Я много работаю с криптовалютами, поэтому решил написать цикл статей по этому поводу. Сегодня я расскажу, как установить и настроить ноду Zcash, а так же сделать ее мониторинг с помощью Zabbix. Информация уникальная и в рунете практически нет ничего на эту тему, так что я решил поделиться.

Содержание:

Цели статьи

1. Установить ноду криптовалюты zcash на свой сервер.
2. Настроить удаленную работу через rpc ноды с авторизацией.
3. Сделать мониторинг ноды с помощью zabbix.

Введение

Я настраивал и работал с нодами всех популярных и не очень криптовалют. Участвовал в запуске нескольких криптобирж. У меня накопился большой опыт работы с ними. Информация эта очень узкая и специфичная. Почитать чей-то опыт практически негде, даже в англоязычном интернете.

Я все изучал и настраивал сам. За основу брал только официальную документацию. В работе нод много нюансов, которые нужно учитывать. Постараюсь поделиться основной информацией, которая у меня есть.

Я уже писал статью про настройку нод криптовалют, а так же их мониторинг, но это все частично устарело, особенно установка. Надо все обновить. В этот раз будет по каждой ноде отдельная статья, чтобы не мешать все в кучу.

Системные требования zcash

Системные требования zcash, которые указаны в документации:

- 64-х битный процессор.
- 5G оперативной памяти.
- 25 Gb диск

Нода zcash нормально запускается на виртуальной машине с 2 CPU, 4 Gb памяти и ssd диском. На момент написания статьи, блокчейн занимал 25 Gb. Места под него должно быть не меньше. С учетом того, что блокчейн постоянно растет, надо делать запас.

Установка ноды zcash

Ставится zcash не сложно, но и не сказать, что просто, так как готовые пакеты есть только под Debian. Я обычно под ноды использую Ubuntu, поэтому собирать буду из исходников. Хотя пакет от Debian скорее всего подойдет и под Ubuntu, но я не люблю экспериментов. Из исходников точно заработает так, как надо.

Для начала установим необходимые для сборки zcash зависимости.

```
# apt install build-essential pkg-config libc6-dev m4 g++-multilib autoconf libtool ncurses-dev unzip git python python-zmq
zlib1g-dev wget curl bsdmainutils automake
```

Копируем к себе репозиторий.

```
# git clone https://github.com/zcash/zcash.git
```

Выбираем ветку с последней версией.

```
# cd zcash/
# git checkout v2.1.0-1
```

Запускаем скрипт, который скачивает необходимые для сборки файлы. Качать будет много, порядка 1,7 Gb.

```
# ./zcutil/fetch-params.sh
```

После этого запускаем сборку.

```
# ./zcutil/build.sh -j$(nproc)
```

Установим скомпилированные бинарники в систему:

```
# cd src
# install -sv zcashd zcash-cli /usr/local/bin/
# install -sv zcash-gtest zcash-tx /usr/local/bin/
```

Обновление ноды zcash обычно выглядит так:

1. Скачиваем и компилируем новые исходники.
2. Останавливаем ноду, заменяем старые исходники.
3. Запускаем ноду.

Если не было больших изменений, приводящих к переиндексации или полному обновлению блокчейна, то все пройдет быстро. В том случае, если нода запустит индексацию блокчейна, то простой может быть значительный, на некоторых нодах до нескольких часов.

Когда нода в работе, обновление надо обязательно предварительно тестировать и смотреть, к чему оно приводит. В проде я обычно держу 2 ноды — основную и резервную. На резервной тестируется обновление. Если все OK, она становится основной, а рабочая уходит в резерв. И так каждый раз ротируется при обновлении.

Настройка и удаленный доступ

Теперь создадим фай конфигурации ноды zcash в директории *~/.zcash*.

```
# mkdir ~/.zcash
# mcedit ~/.zcash/zcash.conf
```

```
rpcuser=username
rpcpassword=password
addnode=mainnet.z.cash
rpcbind=0.0.0.0
rpcallowip=0.0.0.0/0
```

rpcuser	имя пользователя для работы через rpc
rpcpassword	пароль пользователя
addnode	выбор блокчейна, основной mainnet.z.cash или тестовый testnet.z.cash
rpcbind	ip сервера, на котором будет работать сервис rpc
rpcallowip	список разрешенных ip адресов для подключения к rpc

Я для теста разрешил слушать все ip адреса сервера и подключаться отовсюду. Чаще всего это не нужно. Ограничивайте доступ к ноде списком только своих доверенных адресов. Я всегда это делаю, если нода поддерживает такую возможность. Например, parity для ноды ethereum не имеет таких настроек. Дополнительно я всегда ограничиваю доступ к ноде на уровне iptables. Настоятельно рекомендую это делать.

В общем случае ноду лучше запускать не от пользователя root. Более того, некоторые ноды невозможно запустить под рутом, они хотят отдельного пользователя. У меня обычно нода это виртуальная машина, где кроме самой ноды ничего нет. В этом случае нет принципиальной разницы с точки зрения безопасности, от root она работает или нет.

Запуск и проверка работы

Мы установили и настроили ноду zcash. Теперь запускаем ее. Она поддерживает работу как в виде обычного приложения в консоли, так и в режиме демона. Первый раз можете запустить ее прямо в консоли, чтобы тут же увидеть результат работы.

```
# zcashd
```

Нода сразу же начинает качать блоки, что видно в ее интерфейсе. Вообще, zcash одна из немногих нод, где все очень наглядно и понятно. Не надо выискивать информацию о блоках, трафике где-то в логах или в запросах к ноде. Все очень наглядно.

В принципе, можно запускать ноду в этом режиме в screen, тогда можно очень быстро и просто посмотреть ее статус. В таком виде ее можно и в автозапуск добавить. Но надежнее запускать ноды zcash как демон. Она поддерживает такой режим.

```
# zcashd --daemon
```

Нода запускается в виде службы. За ее состоянием можно следить по логу — *~/zcash/debug.log*. Для автозапуска достаточно добавить в cron:

```
# crontab -e
```

```
@reboot zcashd -daemon
```

Состояние ноды можно проверить через консоль с помощью **zcash-cli**.

```
# zcash-cli getinfo
{
  "version": 2010051,
  "protocolversion": 170009,
  "walletversion": 60000,
```

```
  "balance": 0.00000000,
  "blocks": 368,
  "timeoffset": 0,
  "connections": 1,
  "proxy": "",
  "difficulty": 824.4417744689738,
  "testnet": false,
  "keypoololdest": 1579797402,
  "keypoolsize": 101,
  "paytxfee": 0.00000000,
  "relayfee": 0.00000100,
  "errors": ""
}
```


Корректно остановить работу ноды zcash можно командой:

```
# zcash-cli stop
```

Теперь попробуем выполнить удаленный запрос о состоянии ноды через rpc. Для этого идем на любой другой linux сервер, где есть curl. Главное, проверить работу с сервера приложений, где будет крутиться софт для работы с нодой. В консоли выполняем команду:

```
# curl --data-binary '{"jsonrpc":"1.0","id":"curltext","method":"getblockchaininfo","params":[]}' -H 'content-
type:text/plain;' http://username:password@10.20.1.18:8232/
```

8232 — дефолтный порт, который слушает нода, если вы его явно не указывали в конфигурационном файле. Следите за правильностью набора пользователя или пароля. Если укажете неправильно, в ответ ничего не получите, что несколько сбивает с толку. Информация об этом будет в debug.log, но там ее так много, что трудно заметить то, что надо.

В ответ на запрос получите информацию о состоянии ноды в json формате.

Читается в консоли не очень, зато удобно парсить данные, в том числе для системы мониторинга. Об этом будет рассказано далее. Полное описание api для удаленных запросов можно посмотреть в документации — https://zcash.readthedocs.io/en/latest/rtd_pages/zig.html.

Мониторинг работы ноды zcash

Расскажу, как я мониторю работу ноды. Для этого использую Zabbix.

Если у вас еще нет своего сервера для мониторинга, то рекомендую материалы на эту тему. Для тех, кто предпочитает систему CentOS:

1. Установка CentOS 8.
2. Настройка CentOS 8.
3. Установка и настройка zabbix сервера.

То же самое на Debian 10, если предпочитаете его:

1. Установка Debian 10.
2. Базовая настройка Debian.
3. Установка и настройка zabbix на debian.

Здесь я не буду останавливаться на заббиксе подробно. Для этого у меня есть отдельные статьи. Расскажу сразу суть — что и как мониторю.

В одной из недавних версий у Zabbix появились http агенты, с помощью которых можно отправлять post запросы. Это очень упростило многие задачи по мониторингу. То, что я раньше делал с помощью скриптов и curl теперь можно делать прямо в шаблоне и свободно переносить с сервера на сервер в виде экспорта и импорта шаблона.

Что я мониторю в ноде zcash:

1. Работу самой службы zcashd на сервере.
2. Состояние блокчена и разницу между headers и blocks. Первое это количество блоков в блокчейне, а второе — номер блока, который синхронизирован на твоей ноде.
3. Состояние сети блокчейна, активна или нет.

И соответственно триггеры:

1. Служба zcashd не работает.
2. Нода начинает отставать, разница между headers и blocks превышает 50 или 100 блоков.
3. Сеть блокчейна неактивна.

Теперь как это реализовано. В конце будет готовый шаблон.

Таймаут можно побольше ставить, часто 3 секунды не хватает. Этот итем собирает вот такие данные:

Дальше делаем зависимые элементы, для данных из headers и blocks.

И в предобработке выделяем нужный нам JSONPATH.

По аналогии то же самое делается для blocks. Потом делаем простой триггер с их сравнением.

В данном случае разница в 9 блоков была просьба программистов. Я не знаю, какая именно разница критична для работы.

Дальше берется состояние network запросом:

```
# curl --data-binary '{"jsonrpc":"1.0","id":"curltext","method":"getnetworkinfo","params":[]}' -H 'content-type:text/plain;' http://username:password@10.20.1.18:8232/
```

Делается итем по аналогии с blockchaininfo. И для него зависимый элемент и триггер на наличие слова true в данных из JSONPATH по пути

$.result.networks[:1].reachable.

Вот итоговый список итемов и триггеров.

Готовый шаблон с этими элементами — zabbix-zcash.xml. Импорт делался с версии сервера 4.4 В шаблоне нет графиков. Обычно все это смотрится в Latest Data.

На этом по установке, настройке и мониторингу ноды zcash у меня все.

Заключение

> Не понравилась статья и хочешь научить меня администрировать? Пожалуйста, я люблю учиться. Комментарии в твоем распоряжении. Расскажи, как сделать правильно!

Это первая статья из цикла. Я планирую пройтись по всем основным и популярным нодам — bitcoin, bitcoin cash, ethereum, litecoin, dash, cardano, destream, tron, neo, ripple xrp и др.

Если у вас есть желание каким-то образом использовать ноды, открыть криптобиржу или что-то связанное с блокчейном, можете обращаться ко мне. У меня большой практический опыт работы с ними — установка, обслуживание, обновление, мониторинг. Могу подобрать площадку для размещения, запланировать ресурсы, рассчитать затраты на размещение и т.д.