

Smart-Contract-based-Security-Architecture-For-Collaborative-Municipal-System

Pre-Requirement of Core Technologies Installation

- ✓ SDN Controller
- ✓ Blockchain Installation
- ✓ Multichain WebDemo

1. SDN Installation

For Installation of contiki operating system follow the link and install the SDNWISE controller

<https://sdnwiselab.github.io/docs/guides/GetStarted.html>

2.Installation of Multichain Web Demo

MultiChain Web Demo is a simple web interface for [MultiChain](#) blockchains, written in PHP.

<https://github.com/MultiChain/multichain-web-demo>

A. System Requirements

- A computer running web server software such as Apache.
- PHP 5.x or later with the curl and JSON extensions.
- MultiChain 1.0 alpha 26 or later, including MultiChain 2.x.

B. Create and launch a MultiChain Blockchain

If you do not yet have a chain to work with, [Download MultiChain](#) to install MultiChain and create a chain named chain1 as follows:

- **multichain-util create chain1**
- **multichaind chain1 -daemon**

If your web server is running on the same computer as multichaind, you can skip the rest of this section. Otherwise:

multichain-cli chain1 stop

Then add this to ~/.multichain/chain1/multichain.conf:

rpccallowip=[IP address of your web server]
Then start MultiChain again:

multichaind chain1 -daemon

C. Configure the Web Demo

```
cat ~/.multichain/chain1/multichain.conf  
grep rpc-port ~/.multichain/chain1/params.dat
```

In the web demo directory, copy the config-example.txt file to config.txt:

```
cp config-example.txt config.txt
```

In the demo website directory, enter chain details in config.txt e.g.:

```
default.name=Default          # name to display in the web interface  
default.rpchost=127.0.0.1     # IP address or domain of MultiChain node  
default.rpcsecure=0           # set to 1 to access RPC via https (e.g. for MultiChain on  
Azure)  
default.rpcport=12345         # usually default-rpc-port from params.dat  
default.rpcuser=multichainrpc  # username for RPC from multichain.conf  
default.rpcpassword=mnBh8aHp4mun... # password for RPC from multichain.conf
```

Application Design Demonstration

USE CASE Discussion

Use Case that we have implemented involves data exchange between three major government organizations which involves

- 1. National Disaster Management Authority**
- 2. NADRA**
- 3. Pakistan Telecommunication Authority**

NDMA (National Disaster Management Authority)

National Disaster Management Authority (NDMA) is a government authority which keeps track of disasters and comes up with plans to minimize risk from these disasters. NDMA plans at different levels of destruction and has different levels of alert systems within its own website which includes Drought Alert, Flood Alert and so on. In our use case we are currently focusing on air quality and environment alerts.

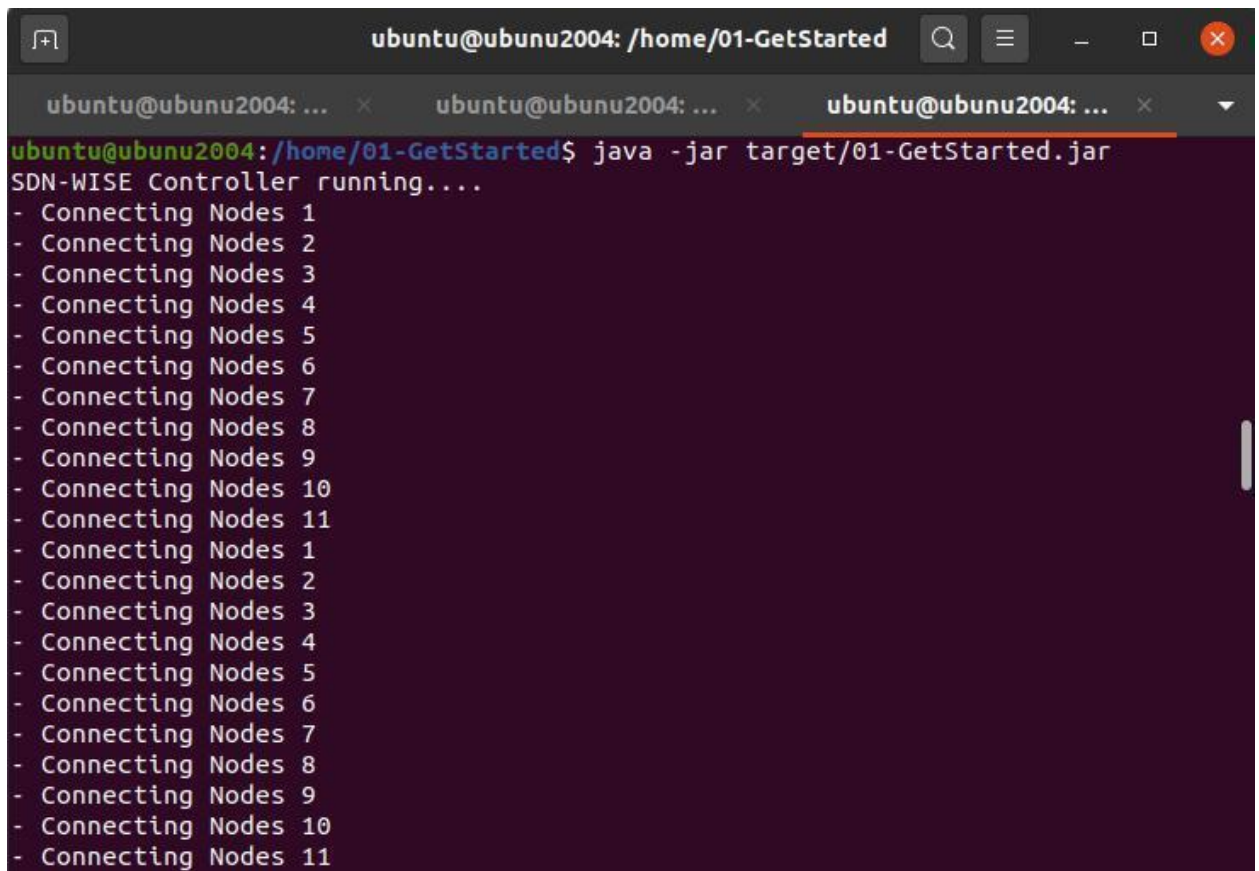
Multiple sensors are placed around the city to keep track of air quality. Air quality is measured in the air quality index. There are different severity levels of air quality and each severity has a different impact on one's life and climate. Climate change is expected to have an adverse impact on Pakistan, as it ranks 7th on the climate risk index. The rapid change in global climate has given rise to many disasters that pose severe threats to human life, property and infrastructure.

Air quality has a direct effect on one's health because each individual is inhaling and exhaling air. Long-term health effects from air pollution include heart disease, lung cancer, and respiratory diseases such as emphysema. Air pollution can also cause long-term damage to people's nerves, brain, kidneys, liver, and other organs. Some scientists suspect air pollutants cause birth defects. Nearly 2.5 million people die worldwide each year from the effects of outdoor or indoor air pollution (NAT GEO).

The simulation of the project has been implemented on Cooja Simulator for 11 IoT Nodes so far, and some UI has been which is intuitive and easy for a human to understand.

SDN Controller UI:

When the smart home user will run the SDN Controller, it will run as shown below in the screenshot:



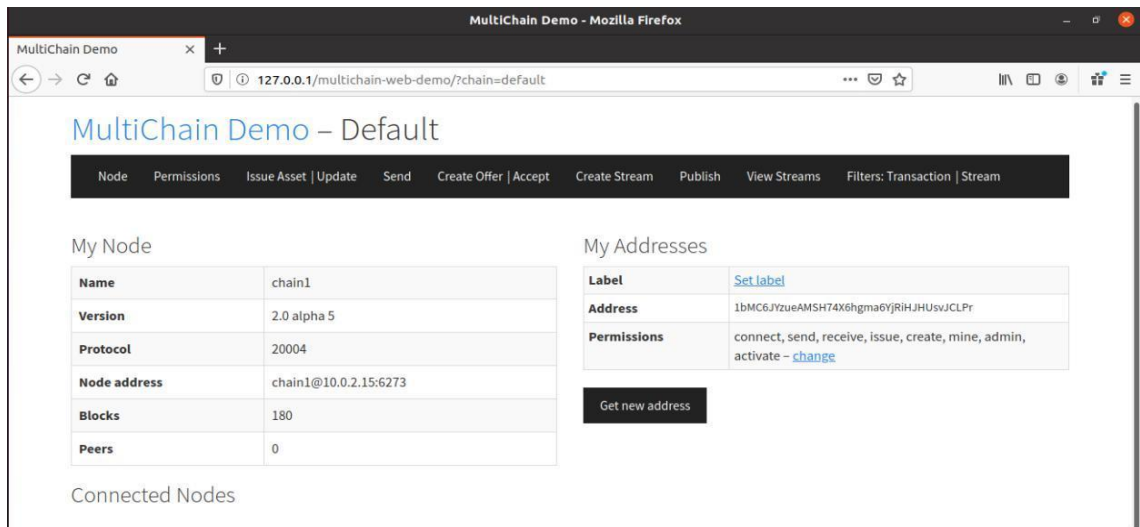
```
ubuntu@ubunu2004: /home/01-GetStarted
ubuntu@ubunu2004: ... x ubuntu@ubunu2004: ... x ubuntu@ubunu2004: ... x
ubuntu@ubunu2004:/home/01-GetStarted$ java -jar target/01-GetStarted.jar
SDN-WISE Controller running....
- Connecting Nodes 1
- Connecting Nodes 2
- Connecting Nodes 3
- Connecting Nodes 4
- Connecting Nodes 5
- Connecting Nodes 6
- Connecting Nodes 7
- Connecting Nodes 8
- Connecting Nodes 9
- Connecting Nodes 10
- Connecting Nodes 11
- Connecting Nodes 1
- Connecting Nodes 2
- Connecting Nodes 3
- Connecting Nodes 4
- Connecting Nodes 5
- Connecting Nodes 6
- Connecting Nodes 7
- Connecting Nodes 8
- Connecting Nodes 9
- Connecting Nodes 10
- Connecting Nodes 11
```

Description:

The System will run SDN Controller first for establishing the connection of SDN Controller with the nodes of the smart home and the controller will connect with all the IoT Nodes of the smart home one by one as shown above in the screenshot.

Multichain Node UI:

When the System user will run Multichain platform to view nodes of the smart home on the chain, the Multichain Platform will look like this as shown in the screenshot below:

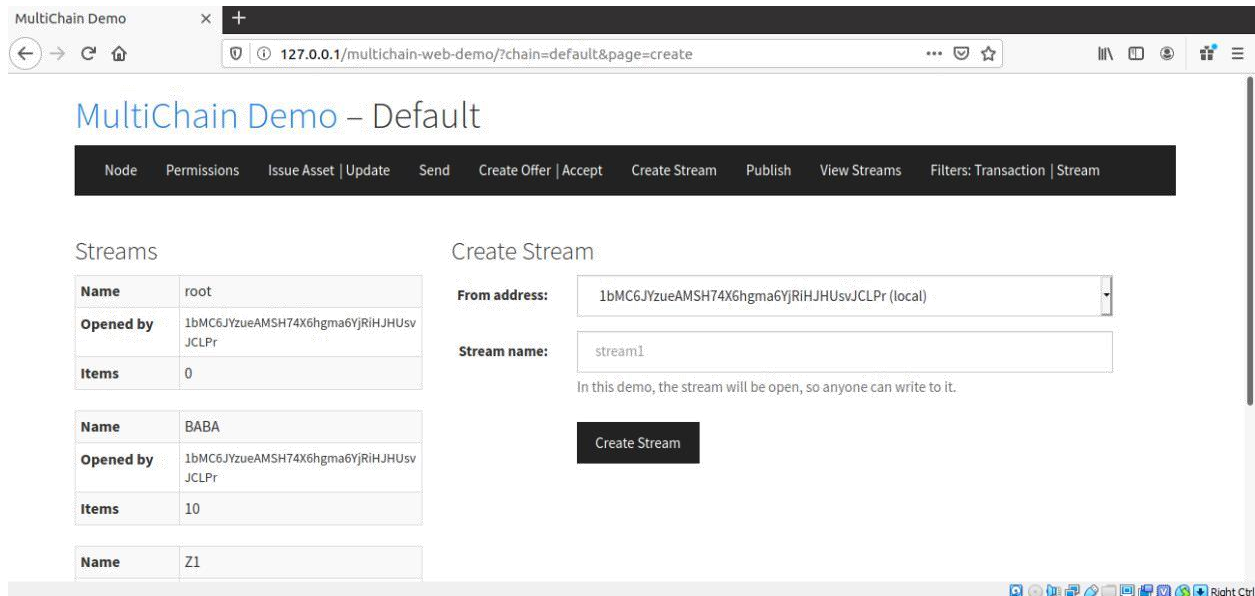


Description:

The System user will be able to view the chain on which the IoT Devices are connected for writing the data on blockchain. The blockchain will be shown in the form of a web demo as shown below in the screenshot.

Multichain Add Data Stream UI:

When user wants to add a data stream on the Multichain he can add like this as shown below in the screenshot

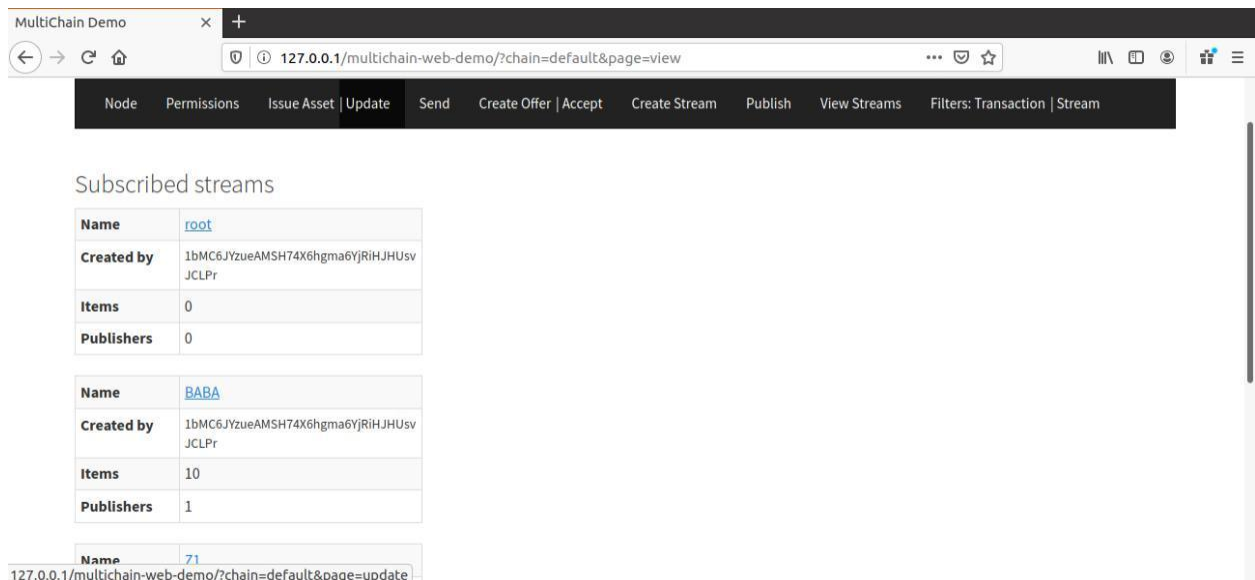


Description:

The System will be able to add the datastream on Multichain by going to the option of ‘Create Stream’ as shown in the screenshot above.

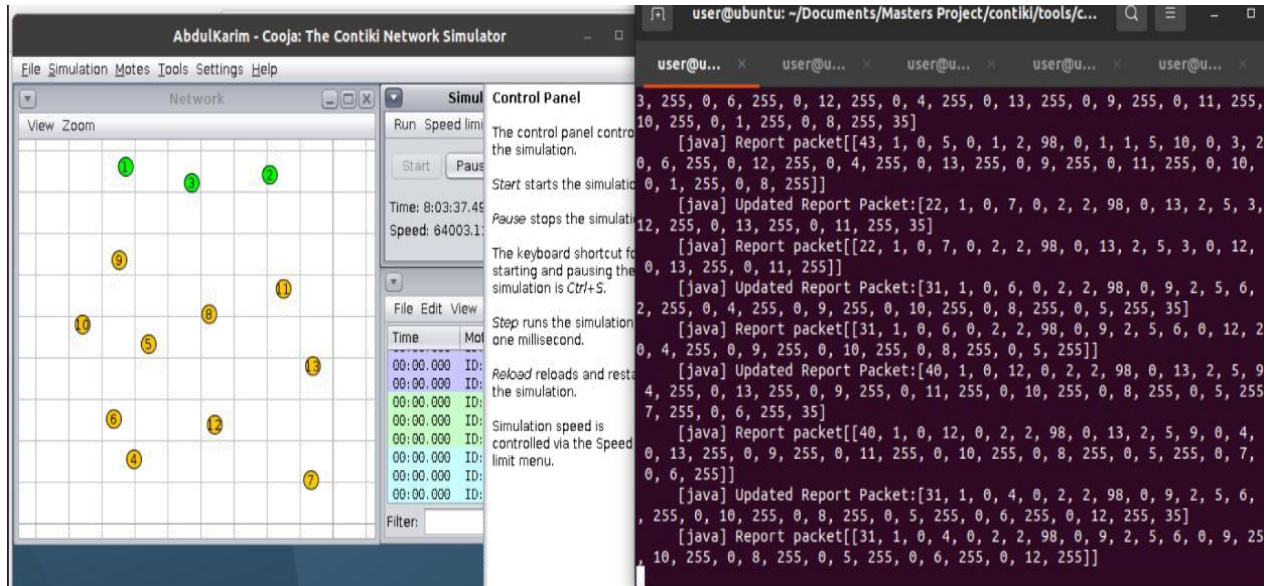
Multichain View Data Stream UI:

When the system wants to view the data streams on the Multichain he can view like this as shown below in the screenshot.



Description:

The System will be able to view the datastream on Multichain by going to the option of View Streams as shown in the screenshot above



In the above picture, we have a simulation of a smart city where we have green nodes as sink nodes and orange nodes as source nodes. Orange nodes are sensors that are sharing data to green nodes which are raspberry pis. Green nodes receive data and share data to multichain and to our frontend dashboard. Beside this simulation we have logged distance between nodes that will be utilised and used in future work.

```

100  462  100  462   0   0   Dload Upload  Total  Spent  Left  Speed
100  462  100  462   0   0  1017    0 --:--:-- --:--:-- --:--:-- 10
Karachi
{"method":"publish","params":["TR","2",{"json":[{"Registration Id":5}, {"Temperature":{"coord":{"lon":67.0011,"lat":24.8607},"weather":[{"id":721,"main":"Haze","description":"haze","icon":"50n"}],"base":"stations","main":{"temp":296.22,"s_like":296.04,"temp_min":296.22,"temp_max":296.22,"pressure":1016,"humidity":75,"visibility":4000,"wind":{"speed":4.12,"deg":210},"clouds":{"all":0},"dt":402669,"sys":{"type":1,"id":7576,"country":"PK","sunrise":1639361273,"sunset":16399475},"timezone":18000,"id":1174872,"name":"Karachi","cod":200}},{{"Message":"Mild Temperature"}]}], "id":"45776356-1639402672","chain_name":"chain4"}

be5c306627def67238daa7fb4288607a1a66524b376af44865638b882452ac86
touch: missing file operand
Try 'touch --help' for more information.
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Curr
      Dload Upload  Total  Spent  Left  Speed
100  462  100  462   0   0   881    0 --:--:-- --:--:-- --:--:-- 8
Karachi
{"method":"publish","params":["TR","3",{"json":[{"Registration Id":0}, {"Temperature":{"coord":{"lon":67.0011,"lat":24.8607},"weather":[{"id":721,"main":"Haze","description":"haze","icon":"50n"}],"base":"stations","main":{"temp":296.22,"s_like":296.04,"temp_min":296.22,"temp_max":296.22,"pressure":1016,"humidity":75,"visibility":4000,"wind":{"speed":4.12,"deg":210},"clouds":{"all":0},"dt":

```


This logs the temperature and other details that are being passed onto multichain and displayed on our dashboard panel. Currently we are passing a few details but we will be collecting more data from the environment and will implement a real world use case after collecting these d

NodePermissionsIssue Asset | UpdateSendCreate Offer | AcceptCreate StreamPublishView StreamsFilters: Transaction | Stream

Subscribed streams

Name	root
Created by	1aZjQ3iU9QTR9KbvvHDcDRQsaptfsjvD4ziY95
Items	0
Publishers	0

Name	TR
Created by	1aZjQ3iU9QTR9KbvvHDcDRQsaptfsjvD4ziY95
Items	9
Publishers	1

Other streams

Stream: TR – 9 of 9 items

Publishers	1aZjQ3iU9QTR9KbvvHDcDRQsaptfsjvD4ziY95
Key(s)	3
JSON data	[{ "Registration Id": 0 }, { "Temperature": { "coord": { "lon": 67.0011, "lat": 24.8607 }, "weather": [{ "id": 721, "main": "Haze", "description": "haze", "icon": "50n" }], "base": "stations", "main": { "temp": 296.22, "feels_like": 296.04, "temp_min": 296.22, "temp_max": 296.22, "pressure": 1016, "humidity": 56 }, "visibility": 4000 }]



This is our dashboard panel that gives visual data representation of data recorded from the environment using sensor

CODE DEMONSTRATION

Run the Simulation Command

- ✓ **Execute all these command at the root folder**
 - 1. Sudo ant compile**
 - 2. Sudo ant run** (to run cooja to execute the Use case in simulated Environment)
 - 3. Link Controller to Cooja Nodes**
 - 4. mvn package**
 - 5. java -jar target\01Getstarted.jar**
 - 6. Run the Application which is build on python application**