Lab solutions for 0209

1

Formed this sql in mysql cli:

```
SELECT COUNT(*)
FROM user
```

Made it into this bash command: `echo 'SELECT COUNT(*)FROM user'| mysql -u root -pdynamitt bookfacedb`

Made it into a bash function that takes a table as argument

2

Script is found in this directory with the name "task2.sh"

3

Script is found in this directory with the name "task3.sh"

Couldn't get rid of the ls error message. (Tried sending the output to `/dev/null`)

4

A script version is found in this directory with the name "task4.sh". Could easily be turned into an alias by either copying it into .bashrc or making the alias refer to the script.

5

Followed the instructions. `openstack server list` works.

6

Did that.

7

Did that. Apparently adding a floating IP that is bound to another server moves it from one server t the other.

Yes, it can be useful when updating the entrypoint to the system. (Creating a new balanacer and moving old floating ip to new one)

8

Script is found in this directory "task8_in.sh" and "task8_out.sh". They move the floating IP users use to access our website between the balancer and a placeholder server.

9

wc prints newline, bytecount and wordcount for each file

10

- The first command shows partitions and disk usage
- The second command shows directories sorted after disk usage (ascending)

11

- `mkdir /home/ubuntu/supperask` creates a directory named superrask.
- `mount -t tmpfs -o size=20m tmpfs /home/ubuntu/superrask` makes the directory a sticky directory. Meaning that the directory can not be changed by users.
- `chown ubuntu:ubuntu /home/ubuntu/superrask` lets the user ubuntu have permissions to change the sticky directory.
- `umount /home/ubuntu/superrask` reverts the superrask directory back to a normal directory.

Lab solutions 02-16

1

Made bashrc function that spins up new instance with given name. Used this function to spin up new instance with name "backup".

Generated key in "db1" with `ssh-keygen`. Manually copied generated public key from db1 to authorized keys in backup.

Check that it works -> it does!

2

> Useful commands from Kyrre: `sfdisk -l`, `date`, `mkfs`, formattere disk: `<filsystem> /dev/vdb`

- Made volume "backupVolume" with size 5G and attached it to backup instance (device `/dev/backups`).
- formatted backup to mkfs.ext4 vdb with `mkfs.ext4 /dev/mkfs.ext4`
- Made backups directory and mounted with `mount /dev/vdb /backups`

TODO: Mount on startup

3

Stuff needed for a "full" backup of mysql database:

mysqldump (`mysqldump --opt --master-data=2 --flush-logs --all-databases > backup.sql`)

```
/var/log/syslog
/var/log/mysql/*
/etc/logrotate.d/mysql-server
```

4

Script can be found in this directory with the name "db1_fullBackup.sh"

5

1. What files are being rotated: `/var/log/mysql/mysql.log /var/log/mysql/mysql-slow.log /var/log/mysql/mariadb-slow.log /var/log/mysql/error.log`
2. When are the files rotated: Once a week. Daily backup is taken, and 7 archieved logs are kept.
3. How many rotated files we have: 7

6

TBA

7

As per right now we can't because we're missing keys going that way. If we added the key, we would also have to add a database user with the right privileges.

8

Taking backup directly from backup instance

Advantages:

- With multiple database-servers, the script will only live on one server
- The backup server decided when it wants to take backup

Disadvantage:

- Encryption not for free
- Database might not be ready for backup

9

The command for starting a new binary log file is `mysqladmin flush-logs`

10

From: https://dev.mysql.com/doc/refman/5.7/en/flush.html

If binary logging is enabled, the sequence number of the binary log file is incremented by one relative to the previous file. If relay logging is enabled, the sequence number of the relay log file is incremented by one relative to the previous file.

11

Do not stop at the end of requested binary log from a MySQL server, but rather continue printing to end of last binary log

12

```
mysqlbinlog --start-datetime="2018-02-14 09:00:00"--stop-datetime="2018-02-15
 23:59:59"
```

13

The way we interpreted the task was that we were supposed to show that we could figure out the exact commands that were run at a position in the binlogs. We chose position 32547436. The commands that were run were:

```
SET @@session.foreign_key_checks=1, @@session.unique_checks=1/*!*/;
SET @@session.sql_mode=1411383296/*!*/;
/*!\C utf8 *//*!*/;
SET @@session.character_set_client=33,@@session.collation_connection
    =33,@@session.collation_server=8/*!*
/;
GRANT SELECT, UPDATE, INSERT, CREATE, DROP, DELETE ON bookfacedb.* TO '
    harry'@'10.10.%' IDENTIFIED BY 'k
aboom'
/*!*/;
```

(The whole binlog entry:)

```
# at 32547436
#180316 12:32:58 server id 1  end_log_pos 32547625 CRC32 0xaf7a3b10
        Query    thread_id=16     exec_tim
```

```
e=0     error_code=0
SET TIMESTAMP=1521203578/*!*/;
SET @@session.foreign_key_checks=1, @@session.unique_checks=1/*!*/;
SET @@session.sql_mode=1411383296/*!*/;
/*!\C utf8 *//*!*/;
SET @@session.character_set_client=33,@@session.collation_connection
    =33,@@session.collation_server=8/*!*
/;
GRANT SELECT, UPDATE, INSERT, CREATE, DROP, DELETE ON bookfacedb.* TO '
    harry'@'10.10.%' IDENTIFIED BY 'k
aboom'
/*!*/;
```

14

Backup Policy:

- A full backup is taken every monday midnight.
- Incremental backups are taken every days at midnight, except monday midnights.
- If the bandwidth is too low at monday midnights, an incremental backup is taken instead of a full backup.

15

In case someone does `delete * from posts` by accident:

- Copy latest mysqldump to db1 from backups
- Execute the commands in mysqldump on the database

(Could we have used bin-log files here? Sure, but we're not confident enough with them to trust that approach)

In case we need to insert everything on a new server because the old one broke:

- Make new database server using our osnew bash function
- Follow instructions from appropriate lab to properly install MariaDB
- Copy latest mysqldump to new database instance from backups
- Execute the commands in mysqldump on the database

16

Concatenated them all to a file with the command: `sudo cat mariadb-bin.* | sudo tee megafile > /dev/null`

Tried opening the new megafile with `mysqlbinlog megafile`. Contains this:

```
/*!50530 SET @@SESSION.PSEUDO_SLAVE_MODE=1*/;
/*!40019 SET @@session.max_insert_delayed_threads=0*/;
/*!50003 SET @OLD_COMPLETION_TYPE=@@COMPLETION_TYPE,COMPLETION_TYPE=0*/;
DELIMITER /*!*/;
# at 4
#180316 11:31:18 server id 1  end_log_pos 256 CRC32 0x7f6e06c3  Start: binlog v 4, server v 10.2.13-MariaDB-10.2.13+maria~xenial-log created 180316 11:31:18 at startup
ROLLBACK/*!*/;
BINLOG '
BqurWg8BAAAA/AAAAAABAAAAAAQAMTAuMi4xMy1NYXJpYURCLTEwLjIuMTMrbWFyaWF+eGVuaWFs
LWxvZwAAAAAAAAAAAAAGq6taEzgNAAgAEgAEBAQEEgAA5AAEGggAAAAICAgCAAAACgoKAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAEEwQADQgICAoKCgHDBm5/
'/*!*/;
# at 256
#180316 11:31:18 server id 1  end_log_pos 285 CRC32 0x37b97081  Gtid list []
# at 285
#180316 11:31:18 server id 1  end_log_pos 330 CRC32 0xb69d1942  Binlog checkpoint mariadb-bin.000001
# at 330
#180316 11:33:57 server id 1  end_log_pos 353 CRC32 0xfc6abcdf  Stop
ERROR: Error in Log_event::read_log_event(): 'Event invalid', data_len: 1, event_type: 190
ERROR: Could not read entry at offset 353: Error in log format or read error.
DELIMITER ;
# End of log file
ROLLBACK /* added by mysqlbinlog */;
```

```
/*!50003 SET COMPLETION_TYPE=@OLD_COMPLETION_TYPE*/;
/*!50530 SET @@SESSION.PSEUDO_SLAVE_MODE=0*/;
```

Our conclusion is that this did not work. Why? We don't know, but we're guessing it has to do with the format of the bin-log files, as that's what the output is pointing towards.

Lab solutions for 02-23

1

Determined size of database

Determined size of database with the command `sudo du -h /var/lib/mysql`. Gave output:

```
220K	/var/lib/mysql/performance_schema
27M	/var/lib/mysql/bookfacedb
1.1M	/var/lib/mysql/mysql
200M	/var/lib/mysql
```

From this we determine that a memcache of size 256 is realistic.

Made new instance for memcache

Used our osnew function to make a new tiny instance called “cache”.

Installed memcache with the command `apt-get install memcached`. Edited `etc/memcached.conf` to listen to 0.0.0.0 and use 256 for caching. Restarted with `service memcached restart`.

2

Added memcache support to www3 with `apt-get install php-memcache libmemcached11 libmemcache-dev`. Restarted apache2.

Enabled memcache support in bookface by adding these three lines in `config.php`:

```
$memcache_enabled = 1;
$memcache_enabled_pictures = 1;
$memcache_server = "10.10.0.187";
```

3

Munin has not yet been covered, so we’ll wait with installing it.

4

Yes you can choose to not use memcache by setting a GET paramter

5

Stopped memcahced. The site still works fine.

6

Turned off image support by altering the config file. Tried measuring the speed with the magic command (from stackoverflow) `curl -Lo /dev/null -skw "\ntime_connect: %{time_connect}s\ntime_namelookup: %{time_namelookup}s\ntime_pretransfer: %{time_pretransfer}\ntime_starttransfer: %{time_starttransfer}s\ntime_redirect: %{time_redirect}s\ntime_total: %{time_total}s\n\n"10.212.136.167`

We saw no real difference..

7

With our current setup with would be more hassle than it’s worth. When we eventually transition to a Docker Swarm solution we’ll probably use a memcache container.

Lab solutions for 03-02

1

1

The upper 5% percentile tells us that there are rarely 27000 or more users.

2

We can see that in the beginning there were regularly around 13000 users, while at the end there were regularly 20000 users

3

We can see how likely / unlikely it is that we will have X amount of players. Buying hardware to support 40000 players might not be smart when it only happens 1% of the time.

2

The script is in this directory with the name `log_incomming_connections.sh`. It looks like this:

```bash
#!/bin/bash

# curl incomming connections
CURRCONN=$(curl -su someuser:password "http://10.212.136.82:1936/;csv"
    | grep FRONTEND | head -1 | awk -F',' '{print $5}')

# determine unixtime
UNIXTIME=$(date +%s)

# echo time,connections
echo "$UNIXTIME,$CURRCONN"
```

Lab solutions for 03-09

1

Made new Harbor account on `10.212.136.160`:

```
Username: TnT
Email: trondhth@stud.ntnu.no
First and last name: Tango November Tango
Password: dynamittH4rry
```

Made a new private project called `docker`

2

On both docker VM and manager VM:

```
curl http://10.212.136.140/harbor.crt > /usr/local/share/ca-
    certificates/harbor.crt
update-ca-certificates
service docker restart
```

Logged in to Harbor via shell with `docker login 10.212.136.160`

bookfaceimage can now be pulled down to manager VM.

Uploaded bookfaceimage to harbor with `docker tag bookfaceimage 10.212.136.160/docker/bookfaceimage:latest docker push 10.212.136.160/docker/bookfaceimage:latest` It works!

3

Installed docker on www1, www2, www3. On manager: Initiated docker master by typing `docker swarm init` Return token: `docker swarm join --token SWMTKN-1-5d0xxilhf03y8ds2zb1gdfrtj9duqm71-2096wfbcx0xm0ssjk34hi6941 10.10.0.70:2377` Added the token to all thre www servers. On manager: Checked if all nodes are connected by typing `docker node ls`. It works!

This process can be automated by for example create a script that stores the token in a file, and then uses `scp` and `ssh` to get it to run locally on the worker.

4

Started bookface webservers as a service with `docker service create -d ---replicas =3 --name=bookface --with-registry-auth -p 3000:80 10.212.136.160/docker/ bookfaceimage:latest` Checked status on all three replicas by typing `docker service ls`. Showing 3/3!

Went to balancer and haproxy.cfg and added `server dockerMaster 10.10.0.70:3000 check` Also changed www1, www2 and www3 to listen to port 3000.

The server that is the docker master is manager, so haproxy points to this. Also haproxy points to www1. www2 and www3 as these are the docker containers that runs the whole infrastructure.

If one of the swarm- servers are taken down, the users of bookface won't notice anything significantly, because there are other servers in the swarm that routes the traffic, functioning as a fail-over.

5

TBA