Preparations before Deployment

System Configuration Requirements

1) Minimum system configuration is 4-core CPU, 8G memory, 50G available disk space, and 2M bandwidth.
2)Recommend system configured to 8-core CPU, 16G memory, 200G available hard disk space, 10 M bandwidth.

Login system, input ping www.baidu.com verify network is normal.

ICW Layout Node Tutorial

Create nodes with this document. The checklist informs you of the requirements for creating the node and provides a summary of the steps required.

1. Download and upload the icw_wallet. tar file to the server at http://8.210.21.144:1999/dist/ICW_Wallet.tar.

Note: It can also be downloaded directly using the command.

1) Login the system and run the following command to install the two software

yum -y install wget
yum -y install vim

2) Download and **install ICW_Wallet. tar on Linux**

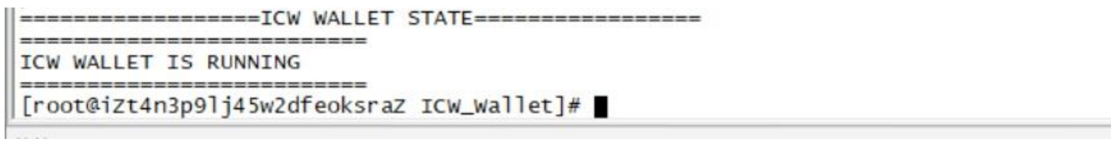
wget http://8.210.21.144:1999/dist/ICW_Wallet.tar

3) Decompression

Input command: tar –xvf ICW_Wallet .tar

4) Initiating by entering ICW_Wallet (working with copied command)

Input command: cd ICW_Wallet
Start node
Run command ./start to start the wallet
Run command ./check-status to check the node is started.
./check-status

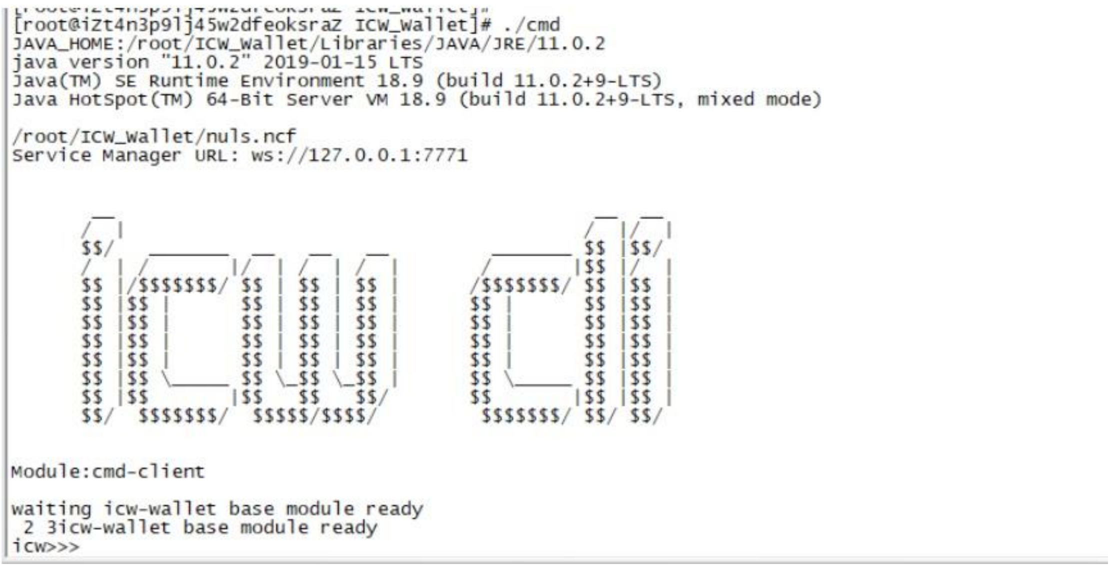


```
==================ICW WALLET STATE==================
===========================
ICW WALLET IS RUNNING
===========================
[root@iZt4n3p9lj45w2dfeoksraZ ICW_Wallet]#
```

The above figure shows successful startup.

Once the node is started, input. / CMD and check whether the node is working as expected. That is, you can send CLI commands to nodes.

Run command ./cmd



```
[root@iZt4n3p9lj45w2dfeoksraZ ICW_Wallet]# ./cmd
JAVA_HOME:/root/ICW_Wallet/Libraries/JAVA/JRE/11.0.2
java version "11.0.2" 2019-01-15 LTS
Java(TM) SE Runtime Environment 18.9 (build 11.0.2+9-LTS)
Java HotSpot(TM) 64-Bit Server VM 18.9 (build 11.0.2+9-LTS, mixed mode)

/root/ICW_Wallet/nuls.ncf
Service Manager URL: ws://127.0.0.1:7771


      __                                             __   __
     /  |                                           /  | /  |
     $$/   _______  __   __   __        _______ $$ | $$/
     /  | /       |/  | /  | /  |      /       |$$ | /  |
     $$ |/$$$$$$$/ $$ | $$ | $$ |     /$$$$$$$/ $$ | $$ |
     $$ |$$ |      $$ | $$ | $$ |     $$ |      $$ | $$ |
     $$ |$$ |      $$ | $$ | $$ |     $$ |      $$ | $$ |
     $$ |$$ |      $$ | $$ | $$ |     $$ |      $$ | $$ |
     $$ |$$ |      $$ | $$ | $$ |     $$ |      $$ | $$ |
     $$ |$$ \_____ $$ \_$$ \_$$ |     $$ \_____ $$ | $$ |
     $$ |$$       |$$   $$   $$/      $$       |$$ | $$ |
     $$/  $$$$$$$/  $$$$$/$$$$/        $$$$$$$/ $$/  $$/


Module:cmd-client

waiting icw-wallet base module ready
 2 3icw-wallet base module ready
icw>>>
```

5) Waiting for your node to synchronize with the blockchain

Use the command Network Info to determine whether synchronization is complete or how close you are to completing synchronization.

```
icw >>>network info
icw >>> network info
{
"localBestHeight" : 4837,
"netBestHeight" : 53905,
"timeOffset" : 995,
"inCount" : 0,
"outCount" : 5
}
icw>>>
```

Synchronization is complete when localBestHeight equals netBestHeight. This is a case in point.

```
icw>>> network info
{
"localBestHeight" : 53920,
"netBestHeight" : 53920,
"timeOffset" : 2,
"inCount" : 0,
"outCount" : 20
}
```

Note: The speed of synchronizing the height of the block is related to the network speed of the machine.

6) Import your packaged account

Once your node is synchronized with the blockchain, import the packaged account.

Run the import command to import the on-chain address private key and package the address private key. Your address will be displayed after the import is successful. The following is an example.

Import the on-chain address

icw>>> import b54db432bba7e13a6c4a28f65b925b18e63bcb79143f7b894fa735d5d3d09db5

Please enter the password (password is between 8 and 20 inclusive of numbers and letters), If you do not want to set a password, return directly.
Enter your password:**********
Please confirm new password:**********
tNULSeBaMkrt4z9FYEkkR9D6choPVvQr94oYZp

Note: The private key following import needs to be replaced with the user's own private key.

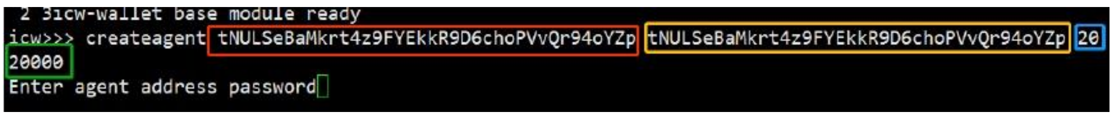
7) Import package address

icw>>> import 12ef0edd5ed75df21bd01f1a224bc19bfd7694e52058445da918cefb27a29507

Please enter the password (password is between 8 and 20 inclusive of numbers and letters), If you do not want to set a password, return directly.
Enter your password:**********
Please confirm new password:**********
tNULSeBaMkrt4z9FYEkkR9D6choPVvQr94oYZp

Note: The address after import needs to be replaced with the user's own address.
The passwords entered in the preceding two steps are used to create a node.
The password for the package address must be ICW123456

8) Create Node
Createagent



```
 2 3icw-wallet base module ready
icw>>> createagent tNULSeBaMkrt4z9FYEkkR9D6choPVvQr94oYZp tNULSeBaMkrt4z9FYEkkR9D6choPVvQr94oYZp 20
20000
Enter agent address password
```

Red box for the chain (agents) address (recommended for wallet ICW address), yellow boxes for packaging addresses, blue box as the commission rate, green box for the deposit.

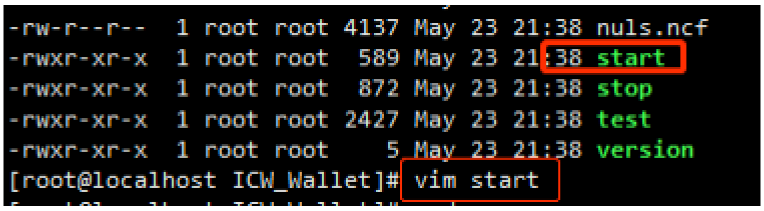
Note: The above addresses are all user's own addresses.
Enter the password for the import chain address input password.
Creation completed

The following instructions for the above parameters:
Create agent < agent address > < Package address > < commission rate > < Deposit >
Agent address (chains) : users
Package address: provided by the user
Commission rate :10-100 adjustable.
Deposit: 20,000 ICW
Reward address: User provided

Comment:

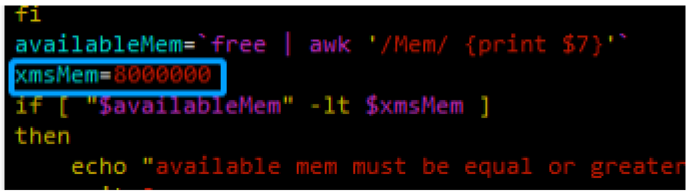
Available memory more than 8G, do not need to perform the following actions.
If you select the minimum configuration, modify the startup file .

Run command: cd /root/ICW_Wallet
vim start



```
-rw-r--r--  1 root root 4137 May 23 21:38 nuls.ncf
-rwxr-xr-x  1 root root  589 May 23 21:38 start
-rwxr-xr-x  1 root root  872 May 23 21:38 stop
-rwxr-xr-x  1 root root 2427 May 23 21:38 test
-rwxr-xr-x  1 root root    5 May 23 21:38 version
[root@localhost ICW_Wallet]# vim start
```

After entry, modify xmsMem parameter = 8000000 to 4000000



```
fi
availableMem=`free | awk '/Mem/ {print $7}'`
xmsMem=8000000
if [ "$availableMem" -lt $xmsMem ]
then
    echo "available mem must be equal or greater
```

Note: please ensure smooth network when synchronizing the block height, do not turn it off.