useful_links for salt and jinja:

http://docs.saltstack.com/en/latest/ref/states/all/
http://docs.jinkan.org/docs/jinja2/templates.html#id6
http://docs.saltstack.cn/zh_CN/latest/topics/yaml/index.html

1) salt installation and configuration

salt-minion id is set default to hostname of current node after installation, you can change minion_id by modifying file /etc/salt/minion_id, make sure minion-id are not duplicated among all the minions managed by salt-master. Else if two salt-minion ‘node1’ and ‘node2’ have the same minion-id ‘node1’, run one salt command like ‘salt node1 state.sls pitrix.hyper pitrix’ may install hyper on both ‘node1’ and ‘node2’.

salt-master authenticate salt-minion by using key, after salt-minion is configured with the correclt salt-master, it will send it’s pub keys to salt-maser, two wasy of authentication by master:

a) automatically authenticate

configure /etc/salt/master.d/pitrix.conf, there is a option:

autosign_file: /etc/salt/autosign.conf

/etc/salt/autosign.conf records which node will be autosiged.

our installation scripts will record all nodes to autosign.conf you have configured in following files:

/pitrix/deployer/salt/pillar/ks_nodes.sls
/pitrix/deployer/salt/pillar/physical_nodes.sls

At the end of pxe installation, the salt-minion nodes being pxed will try to connect to salt-master, and salt-master will autosign those nodes.

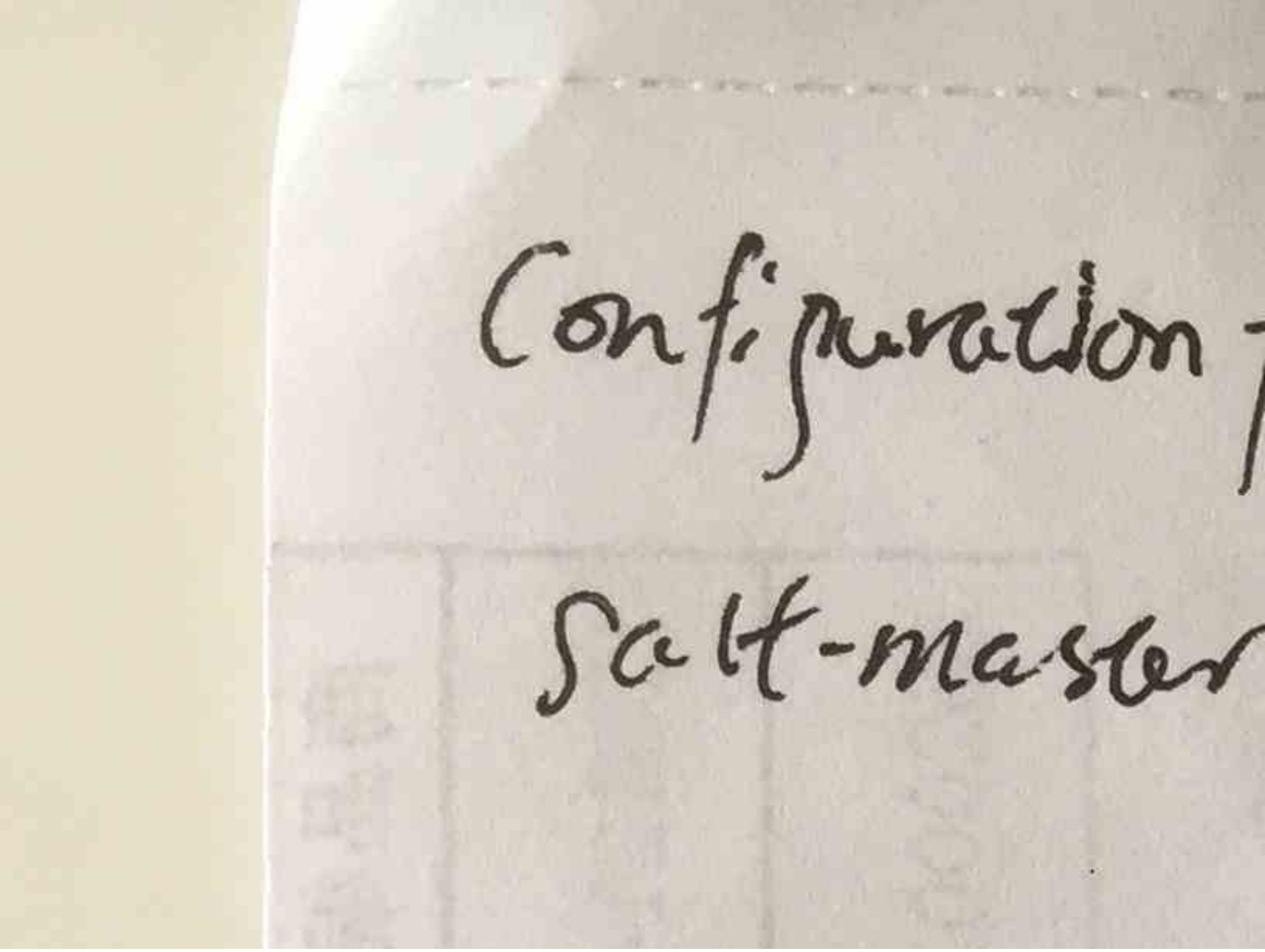
b) manually authenticated

salt-key -L to get the unaccepted salt-minion key
salt-key -A to accept all unaccepted salt-minion key
salt-key -a node1 to accept a special minion’s key
salt-key -d node1 to delete a special minion’s accepted key
salt-key -D to delete all minion’s accepted keys

Note:

In some special case, you have installed a node named ‘node1’, and it’s salt key has been accepted by salt-master, then you reinstall ‘node1’ operating system, and reinstall salt-minion on node1, run ‘salt-run manage.status’ on salt-master will find ‘node1’ is down, this is because salt-master still caches ‘node1’ old salt-key, do following to resolve this problem:

salt-key -d node1 on master
/etc/init.d/salt-minion restart on minion



configuration files on salt-master:

/etc/salt/master.d/pitrix.conf

/etc/salt/autosign.conf

configuration files on salt-minion:

/etc/salt/minion.d/pitrix.conf

/etc/salt/minion_id

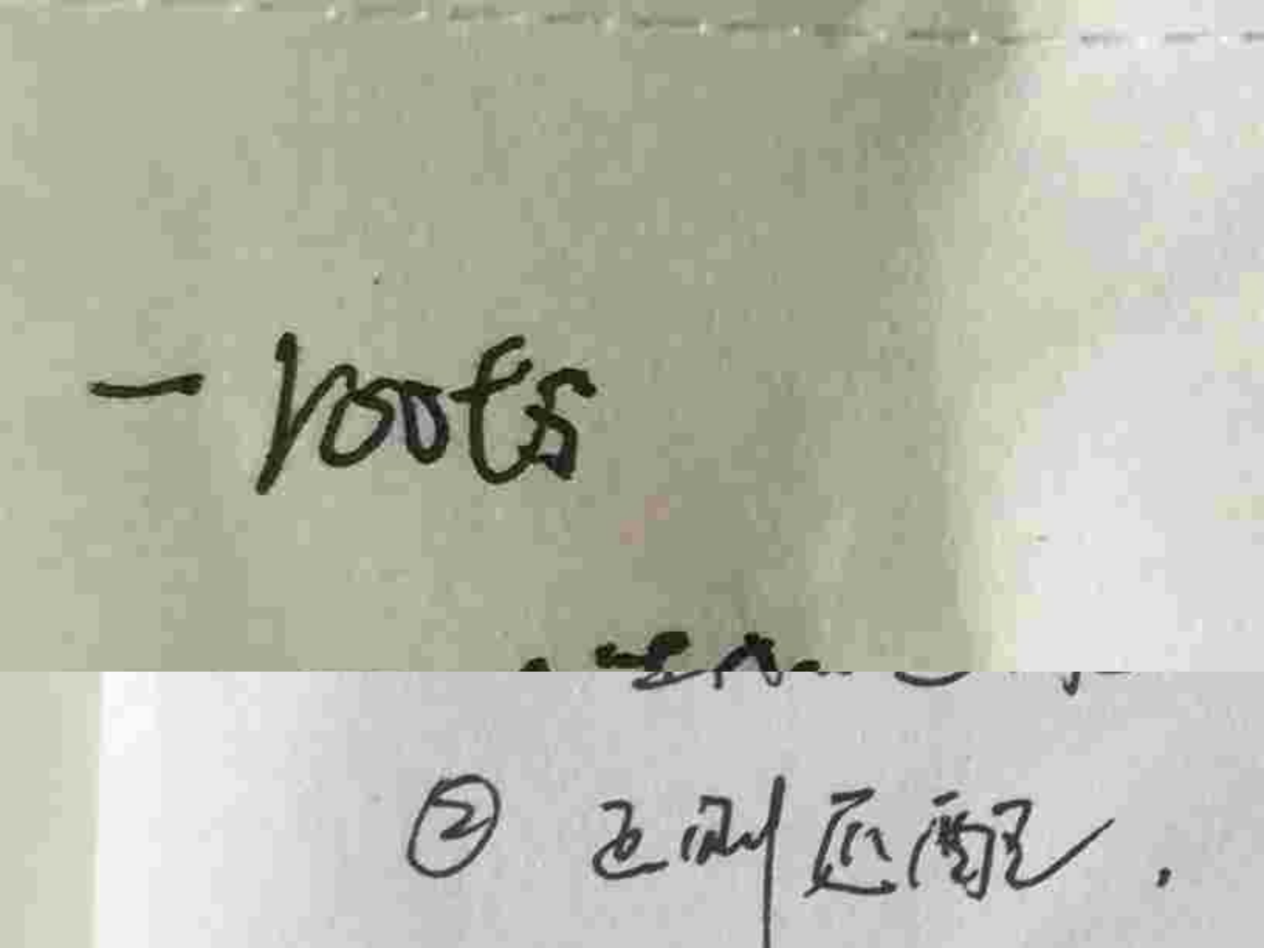
/etc/salt/pki -> salt-minion will store its key in this folder, also cache master's pub key here

2) how to name scripts when running 'salt target state.sls' command ?

/etc/salt/master.d/pitrix.conf defines where to store pillar and salt scripts

name the salt scripts according to the directory structure of salt scripts, like

salt node1 state.sls postgresql.master pitrix



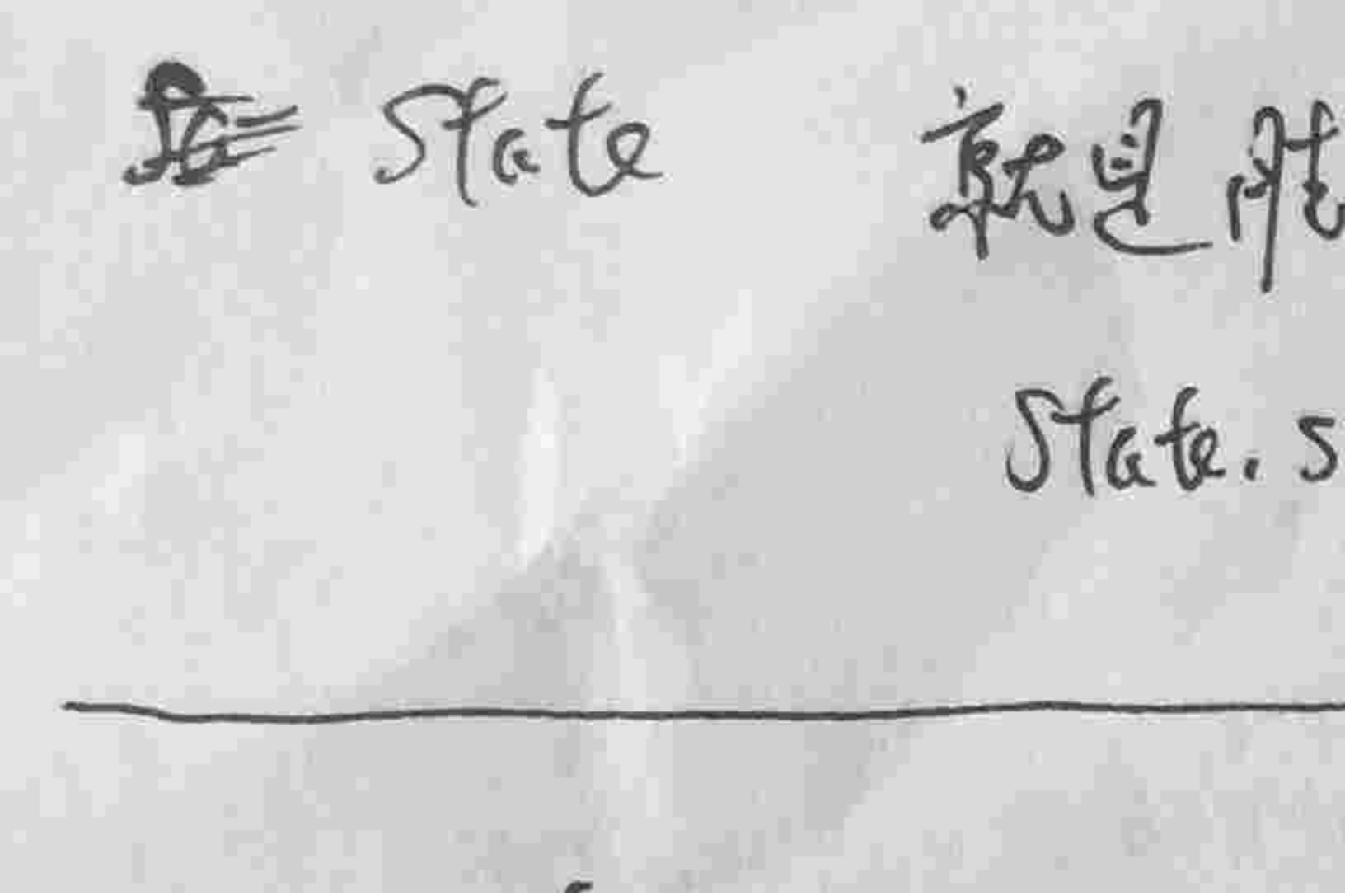
salt node1 state.sls pitrix.hyper pitrix
salt node1 state.sls pitrix.hyper.ovs pitrix

3) How does salt target minions ?

4) code structure

Each salt state folder has a init.sls files, like pitrix/hyper has init.sls, if you call pitrix.hyper directory, it will call pitrix.hyper.init by default. Else if you just want to install ovs on hyper, run ‘salt node1 state.sls pitrix.hyper.ovs pitrix’ instead.

5) mind difference between grain,mine,pillar



grain -> for each salt minion, and call only be read by minion itself
pillar -> for the whole salt topology, can be read by every node
mine -> subset of grain, especially useful if some nodes want to read other nodes' grain

6) remember to sync resource after change
if pillar is changed, please sync pillar for all nodes:
salt '*' saltutil.sync_pillars pitrix
if node's role is changed, need to sync grains for node(s):
salt node1 state.sls pitrix.salt pitrix
if you have modified modules, need to sync modules for all nodes:
salt '*' saltutil.sync_modules pitrix