Telegram Crypto Trading Bot Deployment Guide

For Ionos Cloud Server (Ubuntu 18.04)

This guide will walk you through deploying your Telegram crypto trading bot on an Ionos cloud server running Ubuntu 18.04. The instructions are designed to be easy to follow even if you have no programming experience.

Table of Contents

1. Connecting to Your Ionos Server via SSH
2. Setting Up the Python Environment
3. Installing Required Dependencies
4. Transferring the Bot Code to the Server
5. Configuring the Bot with API Keys
6. Setting Up the Bot for 24/7 Operation
7. Monitoring and Maintaining the Bot
8. Troubleshooting

1. Connecting to Your Ionos Server via SSH

For Windows Users:

1. **Download and Install PuTTY**:
    - Download PuTTY from here
    - Install it by following the on-screen instructions
2. **Connect to Your Server**:
    - Open PuTTY
    - Enter your server's IP address in the "Host Name" field
    - Keep the port as 22
    - Click "Open"
    - When prompted, enter your username (usually "root" or the username provided by Ionos)
    - Enter your password when asked

For Mac/Linux Users:

1. **Open Terminal** and use the SSH command:

```
ssh username@your_server_ip
```

Replace `username` with your server username (usually "root" or the username provided by Ionos) and `your_server_ip` with your server's IP address.

Setting Up SSH Keys (Optional but Recommended):

SSH keys provide a more secure way to log in than using a password.

On Windows (using PuTTY):

1. **Generate SSH Keys**:
    - Download and open PuTTYgen (comes with PuTTY)
    - Click "Generate" and move your mouse randomly
    - Set a passphrase (optional but recommended)
    - Click "Save private key" to save it to your computer
    - Copy the public key text from the top box
2. **Add the Public Key to Your Server**:
    - Log in to your server using PuTTY
    - Create or edit the authorized_keys file:

    ```
    mkdir -p ~/.ssh
    chmod 700 ~/.ssh
    nano ~/.ssh/authorized_keys
    ```

    - Paste your public key and press Ctrl+X, then Y to save
    - Set proper permissions:

    ```
    chmod 600 ~/.ssh/authorized_keys
    ```

On Mac/Linux:

1. **Generate SSH Keys**:

```
ssh-keygen -t rsa -b 4096
```

2. **Copy the Key to Your Server**:

```
ssh-copy-id username@your_server_ip
```

2. Setting Up the Python Environment

Our crypto trading bot requires Python 3.8 or newer. Ubuntu 18.04 comes with Python 3.6 by default, so we'll need to install a newer version.

1. **Update Your System**:

```
sudo apt update
sudo apt upgrade -y
```

2. **Install Required Packages for Python Installation**:

```
sudo apt install -y software-properties-common build-essential libssl-dev libffi-dev python3
```

3. **Add the Python PPA and Install Python 3.10**:

```
sudo add-apt-repository ppa:deadsnakes/ppa -y
sudo apt update
sudo apt install -y python3.10 python3.10-venv python3.10-dev python3-pip
```

4. **Verify Python Installation**:

```
python3.10 --version
```

You should see output showing Python 3.10.x

5. **Create a Virtual Environment**:

```
mkdir -p ~/telegram_crypto_bot
cd ~/telegram_crypto_bot
python3.10 -m venv venv
```

6. **Activate the Virtual Environment**:

```
source venv/bin/activate
```

Your command prompt should now show `(venv)` at the beginning, indicating the virtual environment is active.

3. Installing Required Dependencies

Now let's install the necessary system and Python packages for the bot.

1. **Install System Dependencies**:

```
sudo apt install -y git curl wget nodejs npm
```

2. **Install Node.js and PM2** (for keeping the bot running 24/7):

```
curl -sL https://deb.nodesource.com/setup_16.x | sudo -E bash -
sudo apt install -y nodejs
sudo npm install pm2 -g
```

3. **Install Python Dependencies** (make sure your virtual environment is activated):

```
pip install --upgrade pip
pip install python-telegram-bot ccxt python-dotenv requests pandas numpy matplotlib websocke
```

This installs the common packages needed for crypto trading bots. If your bot has a `requirements.txt` file, you'll use that instead in step 4 of the next section.

4. Transferring the Bot Code to the Server

You have several options to transfer your bot code to the server:

Option 1: Using Git (if your code is in a Git repository)

```
cd ~/telegram_crypto_bot
git clone https://your-repository-url.git bot
cd bot
```

Replace `https://your-repository-url.git` with your actual Git repository URL.

Option 2: Using SCP (from Mac/Linux)

On your local machine (not the server), open a terminal and run:

```
scp -r /path/to/your/bot/code username@your_server_ip:~/telegram_crypto_bot/bot
```

Replace `/path/to/your/bot/code` with the path to your bot code on your local machine.

Option 3: Using WinSCP (from Windows)

1. **Download and Install WinSCP** from here
2. **Connect to Your Server**:
   - Open WinSCP
   - Enter your server details (host, username, password)
   - Click "Login"
3. **Transfer Files**:
   - Navigate to your bot code on your local machine (left panel)
   - Navigate to `~/telegram_crypto_bot` on the server (right panel)
   - Create a new folder called "bot" on the server
   - Drag and drop your bot files from the left panel to the right panel

After Transferring the Code

If your bot has a `requirements.txt` file, install the specific dependencies:

```
cd ~/telegram_crypto_bot/bot
pip install -r requirements.txt
```

5. Configuring the Bot with API Keys

Most crypto trading bots require API keys for Telegram and various exchanges (BTCC, Coinbase, Photon-SOL).

1. **Create or Edit the Configuration File**:

```
cd ~/telegram_crypto_bot/bot
nano .env
```

2. **Add Your API Keys**:

```
# Telegram Bot Token (from BotFather)
TELEGRAM_BOT_TOKEN=your_telegram_bot_token

# Exchange API Keys
BTCC_API_KEY=your_btcc_api_key
BTCC_API_SECRET=your_btcc_api_secret

COINBASE_API_KEY=your_coinbase_api_key
COINBASE_API_SECRET=your_coinbase_api_secret

PHOTON_SOL_API_KEY=your_photon_sol_api_key
PHOTON_SOL_API_SECRET=your_photon_sol_api_secret

# Other Configuration
```

```
USER_ID=your_telegram_user_id
```

Replace all the `your_*` values with your actual API keys and information.

3. **Save the File**: Press `Ctrl+X`, then `Y`, then `Enter` to save and exit.
4. **Set Proper Permissions** (for security):

```
chmod 600 .env
```

6. Setting Up the Bot for 24/7 Operation

We'll use PM2 to keep your bot running continuously, even after you log out or if the bot crashes.

1. **Identify Your Bot's Main File**: Look for the main Python file that starts your bot (often named `bot.py`, `main.py`, or similar).
2. **Create a PM2 Startup Script**:

```
cd ~/telegram_crypto_bot
nano start_bot.sh
```

3. **Add the Following Content**:

```
#!/bin/bash
cd ~/telegram_crypto_bot/bot
source ../venv/bin/activate
python3 main.py
```

Replace `main.py` with the actual name of your bot's main file.

4. **Make the Script Executable**:

```
chmod +x start_bot.sh
```

5. **Start the Bot with PM2**:

```
pm2 start ~/telegram_crypto_bot/start_bot.sh --name crypto-bot
```

6. **Configure PM2 to Start on System Boot**:

```
pm2 save
pm2 startup
```

Copy the command that PM2 outputs and run it. It will look something like:

```
sudo env PATH=$PATH:/usr/bin /usr/lib/node_modules/pm2/bin/pm2 startup systemd -u your_usern
```

7. Monitoring and Maintaining the Bot

Checking Bot Status

1. **View All Running Processes**:

```
pm2 list
```

2. **View Detailed Information About Your Bot**:

```
pm2 info crypto-bot
```

3. **View Bot Logs**:

```
pm2 logs crypto-bot
```

To see only the last 200 lines:

```
pm2 logs crypto-bot --lines 200
```

4. **Monitor CPU and Memory Usage**:

```
pm2 monit
```

Press `Ctrl+C` to exit the monitor.

Managing the Bot

1. **Restart the Bot**:

```
pm2 restart crypto-bot
```

2. **Stop the Bot**:

```
pm2 stop crypto-bot
```

3. **Start the Bot (if stopped)**:

```
pm2 start crypto-bot
```

4. **Remove the Bot from PM2**:

```
pm2 delete crypto-bot
```

Updating the Bot

When you need to update your bot:

1. **Stop the Bot**:

```
pm2 stop crypto-bot
```

2. **Update the Code** (using git pull or by uploading new files)

```
cd ~/telegram_crypto_bot/bot
git pull
```

Or upload new files using SCP or WinSCP as described earlier.

3. **Install Any New Dependencies**:

```
source ~/telegram_crypto_bot/venv/bin/activate
pip install -r requirements.txt
```

4. **Restart the Bot**:

```
pm2 restart crypto-bot
```

8. Troubleshooting

Common Issues and Solutions

Bot Crashes Immediately

1. **Check the Logs**:

```
pm2 logs crypto-bot
```

2. **Common Causes and Solutions**:
    - **Missing Dependencies**: Make sure all required packages are installed
      ```
      source ~/telegram_crypto_bot/venv/bin/activate
      pip install -r requirements.txt
      ```
    - **Configuration Issues**: Verify your `.env` file has all required API keys
    - **Permission Issues**: Check file permissions
      ```
      chmod 600 .env
      chmod +x start_bot.sh
      ```

Connection Issues with Exchanges

1. **Check Internet Connectivity**:

```
ping google.com
```

2. **Verify API Keys**: Double-check that your API keys are correct and have the necessary permissions.
3. **Check Exchange Status**: Visit the exchange's status page to see if they're experiencing issues.

Server Resource Issues

1. **Check System Resources**:

```
top
```

2. **Check Disk Space**:

```
df -h
```

3. **Monitor Memory Usage**:

```
free -m
```

Ionos-Specific Troubleshooting

Accessing the Ionos Control Panel

1. Log in to your Ionos account at https://www.ionos.com/
2. Navigate to the "Server & Cloud" section
3. Select your server to access its control panel

Rebooting Your Server from the Control Panel If you can't access your server via SSH:

1. In the Ionos control panel, find your server
2. Click on the "Restart" or "Reboot" option
3. Confirm the action

Using the Ionos Rescue System If your server has serious issues:

1. In the Ionos control panel, find your server
2. Select "Rescue" or "Recovery" option
3. Follow the instructions to boot into rescue mode
4. Once in rescue mode, you can fix file system issues or reset passwords

Firewall Issues Ionos may have a firewall that blocks certain connections:

1. Check if the firewall is blocking connections:

```
sudo ufw status
```

2. If needed, allow specific ports:

```
sudo ufw allow 22/tcp  # For SSH
```

Getting Additional Help

If you're still experiencing issues:

1. **Contact Ionos Support**: They can help with server-specific issues
2. **Check Bot Documentation**: Refer to your bot's documentation for specific troubleshooting steps
3. **Search Online Forums**: Platforms like Stack Overflow or Reddit may have solutions to common issues

Conclusion

Congratulations! Your Telegram crypto trading bot should now be running 24/7 on your Ionos cloud server. Remember to regularly check the logs and monitor the bot's performance to ensure everything is working correctly.

For security reasons, consider: - Regularly updating your server with `sudo apt update && sudo apt upgrade` - Setting up a firewall with `ufw` - Implementing fail2ban to prevent brute force attacks

Happy trading!