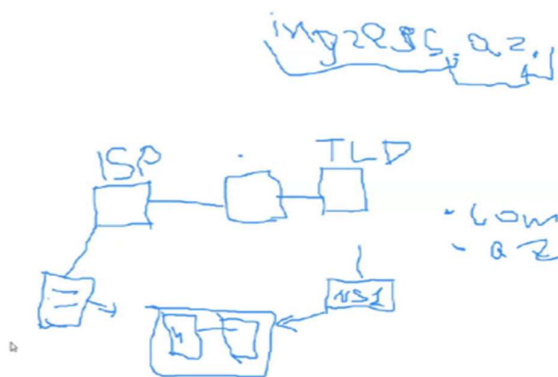


## DNS

DNS nə üçündür? Biz browserdə saytlara girmək üçün IP yazmalıyıq. Amma IP çox uzun olduğu üçün onu yadda saxlamaq olmur. Çünki rəqəmlər yadda saxlamaq çətinidir. Buna görə də biz DNS istifadə edirik. DNS IP-ni ada, adıda IP-yə resolv edir. DNS necə işləyir?

1. İlk olaraq browser-in cache-nə baxır. (TTL vaxtı qədər cache-də qalır.)
2. Daha sonra local host faylına baxır.
3. OS səviyyəsində cache baxır.
4. DNS server varsa ona baxır.
5. Daha sonra ISP-yə baxır.
6. 13 root DNS server-ə baxır. (Burada Top-Level-Domain məlumatlarını öyrənir.)
7. Name Server-ə yönləndirilir.



Biz burada Name Server-in Public IP-sini Domain aldığımız sayta qeyd edirik. Təhlükəsizlik üçün master Name Server qurulur. Və bu global-a çıxarılmır. Bunun yerinə Slave Name Server qurulur, və global-a çıxarılır.

#nslookup google.com

```
[root@localhost ~]# nslookup google.com
Server:         192.168.149.2
Address:        192.168.149.2#53

Non-authoritative answer:
Name:   google.com
Address: 142.251.140.14
Name:   google.com
Address: 2a00:1450:4017:813::200e
```

#dig yandex.ru

```
[root@localhost ~]# dig yandex.ru

;<<<> DiG 9.16.23-RH <<<> yandex.ru
;; global options: +cmd
;; Got answer:
;;->HEADER<<- opcode: QUERY, status: NOERROR, id: 41581
;; flags: qr rd ra; QUERY: 1, ANSWER: 4, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; MBZ: 0x0005, udp: 1232
;; QUESTION SECTION:
;yandex.ru.                IN      A

;; ANSWER SECTION:
yandex.ru.                 5      IN      A      5.255.255.70
yandex.ru.                 5      IN      A      5.255.255.77
yandex.ru.                 5      IN      A      77.88.55.88
yandex.ru.                 5      IN      A      77.88.55.60

;; Query time: 8 msec
;; SERVER: 192.168.149.2#53(192.168.149.2)
;; WHEN: Wed Jan 17 20:36:29 +04 2024
;; MSG SIZE rcvd: 102
```

```
# dig +trace yandex.ru
```

```
[root@localhost ~]# dig +trace yandex.ru

; <<> DiG 9.16.23-RH <<> +trace yandex.ru
;; global options: +cmd
.      5      IN      NS      j.root-servers.net.
.      5      IN      NS      k.root-servers.net.
.      5      IN      NS      l.root-servers.net.
.      5      IN      NS      m.root-servers.net.
.      5      IN      NS      a.root-servers.net.
.      5      IN      NS      b.root-servers.net.
.      5      IN      NS      c.root-servers.net.
.      5      IN      NS      d.root-servers.net.
.      5      IN      NS      e.root-servers.net.
.      5      IN      NS      f.root-servers.net.
.      5      IN      NS      g.root-servers.net.
.      5      IN      NS      h.root-servers.net.
.      5      IN      NS      i.root-servers.net.
.      5      IN      RRSIG  NS 8 0 518400 20240130050000 20240117040000 30f

ru.    172800 IN      NS      a.dns.ripn.net.
ru.    172800 IN      NS      f.dns.ripn.net.
ru.    172800 IN      NS      b.dns.ripn.net.
ru.    172800 IN      NS      d.dns.ripn.net.
ru.    172800 IN      NS      e.dns.ripn.net.
ru.    86400  IN      DS      18274 8 2 AB3501703F39EB42CEE14C6273247938033EEEA9F5CAA70B38580BF 4B
D3E87B
ru.    86400  IN      RRSIG  DS 8 1 86400 20240130050000 20240117040000 30903 . 3cdreVYXGcD3coCwbY
```

```
YANDEX.RU. 345600 IN      NS      ns1.yandex.ru.
YANDEX.RU. 345600 IN      NS      ns2.yandex.ru.
J20C00KHUA3CUMNKST289FF06U25Q91.ru. 3600 IN NSEC3 1 1 0 - J21C11SHOOTMOEQKPRM91CBAGL4886M6 NS SOA RRSIG DNSKEY NSEC3
PARAM
J20C00KHUA3CUMNKST289FF06U25Q91.ru. 3600 IN RRSIG NSEC3 8 2 3600 20240127131355 20231215181945 44301 ru. LCkuUEjOZa6
VnLNRP1nlyS8dPoQ95vLMZ2XXTAcLghnpkyVcvZeJMuIi JGkF92VYfAJTZ/19vAj86qZAgRur65djtZjRKLIAqqJbcnp7vk5wsodv V6LoFcgyu3NlF/
otqVMScVZCeNVSZTrk4TmZyC828qTPx0j+LTy3tP5 rvE=
VJH3PPLSTIU7RJRUH4A0TH2P9E83M9P8.ru. 3600 IN NSEC3 1 1 0 - VJDEGE01PD6MC6EJ36UVD2G04HB9Q7DR NS DS RRSIG
VJH3PPLSTIU7RJRUH4A0TH2P9E83M9P8.ru. 3600 IN RRSIG NSEC3 8 2 3600 20240215192320 20240116182007 44301 ru. tFEyAs1YXW/
FTjwZ5C0mj5rXlaG27EMrfyfu+mw9eyfyv1TiGnAGLg Tm+jw72neb2izriq335Zvn3Hek0Am5kaPdJo5nAuduKeIr8GCKa/beRz GuKTpHZxdq70K0
VXaxP3jB0jDp;03oovYHN+U1T1Hd0+KNUrh/F0W9NN 0VI=
;; Received 669 bytes from 193.232.142.17#53(e.dns.ripn.net) in 112 ms

yandex.ru. 300 IN      A      5.255.255.70
yandex.ru. 300 IN      A      77.88.55.60
yandex.ru. 300 IN      A      5.255.255.77
yandex.ru. 300 IN      A      77.88.55.88
yandex.ru. 604800 IN     NS      ns2.yandex.ru.
yandex.ru. 604800 IN     NS      ns1.yandex.ru.
;; Received 254 bytes from 213.180.193.1#53(ns1.yandex.ru) in 68 ms
```

```
# dig -x 8.8.8.8
```

```
[root@localhost ~]# dig -x 8.8.8.8

; <<> DiG 9.16.23-RH <<> -x 8.8.8.8
;; global options: +cmd
;; Got answer:
;; -->HEADER<<- opcode: QUERY, status: NOERROR, id: 58106
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; MBZ: 0x0005, udp: 1232
;; QUESTION SECTION:
;8.8.8.8.in-addr.arpa.      IN      PTR

;; ANSWER SECTION:
8.8.8.8.in-addr.arpa. 5      IN      PTR      dns.google.

;; Query time: 11 msec
;; SERVER: 192.168.149.2#53(192.168.149.2)
;; WHEN: Wed Jan 17 20:39:26 +04 2024
;; MSG SIZE rcvd: 73
```

## Install DNS (BIND)

```
yum -y install bind bind-utils  
firewall-cmd --add-service=dns --permanent;firewall-cmd --reload
```

systemctl status named (distroya görə dəyişkənlik göstərir)

## Configure DNS (BIND)

```
vim /etc/named.conf
```

```
listen-on port 53 { 127.0.0.1; 192.168.149.129; };  
allow-query { localhost; any; };
```

( Allow-query kimlərdən gələcək olan requestləri resolve edəcəyini göstərir. Any yazıldığı üçün istənilən hostdan gələn request resolve ediləcək.)

Recursion – Əgər no olarsa, “məndə əgər A recordu varsa cavab qaytar, əgər yoxdusa ilişib qalır”. Əgər yes olarsa DNS flow recursion sayılır. Forwarder – Əgər məndə yoxdursa təyin olunmuş server-ə gedir. Məsələn 8.8.8.8 .

```
recursion yes;
```

## Create Forward Zones

```
zone "ingress.local" IN {
```

```
type master;
```

```
file "/var/named/fwd.ingress.local.db";
```

```
allow-update { none; };
```

```
zone "ingress.local." IN {  
    type master;  
    file "ingress-forward.local.db";  
    allow-update {none;};  
};
```

Servisin check olunması üçün istifadə edilir

```
[root@localhost ~]# named-checkconf
```

Ingress.local. domain-ə aid ingress-forward.local.db zone-sini kontrol elə.

```
[root@localhost named]# named-checkzone ingress.local. ingress-forward.local.db  
zone ingress.local/IN: loaded serial 0  
OK
```

## Create Forward Zone Files

```
vim /var/named/fwd.ingress.local.db
```

```
$TTL 3H
@ IN SOA @ ingress.local. (
    0 ; serial
    1D ; refresh
    1H ; retry
    1W ; expire
    3H ) ; minimum
    NS @
    A 127.0.0.1
    AAAA ::1
@ IN NS node01.ingress.local.
```

```
node01 IN A 10.55.8.21
node02 IN A 10.55.8.22
node07 IN CNAME node02.ingress.local.
mail IN A 10.55.8.23
@ IN MX 10 mail.ingress.local
```

```
$TTL 3H
@ IN SOA ingress.local. (
    00 ; serial
    1D ; refresh
    1H ; retry
    1W ; expire
    3H ) ; minimum
    NS
    A 127.0.0.1
    AAAA ::1
master IN A 192.168.149.129
slave IN A 192.168.149.135
test IN A 192.168.149.192
```

```
[root@localhost named]# nslookup test.ingress.local
Server: 192.168.149.129
Address: 192.168.149.129#53

Name: test.ingress.local
Address: 192.168.149.192
```

## Create Reverse Zone

```
zone "149.168.192.in-addr.arpa"  
    IN {  
        type master;  
        file "192.168.149.zone";  
        forwarders {};  
    };
```

```
zone "149.168.192.in-addr.arpa." IN {  
    type master;  
    file "192.168.149.zone";  
    forwarders {};  
};
```

## Create Reverse Zone Files

```
$TTL 3H  
@ IN SOA @ ingress.local. (  
    2 ; serial  
    1M ; refresh  
    1H ; retry  
    1W ; expire  
    3H) ; minimum  
; owner TTL CL type  
RDATA  
600 IN NS  
ns1.ingress.local.  
131 IN PTR  
master.ingress.local.  
132 IN PTR  
mailserver.ingress.local.  
130.8.55.10.in-addr.arpa. IN  
PTR slave.ingress.local.  
134 IN PTR  
server2.ingress.local.
```

```
$TTL 3H  
@ IN SOA @ ingress.local. (  
    002 ; serial  
    1M ; refresh  
    1H ; retry  
    1W ; expire  
    3H) ; minimum  
; owner TTL CL type RDATA  
600 IN NS ns1.ingress.local.  
135 IN PTR slave  
129 IN PTR master
```

## Slave configuration

```
allow-transfer { localhost; 192.168.149.129; }; add to master BIND  
yum -y install bind bind-utils  
firewall-cmd --add-service=dns --permanent; firewall-cmd --reload
```

```
zone "ingress.az" IN {  
    type slave;  
    file "/var/named/fwd.ingress.az.db";  
    masters { 10.55.8.24; };  
    masterfile-format text;
```

```
zone "ingress.local." IN {  
    type slave;  
    file "ingress.local.db";  
    masters { 192.168.149.129; };  
    masterfile-format text;  
};
```

```
zone "8.55.10.in-addr.arpa" IN {  
    type slave;  
    file "10.5.8.zone";  
    masters { 10.55.8.22; };  
    masterfile-format text;
```

```
zone "149.168.192.in-addr.arpa." IN {  
    type slave;  
    file "192.168.149.zone";  
    masters { 192.168.149.129; };  
    masterfile-format text;  
};
```

- **Primary Name Server** – The nameserver that contains the original zone file and not an AXFR transferred copy.
- **Hostmaster Email** – Address of the party responsible for the zone. A period “.” is used in place of an “@” symbol. For email addresses that contain a period, this will be escaped with a slash “/”.
- **Serial Number** – Version number of the zone. As you make changes to your zone file, the serial number will increase.
- **Time To Refresh** – How long in seconds a nameserver should wait prior to checking for a Serial Number increase within the primary zone file. An increased Serial Number means a transfer is needed to sync your records. Only applies to zones using [secondary DNS](#).
- **Time To Retry** – How long in seconds a nameserver should wait prior to retrying to update a zone after a failed attempt. Only applies to zones using [secondary DNS](#).
- **Time To Expire** – How long in seconds a nameserver should wait prior to considering data from a secondary zone invalid and stop answering queries for that zone. Only applies to zones using [secondary DNS](#).
- **Minimum TTL** – How long in seconds that a nameserver or resolver should cache a negative response.