Network Metrix

marcin@seremak.org
v. 2016-09-30

1. Libraries used in the project

- **psutil** - for getting operating system metrics
- **paramiko** - for SSH and SFTP
- **click** - command line
- **pytest** - unittest
- **pycrypto** - encrypt/decrypt client-server messsages
- **sqlite3** - relational database

2. Modules

Module	Description	Dependencies
config.py	Singleton configuration object available for all components	
database.py	Relational database object available on demand and as singleton	config.py
client.py	Script to be executed on remote host.	
node.py	Class that represents remote machine. Has `upload` and `execute` methods. Responsible for storing metrics in relational database and sending alerts. It uses secret key hardcoded in client.py for decrypting messages.	config.py, database.py, client.py
server.py	Entry point. Driven by command line, execute commands on remote nodes, on database and on config.	config.py, database.py, node.py, client.py

3. **Upload** command

1. Read configuration from config.xml. In particular:
    - list of remote `hosts`
    - `script` file name
    - `concurrency` parameter
2. Put `script` on each remote `host` using SFTP. Target directory is configured in config.xml for each host.
3. To speed up, hosts are processed in parallel using multiprocessing pool. Size of pool is configured via `concurrency` parameter in config.xml

4. **Execute** command

1. Read configuration from config.xml. In particular:
    a. list of remote `hosts`
    b. `script` file name
    c. `concurrency` parameter
2. Execute `script` on each remote `host` using SSH.
3. `Script` writes encrypted message with system statistics results to stdin. Secret key for (en)(de)cryption is hardcoded in `script`.

```
(crossover)[master]~/clustermetrix/Source$ python client.py
N2ceeDWRHGrY07tFn7c7MiLo91OaKvvsaEFzf51xspvqqfvYawsaSMgEJpEtM898iEHEU8Jpi18z2mDz0CEGmg==
```

4. Server reads stdout and decrypt the message.
5. Server inserts results to relational database. New database connection is opened for this operation to prevent database from being a bottleneck.
6. Server sends email alerts if results match the limits of configured alerts

```
(crossover)[master]~/clustermetrix/Source$ python server.py execute -h f4
2016-09-30 21:54:54,757 INFO [f4] SUCCESS! Script `/tmp/client.py` executed
2016-09-30 21:54:54,764 DEBUG [f4] Metric `uptime_sec`->`2128130` saved
2016-09-30 21:54:54,772 DEBUG [f4] Metric `mem_percent`->`4.9` saved
2016-09-30 21:54:54,794 DEBUG [f4] Metric `cpu_percent`->`38.5` saved
2016-09-30 21:54:54,813 INFO Mail `ALERT [f4] cpu_percent is 38.5>20` sent to `marcin@seremak.org`
```

7. To speed up, hosts are processed in parallel using multiprocessing pool. Size of pool is configured via `concurrency` parameter in config.xml