# **Step 1: Download and Install IPFS**

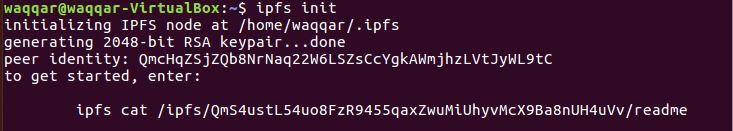
# **Prerequisites: -**

1. **Go** must be preinstalled.
2. We need to have **at least TWO Nodes** to test this.

* Visit <https://dist.ipfs.io/#go-ipfs> and download **go-ipfs.**
* After downloading, untar the archive, and move the ipfs binary somewhere in your executables $PATH using the install.sh script:
* tar xvfz go-ipfs.tar.gz (tar xvfz <filename.extension>)
* cd go-ipfs
* sudo ./install.sh

# **Step 2: Initialize the nodes**

* **ipfs init**



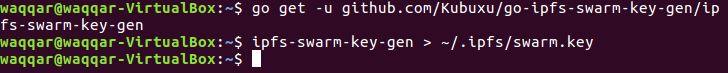
# **Step 3: Creating a Private Network**

Run the following command to install the swarm.key generation utility **(only one Node 1):**

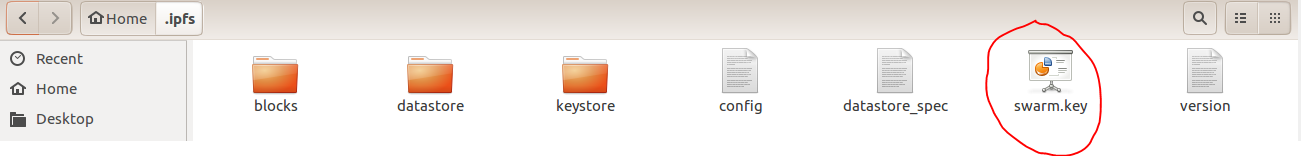
* **go get -u github.com/Kubuxu/go-ipfs-swarm-key-gen/ipfs-swarm-key-gen**

Now run this utility in **one of your node** like this:

* **ipfs-swarm-key-gen > ~/.ipfs/swarm.key**



Copy the file generated to the IPFS directory of each node **(in this case only Node 2).**

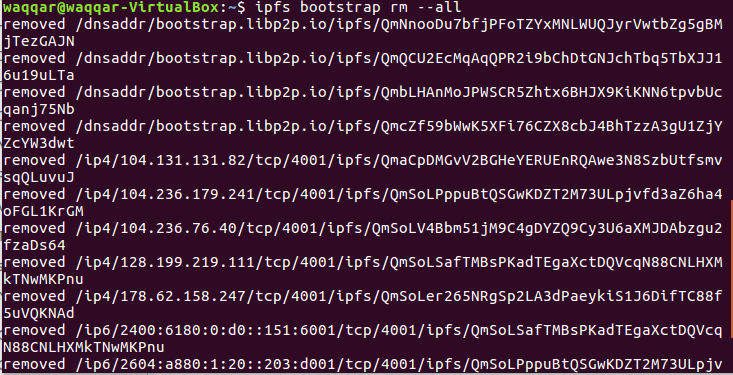


By default, the IPFS directory will be at **$HOME/.ipfs**

# **Step 4: Bootstraping IPFS Node**

First, to prevent your node from even trying to connect to the default bootstrap nodes, run:

* **ipfs bootstrap rm --all**



We have to do this on **both nodes**.

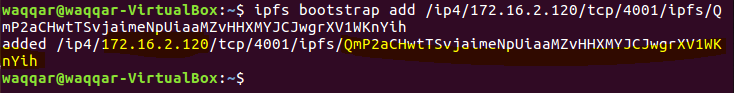
Then add your own bootstrap peers with:

* **ipfs bootstrap add <multiaddr>**

*For example:*

* *ipfs bootstrap add /ip4/<IP Address>/tcp/4001/ipfs/<Peer ID>*

*Replace <IP Address> and <Peer ID> with the IP Address and Peer ID respectively of the node you want to connect.*



This is how we bootstrap **Node 2 in Node 1**. We must do the same on Node 2, by replacing the **IP address and Peer ID** with that of Node 1’s (highlighted).

The bootstrap node gets added to the **config** file present in the IPFS Path of **each node** (~./ipfs) 

To be extra cautious, you can also set the **LIBP2P\_FORCE\_PNET** environment variable to 1 to force the usage of private networks. If no private network is configured, the daemon will **fail to start.**

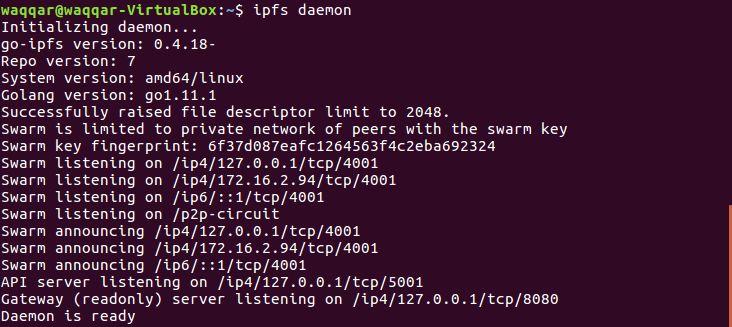


We should do this **both the Nodes.**

# **Step 5: Start the node and test**

Start the IPFS Daemon by running the following command **on each node:**

* **ipfs daemon**



Test if your private network by checking the swarm peers you are connected to by executing the following command:

* **ipfs swarm peers**

If everything goes well, then you should see a list of peers you are connected to in your Private IPFS Network.

On **Node 1,**

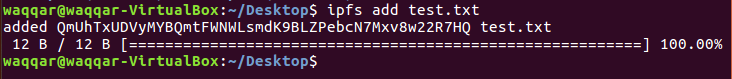


Consequently, Node 1 recognizes Node 2 as its bootstrap node and vice versa.

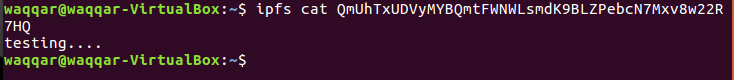
On **Node 2,**



File uploaded through **Node 1**: -



File retrieved through **Node 2**: -



# **Points to remember**

* Only **one swarm.key** can be generated by a node which is necessary to facilitate private network.
* If we don’t use the **swarm.key** then any node will be able to connect to our network **without permission.**
* Private Network is still an **Experimental** Feature of IPFS.
* Pubsub is also **experimental and disabled** by default.
* In a private network, the nodes must **pin** the file created/fetched. Otherwise on garbage collection, the file will be **lost**.
* Apparently, maximum Block Size is **262.2KB**.
* There are 4 basic types of pins:  
  \* **Direct** - Objects pinned directly  
  \* **Recursive** - Roots of recursive pins (like direct, but also pin the children of the object)  
  \* **Indirect** - Objects referenced by recursive pins  
  \* **Internal** - internal, most of the time you don't see those
* By default, **Garbage Collection** takes place every **1 Hour**.
* Because of legality and undesirability, a file uploaded by any provider **doesn’t get automatically distributed**. It is such that only when a requester node requests for that particular file, IPFS fetches it from the provider node.

# **Reference**

<https://www.techulator.com/resources/17466-Step-by-step-guide-to-deploy-a-private-IPFS-cluster-on-Ubuntu.aspx>