**SingularityNET(AGI)**

De

Ever wonder what will happen if there is a creation of an open marketplace for Artificial intelligence where everything can learn from itself

SingularityNET is building the world’s first decentralised AI as a Service Platform

the world 1st Decentralised market place for AI

its a new Ecosystem for AI agents to exchange data, create service and collaborate between themselves

any potential customer can simply browse the singularityNET catalog of AI service or they can request their specific needs to the SingularityNET

SingularityNET is a combination of

* OpenSource Codes
* Block Chain – more geared toward financial applications
* Smart Contracts
* AGI Token – Specially designed cryptocurrency

What is Smart Contracts

SingularityNET (AGI) lets anyone create, share, and monetize AI services at scale. AI is projected to create **16** **trillion** USD of global GDP value by the year 2030, meaning SingularityNET, at its current valuation, has massive upside potential.

**The Goal of SingularityNET is to address 4 main challenges of AI that exist currently**

1. There are many AI solutions or systems that do specialised tasks vey well, but there is No Artificial General Intelligence or AGI.

So SingulairtyNET wants to develop AGI

1. AI is currently quite fragmented or separated; each Specific system exists in its own world and there is not really a way for them to communicate and work together easily.

So AI Service integration is another big challenges and SingularityNET wants to face and solve this challenge

1. AI systems are overwhelmingly built by and for huge tech companies or government or military applications. But many other potential AI applications that can benefit society could be developed, but they are not being realized(not getting awareness) due to the “Socioeconomic nature of the AI industry”.

But SingularityNET want to change this one, the AI systems developed does not need to be only the system that can solve the problem of companies or government, it has solve also the problem of those ordinary people.

1. Massive companies like IBM, Google and others build AI sytems for their own benefit, means after all that companies will get a money from this systems. So the system they built is controlled by them or Centrally.

But the SingularityNET want to change this one, that AI Systems did not to be centrally controlled by that companies only. It has to be Decentralized.

Also by the developed AI systems only that companies will be richer and richer but in SingularityNET everyone who collaboretes in that Service will get a payment through a Block chain smart contracts and payment is shared to all just according to their work.

So by SingularityNET ordinary people and smaller businesses can no benefits rather than just the single company only.

SingularityNET recently announced the launch of the mainnet Beta Version of its platform on the **Ethereum Network**

**What is Ethereum Network ?**

* Its an open-source, public, blockchain based distributed computing platform and operating system featuring smart contract(scripting) functionality
* Ethereum provides a decentralized virtual machine
* like Bitcoin, Ethereum is a distributed public block chain, but the difference is bitcoin is nothing just currency but ethereum is far more than that

**SingularityNET**

AI + blockchain in pursuit of maximizing NET benefits for participants and all Life

1. Blockchain for smarter, safer AI
2. AI powered smart transactions
3. AI marketplace
4. Block chain data protection
5. Token sale

What is a difference between Opencog and SingularityNET

**SingularityNET**

1 Overview

2 Concepts and Components

3 SingularityNet Foundations

4 Keywords

3 SingularityNet Platforms

* Dapp(Decentralized AI MarketPlace, Decentralized Applications)
* snet-cli
* snet-daemon
* snet-sdk
* snet-devportal

4 SingularityNet Platform Contracts

3 setup a singularityNET

4 setting up Snet-Sdk-Python

**Overview**

its an open source and decentralized network of AI services made accessible through blockchain

developers publish their services to the SNET network, where they can be used by anyone with an internet connection

developers able to charge for the use of their services using the native AGI token

these services can provide different domains such as

image/video, speech, text, time-series, bio-AI, network analysis, etc

developers can also deploy autonomous AI agents that interoperate with other services on the network

Knowledgeable people are realizing that the next few decades will see a transition to a new society and economy in which machine intelligence is the dominant factor

SingularityNET is not only a collection of AIs Service, but its a market.

The path to creating a positive “global brain” is challenging.

A technological singularity could have unprecedented benefits but also unprecedented risk.

The popular press is full of dire warnings about the dangers of artificial general intelligence

among the challenges of SingularityNET

* current set of protocols for collective action
* today’s financial mechanisms and instututions

**Blockchain**, with its natively digital money, is a powerful tool for managing **transactions** in an economy dominated by machine intelligence.

However, Blockchain is just a tool; there are important decisions to be made about how to use it.

At its core, SingularityNET is a set of smart contract templates that AI agents can use

1. to request that AI work be done
2. to exchange data and
3. to supply the results of AI work

SingularityNET is designed both to be highly valuable now and to lay the groundwork for the emergence of a self-modifying, decentralized “artificial cognitive organism”.

SingularityNET provides an automated process that enables any businness to connect existing AI tools to build solution it needs.

Many state of the art AI tools exist only in Github repositoreis created by graduate students or independent researchers. The latest algorithims for image and video analysis, machine translation, automated theorem proving, bioinformatics data analysis, etc are typically avialable on Github, but the friction inherent in installing, configuring and running them limits their use.

Most AI developers are academics, not businness people, and have no easily accessible markteplace to turn to inorder to monetize their clever AI code. As a result, the AI in real-world products tends to lag months to years behind the code.

SingularityNET is a launchpad where developers can quickly deploy their AI models and algorithims into real world applications.

Machine learning tools also require datasets of sufficient size. Creating and managing such large datasets are beyond the means and capabilities of most AI developers, and the closed development model that currently prevails makes it hard for developers to share datasets.

SingularityNET connects these AI tools and datasets to the marketplace, making them accessible to end users and developers. Is a sharing-economy marketplace for AI, so its an open network.

Like Uber and Airbnb, we have identified a large unexploited resource and a large market in need of that resource, and we are launching a tool to connect the two.

The apartments in AirBnBs network do not combine to become meta-apartments, nor

does Uber’s network create meta-cars, but AIs in SingularityNETs network come together to form meta-AIs whose intelligence is more than the sums of their parts.

**Robust and Adaptive Software Architecture**

SingualrityNET is a distributed computing architecture for making new kinds of smart contracts to facilitate market interactions with AI and machine learning tools.

The following design principles are incorporated throught the design

* Interoperability → the network will be interface with multiple blockchain
* Data Sovereignty and Privacy → the network includes user-side controls for sharing personal data. Users remain in control of their data and can share it with the network via smart contracts.
* Modularity → flexible network capabilities make it possible to create custom topologies, AI agent collaborations and failure recovery methods
* Scalability → SNET will securely host both private and public contracts

in SingularityNET services are accessed with AGI token. Token holders can use their tokens to purchase services in the marketplace.

SingulairtyNET Agents can run in the

* Clouds
* on Phones
* Robots and
* Embedded devices

SNET platform contains a number of critical components that work together to enable a decentralized network of AI services.

The core components are designed to allow for a functional, scalable, and extensible system.

SNET developers arrive at current architecture through a careful process

* guided by a few key decisions governing blockchain integration
* AI service integration
* by the goal of building an AI marketplace that is both open & complaint with regulatory and legal requirements
* first, we made the conscious choice to minimize our dependence on our current blockchain and ethereum
* 2nd, on AI services integration, we wanted to abstract away as much of the network as possible, in order to reduce the learning curve and minimize the overhead(indirect cost or expense) associated with providing AI services via the network.

→ the abstraction of the network is achieved with a single flexible tool called **Daemon**, that will help us provide scalability, robustness, distribution, and management features to the entire community

* finally, to make our marketplace compliant with regulations without compromising on openness, we implemented it separately from our fully decentralized registry of AI services currently available on the blockchain.

The price for a single call of Service in SingularityNET is equal to 10^8AGI which means 1COG

AGI token

COG → cost of good sold

**Components of SingularityNET**

the components of SNET platform divided into

* core components and
* auxiliary componets

**Core Components of SNET**

1. **SNET** **marketplace** → its a Dapp(Distributed App) and it provides a front-end for exploring AI services available on the network
   * Users can interact with and call them through a web interface, and rate them afterwards.
2. **Service →** its published to the SingularityNET network and provides a grpc based API for calling it.
   * the service API specification, the IP address where to find the service and pricing information is published to IPFS as **Service** **Metadata**
   * People need to follow the SNET **Naming** **Standards** guidelines
   * **IPFS**
     + Interplanetary File System
     + its a versioned file system which can store files and track versions over time, very much likRopstene, Git
     + it also defines how files move across a network, making it a distributed file system, much like BitTorrent.
     + IPFS and blockchains can work well together, because of their similarity
3. **Software**
   * **SNET-CLI** (SNET Command Line Interface)
     + it help us to interact with the SNET platform
     + it allows crucial tasks such as
       - calling and querying services
       - registering and managing identities
       - publishing services
       - updating registration information
       - notifying the platform of new endpoints
       - managing payment channels and balances

* **SDK**
  + the SNET SDK helps you to integrate SNET Services with your own software
* **SNET Daemon**
  + a developer exposes their service to the network by running the SNET Daemon alongside their service
  + it interacts with the blockchain to facilitate authorization(heyyama) and payment for services
  + it acts as a passthrough for making API calls to the Service
  + it isolates the payment and blockchain interaction so a developer can focus on deploying and improving their service.
    - Daemon API
    - Daemon Channel Storage

**4. Blockchain Contracts**

* **AGI Token**
  + its a Utility token to be exchanged for AI service
  + its an ERC20 token hosted on Ethereum

What is a difference between Coin, Token and Cryptocurrency?

* + Many people erroneously use these words interchangeably, but they are different from each other
  + Cryptocurrencies are digital or virtual currencies that are encrypted(secured) using cryptography. Cryptography refers to the use of encryption techniques to secure and verify the transfer of transactions.
    - Its not controlled by any central authority, the decentralization nature of the blockchain gives the meaning to cryptocurrencies.
  + Cryptocurrencies is divided into two categories **tokens** and **coins**
    - Coin is a form of digital money created through encryption techniques which store values. They may be sent, received or mined, but donot perform any functions beyond acting as money. There are different types of coins like Bitcoin, Litecoin, NameCoin
    - Tokens are digital assets, issued by a blockchain based project, which is used as a payment method inside of its ecosystem, performing similar functions with coins. But it also gives the holder a right to participate in the network means to be a company’s share.

e.g you can think a token as a concert ticket. Only the one who has the ticket can participate in the concert, even if you have a money but if you do not have ticket you cant participate on the concert. And this ticket has valuable on this concert only but your money is independent and has valuable everywhere.

Like this the token is specific to a given project and valuable inside that project only

→ there are two types of tokens

* + - * **Security** **tokens** → designed to be the company’s share
      * **Utility** **tokens →** have certain use cases inside the project

→ Creating a token is much easier than creating a coin, as you don’t have to create new code or modify already existing ones. All you have to do is just using a standard template from platforms like **Ethereum**, which is a block chain

based and allow anyone to create tokens in just a few steps.

* **Registry**
  + registry means a place where registers or records are kept
  + services are published to a publicly accessible central registry on the blockchain
  + the registry maintains a list of active services on the network and where to find a services corresponding metadata
  + the registry has support for grouping services by organisation or team, along with access control for organisation members
* **Escrow**
  + escrow is a way of transfering property from the seller and money from the buyer through use of a third party which is neutral.
  + The escrow contract on the blockchain holds AGI funds in escrow during interaction between an end-user and a service. An end-user places funds in escrow before a service can be called, and remain there until the service has been delivered or the escrow funds timeout.
    - MPE – Multi Party Escrow
    - MPE stateless Client

**5. The Request for AI Portal(RFAI)**

* + lets Users make requests for AI services that they would like to see built and deployed onto the SingularityNET network and give AGI tokens as a reward for the developer who develop that service.
  + The RFAI portal fosters(encourage) the community by enabling Users to incentivize(motivate) developers to publish services

**SingularityNet Foundation**

* Decentralized Platform for AI agents
* Powered by AGI utility token
* perform services for external customers or for other AI agents
* monetize(to convert services into a form of currency) AI creations
  + **e.g** many different AI services are found in the github and you can use them freely but what about the AI service developer want it to change into a form of currency or buying and selling, Snet is for this. Just developing a market place like merkato and then anyone who wants can provide his own service in that markets and gets money from a person who use his service.
* A society and economy of AIs
* Network governance and economic logic
* Customers can pay in USD, Euro, Yen, Won or other means.

**Keywords**

* there are a couple of words that we would mention every times, so let get them out of the way now.
* **Service**
  + the AI service itself, its just any functional program
  + Service needs to have
    - Service type(API Protocols) → to define input and output, and to define endpoints
      * + grpc based approach or
        + jsonrpc based approach or
        + process based approach
        + REST based approach
        + in alpha days we were using jsonrpc. its one of the primary ways people do interact with ethereum based platforms also
        + currently we are using grpc based approach
    - Service encoding → to serialize and deserialize data and send data over the network.
      * + protobuf
        + json
        + xml
        + currently we are using protobuf file for the service encoding. But during our Alpha version we had using json.
* **Service Description**
  + describes about the structure of inputs and outputs data
  + what is the input format data
  + what is the structure of the response
  + how to represent this input and output programmatically

e.g if we do emotion recognition, what expect

* + - * inputis image and
      * output is faces detected and their emotion.
  + But the problem is how we represent these real input programmatically
    - for this we use **google protobufs**

e.g

message Person {

required string name = 1;

required int32 id = 2;

optional string email = 3;

}

* **Platform**
  + this is the Infrasructure that manages the service. In terms of
    - Payment Mechanism (Blockchain)
    - Rating of the Service
    - The Scale of the Service
    - The Marketplace API
    - etc
* **Testnet**
  + we use simulation of the etherum platform, Not the Mainnet
    - Mainnet is where real transactions are mined and written to the block chain
    - Testnet Simulation like Ropsten and Kovan are used to enable developers work on top of while developing their applications. Transactions in this layer are meaningless
    - there are around 4 testnet Ropsten, Kovan, Rinkeby, Goerili
    - also you can use Localhos 8545 or you can create your own Custom RPC
* **Endpoints**
  + endpoints is just means the address of the service means from where we can get that service, the IP-address and Port of the Service.

**SingularityNet Platforms**

* **Dapp(Decentralized AI Marketplace)**
  + this is the place where we start interacting with the SingularityNET
  + check the beta version of singularitynet https://beta.singularitynet.io
* **Snet-Cli**
* **Snet-Daemon**
  + this is the tool that works all the backend work
  + its the tool that resides between the Blockchain and the Service’s endpoint
  + it takes requests from user and validates it, then if everything is ok, then it passes all this requests to the Service.
  + Then Service process the request that it received from Daemon and give back a response to Daemon, then Daemon delivers the response to the User.
  + Snet-daemon is the engine that is going to be connected with the client code that enable the AI service interact with the blockchain to enable authentication, fund reserve and scaling of the service.
  + The Daemon interacts with the
    - Registry → to get the service metadata and
    - IPFS → to get the Protobuf API
    - ETCD → to store all signed payments from users requests
  + Take an example of real scenarios in hotel, when you go into the hotel to get some service
    - the Receptionist(astanagaji) came and take your requests and validate your requests, then he/she pass your requests to the chefu, then chefu process your requested food and give it to receptionist, then receptionist delivers the food to you. So
    - Snet-Daemon is like this Receptionist or just taqotaatarii and
    - Service is like the food.
    - The Snet-Cli is the place where you can order that food so its just like a Cafe
    - in Snet you can Order the service by using three things,
      * + by Snet-Cli( Clients commands)
        + by Snet-SDK(by using the SDK)
        + by DApp( by using the web page of the beta singularity net)
    - similarily you can order the food
      * + by going to cafe (like the web form or clients commands)
        + to your workplace or to your home (like to your application, Snet-SDK)
        + through another person (through another service, or SDK)
* **Snet-Sdk**
* **Snet dev portal**
  + this is the Resources Page, just to give a more detailed overview of the platform
  + it gives a step by step tutorials to building a service in the numerous languages and how to connect it with the various tools available.

**SingularityNet Platform Contracts**

Contracts(wuli, ragaa) is a legally agreement between two persons or parties typically involves the exchange of goods, services, money or pomises of any of those.

**Tokens**

* ERC-20 is a technical standard used for smart contracts on the Ethereum blockchain for impelementing tokens
* we have our own token called AGI Token

**Registry**

* its the smart contracts used for registering or keeping infromation about the service, the token and various assorted information to make a viable call to the endpoint. Namely
  + Name and Description
  + IPFS hash to Protobuf file(API)
    - protobuf – is a protocol buffers used in making programs to communicate with each other over a wire or used for storing data.
  + Endpoints
  + Price (AGI)
  + Payment Address.

**Multiparty** **Escrow**(**MPE**)

* coupled with our atomic unidirectional payment channels,
* it enables scalable payments in the platform by minimizing the number of on-blockchain transactions needed between clients and AI service owners

Other parts of the infrastructure we won’t cover here

* the **Reputation** **System**
  + how do we make sure the services that are in the marketplace are as reputable(amanama, trusted) as possible based on mechanisms that avoid deceit
  + most of the time in the real market place there are some fake services or products and its difficult to make sure that they are really applicable products similarly here in SingularityNet this may also happen
* **DAIA**(Decentralized AI Alliance) [link](https://medium.com/daia)
  + a collection of distributed AI companies working together to bring together a robust distributed AI to the market.

**Setup SNET**

During installation of SNET-CLI the ff are also installed together

pip install snet-cli

Installing collected packages:

pyyaml, pillow, six, protobuf, grpcio, grpcio-tools, lru-dict, eth-hash, toolz, cytoolz, eth-typing, eth-utils, hexbytes, attrdict, pycryptodome, eth-keys, eth-keyfile, rlp, eth-rlp, eth-account, parsimonious, eth-abi, websockets, chardet, idna, certifi, urllib3, requests, web3, grpcio-health-checking, argcomplete, hidapi, python-u2flib-host, ecpy, pymultihash, pbkdf2, mnemonic, rfc3986, future, base58, ipfsapi, ecdsa, libusb1, click, pyblake2, trezor, pycoin, attrs, pyrsistent, jsonschema, jsonrpcclient, snet-cli

**Note**

you can install many snet-cli in different environment, but when you interact with snet-cli commands they display the same things because their informations are just loaded from internet so it depends on the account of the metamask if its the same they are the same.

But if the account is different they may also different even in the same environment.

**3. How to Call Snet Service by CLI**

# *Call service with 'org\_id + service\_id + method\_name' not whit 'channel\_id + method'*

*snet client call ORG SERVICE\_NAME METHOD\_NAME.*

*In the current version snet client call has the following parameters:*

*channel\_id, price, endpoint, method, params.  
It might be more convenient for users to have the following parameters: price, org\_id, service\_id, method.*

* we could remove price. But we probably should keep it for cross check.
* endpoint could be made optional. By default, we read endpoint from metadata and use round-robin in case of multiply endpoints
* channel\_id is optional. We need it only in case we have several initialized channel for the same service.

*To implement it, we need to have a way to find all initialized channels which was open for specific service.*

To findout what channel a service belongs

snet channel print-initialized-filter-group

to findout all initialized channels

snet channel print-initialized

to see current network, identity and ipfs

snet session

In case of several channels for the given sender and group\_id, everybody (snet-cli, dApp, all SDK) should select the same channel by default.

In the current version snet-cli and dApp always select channel with smallest channel\_id. We should make sure that we do it in SDK.  
[singnet/snet-dapp#223](https://github.com/singnet/snet-dapp/issues/223)

It adds proper support for secure channels (https) and it significantly simplifies the life for users by hiding channel\_id (and endpoints).

Example of workflow:

snet network ropsten

# open new channel only if you haven't done it already

# It will not open new channel if you already have one!

snet channel open-init DappTesOrganization DappTesthttpsService 0.00001 +8days

# add funds if needed (for service which was opened for specific service)

snet channel extend-add-for-service DappTesOrganization DappTesthttpsService --amount 0.00001 --expiration +8days

# print list of initialized channels

snet channel print-initialized

# make a call (you can use -y to remove price warning)

snet client call DappTesOrganization DappTesthttpsService mul '{"a":12,"b":7}'

# try claim timeout for ALL your channels (channels which have your as a sender)

snet channel claim-timeout-all

To get the metadata like service\_ip and port of a service

snet service print-metadata <orgid> <service\_id>

e.g snet service print-metadata snet network-analytics-nodeimportance

**4 Setting up Snet-Sdk-python**

Snet-sdk allows you

to import compiled (collected from dt source) client libraries for your service.

To make calls to Snet Services programmatically from your application by setting up state channels with the provides of those service.

Making gRPC calls to the SingularityNET daemons for those services by selecting a channel with sufficient funding and supplying the appropriate metadata for authentication.

Channel →

After you have installed the snet-sdk in your current environment and it’s in your PYTHONPATH,

1. import it and create instance sdk

From snet\_sdk import Snet

import config

snet = Snet(private\_key=config.private\_key)

or snet = Snet(private\_key = pk)

2. then Instantiate Clients for SingularityNET Services by using the created instance of sdk.

**Note:** to interact wz the Snet Services by Sdk, the Sdk needs to be supplied the compiled Client libraries and a reference to their path on your file system

to generate the Client libraries you need to install Snet CLI

for the installation of snetcli the python virtual enviroment is good.

Then to generate client library

snet sdk generate-client-library python <org\_id> <service\_id>

you can specify the directory by cd, but if you just generate it in the home directory

Now, by default the Sdk is going

3.

**Questions**

1 How can I install the sdk-python, I think there are two ways.

1 Just by typing, pip3 install snet\_sdk

2 by cloning the github page,

so which one should I follow.

1. On the first line of code “from snet\_sdk import Snet” is this “snet\_sdk” is from the pythonpath or just from the cloned sdk-python github page.

2. In installing sdk-python, Do I need to install it in the snetcli environment of just in another stand alone environment.

What I do is I just try both, but I think just setting up in the snetcli environment is best.

3. Do we need to generate the compiled client library for each Services we want to interact with using Sdk

4 ReadMe it was written that, by default the sdk is going to look for the generated client libraries in the ./grpc/<org\_id>/<service\_id> but in my machine .grpc is not found. Instead the library is installed just in the folder called client\_libraries

so how can I specify this path in my python code by what specific code ?

The first issue is fixed by

<https://github.com/singnet/snet-sdk-python/pull/5>

$snet channel open-init snet translation 0.0001 +10days

$snet client call snet translation translate '{"text":"Hello World.", "source\_language":"en", "target\_language":"de"}' -y

$snet client call snet example-service add '{"a":21,"b":3}' -y

snet client call snet network-analytics-bipartite

$snet client call snet --channel-id=6249 example-service add '{"a":4, "b":5}'

<https://github.com/singnet/snet-sdk-python/pull/18>

5 Calling a Service from Snet CLI

To create or open a channel

snet

To check the list of initialized channels

snet channel print-initialized

To get current channel state

snet client get-channel-state 6191

**Snet Cli Commands**

Currently it contains 15 commands

1. account
2. channel
3. client
4. contract
5. identity
6. network
7. organization
8. sdk
9. service
10. session
11. set
12. treasure
13. unset
14. version

and others

--h and --print-traceback

**snet channel**

* commands contains other 15 commands

1. **init**

* is used to only Initialize channel using service metadata, and it gets this service\_metadata from Registry by itself we do not need to provide for it.
* service\_metadata is the json file that contains the information about that service.
* requires threee postional arguments
* just display the group name the given service is belongs to

e.g $ snet channel init snet example-service 2415

output #group\_name

default\_group

1. **init-metadata**

* this is used to only Initialize channel using service metadata, for this we need to provide service\_metadata
* requires three positional arguments (ORG\_ID, SERVICE\_ID, CHANNEL\_ID )
* and it needs “service\_metadata.json” of the service

e.g $ snet channel init-metadata snet example-service 2415

1. **open-init**

* is used to open and initialize channel
* using service metadata, it gets the service\_metadata from Registry we do not need to provide for it.
* this is used to initialize the channel for each service
* it requires 5 positional arguments (ORG\_ID, SERVICE\_ID, AMOUNT(AGI token), EXPIRATION(expiration date)
* also has other optional arguments

e.g $ snet channel open-init snet example-service 0.001 +10days

* for the service that has already initialized channel to create another new channel for it just pass the optional arguments of open-init which is –open-new-anyway

e.g $ snet channel open-init –open-new-anyway snet example-service 0.001 +10days

1. **open-init-metadata**

* this is used to open and initialize service channel using service metadata, for this we need to provide the service\_metadata

1. extend-add
2. extend-add-for-service
3. block-number
4. print-initialized

**snet session**

displays what that is the current

identity, ipfs, and network.

**snet service**

* contains around 22 different commands

**1. print-metadata**

* this is used to print the metadata or information about the given service in a given organization. Information like service\_description, endpoints and others
* it needs 2 positopen\_init\_channel\_registryional arguments (ORG\_ID and Service\_ID)

e.g $ snet service print-metadata snet example-service

2.

**snet version**

displays the version

**Questions to ask**

1. In Components of the SNET I understand the document or written tutorials, but I couldn’t understand the diagram that depicts the components of SNET

2. which docker image and Metamask I need to install to setup the SNET

3. before installing the docker image I want to read the SingualrityNET whitepaper 2.0 how is it?

4 what is the difference between GRPC, JSONRPC and Protobuf

4 what is the gas\_price, gasUsed ?

e.g you can see when you initialize a channel for a given service

5. can I know the Identity fo a person or who he is by his Metamask Public Address ?

6 what is a difference between Service URL, Servie Name and Service tag ? [link](https://dev.singularitynet.io/docs/concepts/naming-standards/)

7 what does mean repeated keyword in protobuf field ?

8

**Issues during installation**

1. in How to publish a SingularityNET service page I faced an error in Step 8, couldn’t Publish a Service on SingularityNET

root@326b4895849a:/opt/singnet/example-service# snet service publish $ORGANIZATION\_ID $SERVICE\_ID -y

# Calculating gas price. It might take ~60 seconds.

# gas\_price = 1.969292 GWei

Error: {'code': -32000, 'message': 'gas required exceeds allowance or always failing transaction'}

If you want to see full Traceback then run:

snet --print-traceback [parameters]

root@326b4895849a:/opt/singnet/example-service#

another problem here is the Snet account address created here and my metamask account address is not the same I dont know where it gets this account.

Check it with “snet account print” or “snet account balance”

5. in How to Write a SingularityNet Services in Python page, also I faced an error in Step 8, couldn’t create SNET identity. But even I create this identity in How to Publish just by following the tutoirals, but in this I couldn’t

it displays an error of Identity\_type

error: the following arguments are required: IDENTITY\_TYPE

usage: snet identity create [-h] [--mnemonic MNEMONIC]

[--private-key PRIVATE\_KEY]

[--keystore-path KEYSTORE\_PATH]

[--network NETWORK] [--wallet-index WALLET\_INDEX]

IDENTITY\_NAME IDENTITY\_TYPE

6. in Deploying SingularityNEt locally, in step of Setting up snet command line interface

./scripts/blockchain: No such file or directory

7. I fully install all libraries and dependencies in how to deploy local version of SingularityNET but after finishing it I dont know how to start it and use it.

8. I install snet-sdk by two ways

1 just by pip3 install snet-sdk

but this puts in the library as snet.sdk and this is not imported

2 by cloning github repositiories of snet-sdk-python then follow the installing part at the end of readme.

But the development/test blockchain dependencies could not be installed

./scripts/blockchain install raise an error of

FileNotFoundError: [Errno 2] No such file or directory: '/home/amante/snet/snet-sdk- python/snet\_sdk/resources/contracts'

but the package is installed correctly

pip install -e .

and in python environment this mention snet-sdk the folder which exctracted from github.

And the problem is this folder does not have contracts files in snet\_sdk/resources folders

which pointed in \_\_init\_\_.py line 136

9 for the config part I installed it with pip3 install config, but this config does not have private key

and as I think here what we need is the config of snet, but I do not know how to import this config of snet.

So I just provide the private key in this python file.

There is also another thing that I change here, which is the code take your private key and from this private key it extracts your public key and adds “0x” before this public key to make it hex, but by default the extracted public key is hex, so it raise an error, so what I change is I just delete the code that add “0x” to the public key in

snet-sdk-python/snet\_sdk/\_\_init\_\_.py or Snet Class line 101 and 115

10 by creating the python function call.py inside the github repositories of snet-sdk-python with the following code

from snet\_sdk import Snet

snet = Snet(private\_key='4e51e1cc30aae65758d4eb9edc46dad7b34a8f25ffa941e1dcbb15a34935a488')

client = snet.client("snet", "network-analytics-nodeimportance")

10 I make the snet sdk generate-client-library python snet network-analytics-nodeimportance

in three postion,

– home directory

– snetcli environment and

– in snet-sdk-python directory

11 Even calling snet service through terminal is not working correctly

snet client call snet network-analytics-nodeimportance CentralNodes centralnodes.json -y

Error: Cannot find initialized channel for service with org\_id=snet service\_id=network-analytics-nodeimportance and signer=0xE2A22dC95Ed0D02eA3Ea79964Da10c1A4A2515da

SingularityNet installation

to install

**TO create identity with private key**

sudo snet identity create [identity-name] key –private-key [private-key]

To open the virtual environments snet-cli was installed

cd snet/

source snetcli/bin/activate

to deactivate it →

to list .files or hidden files in ubuntu

ls -a

sudo nautilus .snet/

Questions

1. what is the difference between Opencog and SingularityNET

2. block chain is the one who stores your security information, but who controls the block chain ? Or how can we fully believe the block chain.

3. in block chain is it true that one no can see our key and no one knows our password except us

4. what is agas\_price ? Because when I create it says It might take ~60sec to calculate the gas price and it calculate it as gas\_price = 0.110464 Gwei

5. what is a protobuf file

6 what is channel in snet

**Error Faced**

1 Installation Error

2 No Initialized Channel

how to solve

open initial channel by

snet channel open-init <org-id> <service-name> <AGI amount> <expiration-time>

e.g

snet channel open-init snet example-service 0.0001 +10days

4 ChannelStateRequest “no\_current\_block”

how to solve  
just I create new python environemnt and install all from scratch

snet-cli pip install snet-cli

nodejs+npm pip install npm

sdk too

git clone

./scripts/blockchain install

pip install -e .

3 details = "only channel signer can get latest channel state"

(newsnetcli) amante@amante-OptiPlex-9020:~/snet$ snet client call snet translation translate '{"text":"Hello World.", "source\_language":"en", "target\_language":"de"}' -y

Error: <\_Rendezvous of RPC that terminated with:

status = StatusCode.UNKNOWN

details = "only channel signer can get latest channel state"

debug\_error\_string = "{"created":"@1565078684.486817456","description":"Error received from peer ipv4:34.216.72.29:6308","file":"src/core/lib/surface/call.cc","file\_line":1052,"grpc\_message":"only channel signer can get latest channel state","grpc\_status":2}"

>

If you want to see full Traceback then run:

snet --print-traceback [parameters]

how to solve

4 details = "failed to connect to all addresses"

(newsnetcli) amante@amante-OptiPlex-9020:~$ snet client get-channel-state 6191 tz-services-1.snet.sh:2234

Error: <\_Rendezvous of RPC that terminated with:

status = StatusCode.UNAVAILABLE

details = "failed to connect to all addresses"

debug\_error\_string = "{"created":"@1565086594.263649708","description":"Failed to pick subchannel","file":"src/core/ext/filters/client\_channel/client\_channel.cc","file\_line":3528,"referenced\_errors":[{"created":"@1565086594.263348673","description":"failed to connect to all addresses","file":"src/core/ext/filters/client\_channel/lb\_policy/pick\_first/pick\_first.cc","file\_line":399,"grpc\_status":14}]}"

>

If you want to see full Traceback then run:

snet --print-traceback [parameters]

(newsnetcli) amante@amante-OptiPlex-9020:~$

4 in sdk, Hexa Error

how to solve,

remove the string (“0x” +) in line 101 and 115

5 in sdk No Contracts

how to solve, just follow the issue #19

add some codes

6 in sdk, details = "failed to connect to all addresses"

grpc.\_channel.\_Rendezvous: <\_Rendezvous of RPC that terminated with:

status = StatusCode.UNAVAILABLE

details = "failed to connect to all addresses"

debug\_error\_string = "{"created":"@1565079313.738845748","description":"Failed to pick subchannel","file":"src/core/ext/filters/client\_channel/client\_channel.cc","file\_line":3528,"referenced\_errors":[{"created":"@1565079313.738599037","description":"failed to connect to all addresses","file":"src/core/ext/filters/client\_channel/lb\_policy/pick\_first/pick\_first.cc","file\_line":399,"grpc\_status":14}]}"

>

How did I solve,

7 Sdk Block chain install warnings

npm WARN blockchain No description

npm WARN blockchain No repository field.

npm WARN blockchain No license field.

9 import config error

→ it cannot import the ./snet/config file

how to solve

just take the private key from the user in command line

8 in sdk, import error from client.grpc.service

how to solve just,

create pythonAPI in the directory where the client library is found and call it directly

by importing them as follows

import example\_service\_pb2 as ep

import example\_service\_pb2\_grpc as epg

then

stub = epg.CalculatorStub(client.grpc\_channel)

9

Error 2

EEEROOOR

<\_Rendezvous of RPC that terminated with:

status = StatusCode.UNKNOWN

details = "Exception calling application: (<StatusCode.UNKNOWN: (2, 'unknown')>, "(<StatusCode.UNKNOWN: (2, 'unknown')>, 'Element of the input array edges at zero-indexed poistion 0 does not contain two nodes')")"

debug\_error\_string = "{"created":"@1565187544.076646277","description":"Error received from peer ipv4:159.69.56.49:2224","file":"src/core/lib/surface/call.cc","file\_line":1052,"grpc\_message":"Exception calling application: (<StatusCode.UNKNOWN: (2, 'unknown')>, "(<StatusCode.UNKNOWN: (2, 'unknown')>, 'Element of the input array edges at zero-indexed poistion 0 does not contain two nodes')")","grpc\_status":2}"

>

10 Traceback (most recent call last):

File "call.py", line 8, in <module>

identity = Snet(private\_key='4e51e1cc30aae65758d4eb9edc46dad7b34a8f25ffa941e1dcbb15a34935a488')

File "/home/amante/snet/snet-sdk-python/snet\_sdk/\_\_init\_\_.py", line 123, in \_\_init\_\_

times = [block.timestamp for block in list(map(lambda n: self.web3.eth.getBlock(n), range(latest.number, latest.number-20, -1)))]

File "/home/amante/snet/snet-sdk-python/snet\_sdk/\_\_init\_\_.py", line 123, in <listcomp>

times = [block.timestamp for block in list(map(lambda n: self.web3.eth.getBlock(n), range(latest.number, latest.number-20, -1)))]

AttributeError: 'NoneType' object has no attribute 'timestamp'

Solution

Just leave this one and run it again