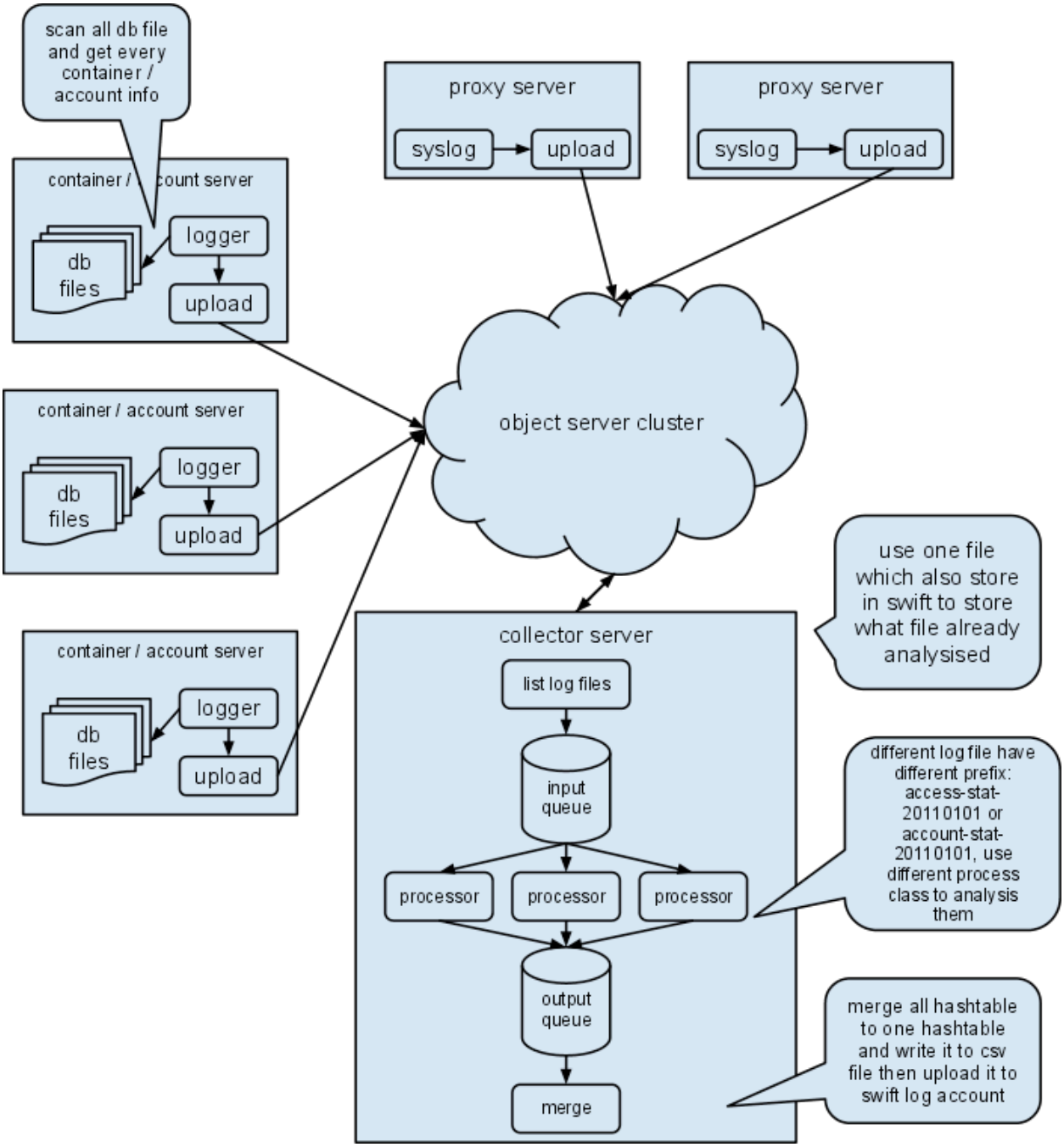
openstack swift slogging system

1. architecture:




2. process:

1. deploy access_loger in proxy server, set crontab run it every hour
2. deploy account_loger in account server, set crontab run it every hour
3. deploy container_loger in container_server, set crontab run it every hour
4. create a new account to store all the log file

5. deploy uploader in every server (except object server), set crontab run it every hour, it can upload all log file to log account in swift

6. deplog log_processor in another server, set crontab run it every hour, it can collect all log from log account in swift

7. log_processor will sum all of the disk space usage info (if we have 3 account server, there is 3 account log file in each server)

3. risk:

1. all calculate use memory, if we have a big system, it will become slow, because have to switch between swap.
2. single point in the whole system, can not dispath the job of process the logs to other server.
3. use a single file to store what files already be processed, and store it in swift , every hour must download it again, and when analysis finish refresh it and upload it again, if this system run a long time , this file will be very big:

self.processed_files_filename = 'processed_files.pickle.gz'

4. if one of the account server crash or network partition, the value of each account disk usage info will not correct, because the list of log file will not find one of the account log file , and merge code just sum all the value of each account.

4. refer:

http://notmyname.github.com/slogging-debian/1.1.2/overview_stats.html