目录

一、环境需求

要求各机器之间可以 ping 通

✧ 采集机 A：
- Centos 6+
- Python 2.7+

✧ 物理机 B：
- Linux 各版本
- Libvirtd

✧ 测试机 C：
- Windows

二、程序安装

1. 在 A 机上安装 pip：

   安装 python2.7+后执行以下步骤来安装 pip

   ```
   # wget https://bootstrap.pypa.io/get-pip.py
   # python get-pip.py
   ```

2. 在 A 机上安装依赖包：

   ```
   # yum install python-libvirt
   # pip install gevent
   # pip install web.py
   # pip install mimerender
   # pip install mysql-python
   ```

3. 在 A 机上配置无密码登陆 B 物理机：

   1) 在 A 机下生成公钥/私钥对，按一次回车，它在~/.ssh 生成 id_rsa 和 id_rsa.pub

      ```
      # ssh-keygen -t rsa -P ''
      ```

   2) 把 A 机下的 id_rsa.pub 复制到 B 机的~/.ssh/目录下，重命名为 authorized_keys

      ```
      # scp ~/.ssh/id_rsa.pub root@B 机 IP:~/.ssh/authorized_keys
      ```

   3) 测试无密码登陆： ssh B 机 IP 如果不需要输入密码则配置成功

4. 建立项目数据库

   在 A 机数据库中运行 cloud_monitor.sql 文件

   ```
   # cd libvirt_monitor_server
   # mysql –u root –p
   >source sql/cloud_monitor.sql
   ```

5. 在 cloud_config 表中添加配置项

Key	value	默认值
interval_check	抓取数据间隔（秒）	200
interval_travelsal	连接物理机，获取虚拟机列表间隔（秒）	200
host	物理机 IP	无

其中 host 项可添加多条数据，每条数据一个 IP

6. 系统配置

在 code/cloud_monitor_setting.py 中填写各项配置，示例如下

```
# 数据库配置
db_engine = 'mysql'
db_server = '数据库 IP'
db_username = '用户名'
db_password ='密码'
db_database = 'cloud_monitor'
# API 端口配置
api_server_port = 9898
```

三、测试运行

1. 在采集机 A 上运行持续监控后台程序

```
# cd libvirt_monitor_server
# chmod 755 start-monitor.sh
# ./ start-monitor.sh
```

监控程序日志将输出到 log 目录下的 daemon.log 和 monitor.log 文件，运行后查看日志，若无错误输出说明运行成功。

2. 在采集机 A 上运行 API 程序

```
# chmod 755 start-api-server.sh
# ./ start-api-server.sh
```

API 程序日志将输出到 log 目录下得 api.log 文件，运行后查看日志，若无错误输出说明运行成功。

3. 页面展示

完成上述步骤后在测试机 C 上

1) 修改 html/index.html，第 29 行，修改为采集机 A 的 IP 及端口

```
var api_url = "http://173.26.100.211:9898/instances";
```

2) 修改 html/graph.html，第 80 行，修改为采集机 A 的 IP 及端口

```
var api_url = "http://173.26.100.211:9898/instances";
```

3) 在浏览器中打开 index.html，可以看到云平台所有虚拟机列表，点击每条最后一项 graph 链接，进入虚拟机信息图表页面，展示如下说明配置成功。

4) 在 graph 页面中可选择时间段展示虚拟机信息。

四、API 接口

以下 localhost 表示采集机 A 的 IP

1. 获取所有虚拟机列表

http://localhost/instances	
HTTP method	GET
返回值	id：虚拟机编号 uuid：虚拟机唯一标识 ip：虚拟机所在物理机 IP enable：是否抓取数据,1-是，0-否
返回值 json 示例	

2. 设置是否抓取指定虚拟机

http://localhost/enable/虚拟机 uuid	
HTTP method	POST
参数	enable：是否抓取数据 1-是，0-否
返回值	
返回值 json 示例	{'message':'success'}

3. 删除指定虚拟机

http://localhost/instances/虚拟机 uuid	
HTTP method	DELETE
参数	
返回值	
返回值 json 示例	{'message':'success'}

4. 按时间起始结束，获取指定虚拟机数据

http://localhost/ instances /虚拟机 uuid	
HTTP method	POST
参数	stime：起始时间 etime：结束时间
返回值	time：数据抓取时间（精确到分） uuid：虚拟机唯一标识 name：虚拟机名称 host：所属物理机 IP state：物理机状态 0 no state 1 the domain is running 2 the domain is blocked on resource 3 the domain is paused by user 4 the domain is being shut down 5 the domain is shut off 6 the domain is crashed 7 the domain is suspended by guest power management 8 NB: this enum value will increase over time as new events are added to the libvirt API. It reflects the last state supported by this version of the libvirt API. number_cpus：cpu 数量 cpu_usage：cpu 使用量（%） max_memory：最大内存（KB） memory_usage：内存使用量（KB） vir_interfaces：rx_packets：下载包数 tx_packets：上传包数 rx_bytes：下载数据大小（bytes） tx_bytes：上传数据大小（bytes） rx_errs：下载错误数 tx_errs：上传错误数 rx_drop：下载丢弃 tx_drop：上传丢弃 vir_disks：rd_req：读请求数 wr_req：写请求数 allocation：已分配 rd_bytes：读数据大小（bytes） wr_bytes：写数据大小（bytes） errs"：错误数 capacity：硬盘容量 physical：物理磁盘用量

返回值 json 示例	{
	"time": "2015-11-14 15:54",
	"uuid_string": "ba16b497-d49d-4a0b-a00b-246209076bfa",
	"name": "instance-00000014",
	"host": "173.26.100.211",
	"state": 1,
	"number_cpus": 1,
	"cpu_usage": "4.667",
	"max_memory": 524288,
	"memory_usage": 524288,
	"vir_interfaces": {
	"tap9d6c50ed-03": {
	"rx_packets": 0,
	"tx_packets": 0,
	"rx_bytes": 0,
	"tx_bytes": 0,
	"rx_errs": 0,
	"tx_errs": 0,
	"rx_drop": 0,
	"tx_drop": 0
	}
	},
	"vir_disks": {
	"vda": {
	"rd_req": 0,
	"wr_req": 1.331264531878126,
	"allocation": 4595712,
	"rd_bytes": 0,
	"wr_bytes": 2726.429761286402,
	"errs": 0,
	"capacity": 1073741824,
	"physical": 4653056
	}
	}
	}

5. 设置抓取时间间隔（秒）

http://localhost/interval/check	
HTTP method	POST
参数	interval：时间间隔（秒）
返回值	
返回值 json 示例	{'message':'success, new check interval is xxx' }

6. 设置获取虚拟机列表间隔（秒）

http://localhost/interval/travelsal	
HTTP method	POST
参数	interval：时间间隔（秒）
返回值	
返回值 json 示例	{'message':'success, new check interval is xxx' }

7. 添加物理机 IP

http://localhost/host/add	
HTTP method	POST
参数	host：物理机 IP
返回值	
返回值 json 示例	{'message':'success'}

8. 删除指定 IP 的物理机

http://localhost/host/虚拟机 IP	
HTTP method	POST
参数	
返回值	
返回值 json 示例	{'message':'success'}

详细调用示例请查看 index.html 和 graph.html