Hizoco Linux 系统容器部署说明

前言：

由于 Hizoco 链是总体上参考了 CHIA PoST 共识并兼容 CHIA 的 PoS 算力能实现双挖，因此构建 Hizoco 链节点并进行算力竞争出块，部分组件会兼容借用 CHIA 链的工具（并不影响使用 CHIA 链）

部署总体流程：

一、系统需求

二、部署 CHIA

三、Hizoco 节点容器部署与快速初始化

四、算力接入

五、可选的群集化部署和嵌入式收集器部署

六、附录及密钥管理

一、系统需求，建议使用支持 Docker 容器环境的 Linux 应用环境，硬件条件建议至少 6 核的 X86 处理器，16G 内存，64G 以上的硬盘空间，以下为示例环境

操作系系统 Ubuntu 22.04 LTS

Docker 环境：docker.io 24.05

#安装 docker：

sudo apt-get install docker.io

#启用自动启动 docker 服务：

sudo systemctl enable docker;sudo systemctl start docker

二、部署 CHIA，如前言所述，Hizoco 链兼容 CHIA 链的 POS 算力证明，因此可以直接使 CHIA 官方版本的 harvester(收割机)作为 Hizoco 链的 PoC(容量证明)收集器。如果用户已经安装有 CHIA 可以跳过此步。

Ubuntu 安装 CHIA 并设置自动启动

```
# Install packages
sudo apt-get update
sudo apt-get install ca-certificates curl gnupg

# Add GPG key
curl -sL https://repo.chia.net/FD39E6D3.pubkey.asc | sudo gpg --dearmor -o
/usr/share/keyrings/chia.gpg

# Set up repository
echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/chia.gpg]
https://repo.chia.net/debian/ stable main" | sudo tee /etc/apt/sources.list.d/chia.list >
/dev/null
sudo apt-get update
```

```
# Install chia-blockchain
sudo apt-get install chia-blockchain chia-blockchain-cli

# Use chia-blockchain-cli instead for CLI only

#最后设置以当前用户自动启动 CHIA 的节点服务：
chia init
sudo systemctl enable chia-full-node@${LOGNAME}
sudo systemctl enable chia-timelord@${LOGNAME}
sudo systemctl enable chia-farmer@${LOGNAME}
sudo systemctl enable chia-harvester@${LOGNAME}
sudo systemctl start chia-full-node@${LOGNAME}
sudo systemctl start chia-timelord@${LOGNAME}
sudo systemctl start chia-farmer@${LOGNAME}
sudo systemctl start chia-harvester@${LOGNAME}
```

CHIA 节点运行起来后，需要一定的时间同步链上数据，CHIA 的官方网站上也提供有最近期的节点数据可下载导入以减少同步时间。

关于导入已有的密钥，P 图生成 plot 文件等，或是接入当前已 P 好的算力到安装好的 CHIA 节点上等操作，可参照 CHIA 官方文档或网上教程等进行，由于 P 图生成 plot 文件并收集除了官方的 P 图工具以外，还有 GPU 加速与压缩 P 图及收集等优化的第三方工具等多种情况，本说明按 CHIA 官方工具作介绍，具体第三方工具请部署者参考进行。

三、Hizoco 节点容器部署与快速初始化

首先下载最近的 Hizoco 容器镜像备档：

Web 方式：

```
wget http://hizoco.net/public/archive/zoco.tar.gz
wget http://120.53.234.130/zoco.tar.gz
```

或是使用 BT 下载种子后 P2P 下载。

```
有必要先安装 tranmission
sudo apt-get install transmission-cli
wget http://hizoco.net/public/archive/zoco.torrent
tanssmission-cli ./zoco.torrent
```

或是其它手段下载 zoco.tar.gz

导入 docker 镜像

```
sudo docker import ./zoco.tar.gz hizoco:current
```

列出可用的容器镜像：

```
sudo docker images
输出中应有一行：
hizoco          current     f2e60dc47747     11 seconds ago     1.95GB
```

其中哈希、导入时间，大小等视实际情况可能不同

使用刚刚导入并宣称的仓库及 ID 的镜像创建容器实例：

sudo docker create --name zoco --restart always hizoco:current "bash" "-c" "supervisord -n -c /etc/supervisor/supervisord.conf"

--name 后的容器名称可以自定义。

以下假定容器名字如上文为 zoco

启动容器

sudo docker start zoco

初次启动后要初始化一次 hizoco 节点随后与 CHIA 算力节点通讯的 CA 证书及运行参数

sudo docker exec -it zoco "init_hizoco.sh"

正常情况下，会显示以下

CHIA_ROOT is set to /root/.gozoco/root

Chia directory /root/.gozoco/root

No keys are present in the keychain. Generate them with 'chia keys generate'

/root/.gozoco/root already exists, no migration action taken

初始化完成后，重启一次容器：

sudo docker restart zoco

正常地启动后，在宿主系统中可以看到两个进程：

兼容 CHIA POT 算法的 POT 验证进程：chia_timelord

兼容以太坊 EVM 及 ETH 协议的节点进程：geth

基于当前镜像创建的容器内部没有 hizoco 链的数据，需要大量的时间和处理器负荷来同步创世以来的数据，因此和 CHIA 一样，强烈创建从 hizoco 官方网站下载最近的链数据以减少同步时间。

按需选择，WEB 下载：

全数据包括备档（可用于构建自用的区块浏览器），在宿主系统上运行：

wget http://hizoco.net/public/archive/hizoco_data_full_archive.tar.gz -O zoco_data.tgz

完整数据

wget http://hizoco.net/public/archive/hizoco_data_full.tar.gz -O zoco_data.tgz

快照数据

wget http://hizoco.net/public/archive/hizoco_data.tar.gz -O zoco_data.tgz

创建解压目录并解压

mkdir zocodata

tar -xf zoco_data.tgz -C zocodata

建议先关闭运行中的节点容器：

sudo docker stop zoco

将解压出来的数据复制到容器中：

sudo docker cp -a zocodata zoco:/root/.gozoco/

重新启动节点

sudo docker start zoco

（可选），启动后可以清除节点缓存

sudo docker exec zoco "bash" "-c" "rm -rf /root/.gozoco/zocodata/geth/node*"

使用以下命令可在容器运行时连接到容器的 shell 上。

sudo docker exec -it zoco bash

连到到容器的 shell 上后，可使用以下命令连到到节点的控制台：

geth --datadir=/root/.gozoco/zocodata attach

其它：

可选的节点组件：

Hizoco 节点一共有以下组件：

1、EVM 和以太坊协议栈支持的节点进程，下称为节点服务。

2、PoT(间延证明)验证及 PoT 生成调度器，下称为验证器服务。

3、PoT 生成器。

4、CHIA 兼容的 PoC 过滤器，下称为过滤器。

5、CHIA 兼容的 BLS 私钥管理器。

6、CHIA 兼容的 PoC 收集器(不在容器中运行)。

仅与主网同步链接据，并查看账户、交易等的节点，仅需要：

节点服务、验证器服务

依据本地算力，依据 PoT 对应进行 PoC 查找并试图挖矿争取记账权的节点，需要：

节点服务、验证器服务、私钥管理器、过滤器、PoC 收集器

如果不信任网络上的其它节点会依据协议为自己找到的 PoC 生成完成注入的 PoT(IP)，则需要完整部署以上所有节点组件，并耗费本地处理器算力生成 PoT。

虽然不使用本地处理器生成 PoT 也一样可以正常出块，但是仍然建议节点运行者有条件的情况下在本地部署 PoT 生成器，可以有效保证自己拥有的算力能有机会出块。

新的镜像已经直接默认启动节点、验证器、私钥管理器、过滤器、收集器。

如果将容器的 TCP/UDP 30305 端口透到公网 IP 上许可入站访问，将极大的可以优化 P2P 组网的效率，部署者可以依据自身的网络配置情况或使用 NAT、或配置防火墙，或组建 VPN 等手段自动配置。

完成以上配置后，hizoco 节点容器已可以正常与主网通讯并同步，正常情况下，由于导入的最近数据与实际链上数据的差异，会有一小段同步数据的过程，其间处理器占用会比较高，同步完成后则会下降。

四、算力接入，本说明假定部署环境中已经正常运行 CHIA 链节点并已 P 好一定的 plot 文件进行挖矿，以下说明如何前已有的算力接入到 hizoco 节点中进行双挖：

从容器中将 CA 证书目录复制到当前用户

```
mkdir ~/.zoco
sudo docker cp -a zoco:/root/.gozoco/root/config/ssl/ca ~/.zoco/ca
sudo chown ${LOGNAME}:${LOGNAME} -R ~/.zoco/
```

更改 CHIA_ROOT 路径指向到新的目录。

```
export CHIA_ROOT=~/.zoco
```

初始化 harvester 的 SSL 证书和配置

```
chia init
chia init -c ~/.zoco/ca
```

从容器中将示例配置文件复制到宿主系统

```
sudo docker cp -a zoco:/root/.gozoco/root/config/config.yaml
${CHIA_ROOT}/config/
sudo chown ${LOGNAME}:${LOGNAME} -R ${CHIA_ROOT/config}/config.yaml
```

查看容器的 IP:

```
sudo docker inspect zoco | grep IPAddress
```

以下修改 CHIA_ROOT 目录下的 config/config.yaml，重点是 harvester 段下的 farmer_peer 属性到容器的目录，视部署者使用的文本编辑器而定

```
vi ${CHIA_ROOT}/config/config.yaml
```

或是使用 nano 编辑器

```
nano  ${CHIA_ROOT}/config/config.yaml
```

修改配置文件中的内容：

```
farmer_peer:  容器 IP
farmer_port: 8447
```

为了防止与 CHIA 的 harverster 产生端口冲突，可以将 harvester 段下的 RPC 属性关闭：

```
start_rpc_server: false
```

或是改用另一个 RPC 端口。

```
rpc_port: 18560
```

最后保存修改的配置文件。

向容器中的过滤器增加使用的密钥（在宿主的 shell 上执行）：

```
sudo docker exec -it zoco add_key.sh
```

随后输入所 P 图文件对应的私钥助记词并命名

从容器中将收集器的示例 systemd 脚本复制到本地：

sudo docker cp zoco:/usr/lib/systemd/system/zoco-harvester@.service /usr/lib/systemd/system

设置收集器服务自动启动

sudo systemctl enable zoco_harvester

启动收集器服务

sudo systemctl start zoco_harvester

对应地，运行以下指令将存放在 plot 文件的路径加入到收集器的配置中：

chia plots add -d PLOT 文件路径

例如 plot 文件放在/media/ubuntu/22T/final 下：

$chia plots/media/ubuntu/22T/final

（可选）可以运行收集器的检查工具查看是否能读取到有效 PoC 证据：

chia plots check

检查的时间比较长，如果没有出现明显的警告输出则正常，可以用 CTRL-C 键中止检查。

至此 hizoco 节点已经有效接入算力并能参与共识挖矿。

五、可选的群集化部署和嵌入式收集器部署

以上说明均假设节点、算力等都在同一台机器上，依据组件的低耦合设计，其中过滤器和收集器都可以进行群集部署，一般的大型群集可以选择部署一个完整的节点，并部署多台收集器。

Hizoco 的收集器群集基本和 CHIA 的收割机群集一致，原则上只要保证网络内的收集器能有效访问到容器节点的 8447 端口并持有效证书访问即可，因此群集化收集 PoC 证据的部署请依据群集网络的配置并参考 CHIA 进行，主要包括：

1、设置网络以保证群集节点都能访问到容器的 TCP 协议 8447 端口。

2、将主节点上的.zoco/ca 目录复制到群集节点上，比如~/new_ca

3、设置环境变量 CHIA_ROOT 到新的位置，比如 ~/.zoco

4、运行 chia init -c ~/.zoco/ca 使用根证书初始化群集节点以生成收集器证书,

5、从主节点上复制~/.zoco/config/config.yaml 到群集节点上并正确配置 farmer_peer 的 IP 配置到主节点的对应地址和端口上。

由于群集化部署视部署者的网络而定，因此请部署者针对自身的网络环境作设置。

附录 1 API 与网络端口说明

1、P2P 组网，主节点进程同时使用 TCP 和 UDP 协议侦听在容器 IP 的 30305 端口上并试图联接其它节点的 P2P 端口以交换数据同步区块链，建议容器环境的使用者按自己的网络情况对公网开放 P2P 端口以便其它节点能与当前点节更好地连接，减少因为网络原因与网络失步的可能性。

2、API 端口，主节点进程以 TCP 协议在容器 IP 的 8045，8046 端口上提供 HTTP RPC 和 WebSocket RPC，默认情况下除了本地网络的钱包前端（包括并且不限于 MetaMask 浏览器插件、App、Hizoco 钱包等）希望直接访问本地节点来使用 HiZOCO 区块链之外，并不需要对外开放这两个端口。

PoC 过滤器进程使用 SSL 协议基于 TCP 侦听容器 IP 的 8447 端口，运行在容器宿主系统上的，或是宿主环境的局域网内的多个 PoC 收集进程均应该能正常地地访问到，对于单机节点（所有 POC 算力存储都直接挂载在宿主系统上）来说，不需要特别作网络设置也不需要对外开放 8447 端口，对于局域网甚至是互联网群集节点来说（有若干台运行了 POC 收集进程的独立系统并分别挂载了 POC 算力存储），请根据实际部署的网络情况来保证这些收集器都能访问到容器的 8447 端口。

附录 2 钱包与前端软件

PC 端可以直接使用 Metamask 浏览器扩展，访问项目的首页 https://hizoco.net 即可自动增加 hizoco 链。同时支持的 web3 浏览器扩展还有 OKX web 钱包等。

项目方提供了安卓端的 APP 下载：

https://hizoco.net/public/wp_apk/zoco_app-release.apk

也可以使用 Metamask APP 及 TrustWallet 等手机端软件，请按手机系统种类和应用软件商店自行决定使用何种手机端钱包。

附录 3 密钥管理

要在 Metamask 等钱包中导入私钥，所使用的 BIP 助记词不一样，只能使用私钥，导出目前节点使用的私钥的方法为：

```
sudo docker    exec -it zoco "bash" "-c" "env CHIA_ROOT=/root/.gozoco/root
/usr/bin/chia keys show --show-mnemonic-seed --json"
```

在宿主执行该行命令，将会以 JSON 格式导出先前导入和生成的密钥。

其中”farmer_sk” 所指属性则为私钥，有些钱包软件导入私钥时需要在前面加上“0x”前缀以表示附后为十六进制指示的密钥。

以明文展示密钥私钥存在风险！请确保使用该命令时使用的系统足够安全，没有旁人在窥视你的屏幕！