**Compiling XODE**

**Setting up your Github environment**

Check Existing SSH Keys: Ensure that you have an existing SSH key. Run the following command to list your existing keys:

$ ls -al ~/.ssh

**Look for files named id\_rsa or id\_rsa.pub.**

Generate a New SSH Key (If Needed): If you don't have an SSH key or want to generate a new one, you can use the following command:

$ ssh-keygen -t rsa -b 4096 -C "your\_email@example.com"

**Follow the prompts to generate a new key. Make sure to replace "your\_email@example.com" with your actual email address.**

**Add SSH Key to SSH Agent: Start the SSH agent and add your private key:**

$ eval "$(ssh-agent -s)"

$ ssh-add ~/.ssh/id\_rsa

**Copy the SSH Key to Clipboard: Use the following command to copy your SSH key to the clipboard:**

$ cat ~/.ssh/id\_rsa.pub

Add SSH Key to GitHub:

Go to your GitHub account settings.

Navigate to "SSH and GPG keys" or "SSH keys."

Click on "New SSH key" or "Add SSH key."

Paste your key into the provided field and give it a title.

**Clone Xode-Blockchain**

$ git clone git@github.com:Blockspace-Corporation/xode-blockchain.git

$ cd xode-blockchain

**Setting up your Rust environment**

**Install the build-essential package.**

$ sudo apt install build-essential

**Install support packages for Rust**

$ sudo apt install --assume-yes git clang curl libssl-dev protobuf-compiler

$ sudo apt install --assume-yes git clang curl libssl-dev llvm libudev-dev make protobuf-compiler

**Download the rustup installation program and use it to install Rust by running the following command.**

$ curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh

**Update your current shell to include Cargo by running the following command**

$ source $HOME/.cargo/env

**Configure the Rust toolchain to default to the latest stable version by running the following commands**

$ rustup default stable

$ rustup update

**Add the nightly release and the nightly WebAssembly (wasm) targets to your development environment by running the following commands:**

$ rustup update nightly

$ rustup target add wasm32-unknown-unknown --toolchain nightly

$ rustup default nightly-x86\_64-unknown-linux-gnu

**Verify the configuration of your development environment by running the following command.**

$ rustup show

$ rustup +nightly show

**Install WASM support for smart contracts**

$ rustup target add wasm32-unknown-unknown

**Compile Xode and run the node**

$ cargo build

**Compiling for Raspberry Pi (aarch64) environment**

**Install cross crate**

$ cargo install cross

**Install Docker**

$ sudo apt update

**Install a few prerequisite packages which let apt use packages**

$ sudo apt install apt-transport-https ca-certificates curl software-properties-common

**Add the GPG key for the official Docker repository to your system**

$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg

**Add the Docker repository to APT sources**

$ echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu $(lsb\_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

**Update again the packes**

$ sudo apt update

**Make sure you are about to install from the Docker repo**

$ apt-cache policy docker-ce

**Install docker**

$ sudo apt install docker-ce

**Check docker status**

$ sudo systemctl status docker

**Make docker usable without having to use sudo**

$ sudo usermod -aG docker ${USER}

$ su - ${USER}

$ groups

**Now run cross compile**

$ cross +nightly build --target aarch64-unknown-linux-gnu --release

**Running XODE using Zombienet**

**Test of the xode-node binary is running and fully compiled**

$ ./target/release/xode-node –version

**Download the Polkadot binary (make sure you are in the zombienet directory with yout download the this binary)**

$ wget https://github.com/paritytech/polkadot/releases/download/v1.0.0/polkadot

**Download the Zombienet binary (make sure you are in the zombienet directory with yout download the this binary)**

$ wget https://github.com/paritytech/zombienet/releases/download/v1.3.109/zombienet-linux-x64

**Make the two binaries executable**

$ chmod +x polkadot

$ chmod +x zombienet-linux-x64

**Run XODE using Zombienet**

$ ./zombienet-launch.sh