Ontology python 测试框架基础使用介绍

一、 目录结构

1. 测试服务端及相关工具文件如图 1：

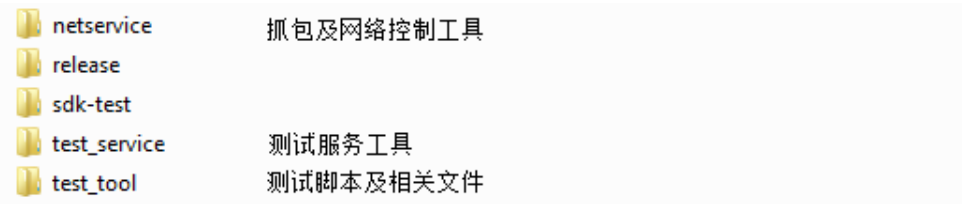

图 1

2. 客户端测试框架如图 2：

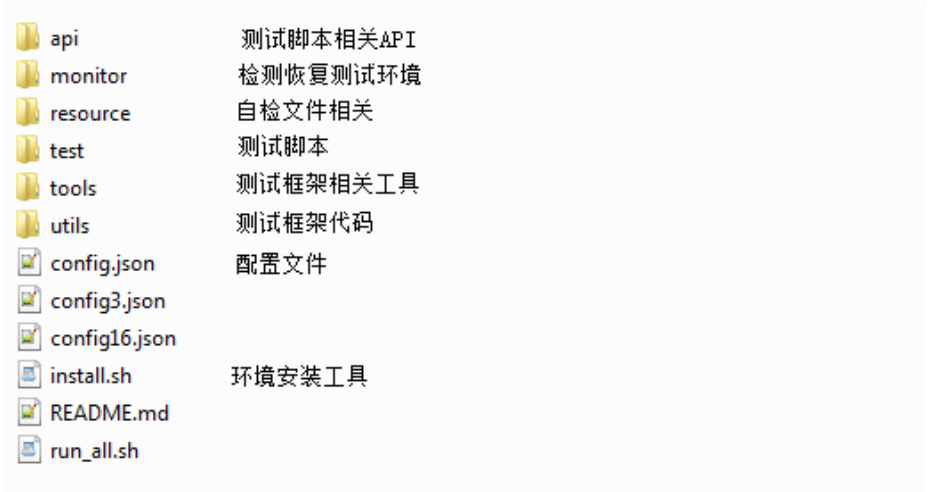

图 2

3. 客户端测试用例及相关工具如图 3：

图 3

二、 测试环境搭建及运行测试脚本

使用自检文件配置：

1. 在 GitHub 上下载 ontology 测试相关文件，下载文件 ontology-test 打开后如图 1 所示。其中 test_service 中有测试服务工具。在 test_tool 中包含测试脚本、测试框架相关代码工具、自检文件、检测恢复等文件，如图 2 所示。
2. 首先将下载的 ontology-test 中的 test_service 配置到/home/ubuntu/ontology/下（必须与 node 文件夹同级目录），然后按照以下步骤安装所需 python 模块并开启测试相关服务：

   $ cd /home/ubuntu/ontology/test_service/

   $ chmod 777 install.sh

   $ ./install.sh

   $ chmod 777 run.sh

   $ ./run.sh
3. 接着将 ontology-test 下的 netservice 置于/home/ubuntu/ontology/下（与 node 文件夹同级目录），然后按以下步骤安装相关模块：

   $ cd /home/ubuntu/ontology/netservice/

   $ chmod 777 environment.sh

   $ ./ environment.sh
4. 对所有 16 台服务器配置以上 2~3 的所有步骤。
5. 将 ontology-test 下的 netservice 和 test_tool 置于/home/ubuntu/ontology/下（与 node 文件夹同级目录），然后按以下步骤安装相关模块：

   $ cd /home/ubuntu/ontology/test_tool/

   $ chmod 777 install.sh

   $ ./install.sh
6. 使用自检文件核对并修改测试环境。在/home/ubuntu/ontology/test_tool/tools/init_selfcheck 路径下包含了自检文件及其配置文件，另外在 ontology-test\test_tool\resource 下有自检文件中需要使用的最新文件。在实现自检之前需要修改配置文件中的内容，配置修改示例如下：

```
{
    "resource":{
        "root":"/home/ubuntu/ontology/test_tool/resource",
        "ontology_source_name": "ontology",
        "wallet_source_name": "wallet",
        "onto_config_source_name": "onto_config.json",
        "test_config_source_name": "test_config.json",
        "sigsvr_source_name": "sigsvr",
        "abi_source_name": "abi",
        "test_service_source_name": "rpcserver.py"
    },

    "node":{
        "root":"/home/ubuntu/ontology/node/",
        "onto_name": "ontology",
        "wallet_name": "wallet.dat",
        "onto_config_name": "config.json",
        "sigsvr_name": "sigsvr",
```

```
            "abi_name": "abi"
        },

        "test_config_path": "/home/ubuntu/ontology/test_tool/config.json",
        "default_node_args": " --ws --rest --loglevel=0 --enableconsensus --networkid=6666
--gasprice 0 --gaslimit 20000 ",
        "test_service_path": "/home/ubuntu/ontology/test_servicer/rpcserve.py"
}
```

该配置文件内容的 resource 中的 root 为自检文件相关 resource 文件夹的路径， node 中的 root 为 node 文件夹路径，test_config_path 为 test_tool 下 test_config.json 所在路径（test_tool 路径按实际存放路径填写），而 test_service_path 为测试服务 rpcserver.py 文件所在路径（必须为 /home/ubuntu/ontology/test_service/ rpcserve.py）。若 test_tool 存放路径不同则按自己的实际路径填写。若将 test_tool 文件夹配置在/home/ubuntu/ontology/test_tool 路径，则填入内容如上所示。***若改变 test_tool 文件夹和 node 文件夹的路径，以上 resource 和 node 中的 root 都需要相应的变化。当然如若节点服务器内 node 文件夹中的 config.json，sigsvr，abi，ontology，wallet.dat 等作出命名上改动，在配置文件中" node"下的"onto_name", "wallet.dat","config.json","sigsvr_name","abi_name"的内容都需要做相应变化，其他配置默认即可。

除此以外/home/ubuntu/ontology/test_tool 下的 config.json 内容需要修改如下：

```
 "RPC_URL": "http:// 127.0.0.1:20336/jsonrpc",
"CLIRPC_URL": "http://127.0.0.1:20000/cli",
"RESTFUL_URL": "http:// 127.0.0.1:20334",
"WS_URL": "ws:// 127.0.0.1:20335",
"NODES": [{
        "ip": "139.219.128.177",
        "wallet" : "wallet00.dat"
},
{
        "ip": "139.219.141.110",
        "wallet" : "wallet01.dat"
},
…
],
 "MULTI_SIGNED_ADDRESS" : "AJroHBLRsfPTPQocSKjpADp4AKybg8GZuz",
 "INIT_AMOUNT_ONG" : "1000000000000000",
 "DEFAULT_NODE_ARGS" : "--ws --rest --loglevel=0 --enableconsensus --networkid=299
--gasprice 0 --gaslimit 20000 ",
 "NODE_PATH" : "/home/ubuntu/ontology/node"
}
```

其中 NODES 中的 ip 需要改为实际 16 个节点服务器的 ip 地址（前七个默认为创世共识节点），NODE_PATH 为节点中的 node 文件夹路径，其他配置默认即可。上面的 NODE_PATH 也已经为笔者当前所在 node 文件夹位置所以只需要将 ip 改为实际节点服务器对应的 ip。RPC_URL，CLIRPC_URL，RESTFUL_URL，WS_URL 修改如下：

```
 "RPC_URL": "http:// 127.0.0.1:20336/jsonrpc",
```

"CLIRPC_URL": "http://127.0.0.1:20000/cli",
"RESTFUL_URL": "http:// 127.0.0.1:20334",
"WS_URL": "ws:// 127.0.0.1:20335",

而且/home/ubuntu/ontology/test_tool/resource 下的 onto_config.json 内容需要修改如下：


```
"ConsensusType": "vbft",
    "VBFT": {
        "peer_handshake_timeout": 10,
        "max_block_change_view": 1000,
        "admin_ont_id": "did:ont:AG4pZwKa9cr8ca7PED7FqzUfcwnrQ2N26w",
        "peers": [
            {
                "address": "AG4pZwKa9cr8ca7PED7FqzUfcwnrQ2N26w",
                "peerPubkey":
"02e3ff56206295e71295aeaf47b7f3681f93fb98b692a66b103feb7b756667c74d",
                "initPos": 0,
                "index": 1
            },
            {
                "address": "AUSyZx1BoQbTy3avMKygvJrkw8mRbzt6Aa",
                "peerPubkey":
"02ca35fd7f3d78f01a423e17a85fe122715c2bcece7c5c8c851adcb265eb59486c",
                "initPos": 0,
                "index": 2
        },
        ……
    ],
        "vrf_value":
"1c9810aa9822e511d5804a9c4db9dd08497c31087b0daafa34d768a3253441fa20515e2f30f81741102af
0ca3cefc4818fef16adb825fbaa8cad78647f3afb590e",
        "min_init_stake": 10000,
        "hash_msg_delay": 10000,
        "l": 112,
        "k": 7,
        "c": 2,
        "n": 7,
        "block_msg_delay": 10000,
        "vrf_proof":
"c57741f934042cb8d8b087b44b161db56fc3ffd4ffb675d36cd09f83935be853d8729f3f5298d12d6fd28d
45dde515a4b9d7f67682d182ba5118abf451ff1988"
    },
    "SeedList": [
        "139.219.128.177:20338",
        "139.219.141.110:20338",
        "139.219.141.33:20338",
```

"139.219.136.233:20338"

]

}

注意 onto_config.json 中 ConsensusType 为共识机制根据实际情况为“vbft”或“dbft”。admin_ont_id 中的内容为“did:ont:”加上 index 为 1 的节点 address。peers 中填入七个节点的相关信息，address 为节点 wallet 中的 address，peerPubkey 为节点 pubkey，index 为节点 pubkey，initPos 为设置的 initpos。SeedList 中填入节点 index 为 1 到 4 的节点 ip 地址格式如上。

最后根据如下操作完成自检：

$ cd /home/ubuntu/ontology/test_tool/tools/init_selfcheck

$ chmod 777 init_selfcheck.py

$ python3 init_selfcheck.py

自检文件会检查 ontology 的执行权限、版本和 MD5，abi 下所有文件 MD5，测试服务版本以及 sig 的执行权限、版本和 MD5，并将 test_config.json 和 onto_config.json 与 wallet（ontology-test\test_tool\resource 下的）依次进行确认和修正，修正结束后将 onto_config.json 和 wallet 文件夹覆盖到 16 个节点服务器下，最终重启所有节点服务器和 sig 服务并检查重启后节点数量是否正常。

在使用自检文件后即可开始执行相关测试例，如何使用见**脚本测试步骤。**

*****如果不使用自检文件启动节点服务器，而进行手动配置，可以按照以下步骤：**

1. 在 GitHub 上下载 ontology 测试相关文件，下载文件 ontology-test 打开后如图 1 所示。其中 test_service 中有测试服务工具，在 test_tool 中包含测试脚本、测试框架相关代码工具、自检文件、检测恢复等文件（如图 2）。
2. 首先将下载的 ontology-test 的 test_tool 和 netservice 配置到/home/ubuntu/ontology/下（与 node 文件夹同级目录），然后按照以下步骤安装所需 python 模块：

   $ cd /home/ubuntu/ontology/test_tool

   $ chmod 777 install.sh

   $ ./install.sh

   $ cd /home/ubuntu/ontology/netservice/

   $ chmod 777 environment.sh

   $ ./ environment.sh
3. 接着将 ontology-test 下的 test_service 置于/home/ubuntu/ontology/下（必须与 node 文件夹同级目录），然后按以下步骤安装相关模块并开启测试相关服务：

   $ cd /home/ubuntu/ontology/test_service/

   $ chmod 777 install.sh

   $ ./install.sh

   $ chmod 777 run.sh

   $ ./run.sh
4. 启动节点服务器的 sigsvr 签名服务器，使用默认设置即可：

   $ cd /home/ubuntu/ontology/node/

   $ chmod 777 startsisvr.sh

   $ ./ startsisvr.sh
5. 修改/home/ubuntu/ontology/test_tool 下的 config.json：

   1) 将其中的"RPC_URL"，"CLIRPC_URL"，"RESTFUL_URL"，"WS_URL"中的 ip 改为本机地址，当然也可以修改成：

"RPC_URL": "http://127.0.0.1:20336/jsonrpc",
"CLIRPC_URL": "http://127.0.0.1:20000/cli",
"RESTFUL_URL": "http://127.0.0.1:20334",
"WS_URL": "ws://127.0.0.1:20335",

2) NODES 中的十六个节点需要修改为实际节点的 IP 与 Wallet，修改后如下所示，注意前七个为创世节点：

"NODES": [{
"ip": "139.219.128.177",
"wallet" : "wallet00.dat"
},
{
"ip": "139.219.141.110",
"wallet" : "wallet01.dat"
},
...

3) 将 NODE_PATH 中的路径修改为节点服务器中 node 文件夹所在地址，修改后示例如下（只要不改 node 文件夹地址，即保留以下示例内容即可）：

"NODE_PATH" : "/home/ubuntu/ontology/node"

4) 其他配置默认即可，不需要修改。

6. 修改/home/ubuntu/ontology/node 下的 config.json：

```
"ConsensusType": "vbft",
    "VBFT": {
        "peer_handshake_timeout": 10,
        "max_block_change_view": 1000,
        "admin_ont_id": "did:ont:AG4pZwKa9cr8ca7PED7FqzUfcwnrQ2N26w",
        "peers": [
            {
                "address": "AG4pZwKa9cr8ca7PED7FqzUfcwnrQ2N26w",

"peerPubkey":"02e3ff56206295e71295aeaf47b7f3681f93fb98b692a66b103feb7b756667c74d",
                "initPos": 0,
                "index": 1
            },
            {
                "address": "AUSyZx1BoQbTy3avMKygvJrkw8mRbzt6Aa",
                "peerPubkey":
"02ca35fd7f3d78f01a423e17a85fe122715c2bcece7c5c8c851adcb265eb59486c",
                "initPos": 0,
                "index": 2
            },
            ……
],
        "vrf_value":
"1c9810aa9822e511d5804a9c4db9dd08497c31087b0daafa34d768a3253441fa20515e2f30f81741102af
```

0ca3cefc4818fef16adb825fbaa8cad78647f3afb590e",
"min_init_stake": 10000,
"hash_msg_delay": 10000,
"l": 112,
"k": 7,
"c": 2,
"n": 7,
"block_msg_delay": 10000,
"vrf_proof":
"c57741f934042cb8d8b087b44b161db56fc3ffd4ffb675d36cd09f83935be853d8729f3f5298d12d6fd28d45dde515a4b9d7f67682d182ba5118abf451ff1988"
},
"SeedList": [
"139.219.128.177:20338",
"139.219.141.110:20338",
"139.219.141.33:20338",
"139.219.136.233:20338"
]
}

注意 onto_config.json 中 ConsensusType 为共识机制根据实际情况为“vbft”或“dbft”。admin_ont_id 中的内容为“did:ont:”加上 index 为 1 的节点 address。peers 中填入七个节点的相关信息，address 为节点 wallet 中的 address，peerPubkey 为节点 pubkey，index 为节点 pubkey，initPos 为设置的 initpos。SeedList 中填入节点 index 为 1 到 4 的节点 ip 地址格式如上。其他配置保留默认即可。

7. 对所有 16 台服务器配置以上 2~6 的所有步骤。
8. 启动整个区块链：
   1) 首先找到/home/ubuntu/ontology/test_tool/tools 下的 nodecontroll.py 和 config.json 文件，注意启动整个区块链的节点之前配置 config.json 文件中的内容，peers 中的内容如下所示：

"peers": [
{
"address": "AG4pZwKa9cr8ca7PED7FqzUfcwnrQ2N26w",
"peerPubkey":
"02e3ff56206295e71295aeaf47b7f3681f93fb98b692a66b103feb7b756667c74d",
"initPos": 10000,
"index": 1
},
{
"address": "AUSyZx1BoQbTy3avMKygvJrkw8mRbzt6Aa",
"peerPubkey":
"02ca35fd7f3d78f01a423e17a85fe122715c2bcece7c5c8c851adcb265eb59486c",
"initPos": 10000,
"index": 2
},
.....

注意其中 index 为节点 index，address 为节点钱包地址，peerPubkey 为节点 Pubkey，initPos 为需要设置的 initPos，这里只需要填写前七台节点服务器来作为创世节点。

2) 其中 SeedList 中的为前四台节点服务器地址，SeedList 中的内容格式示例如下所示：

```
"SeedList": [
    "139.219.128.177:20338",
    "139.219.141.110:20338",
    "139.219.141.33:20338",
    "139.219.136.233:20338"
]
```

3) admin_ont_id 为 did:ont:后加当前节点的钱包地址，当前使用的节点为 index 为 1 的节点所以 admin_ont_id 中的内容示例如下：

"admin_ont_id": "did:ont:AG4pZwKa9cr8ca7PED7FqzUfcwnrQ2N26w"

4) ConsensusType 为共识机制：按实际情况填入 vbft 或 dbft。

5) 按以下步骤批量启动节点服务器：

```
$ cd /home/ubuntu/ontology/test_tool/tools
$ python3 nodecontroll.py -a "restart" -n 0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15
#这里为重启所有节点服务器，-a 后还可以填入"start"，但使用 python3 nodecontroll.py
# -a"start" -n 0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15 不会清除之前生成的 log 和 Chain
#如若已经启动过节点，需要在此之前关闭，关闭操作只需将-a 的"restart"修改为"stop"
```

*启动或关闭单个节点可以使用如下步骤：

```
$ cd /home/ubuntu/ontology/node
$ ./start.sh  #或者$ ./stop.sh，并根据需求删除 log 与 Chain
```

脚本测试步骤如下：

测试脚本位于/home/ubuntu/ontology/test_tool/test 下，路径下文件内容如下：

使用 run_alltest.py 可以实现选择性测试相关用例，select.json 为选择性执行测试脚本的配置文件。

在进行测试之前可以先获取所有脚本的执行权限，步骤如下：

```
$ cd /home/ubuntu/ontology/test_tool
$ chmod -R 777 test
$ cd /home/ubuntu/ontology/test_tool/test
$ python3 run_alltest.py #执行所有 case
```

***

另外，如若无需测试所有的 case，可以按照如下规则调用 run_alltest.py 有选择地执行 case：

在 run_alltest 中导入了 test_tool/monitor 下 monitor.py 中的 TestMonitor，monitor 可以在执行 case 时监测环境中的未知错误和恢复测试环境。调用 run_alltest 执行 case 时，可以根据参数选择执行需要测试的 case，当然首先需要到 run_alltest 所在路径：

$ cd /home/ubuntu/ontology/test_tool/test

调用 run_alltest 时，其后可以跟-c(--config)、-t(--type)、-f(--filter)、-m(--monitor)、-e(--exclude)。

1) -c 后跟配置文件，一般 ontology-test 为 select.json。将 select.json 中修改如下：

```
{
    "test_rpc" : false,
    "test_restful" : false,
    "test_websocket" : false,
    "test_benefit_model" : true,
    "test_consensus" : false,
    "test_muti_contract" : false
}
```

在进行如下调用时便可选择只执行分润模型相关测试：

$ cd /home/ubuntu/ontology/test_tool/test

$ python3 run_alltest.py -c select.json

2) -t 后跟 case 类型："base"，"normal"，"abnormal"。-t 为"base"时便可选择所有 case 中方法名中包含_base 的 case，"abnormal"时便可选择所有 case 中方法名中包含_abnormal 的 case 即期望结果为异常的 case，但注意"normal"时是选择所有 case 中方法名中包含_normal 和_base 的所有 case 即期望结果为正常的 case，示例如下：

$ cd /home/ubuntu/ontology/test_tool/test

$ python3 run_alltest.py -t "base"

3) -f 后跟具体的 case，比如需要调用 test_auth 下的第一条 case，可以通过以下步骤实现单条 case 的执行：

$ cd /home/ubuntu/ontology/test_tool/test

$ python3 run_alltest.py -f test_auth_1.test_base_001_initContractAdmin

4) -m 后跟"0"或"1"，0 代表不启用 monitor，1 代表启用 monitor，调用 run_alltest 时无-m 时默认启用 monitor，其调用规则如下：

$ cd /home/ubuntu/ontology/test_tool/test

$ python3 run_alltest.py -m "1" 或 python3 run_alltest.py -m 1

-e 后跟"类名.方法名"与-c 配合，可以用于执行-c 中选定的测试例集合中排除-e 中的那些测试例。比如以下步骤中即为执行分润中的测试例但不包含第一条和第二条测试例，注意配置文件中内容与 1）中相同：

$ cd /home/ubuntu/ontology/test_tool/test

$ python3 run_alltest.py -c select.json –e test_benefit_model_1.test_base_001_benefit, test_benefit_model_1.test_normal_002_benefit