Rocky Enterprise Linux 9.2 Manual Pages on command 'docker-network-inspect.1'

C:\>man docker-network-inspect.1

DOCKER(1) Docker User Manuals DOCKER(1)

NAME

docker-network-inspect - Display detailed information on one or more networks

SYNOPSIS

docker network inspect [OPTIONS] NETWORK [NETWORK...]

DESCRIPTION

Returns information about one or more networks. By default, this command renders all results in a JSON object. For example, if you connect two containers to the de? fault bridge network:

$ sudo docker run -itd --name=container1 busybox

f2870c98fd504370fb86e59f32cd0753b1ac9b69b7d80566ffc7192a82b3ed27

$ sudo docker run -itd --name=container2 busybox

bda12f8922785d1f160be70736f26c1e331ab8aaf8ed8d56728508f2e2fd4727

The network inspect command shows the containers, by id, in its results. You can specify an alternate format to execute a given template for each result. Go's text/template ?http://golang.org/pkg/text/template/? package describes all the de? tails of the format.

$ sudo docker network inspect bridge

```
[
    {
        "Name": "bridge",
        "Id": "b2b1a2cba717161d984383fd68218cf70bbbd17d328496885f7c921333228b0f",
```

```
"Scope": "local",
"Driver": "bridge",
"IPAM": {
    "Driver": "default",
    "Config": [
        {
            "Subnet": "172.17.42.1/16",
            "Gateway": "172.17.42.1"
        }
    ]
},
"Internal": false,
"Ingress": false,
"Containers": {
    "bda12f8922785d1f160be70736f26c1e331ab8aaf8ed8d56728508f2e2fd4727": {
        "Name": "container2",
        "EndpointID": "0aebb8fcd2b282abe1365979536f21ee4ceaf3ed56177c628eae9f706e00e019",
        "MacAddress": "02:42:ac:11:00:02",
        "IPv4Address": "172.17.0.2/16",
        "IPv6Address": ""
    },
    "f2870c98fd504370fb86e59f32cd0753b1ac9b69b7d80566ffc7192a82b3ed27": {
        "Name": "container1",
        "EndpointID": "a00676d9c91a96bbe5bcfb34f705387a33d7cc365bac1a29e4e9728df92d10ad",
        "MacAddress": "02:42:ac:11:00:01",
        "IPv4Address": "172.17.0.1/16",
        "IPv6Address": ""
    }
},
"Options": {
    "com.docker.network.bridge.default_bridge": "true",
    "com.docker.network.bridge.enable_icc": "true",
    "com.docker.network.bridge.enable_ip_masquerade": "true",
```

```
            "com.docker.network.bridge.host_binding_ipv4": "0.0.0.0",
            "com.docker.network.bridge.name": "docker0",
            "com.docker.network.driver.mtu": "1500"
        }
    }
]
```

Returns the information about the user-defined network:

```
$ docker network create simple-network
69568e6336d8c96bbf57869030919f7c69524f71183b44d80948bd3927c87f6a
$ docker network inspect simple-network
[
    {
        "Name": "simple-network",
        "Id": "69568e6336d8c96bbf57869030919f7c69524f71183b44d80948bd3927c87f6a",
        "Scope": "local",
        "Driver": "bridge",
        "IPAM": {
            "Driver": "default",
            "Config": [
                {
                    "Subnet": "172.22.0.0/16",
                    "Gateway": "172.22.0.1"
                }
            ]
        },
        "Containers": {},
        "Options": {}
    }
]
```

docker network inspect --verbose for swarm mode overlay networks shows service-spe?
cific details such as the service's VIP and port mappings. It also shows IPs of
service tasks, and the IPs of the nodes where the tasks are running.

Following is an example output for an overlay network ov1 that has one service s1

attached to. service s1 in this case has three replicas.

```
$ docker network inspect --verbose ov1
[
    {
        "Name": "ov1",
        "Id": "ybmyjvao9vtzy3oorxbssj13b",
        "Created": "2017-03-13T17:04:39.776106792Z",
        "Scope": "swarm",
        "Driver": "overlay",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": null,
            "Config": [
                {
                    "Subnet": "10.0.0.0/24",
                    "Gateway": "10.0.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "Containers": {
            "020403bd88a15f60747fd25d1ad5fa1272eb740e8a97fc547d8ad07b2f721c5e": {
                "Name": "s1.1.pjn2ik0sfgkfzed3h0s00gs9o",
                "EndpointID": "ad16946f416562d658f3bb30b9830d73ad91ccf6feae44411269cd0ff674714e",
                "MacAddress": "02:42:0a:00:00:04",
                "IPv4Address": "10.0.0.4/24",
                "IPv6Address": ""
            }
        },
        "Options": {
```

```
    "com.docker.network.driver.overlay.vxlanid_list": "4097"
},
"Labels": {},
"Peers": [
    {
        "Name": "net-3-5d3cfd30a58c",
        "IP": "192.168.33.13"
    },
    {
        "Name": "net-1-6ecbc0040a73",
        "IP": "192.168.33.11"
    },
    {
        "Name": "net-2-fb80208efd75",
        "IP": "192.168.33.12"
    }
],
"Services": {
    "s1": {
        "VIP": "10.0.0.2",
        "Ports": [],
        "LocalLBIndex": 257,
        "Tasks": [
            {
                "Name": "s1.2.q4hcq2aiiml25ubtrtg4q1txt",
                "EndpointID": "040879b027e55fb658e8b60ae3b87c6cdac7d291e86a190a3b5ac6567b26511a",
                "EndpointIP": "10.0.0.5",
                "Info": {
                    "Host IP": "192.168.33.11"
                }
            },
            {
                "Name": "s1.3.yawl4cgkp7imkfx469kn9j6lm",
```

```
                "EndpointID": "106edff9f120efe44068b834e1cddb5b39dd4a3af70211378b2f7a9e562bbad8",
                "EndpointIP": "10.0.0.3",
                "Info": {
                    "Host IP": "192.168.33.12"
                }
            },
            {
                "Name": "s1.1.pjn2ik0sfgkfzed3h0s00gs9o",
                "EndpointID": "ad16946f416562d658f3bb30b9830d73ad91ccf6feae44411269cd0ff674714e",
                "EndpointIP": "10.0.0.4",
                "Info": {
                    "Host IP": "192.168.33.13"
                }
            }
        ]
    }
  }
]
```


OPTIONS

-f, --format="" Format the output using the given Go template

-h, --help[=false] help for inspect

-v, --verbose[=false] Verbose output for diagnostics

SEE ALSO

docker-network(1)

Docker Community Feb 2022 DOCKER(1)