DNS CONFIGURATION GUIDE

Open Telekom Cloud

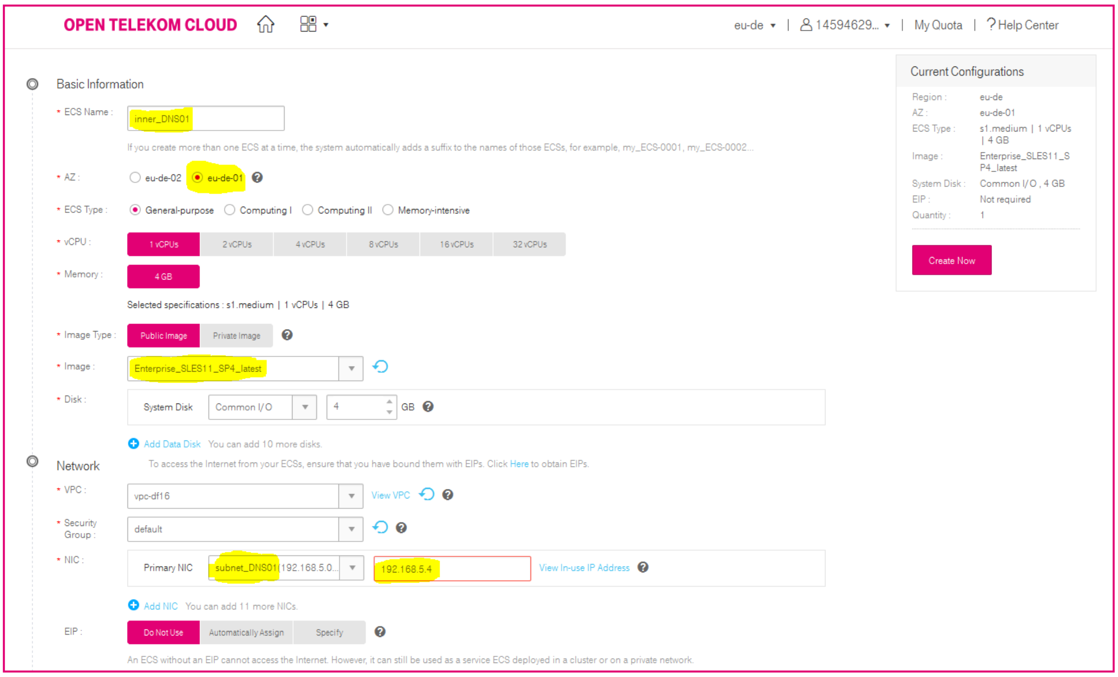
www.telekom.de/opentelekomcloud

T · · Systems·

For this guide we assume that two subnets are already configured. In our example the subnets are called subnet_DNS01 (in AZ eu-de-01) and subnet_DNS02 (in AZ eu-de02). The IP segment is 172.16.10.0/24 and the gateway 172.16.10.1.

DNS SERVER SETUP

Create a SUSE ECS server with subnet_DNS01



Basic Information

- ECS Name : inner_DNS01
  If you create more than one ECS at a time, the system automatically adds a suffix to the names of those ECSs, for example, my_ECS-0001, my_ECS-0002...
- AZ : ○ eu-de-02 ● eu-de-01
- ECS Type : ● General-purpose ○ Computing I ○ Computing II ○ Memory-intensive
- vCPU : [✓ 1 vCPUs] 2 vCPUs 4 vCPUs 8 vCPUs 16 vCPUs 32 vCPUs
- Memory : [✓ 4 GB]
  Selected specifications : s1.medium | 1 vCPUs | 4 GB
- Image Type : [✓ Public Image] Private Image
- Image : Enterprise_SLES11_SP4_latest
- Disk : System Disk Common I/O 4 GB
  Add Data Disk You can add 10 more disks.

Network

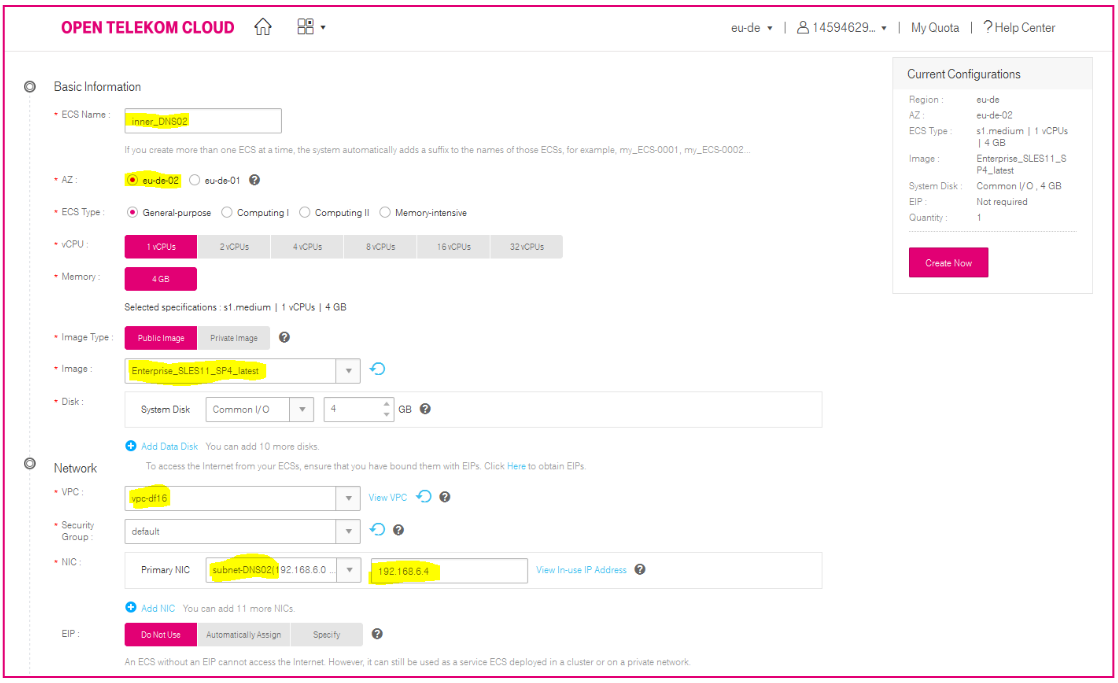
To access the Internet from your ECSs, ensure that you have bound them with EIPs. Click Here to obtain EIPs.

- VPC : vpc-df16 View VPC
- Security Group : default
- NIC : Primary NIC subnet_DNS01(192.168.5.0... 192.168.5.4 View In-use IP Address
  Add NIC You can add 11 more NICs.
- EIP : [✓ Do Not Use] Automatically Assign Specify
  An ECS without an EIP cannot access the Internet. However, it can still be used as a service ECS deployed in a cluster or on a private network.

Current Configurations

Region :	eu-de
AZ :	eu-de-01
ECS Type :	s1.medium | 1 vCPUs | 4 GB
Image :	Enterprise_SLES11_SP4_latest
System Disk :	Common I/O , 4 GB
EIP :	Not required
Quantity :	1

Create a SUSE ECS server with subnet_DNS02



Basic Information

- ECS Name : inner_DNS02
  If you create more than one ECS at a time, the system automatically adds a suffix to the names of those ECSs, for example, my_ECS-0001, my_ECS-0002...
- AZ : ● eu-de-02 ○ eu-de-01
- ECS Type : ● General-purpose ○ Computing I ○ Computing II ○ Memory-intensive
- vCPU : [✓ 1 vCPUs] 2 vCPUs 4 vCPUs 8 vCPUs 16 vCPUs 32 vCPUs
- Memory : [✓ 4 GB]
  Selected specifications : s1.medium | 1 vCPUs | 4 GB
- Image Type : [✓ Public Image] Private Image
- Image : Enterprise_SLES11_SP4_latest
- Disk : System Disk Common I/O 4 GB
  Add Data Disk You can add 10 more disks.

Network

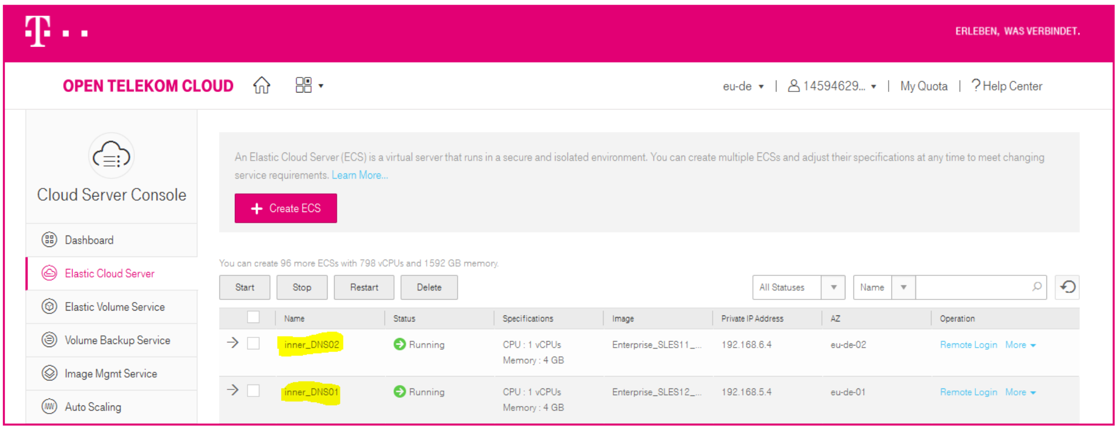
To access the Internet from your ECSs, ensure that you have bound them with EIPs. Click Here to obtain EIPs.

- VPC : vpc-df16 View VPC
- Security Group : default
- NIC : Primary NIC subnet-DNS02(192.168.6.0 ... 192.168.6.4 View In-use IP Address
  Add NIC You can add 11 more NICs.
- EIP : [✓ Do Not Use] Automatically Assign Specify
  An ECS without an EIP cannot access the Internet. However, it can still be used as a service ECS deployed in a cluster or on a private network.

Current Configurations

Region :	eu-de
AZ :	eu-de-02
ECS Type :	s1.medium | 1 vCPUs | 4 GB
Image :	Enterprise_SLES11_SP4_latest
System Disk :	Common I/O , 4 GB
EIP :	Not required
Quantity :	1

After several minutes we get the 2 VMs as DNS server



An Elastic Cloud Server (ECS) is a virtual server that runs in a secure and isolated environment. You can create multiple ECSs and adjust their specifications at any time to meet changing service requirements. Learn More...

You can create 96 more ECSs with 798 vCPUs and 1592 GB memory.

Name	Status	Specifications	Image	Private IP Address	AZ	Operation
inner_DNS02	Running	CPU : 1 vCPUs Memory : 4 GB	Enterprise_SLES11_...	192.168.6.4	eu-de-02	Remote Login More
inner_DNS01	Running	CPU : 1 vCPUs Memory : 4 GB	Enterprise_SLES12_...	192.168.5.4	eu-de-01	Remote Login More

BIND Installation

Before we can start the configuration work, we need to install the BIND software. To do this we login to the previously created VM and install the BIND package by using the yast command. Yast will download the package from default software repository in SUSE Linux OS. The command needs to be run with admin rights (same with the other commands during this guide).

```
yast -i bind
```

DNS CONFIGURATION

Creating a DNS Zone File

At this stage we will need to create a new zone file for a domain otcuser.org. Navigate to /var/lib/named directory and create the subdirectory zones/otcuser/

```
cd /var/lib/named

mkdir -p zones/otcuser

cd zones/otcuser/
```

The directory /var/lib/named/zones/otcuser shall contain a zone file for an otcuser.org domain. If you prefer to use another directory to hold this file you are free to do so. The following zone file db.otcuser.org will hold a DNS record to assist the name server to resolve a fully qualified domain name to an IP address. Create and save db.otcuser.org with the following content:

```
;
; BIND data file for otcuser.org
```

```
$TTL    3h
@           IN      SOA     ns1.otcuser.org. admin.otcuser.org. (
                    1       ; Serial
                    3h      ; Refresh after 3 hours
                    1h      ; Retry after 1 hour
                    1w      ; Expire after 1 week
                    1h )    ; Negative caching TTL of 1 day
@           IN      NS      ns1.otcuser.org.
@           IN      NS      ns2.otcuser.org.
otcuser.org.        IN      A       172.16.10.4
ns1         IN      A       172.16.10.4
ns2         IN      A       172.16.20.4
www         IN      CNAME   otcuser.org.
mail                IN      A       172.16.10.4
ftp         IN      CNAME   otcuser.org.
```

Here is just a quick review of some lines from the above BIND DNS zone file:

- SOA Record: the name server authoritative for the zone `otcuser.org` is `ns1.otcuser.org` and `admin.otcuser.org` is the email address of the person responsible for this DNS zone
- NS Records: two name servers for the `otcuser.org` zone are `ns[1,2].otcuser.org`
- CNAME Record (Canonical Name record): restart the query using the canonical name instead of the original name
- PTR: a DNS record used for a mapping of an IP address to a host name

Address-to-Name Mappings

At this stage the BIND DNS server can only resolve an IP address mapped to the otcuser.org host. What we should do now, is tell our name server the resolution for the opposite direction, which is, to resolve a host from an IP address. For this we are going to need yet another file and that is 172.16.zone with the following content:

```
;
; BIND reverse data file for 16.172.in-addr.arpa
;
$TTL    604800
@       IN      SOA     ns1.otcuser.org. admin.otcuser.org. (
                           1         ; Serial
                           3h       ; Refresh after 3 hours
                           1h       ; Retry after 1 hour
                           1w       ; Expire after 1 week
                           1h )     ; Negative caching TTL of 1 day

        IN      NS      ns1.otcuser.org.
        IN      NS      ns2.otcuser.org.
4.5    IN      PTR     otcuser.org.
```

BIND Configuration for Inner Domain Name

Until now, we have configured both forward DNS records and reverse DNS records. In order to make the DNS server running properly, we also need to insert these commands in the file of /etc/named.conf:

```
listen-on port 53 { any; };
allow-query { any; };
forwarders { 8.8.8.8; 114.114.114.114; };
forward first;
```

If you want to create a master/slave cluster, we recommend you to use DNS cluster to avoid SPOF (Single Point of Failure). You also need to update the configuration in the file of /etc/named.conf.

For the master add the following lines:

```
zone "16.172.in-addr.arpa" IN {
       type master;
       file "/var/lib/named/zones/otcuser/172.16.zone";
      allow-transfer{172.16.20.4;};
};
zone "otcuser.org" IN {
       type master;
       file "/var/lib/named/zones/otcuser/db.otcuser.org";
       allow-transfer{172.16.20.4;};
```

And for the slave:

```
zone "otcuser.org" in {
       type slave;
       masters { 172.16.10.4; };
       file "/var/lib/named/zones/otcuser/db.otcuser.org";
       allow-transfer { none; };
};

zone "16.172.in-addr.arpa" in {
       type slave;
       file "/var/lib/named/zones/otcuser/172.16.zone";
       masters { 172.16.10.4; };
       allow-transfer { none; };
};
```

BIND Configuration for Public Domain Name

Before we can test, if our configuration works properly, we need to configure IP addresses as public DNS servers. This configuration needs to be added to the named.conf.options file. This IP address is used in case that the local DNS server does not know the answer the name resolution query.

```
forwarders {
    100.125.4.25;
    217.150.148.148;
    8.8.8.8;
};
```

Checking BIND's Zone Files and Configuration

Before we attempt to start a BIND name server with a new zone and configuration here are some tools to check, if we mis-configured the service.

To check a configuration file you can run the following command:

```
named-checkconf
```

If no output is produced, your config files looks OK.

To check the DNS zone files, we can use the named-checkzone command:

```
named-checkzone otcuser.org /var/lib/named/zones/otcuser/db.otcuser.org

zone otcuser.org/IN: loaded serial 1

OK
```

Now we check the reverse zone file:

```
named-checkzone 0.168.192.in-addr.arpa /var/lib/named/zones/otcuser/db.172.16.0

zone 0.168.192.in-addr.arpa/IN: loaded serial 2

OK
```

Start / Restart the BIND name server

```
service named start

Starting domain name service...: BIND.
```

Alternatively, if your BIND server is already running use a following command to restart:

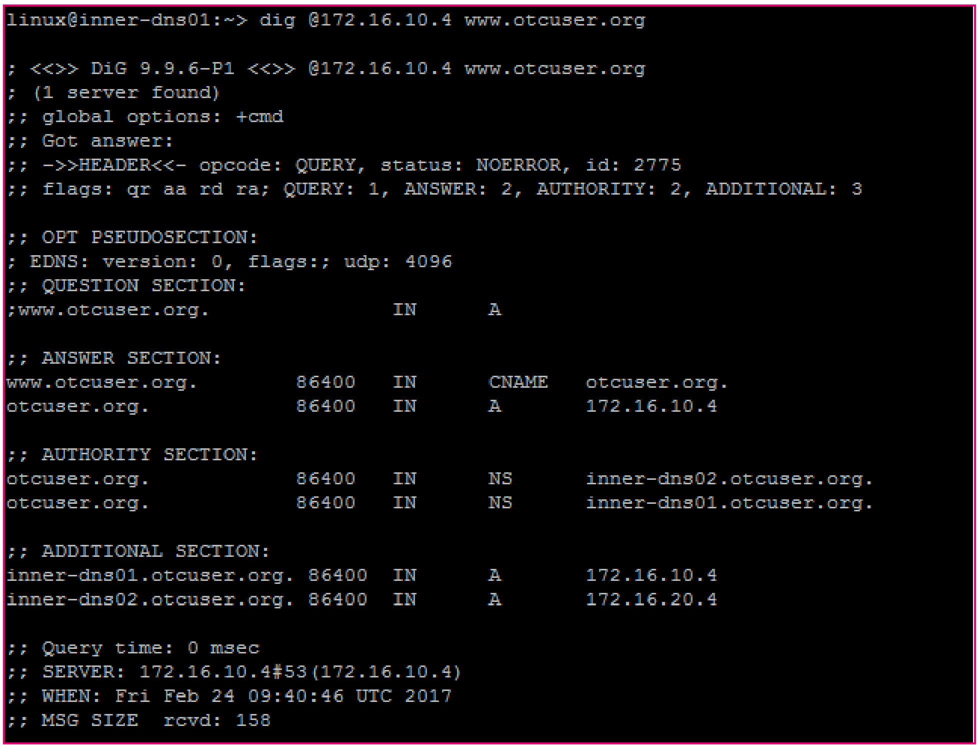
```
service named restart
```

```
Stopping domain name service...: BIND.

Starting domain name service...: BIND.
```

Testing a BIND Server Configuration

The dig command from the dnsutils package is handy to help us testing a new configuration of BIND name server. It can be used from any computer, that has network access, but preferably you should start your testing from localhost. In our case the IP address of the name servers is 172.16.10.4/192.168.20.4. First we will test the host-to-IP resolution:

```
dig @172.16.10.4 www.otcuser.org
```

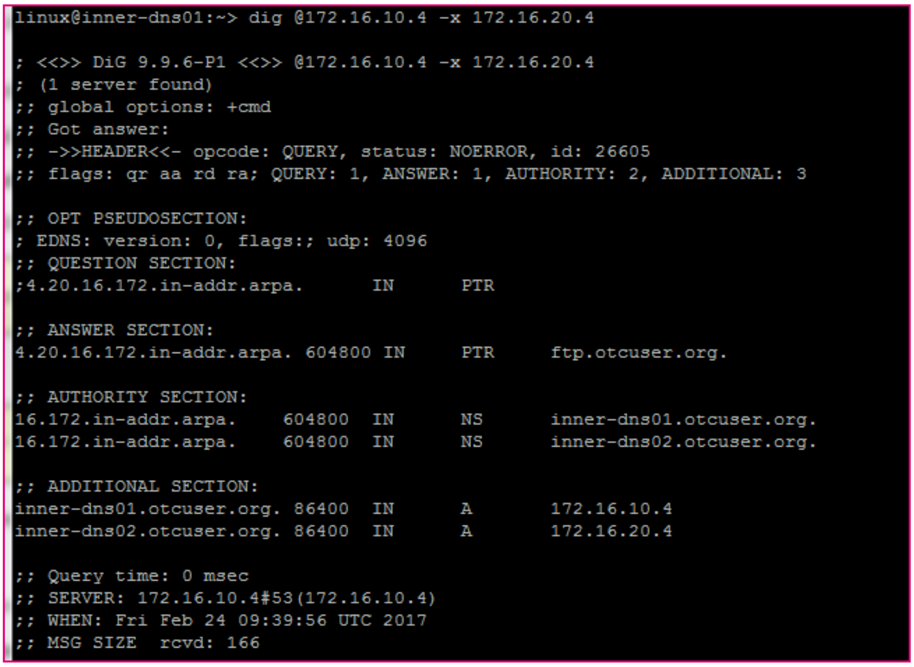


```
linux@inner-dns01:~> dig @172.16.10.4 www.otcuser.org

; <<>> DiG 9.9.6-P1 <<>> @172.16.10.4 www.otcuser.org
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 2775
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 2, ADDITIONAL: 3

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.otcuser.org.               IN      A

;; ANSWER SECTION:
www.otcuser.org.        86400   IN      CNAME   otcuser.org.
otcuser.org.            86400   IN      A       172.16.10.4

;; AUTHORITY SECTION:
otcuser.org.            86400   IN      NS      inner-dns02.otcuser.org.
otcuser.org.            86400   IN      NS      inner-dns01.otcuser.org.

;; ADDITIONAL SECTION:
inner-dns01.otcuser.org. 86400  IN      A       172.16.10.4
inner-dns02.otcuser.org. 86400  IN      A       172.16.20.4

;; Query time: 0 msec
;; SERVER: 172.16.10.4#53(172.16.10.4)
;; WHEN: Fri Feb 24 09:40:46 UTC 2017
;; MSG SIZE  rcvd: 158
```

Next we test the IP-to-host resolution:

```
dig @172.16.20.4 -x 172.16.10.4
```

If you got the right resolved record, you have just created and configured your own DNS zone using BIND name server.



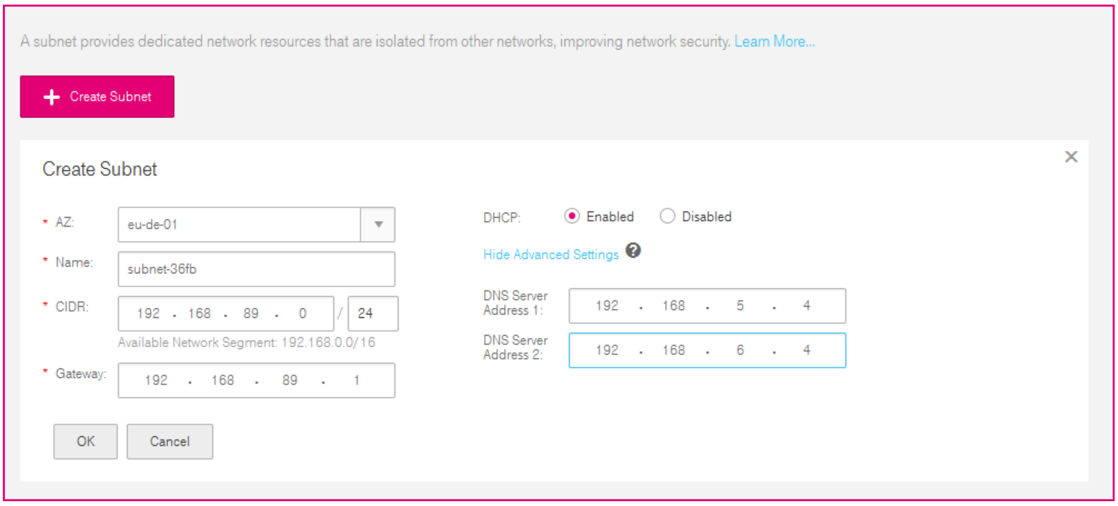
```
linux@inner-dns01:~> dig @172.16.10.4 -x 172.16.20.4

; <<>> DiG 9.9.6-P1 <<>> @172.16.10.4 -x 172.16.20.4
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 26605
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 3

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;4.20.16.172.in-addr.arpa.      IN      PTR

;; ANSWER SECTION:
4.20.16.172.in-addr.arpa. 604800 IN     PTR     ftp.otcuser.org.

;; AUTHORITY SECTION:
16.172.in-addr.arpa.    604800  IN      NS      inner-dns01.otcuser.org.
16.172.in-addr.arpa.    604800  IN      NS      inner-dns02.otcuser.org.

;; ADDITIONAL SECTION:
inner-dns01.otcuser.org. 86400  IN      A       172.16.10.4
inner-dns02.otcuser.org. 86400  IN      A       172.16.20.4

;; Query time: 0 msec
;; SERVER: 172.16.10.4#53(172.16.10.4)
;; WHEN: Fri Feb 24 09:39:56 UTC 2017
;; MSG SIZE  rcvd: 166
```

Now that we have a working DNS server, we can set the name server 172.16.10.4/172.16.20.4 into other VMs or as the default DNS server in the subnet.

Set Default DNS for a New Subnet

Below are two examples for a newly created subnet, which should be used by newly created ECS servers.

The first one is an example about the new subnet in the availability zone eu-de-01. We set the DNS 192.168.5.4 with higher priority for a better reliability.

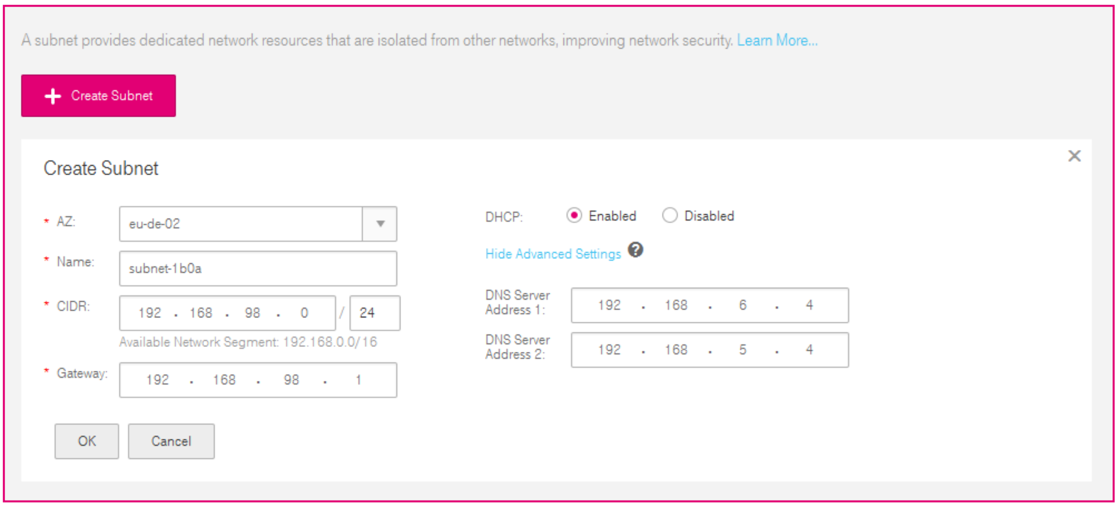


A subnet provides dedicated network resources that are isolated from other networks, improving network security. Learn More...

Create Subnet

* AZ:	eu-de-01
* Name:	subnet-36fb
* CIDR:	192 . 168 . 89 . 0 / 24
	Available Network Segment: 192.168.0.0/16
* Gateway:	192 . 168 . 89 . 1
DHCP:	● Enabled ○ Disabled
DNS Server Address 1:	192 . 168 . 5 . 4
DNS Server Address 2:	192 . 168 . 6 . 4

In the same way, we use the DNS 192.168.6.4 with higher priority in the availability zone eu-de-02.



A subnet provides dedicated network resources that are isolated from other networks, improving network security. Learn More...

* AZ: eu-de-02

* Name: subnet-1b0a

* CIDR: 192 . 168 . 98 . 0 / 24

Available Network Segment: 192.168.0.0/16

* Gateway: 192 . 168 . 98 . 1

DHCP: ● Enabled ○ Disabled

Hide Advanced Settings

DNS Server Address 1: 192 . 168 . 6 . 4

DNS Server Address 2: 192 . 168 . 5 . 4

DNS Maintenance

In the daily operation and maintenance work we may need to add or remove some DNS records, here is a guide for that.

Add a new DNS Record

To add a domain with multiple domain names, we need to do the following configuration work. First, the new zone file should be created, we recommend to create the zone file in the path:

`/var/lib/named/zones/.`

For example,

`/var/lib/named/zones/test/db.test.com` for `zone test.com.`

The following zone file db.test.com will hold a DNS record to assist a nameserver resolve a fully qualified domain name to an IP address. Create and save db.test.com with the following content:

```
;
; BIND data file for test.com
```

```
$TTL    3h
@               IN      SOA     ns1.test.com. admin.test.com. (
                1       ; Serial
                3h      ; Refresh after 3 hours
                1h      ; Retry after 1 hour
                1w      ; Expire after 1 week
                1h )    ; Negative caching TTL of 1 day
;
@               IN      NS      ns1.test.com.
@               IN      NS      ns2.test.com.
test.com.       IN      A       172.16.10.100
ns1             IN      A       172.16.10.100
ns2             IN      A       172.16.20.100
www             IN      CNAME   test.com.
mail            IN      A       172.16.20.100
ftp             IN      CNAME   test.com.
```

Then, we should insert the zone file name into BIND's configuration file `named.conf.local`. To do that we need to add the following lines to this file:

```
zone "test.com" {
        type master;
        file "/var/lib/named/zones/test/db.test.com";
};
```

Before we attempt to make the new zone work, we also should check configuration files by running the following command:

```
named-checkconf
```

To check a DNS zone files to ensure the new added zone has been loaded we can use named-checkzone command:

```
named-checkzone test.com /var/lib/named/zones/test/db.test.com
zone test.com/IN: loaded serial 1
OK
```

Finally we should restart the DNS service or use rndc reload to let the new added record work.

```
service named restart

Stopping domain name service...: BIND9.

Starting domain name service...: BIND9.
```

Remove a DNS Record

To remove a domain record, we need to perform the following steps:

First, remove the record from the zone file. For example, in /var/lib/named/zones/test/db.test.com for zone test.com, we remove the record for domain name mail.test.com, which is marked with red below.

```
;

; BIND data file for test.com
```

```
$TTL    3h

@               IN      SOA     ns1.test.com. admin.test.com. (

                1       ; Serial

                3h      ; Refresh after 3 hours

                1h      ; Retry after 1 hour

                1w      ; Expire after 1 week

                1h )    ; Negative caching TTL of 1 day

;

@               IN      NS      ns1.test.com.

@               IN      NS      ns2.test.com.

test.com.       IN      A       172.16.10.100

ns1             IN      A       172.16.10.100

ns2             IN      A       172.16.20.100

www             IN      CNAME   test.com.

mail            IN      A       172.16.20.100

ftp             IN      CNAME   test.com.
```

To check the DNS zone files after removal, we can use the `named-checkzone` command:

```
named-checkzone test.com /var/lib/named/zones/test/db.test.com

zone test.com/IN: loaded serial 1

OK
```

Finally we should restart the DNS service to let the newly added record work.

```
service named restart

Stopping domain name service...: BIND9.

Starting domain name service...: BIND9.
```

Until now, we have finished all the configuration on DNS servers. With this article, you can get a fully functional DNS service including forward DNS resolution and reverse DNS resolution. Also your DNS service can support high availability features which means even if one of the DNS server will fail, your DNS service will keep running. It is a good way to implement a reliable DNS service in your application on the Open Telekom Cloud.

NOCH FRAGEN?
Weitere Informationen erhalten Sie über:
- Deutschland: 0800 330 44 77 freecall
- International: +800 44 556600 freecall
- Kontaktformular: cloud.telekom.de/e-mail
- telekom.de/open-telekom-cloud

HERAUSGEBER
T-Systems International GmbH
Digital Division
Hahnstraße 43d
60528 Frankfurt am Main, Deutschland