

Discovery of OdysseusNodes

There are different ways to build an network of Odysseus nodes. Some of them are only available, if worker nodes are OdysseusNet nodes, too. The master must always be an OdysseusNet node. A combination of different mechanisms is possible and needs to be configured in the [OdysseusNet configuration](#) (e.g. `<entry key="net.discoverer.name">MulticastOdysseusNodeDiscoverer,BroadcastOdysseusNodeDiscoverer,IPListOdysseusNodeDiscoverer</entry>`)

- [Broadcast](#)
- [Multicast](#)
- [Config File](#)
- [Odysseus Script Command](#)
- [Persisting Nodes](#)

Broadcast

This is only available for cases, where nodes are OdysseusNet nodes and all nodes are in the same subnet.

```
<entry key="net.discoverer.name">BroadcastOdysseusNodeDiscoverer</entry>
```

Further parameters are

- `net.discoverer.interval`: How many milliseconds between two broadcast calls.

Multicast

This is only available for cases, where all node are OdysseusNet nodes and all nodes are in the same subnet.

```
<entry key="net.discoverer.name">MulticastOdysseusNodeDiscoverer</entry>
```

Further parameters are

- `net.discoverer.interval`: How many milliseconds between two multicast calls.
- `net.discoverer.multicast.ip`: Which IP should be used for multicast (default is 239.254.42.95)
- `net.discoverer.multicast.port`: Which port should be used for multicast (default is 9796)

Config File

To initially assign any odysseus node as a worker to a OdysseusNet master, a config file named `nodeIPList.conf.json` must be placed in the [Odysseus Home](#) folder

```
<entry key="net.discoverer.name">IPListOdysseusNodeDiscoverer</entry>
```

DEPRECATED:

```
[{
  "hostname": "192.168.8.124",
  "port": "8888",
  "username": "System",
  "password": "manager"
},{
  "hostname": "192.168.8.128",
  "port": "8888",
  "username": "System",
  "password": "manager"
}]
```

NEW:

```
[{
  "name": "worker_01",
  "serverAddress": "192.168.188.36",
  "serverPort": "18881",
  "username": "System",
  "password": "manager"
},{
  "name": "worker_02",
  "serverAddress": "192.168.188.36",
  "serverPort": "18882",
  "username": "System",
  "password": "manager"
},{
  "name": "worker_03",
  "serverAddress": "192.168.188.36",
  "serverPort": "18883",
  "username": "System",
  "password": "manager"
}]
```

The "name"-attribute is mandatory now.

Odysseus Script Command

With the Odysseus Script Command #ADD_NODE/#REMOVE_NODE we provide a mechanism to add/remove nodes at runtime.

```
#DEFINE HOST 192.168.2.57
#ADD_NODE (name=worker_1) (serverAddress=${HOST}) (serverPort=18881) (username=System) (password=manager)
#ADD_NODE (name=worker_2) (serverAddress=${HOST}) (serverPort=18882) (username=System) (password=manager)
#ADD_NODE (name=worker_3) (serverAddress=${HOST}) (serverPort=18883) (username=System) (password=manager)
```

or

```
#ADD_NODE (name=cluster_1) (serverAddress=192.168.2.162) (serverPort=8888) (username=System) (password=manager)
#ADD_NODE (name=cluster_2) (serverAddress=192.168.2.163) (serverPort=8888) (username=System) (password=manager)
#ADD_NODE (name=cluster_3) (serverAddress=192.168.2.164) (serverPort=8888) (username=System) (password=manager)
#ADD_NODE (name=cluster_4) (serverAddress=192.168.2.165) (serverPort=8888) (username=System) (password=manager)
```

or

```
#REMOVE_NODE (name=worker_1)
#REMOVE_NODE (name=worker_2)
#REMOVE_NODE (name=worker_3)
```

Remark: In a docker environment, do not use **localhost**! **This will not work, when the nodes should interact with each other. Localhost would be interpreted als the localhost of the container!** So it is best du use a fixed IP adress of the node where all containers are started (in a single host enviroment)

Persisting Nodes

By default, the nodes are only available at runtime and need to be connected again in case of a master restart. It is possible to persist node information via configuration:

```
<entry key="net.nodemanager.storetype">filestore</entry>
<entry key="net.nodemanager.filename">C:/Users/Marco/odysseus/store/nodes.store</entry>
```

This is an example for using a filestore. Other stores (e.g. [Redis Feature](#)) are possible, too. In this case provide the required configuration information with the key-prefix: net.nodemanager (e.g. net.nodemanager.host)