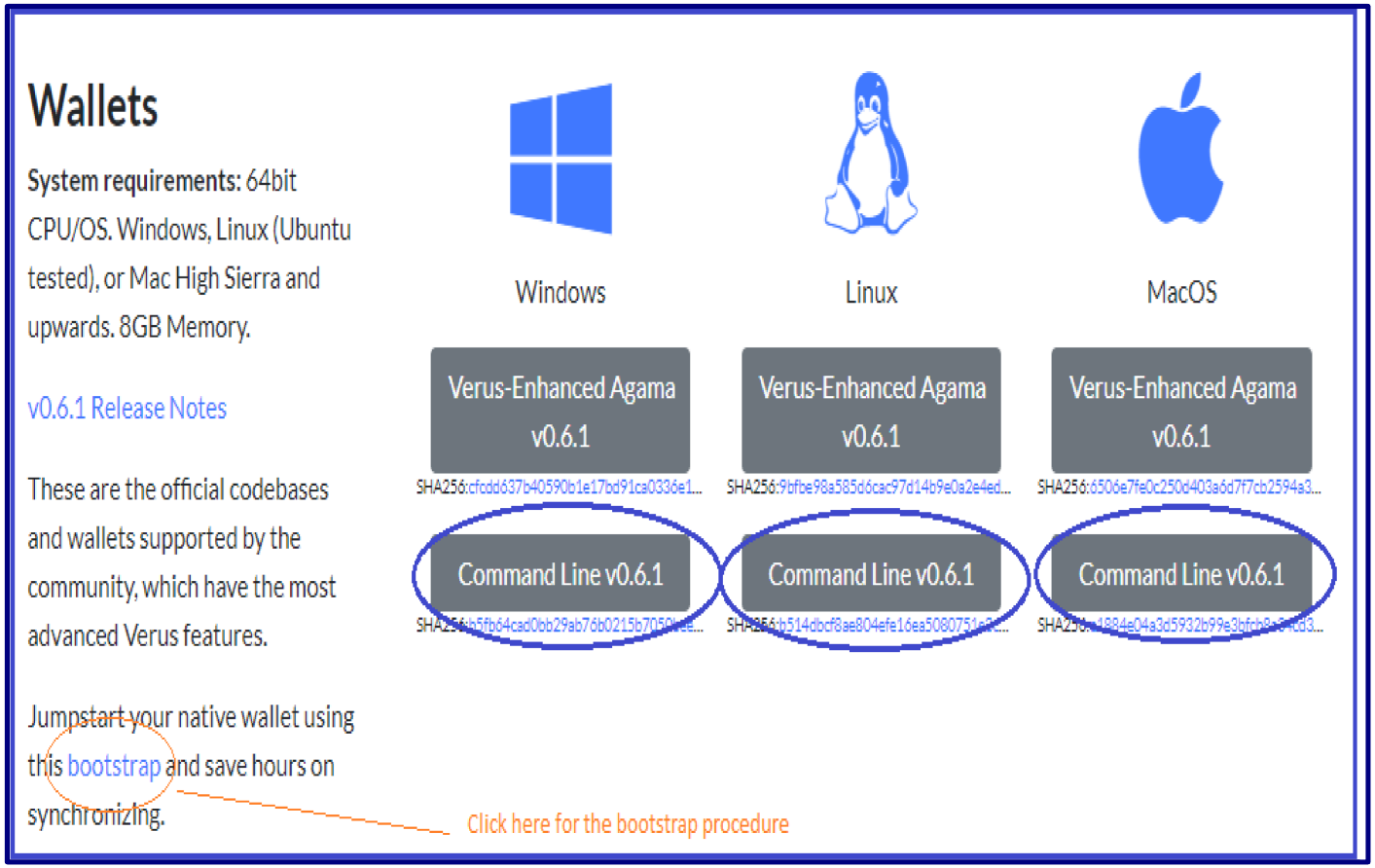
How to create your first Verus ID with CLI -

This guide will explain how to get your first Verus ID using the Command Line Interface. I recommend reading it all down first, and then go through the process of creating an ID.

If you don't the wallet installed yet, check our website at https://veruscoin.io/wallet.html. Scroll down to **Wallets** and choose your operating system.



If you didn't set up your wallet before, please also refer to the guides on that page, especially see the "FAQ and How-to" section at https://veruscoin.io/service.html, and check the **Bootstrap procedure** (marked in above image) to speed up syncing of your wallet at https://veruscoin.io/downloads/how-to/how-to_bootstrap.html

If you have troubles, please visit us on Discord https://discord.gg/VRKMP2S and ask in the **#community-support** channel, so somebody can support you.

Ok, back to topic. This guide will explain how you can get your first ID with a clean and fresh installed CLI wallet. Some of the steps will no longer apply when you already have a working version of that wallet, but you will notice.
Getting a **Verus ID** requires a balance of 100.0002 VRSC in your wallet (or 80.0002 if you have a referral ID). The ID will cost 100 (or 80) VRSC, and there is a transaction fee of 2 x 0.0001 VRSC (at time of writing this guide). So, if you don't have that much, you might want to get that before. If you don't know how, please also see https://veruscoin.io/getVRSC.html for that.

When you want to create an ID:

- start the CLI daemon (The example below will run as a daemon and also stake your balance and mine on 1 thread)
  ```
  ./verusd -mint -gen -genproclimit=1 -daemon 1>/dev/null 2>&1
  ```

- Once the daemon is started, check your blockheight and compare it to the current blockheight on https://explorer.veruscoin.io

  ```
  ./verus checkblockcount
  ```

  which should result in a blocknumber:
  ```
  838996
  ```

- Now it is time to check if you have sufficient funds to acquire an ID. Check your balance with:

  ```
  ./verus z_gettotalbalance
  ```

- which should show an output similar to this:
  ```
  {
   "transparent": "640.0310769",
   "interest": "0.00",
   "private": "0.00",
   "total": "640.0310769"
  }
  ```
- If it's less than 100, you need to get some more before you can create an ID. If you have recently sent funds to your address and that isn't included in this balance, you can wait for the transfer to show up.

- To create an ID, you have to check first if that ID already exists with this command (I'll be using **Demo ID** in this guide):
  `./verus getidentity "Demo ID@"`

  If the identity is not yet registered you get the following output:

  ```
  error code: -5
  error message:
  Identity not found
  ```

  If the identity is already registered, you will be provided with details about that ID
- You can use any character except **\ / : . * ? " < > | @** for your ID
  - And you can not use an empty space as a first character.
  - But don't worry, the system won't let you commit such names, even if you try.
  - What you can use is e.g. an empty space in between, which would allow you to commit a name like "Lucky Joe" (without the quotes of course), but also "Lucky_Joe", "LuckyJoe", and even "Lüçkÿ Jöë" or "幸运 八八八" would work. There is no rule for creating a useful ID, and you can create several IDs if you like. That means, you can create a matching ID for each purpose.

- The command to commit a name for your ID:

```
registernamecommitment "Demo ID" "RFoinkqD2QinMi25mUu5h7k3f4ETaBQWpo" "Verus Coin Foundation@"
```

  - The text between the first pair of quotes is your ID name
  - The text between the second pair of quotes is the address in your wallet that pays the fees to create this ID
  - The text between the third pair of quotes is the optional referral ID, which lowers your purchase price.

  The command above will return you a json formatted message:


```
{
 "txid": "862485a6af4b6989531cf530e5a13b531243c805702de40b0d8086f3527c174c",
 "namereservation": {
 "name": "Demo ID",
 "salt": "739e248f525c8fbbb01c4c480a6c325c10e28ab1a00dc0f81b6583e1b9cab261",
 "referral": "i5v3h9FWVdRFbNHU7DfcpGykQjRaHtMqu7",
 "parent": "",
 "nameid": "iRLC6sRkSACDVRYEoz32iY6kE9ax5xyTyE"
 }
}
```


- The message that the command above returned, contains the information which is needed for the registration of the ID, but the above transaction first has to be mined into the blockchain, so you will have to wait at least 1 block before you can issue the next command. To make sure the transaction is mined into the chain you can check the TXID from that message on the explorer: https://explorer.veruscoin.io

- The command to register your ID:

```
./verus registeridentity '{"txid":
"862485a6af4b6989531cf530e5a13b531243c805702de40b0d8086f3527c174c", "namereservation": { "name":
"Demo ID", "salt": "739e248f525c8fbbb01c4c480a6c325c10e28ab1a00dc0f81b6583e1b9cab261", "referral":
"i5v3h9FWVdRFbNHU7DfcpGykQjRaHtMqu7", "parent": "", "nameid":
"iRLC6sRkSACDVRYEoz32iY6kE9ax5xyTyE"}, "identity":{"name":"Demo ID",
"primaryaddresses":["RFoinkqD2QinMi25mUu5h7k3f4ETaBQWpo"], "minimumsignatures":1,
"privateaddress":
"zs16xdne62g2wl62vq8kmwl03klv4qs5zf3jxmax9vj7f443ymj59ds4qtvj8tp67zvs58fvvpc22n"}}'
```

  - The data shown in red is copied from the response message from the commit command.
  - The data shown in green is your public address that your ID is bound to.
  - The data shown in purple, is your private address, which can be used for private transfers, messaging, voting, polling etcetera.

- After running this command the CLI will respond with a single line containing the TXID:
  de2909a8832a94182037f3ea29f7382b5a2841f505ab20ceb8334e3712127078

- Once this TXID is mined into a block on the blockchain (again: you can use the explorer to check), your ID is registered. You can check the registration using:

```
./verus listidentities
```

The output will show you all identities in your wallet:


```
[
 {
 "identity": {
 "version": 1,
 "flags": 0,
 "primaryaddresses": [
 "RFoinkqD2QinMi25mUu5h7k3f4ETaBQWpo"
 ],
 "minimumsignatures": 1,
 "identityaddress": "iNcmBb1z1pdW2qW4UDhNp1MXg7Rq9Yk8ge",
 "parent": "RQVsJRf98iq8YmRQdehzRcbLGHEx6YfjdH",
 "name": "!",
 "contentmap": {
 },
 "revocationauthority": "iNcmBb1z1pdW2qW4UDhNp1MXg7Rq9Yk8ge",
 "recoveryauthority": "iNcmBb1z1pdW2qW4UDhNp1MXg7Rq9Yk8ge",
 "privateaddress": "zs1sx3yklu4ezq8ht5xnghrjhm4jxs8mcgxaqlky5Oe548mqngv8u8vnptrluq5w9jft2ackpxsh9e"
 },
 "blockheight": 3701,
 "txid": "48ba2a4e1a5e5ee0cada495b268b85679fc43bf573a3ad86aa1e117e37f1b848",
 "status": "active",
 "canspendfor": true,
 "cansignfor": true
 }
]
```

The procedure above creates IDs that have all authorities set to themselves.

- If you want to update your ID after creating, your can use the command **updateidentity**. In our example it would look like this, if I wanted to update the revocation authority and recovery authority to other existing Identities:

```
./verus updateidentity '{"version": 1, "flags": 0, "primaryaddresses":
["RFoinkqD2QinMi25mUu5h7k3f4ETaBQWpo"], "minimumsignatures": 1, "identityaddress":
"iNcmBb1z1pdW2qW4UDhNp1MXg7Rq9Yk8ge", "parent": "RQVsJRf98iq8YmRQdehzRcbLGHEx6YfjdH", "name": "Demo ID",
"contenthashes": [], "revocationauthority": "i5vt1Cx7oDMjfFSNSFpdMFQLoyD9gqDN55", "recoveryauthority":
"iLbcafWjFgvrZ9yGNKofWmBw1DgJyAaVe2", "privateaddress":
"zs1sx3yklu4ezq8ht5xnghrjhm4jxs8mcgxaqlky50e548mqngv8u8vnptrluq5w9jft2ackpxsh9e"}'
```

- The ID addresses in red reflect the changes I made, to assign the authorities to other IDs.
- The `"revocationauthority": "i5vt1Cx7oDMjfFSNSFpdMFQLoyD9gqDN55"` and `"recoveryauthority": "iLbcafWjFgvrZ9yGNKofWmBw1DgJyAaVe2"` can also added to the registration process, to assign these authorities right away. The resulting command would look like this, in our example:

```
./verus registeridentity '{"txid":
"862485a6af4b6989531cf530e5a13b531243c805702de40b0d8086f3527c174c", "namereservation": { "name":
"Demo ID", "salt": "739e248f525c8fbbb01c4c480a6c325c10e28ab1a00dc0f81b6583e1b9cab261", "referral":
"i5v3h9FWVdRFbNHU7DfcpGykQjRaHtMqu7", "parent": "", "nameid":
"iRLC6sRkSACDVRYEoz32iY6kE9ax5xyTyE"}, "identity":{"name":"Demo ID",
"primaryaddresses":["RFoinkqD2QinMi25mUu5h7k3f4ETaBQWpo"], "minimumsignatures":1,
"privateaddress":
"zs16xdne62g2wl62vq8kmwl03klv4qs5zf3jxmax9vj7f443ymj59ds4qtvj8tp67zvs58fvvpc22n",
"revocationauthority": "i5vt1Cx7oDMjfFSNSFpdMFQLoyD9gqDN55", "recoveryauthority":
"iLbcafWjFgvrZ9yGNKofWmBw1DgJyAaVe2"}}'
```

  - The data shown in red is copied from the response message from the commit command.
  - The data shown in green is your public address that your ID is bound to.
  - The data shown in purple, is your private address, which can be used for private transfers, messaging, voting, polling etcetera.
  - The data shown in Dark yellow, are the i-addresses for the revocation and recovery addresses. You can use the ID-name too, instead of the i-addresses.

**) Some notes on IDs

You see that you can enter three IDs, viz. the **main ID**, the **revocation ID**, and the **recovery ID**. Actually they are equal for their own purpose, each of them is an own, normal, independent ID, and can do anything the **main ID** can do on their own. They don't even have to be your IDs. But they just can be assigned to different roles (a.k.a. authority). If you choose to do so, here's what can be controlled by them:

- The **main ID** is the one you would do all transactions with, and other stuff that is still to come. It always has the power to change both of the other IDs.
- The **Revocation ID** can revoke the main ID, but also has control over changing anything that is related to the Revocation ID itself. This means, if you change the Revocation ID to an ID you do not control, you no longer can change the Revocation ID again, except from the wallet that also controls the Revocation ID. This also applies to Recovery ID.
- The **Recovery ID** can of course recover the main ID *when it was revoked*. And (only) in this process, it can also change the *public and private addresses* of the main ID, and it's Revocation ID.
- Both the **Revocation ID** and the **Recovery ID** can be used for more than one main ID, so you only need to create them once. And they can be the same, too, so you can use one other ID both for revocation and recovery.
- **If you want to be able to revoke and recover your main ID, you need at least one more ID for that. The main ID can not revoke itself.** However, if you want to have a totally self-sovereign ID, and control the private and public address it is assigned to in another way, you can use self-assignment.
- **Revocation and Recovery ID(s) should be created before your main ID.** Although there is a way to change them later, it's rather complicated (at time of writing this guide). So if you plan to be able to revoke and recover, it's much easier to create them first.

Now you see what these IDs can control. What they *do not* control, nor revoke or recover, are the underlaying *public (R-) and private (z-) addresses* that you enter on creation. These will always be in control of the wallet that has the private *keys* (a.k.a. WIFs) for them. They will stay unchanged, can't neither be revoked nor recovered. That is only possible for **IDs**.
You will, however, find a new address after creating an ID, which is an *i-address*. This address is tied to an ID, and only exists together with a valid ID. It does not have a private *key*, and can not be restored other than by the associated ID.
For the experienced user it may even be an option to create an ID (e.g., for revocation or recovery) on a new wallet, with a separate wallet.dat file. The wallet.dat can be saved and secured offline, so it is at hand when needed. But you only should do so if you're really sure what you are doing. If you would like to have some support with this, feel free to ask in our Discord https://discord.gg/VRKMP2S.

Now, after you know a bit more about IDs, you can decide how many and which IDs you want to create.

Finally, a few notes on referrals.

When you enter a referral ID, purchasing an ID will cost 80 instead of 100 VRSC (not including transaction fees). At the same time, the referral ID will get 20 VRSC.

Furthermore, when someone is using your ID as a referral, you will get 20, and the referral you were using will again get 20 VRSC (but purchase will not get cheaper than 80 VRSC).

This referral propagation will work up to the 3rd generation, i.e., up to the referrer of the referrer of the referrer. Each of them will have 20 VRSC. But at least 20 VRSC will always go back to the miners and stakers of Verus Coin, and will be added to one of the coming blocks. That's also why mining and staking will become more profitable with each ID that was created.

Thanks for reading, and don't hesitate to ask in our discord if you need support! :)

veruscoin.io