used by unsophisticated attackers

INSERT & UPDATE

- If given at global level (*.*), lead immediately to privilege escalation via mysql.* tables

TRIGGER

- Since 5.1, it can be used to set actions for other users
- Used to be part of SUPER in 5.0

EVENT

- Allows background execution of tasks
- Can be used for timing attacks, injecting bombs, etc

SHUTDOWN

- Can turn server off
- Most isolated/secure privilege out there

GRANT OPTION

- Allows giving same rights as executor's
- Needs 'CREATE USER' privilege to be able to create new users
- Needs access to mysql.* to reset password
- Grants can be revoked from anyone, including 'root', so there would be no way to set them back (except mysql.user edits)

Default users

- "@localhost, "@hostname - access to test
- root@localhost, root@hostname - superuser without password

```
DROP USER "@localhost;
DROP USER "@localhostname;
SET PASSWORD FOR root@localhost = PASSWORD('new password');
DROP USER root@localhostname; -- (or set password)
```

Resource control

- It is minimal
- Users can change session buffers
- Max can be specified:
  - --maximum-sort-buffer-size
- Apparently this isn't documented :(

Security features

- AES_ENCRYPT, DES_ENCRYPT, etc
- SHA1, MD5, etc
- SSL
- Views, triggers, procedures and functions

SSL

- Can request users to have a verifiable client certificate
- ... issued by specific CA
- ... with specific Subject Name
- ... and even ask password on top

SSL Example

- GRANT ALL PRIVILEGES ON test.* TO 'root'@'localhost'
- IDENTIFIED BY 'goodsecret'
- REQUIRE SUBJECT '/C=EE/ST=Some-State/L=Tallinn/
- O=MySQL demo client certificate/
- CN=Tonu Samuel/Email=tonu@example.com'
- AND ISSUER '/C=FI/ST=Some-State/L=Helsinki/
- O=MySQL Finland AB/CN=Tonu Samuel/Email=tonu@example.com'
- AND CIPHER 'EDH-RSA-DES-CBC3-SHA';

SSL will not

- by default check if server's certificate subject name matches server hostname
- can be set for regular clients, but replication setup does not have such option - would have to use server's public key as CA public key
- provide pure SSL port - so SSL will always have to be done by MySQL library

Views, functions, procedures

- Can be executed either in definer or executor security contexts
- Can allow horizontal and vertical table data security
- Can execute procedures on tables user does not have access to

Auditing

- Binlog (unsafe)
- General query log (possible to turn off in 5.1)
- SUPER audit (Google patch, v1)
- Triggers

SQL injections

- MySQL by default does not allow multiple statements (though, can be changed with connection flag) - good! Really!
- Prepared statements are widely used to guard against this
- Escaping rules are character set specific (can of worms)
- INFORMATION_SCHEMA is too revealing

Summary

- It is not that bad (I don't have much work at security team)
- Defaults could be better though
- Developers have great attitude to security issues (thanks Serg!)

Questions?

- domas at sun dot com
- http://dammit.lt/ & http://mysql.com/
- Report security vulnerabilities:
  - security at mysql dot com