

1. Konfigurujemy karty sieciowe na maszynie z Ubuntu

- * pierwsza karta sieciowa – zmostkowana
- * druga karta sieciowa – wewnętrzna intnet

jeżeli posiadamy, możemy przywrócić odpowiednią migawkę z kartami już skonfigurowanymi

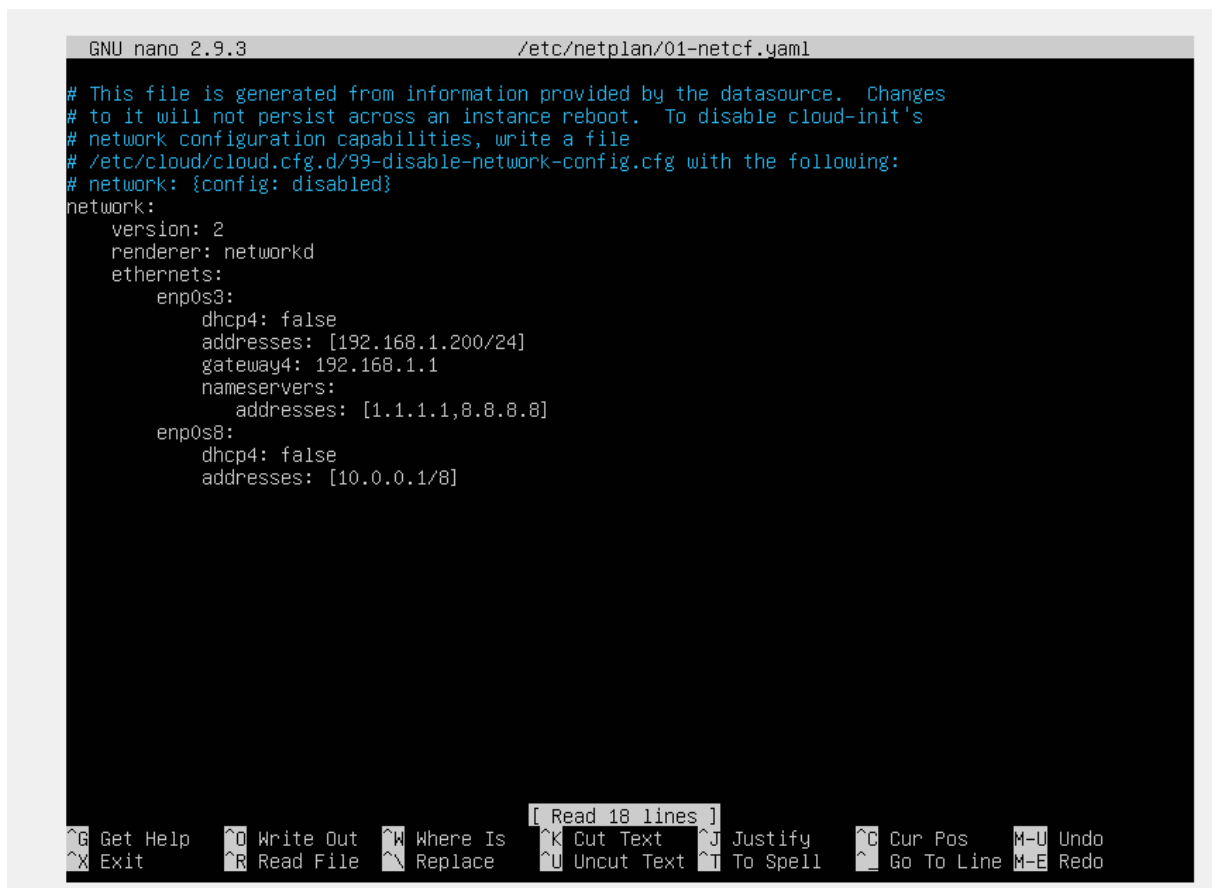
2. Konfigurujemy karty sieciowe na maszynie testowej z WinXP:

- * pierwsza karta sieciowa - zmostkowana

3. Uruchamiamy maszynę z Ubuntu , sprawdzamy ustawienia adresów IP

nano /etc/netplan/01-netcf.yaml

ustawiamy na pierwszej sieciówce adres pasujący do adresacji w naszej sieci domowej, u mnie to:



```
GNU nano 2.9.3 /etc/netplan/01-netcf.yaml
# This file is generated from information provided by the datasource. Changes
# to it will not persist across an instance reboot. To disable cloud-init's
# network configuration capabilities, write a file
# /etc/cloud/cloud.cfg.d/99-disable-network-config.cfg with the following:
# network: {config: disabled}
network:
  version: 2
  renderer: networkd
  ethernets:
    enp0s3:
      dhcp4: false
      addresses: [192.168.1.200/24]
      gateway4: 192.168.1.1
      nameservers:
        addresses: [1.1.1.1,8.8.8.8]
    enp0s8:
      dhcp4: false
      addresses: [10.0.0.1/8]
```

po ustawieniu odpowiednich adresów zatwierdzamy ustawienia za pomocą **netplan apply** i sprawdzamy za pomocą **ip a**, sprawdzamy też za pomocą polecenia ping czy mamy łączność z internetem.

4. Aktualizujemy zawartość repozytoriów:

```
sudo apt update
```

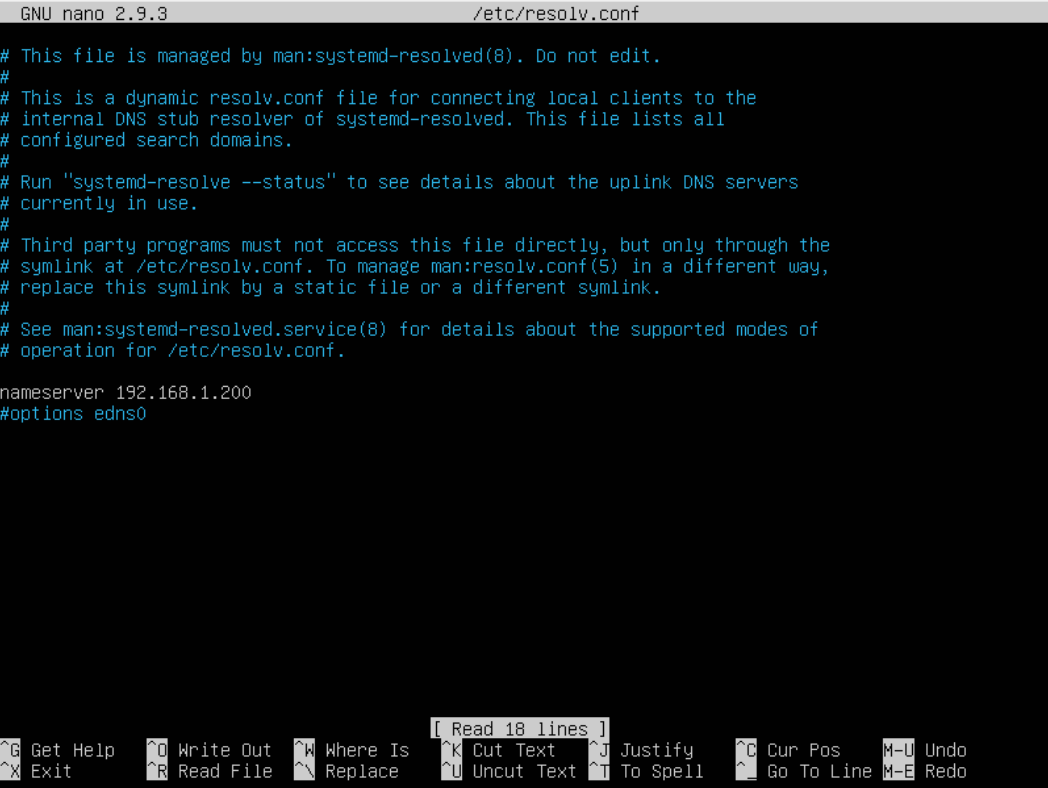
5. Instalujemy serwer DNS:

```
sudo apt install bind9
```

Folder gdzie serwer DNS będzie trzymał swoje pliki konfiguracyjne to:

```
/etc/bind/
```

6. Ustawiamy w pliku `/etc/resolv.conf` nasz adres IP jako adres serwera DNS:



```
GNU nano 2.9.3 /etc/resolv.conf
# This file is managed by man:systemd-resolved(8). Do not edit.
#
# This is a dynamic resolv.conf file for connecting local clients to the
# internal DNS stub resolver of systemd-resolved. This file lists all
# configured search domains.
#
# Run "systemd-resolve --status" to see details about the uplink DNS servers
# currently in use.
#
# Third party programs must not access this file directly, but only through the
# symlink at /etc/resolv.conf. To manage man:resolv.conf(5) in a different way,
# replace this symlink by a static file or a different symlink.
#
# See man:systemd-resolved.service(8) for details about the supported modes of
# operation for /etc/resolv.conf.

nameserver 192.168.1.200
#options edns0

[ Read 18 lines ]
^G Get Help  ^O Write Out  ^W Where Is   ^K Cut Text    ^J Justify    ^C Cur Pos    M-U Undo
^X Exit      ^R Read File  ^\ Replace   ^U Uncut Text ^T To Spell   ^G Go To Line M-E Redo
```

7. Konfiguracja serwera DNS:

Serwer DNS ma następujące pliki konfiguracyjne:

- `/etc/bind/named.conf` - główny plik konfiguracyjny zawierający instrukcje dołączania pozostałych plików konfiguracyjnych
- `/etc/bind/named.conf.options` - opcje serwera DNS
- `/etc/bind/named.conf.local` - definicje stref lokalnych
- `/etc/bind/named.conf.default-zones` - konfiguracja domyślnych stref (localhost, serwery główne - root servers)

Uwaga: w plikach konfiguracyjnych `named.conf.*` znaki `//` oznaczają komentarz jednolinijkowy.

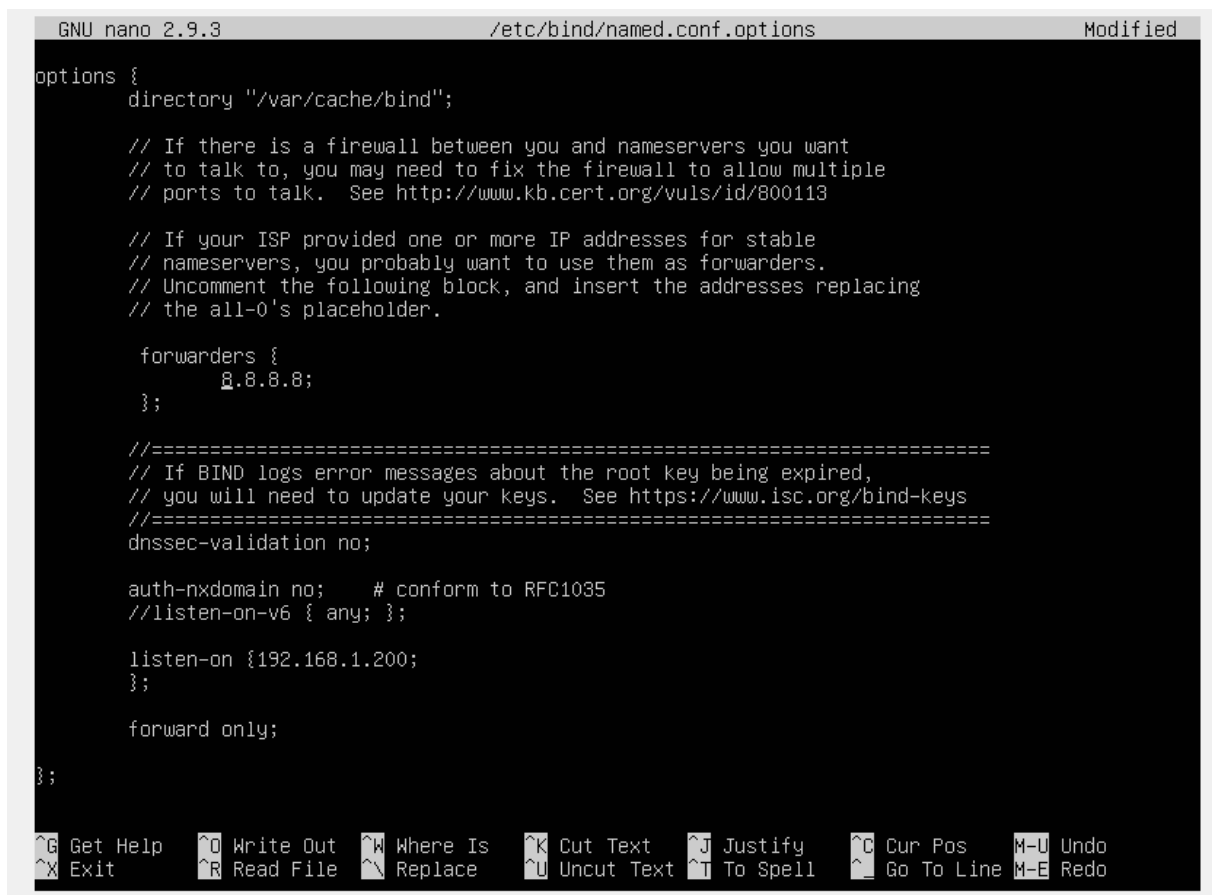
Na początku trzeba zacząć od zrobienia kopii zapasowych tych plików:

```
cp /etc/bind/named.conf /etc/bind/named.conf.bac
cp /etc/bind/named.conf.options /etc/bind/named.conf.options.bac
cp /etc/bind/named.conf.local /etc/bind/named.conf.local.bac
cp /etc/bind/named.conf.default.zones /etc/bind/named.conf.default.zones.bac
```

Teraz możemy przystąpić do faktycznej konfiguracji:

W pliku ***/etc/bind/named.conf.options*** określa się globalną konfigurację serwera DNS.

ustawiamy tutaj na którym interfejsie ma działać nasz serwer DNS, adresy serwerów DNS do których będzie przekazywał zapytania których sam nie potrafi rozwiązać, czy też reguły ACL kto może się odpytywać naszego serwera DNS – ja ustawiłem że serwer ma działać na adresie 192.168.1.200 i ma się odpytywać serwer DNS google, jeżeli czego nie będzie wiedział.



```
GNU nano 2.9.3 /etc/bind/named.conf.options Modified
options {
    directory "/var/cache/bind";

    // If there is a firewall between you and nameservers you want
    // to talk to, you may need to fix the firewall to allow multiple
    // ports to talk.  See http://www.kb.cert.org/vuls/id/800113

    // If your ISP provided one or more IP addresses for stable
    // nameservers, you probably want to use them as forwarders.
    // Uncomment the following block, and insert the addresses replacing
    // the all-0's placeholder.

    forwarders {
        8.8.8.8;
    };

    //=====
    // If BIND logs error messages about the root key being expired,
    // you will need to update your keys.  See https://www.isc.org/bind-keys
    //=====
    dnssec-validation no;

    auth-nxdomain no;    # conform to RFC1035
    //listen-on-v6 { any; };

    listen-on {192.168.1.200;
    };

    forward only;
};

^G Get Help  ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos   M-U Undo
^X Exit      ^R Read File  ^_ Replace   ^U Uncut Text ^T To Spell   ^_ Go To Line M-E Redo
```

Inna przykładowa konfiguracja (z kursu na moodle) z opisem parametrów – oczywiście jeżeli będziemy jej chcieli użyć to musimy dopasować do naszej adresacji:

```
Plik named.conf.options CodeHL 1.7 [Command Prompt]
1 #utworzenie grupy ZaufaniKlienci mogącej odpytywać ten serwer DNS
2 acl ZaufaniKlienci {
3     192.168.0.0/24;
4     localhost;
5     localnets; #lokalne sieci serwera (wszystkie interfejsy)
6 };
7
8 options {
9 #lokalizacja pamięci podręcznej serwera DNS:
10     directory "/var/cache/bind";
11
12 #zezwolenie na odpytywanie tego serwera DNS tylko przez określonych klientów (np. z utworzonej grupy):
13     allow-query {
14         ZaufaniKlienci;
15         10.0.0.0/8;
16     };
17
18 #Adresy IP serwerów DNS, do których będą wysyłane dalsze zapytania.
19     forwarders {
20         8.8.8.8;
21         8.8.4.4;
22     };
23
24 #wybranie adresów IP, na których serwer DNS będzie nasłuchiwał:
25     listen-on {
26         192.168.0.1;
27     };
28
29 #forward (only | first)
30 #forward only - zapytania przesyłane są wyłącznie do adresów określonych w sekcji forwarders
31 #forward first - (domyślnie) jeżeli serwery z sekcji forwarders nie są w stanie udzielić odpowiedzi,
32 #                 serwer podejmie próbę rozwiązania nazwy np. za pomocą serwerów najwyższego poziomu (root)
33     forward only;
34 };
```

Sprawdzanie poprawności konfiguracji.

Polecenie *named-checkconf* sprawdza plik konfiguracyjny serwera DNS bind9 (*/etc/bind9/named.conf.options*) i wyświetla ewentualne błędy.

8. Utworzenie lokalnych stref wyszukiwania

Serwer DNS potrafi wyszukiwać do przodu – zamieniać nazwy domenowe na adresu IP oraz wyszukiwać wstecz – zamieniać IP na nazwy domenowe. Oczywiście, żeby potrafił to zrobić, musimy zdefiniować strefy wyszukiwania:

nano /etc/bind/named.conf.local

W naszym przypadku zdefiniujemy strefę wyszukiwania w przód dla domeny **zsmc.local**, oraz strefę wyszukiwania wstecz dla adresów należących do sieci **192.168.1.0/24**

W pliku *named.conf.local* definiowane są strefy lokalne: wyszukiwania w przód (*ang. forward zone*) i wstecz (*ang. reverse zone*). Serwer bind9 definiuje całe strefy (a nie domeny / nazwy domenowe) czyli grupy nazw domen, subdomen i hostów.

Przykładowa konfiguracja stref z opisem parametrów (będziemy musieli ją przerobić tak żeby pasowała do naszej adresacji:)

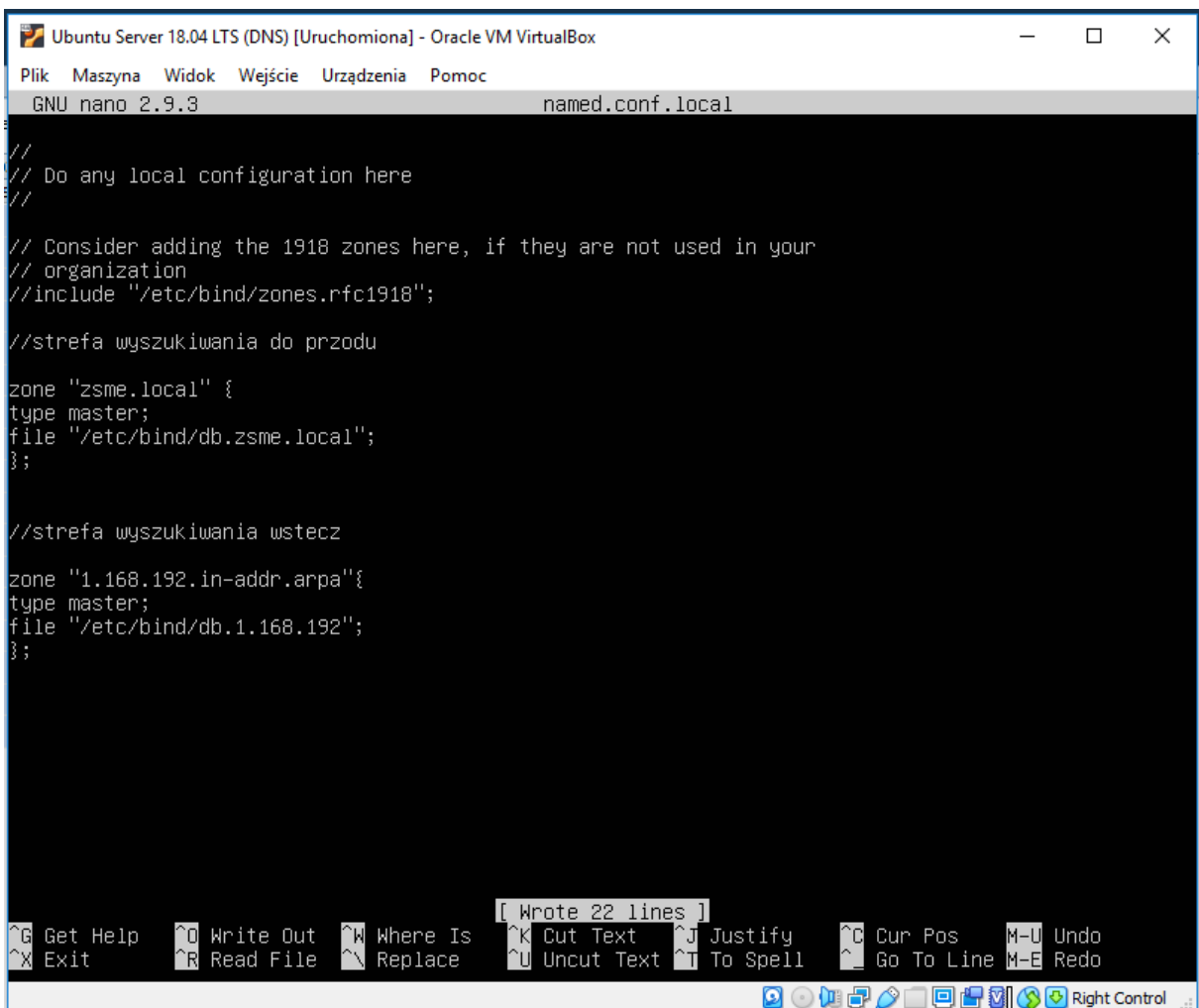
Definicja strefy wyszukiwania w przód i wstecz.

```
Strefa wyszukiwania w przód: named.conf.local CodeHL 1.7 [Command Prompt]
1 #definicja strefy wyszukiwania w przód np.: zsmc.local - 192.168.0.1
2
3 zone "zsmc.local" {
4     #typ strefy: master | slave
5     type master;
6
7     #lokalizacja pliku z definicją danej strefy
8     file "/etc/bind/db.zsmc.local";
9
10    #zezwozenie na wysyłanie danych do dodatkowego serwera DNS
11    #allow-transfer { 192.168.0.2; };
12
13    #serwer dodatkowy - slave - powinien mieć zdefiniowany adres serwera głównego
14    #masters { 192.168.0.1; };
15 };
16
17 #definicja strefy wyszukiwania wstecz np.: 192.168.0.1 - zsmc.local
18 zone "0.168.192.in-addr.arpa" {
19     #typ strefy: master | slave
20     type master;
21
22     #lokalizacja pliku z definicją danej strefy
23     file "/etc/bind/db.0.168.192";
24
25     #zezwozenie na wysyłanie danych do dodatkowego serwera DNS
26     #allow-transfer { 192.168.0.2; };
27 };
```

Typ strefy definiowany parametrem **type** określa czy dany serwer jest serwerem głównym (**master**) czy pomocniczym (**slave**). Serwery pomocnicze stosowane są w celu zapewnienia ciągłej dostępności usługi DNS np. w sytuacji gdy serwer główny ulegnie awarii lub jest przeciążony. Serwer główny wysyła aktualizacje danych do serwerów dodatkowych.

Parametr **allow-transfer** definiuje adresy IP dodatkowych serwerów DNS przechowujących kopie danej strefy.

W naszym przypadku plik z definicją stref będzie wyglądał tak:



```
Ubuntu Server 18.04 LTS (DNS) [Uruchomiona] - Oracle VM VirtualBox
Plik Maszyna Widok Wejście Urządzenia Pomoc
GNU nano 2.9.3 named.conf.local
//
// Do any local configuration here
//
// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";
//strefa wyszukiwania do przodu
zone "zsmc.local" {
type master;
file "/etc/bind/db.zsmc.local";
};
//strefa wyszukiwania wstecz
zone "1.168.192.in-addr.arpa"{
type master;
file "/etc/bind/db.1.168.192";
};
[ Wrote 22 lines ]
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos M-U Undo
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line M-E Redo
Right Control
```

Zwróćcie uwagę na sposób definiowania strefy wstecz – nasza przykładowa strefa wyszukiwania wstecz będzie pozwalała na wyszukiwanie rekordów dla adresów należących do sieci **192.168.1.0/24** – w definicji są wpisane pierwsze trzy oktety, więc dodając rekordy do strefy będziemy wpisywać tylko ostatni oktet adresu. Jeżeli na przykład chcielibyśmy zrobić strefę wyszukiwania wstecz dla adresów z sieci **10.0.0.0/8**, to w definicja strfy wstecz wyglądała by następująco:

```
zone "10.in-addr.arpa" {
    #typ strefy: master | slave
    type master;
    file "/etc/bind/db.10";
};
```

9. Utworzenie plików z konfiguracja stref

Musimy stworzyć pliki w których będziemy dodawać wpisy do stref, nazwy i ich położenie muszą być takie jak zdefiniowaliśmy dla poszczególnych stref w pliku named.conf.local. Żeby nie musieć tworzyć plików od zera, możemy skopiować pliki z przykładową konfiguracją stref które są już w katalogu /etc/bind. Plik konfiguracji strefy zsme.local stworzymy na podstawie db.local

```
cp /etc/bind/db.local /etc/bind/db.zsme.local
```

a plik konfiguracji dla strefy wyszukiwania wstecz stworzymy na podstawie pliku db.127

```
cp /etc/bind/db.127 /etc/bind/db.1.168.192
```

Teraz możemy przejść do konfiguracji poszczególnych opcji w plikach:

Konfiguracja **strefy wyszukiwania do przodu** dla domeny **zsme.local**:

zdefiniujemy wpisy dotyczące hostów:

zsme.local ma pokazywać na adres 192.168.1.200

ns1.zsme.local ma pokazywać na adres 192.168.1.200 i jest też serwerem DNS (rekord typu NS)

xp.zsme.local ma pokazywać na adres 1.150 – nasz testowy komputer z XP

komp1.zsme.local ma pokazywać na to samo co xp.zsme.local – rekord typu CNAM (alias)

router.zsme.local – ma pokazywać na adres naszego routera – 192.168.1.1

W plikach z definicjami stref mogą być stosowane znaki o specjalnym znaczeniu:

Znak średnika ; oznacza komentarz jednolinijkowy

Symbol @ - oznacza aktualnie definiowaną strefę (w tym przypadku zsme.local.)

Kropka - na końcu nazwy rekordu oznacza koniec dołączania. Jeżeli jej nie będzie, serwer dołączy do niej nazwę strefy np.: abcd jest równoznaczne z: abcd.zsme.local.

```
GNU nano 2.9.3 db.zsme.local
;
; BIND data file for local loopback interface
;
$TTL 604800
@ IN SOA ns1.zsme.local. admin.zsme.local. (
    2 ; Serial
    604800 ; Refresh
    86400 ; Retry
    2419200 ; Expire
    604800 ) ; Negative Cache TTL
;
@ IN NS ns1.zsme.local.
@ IN A 192.168.1.200
ns1 IN A 192.168.1.200
xp IN A 192.168.1.150
komp1 IN CNAME xp
router IN A 192.168.1.1
```

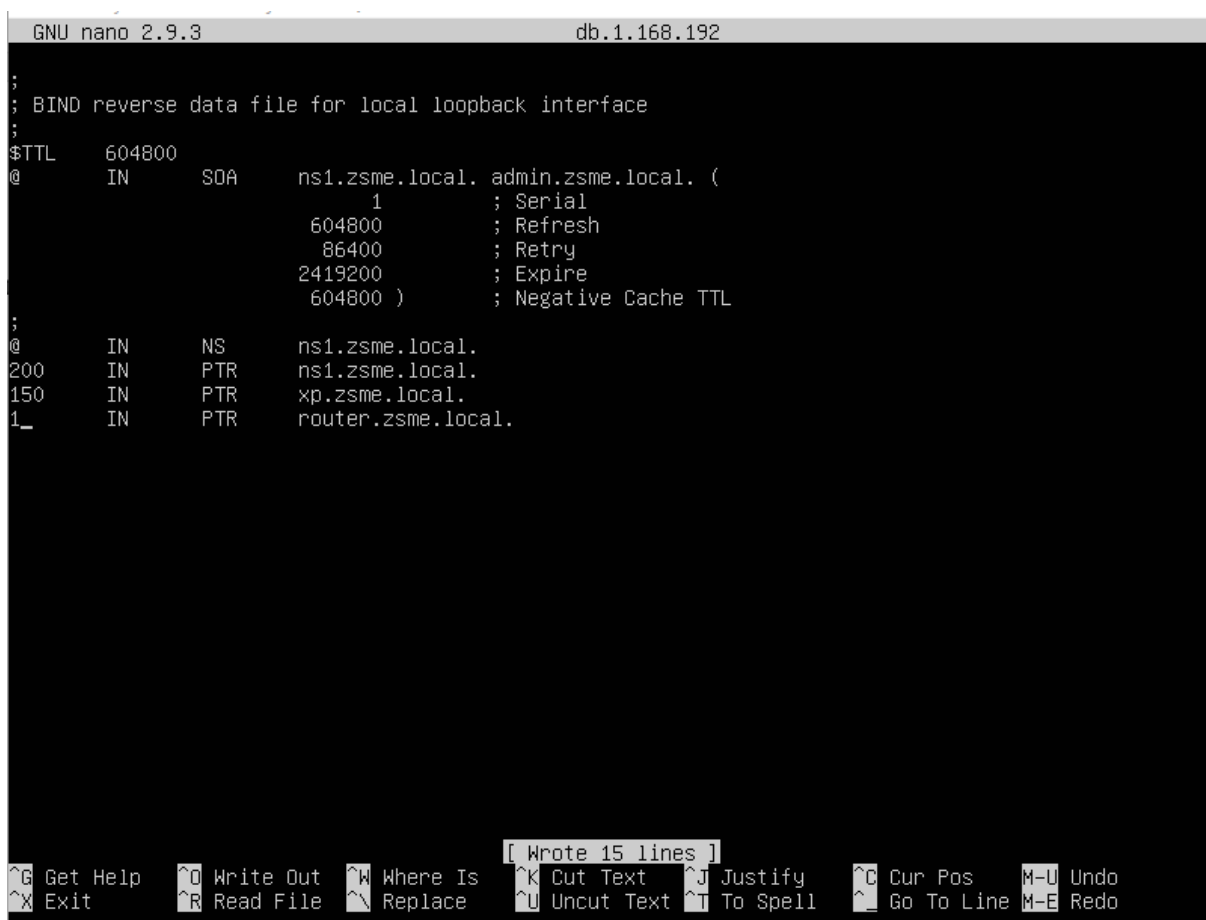
[Wrote 17 lines]

| | | | | | | |
|--------------------|---------------------|--------------------|----------------------|--------------------|----------------------|-----------------|
| ^G Get Help | ^O Write Out | ^W Where Is | ^K Cut Text | ^J Justify | ^C Cur Pos | M-U Undo |
| ^X Exit | ^R Read File | ^_ Replace | ^U Uncut Text | ^T To Spell | ^L Go To Line | M-E Redo |

Konfiguracja strefy wyszukiwania wstecz dla sieci 192.168.1.0/24:

Definiujemy rekordy typu PTR czyli wskaźniki (czyli dany adres IP pokazuje na podana nazwę), zwróćcie uwagę, że podajemy tylko brakujący oktet (bo deklarując strefę w named.conf.locale deklarowaliśmy

zone "1.168.192.in-addr.arpa" - jak widać, **brakuje** nam tylko **jednego oktetu**, stąd dodając rekordy wpisujemy tylko ten oktet – **serwer sklepi to w całość**.



```
GNU nano 2.9.3 db.1.168.192
;
; BIND reverse data file for local loopback interface
;
$TTL      604800
@         IN      SOA      ns1.zsme.local. admin.zsme.local. (
                        1          ; Serial
                        604800     ; Refresh
                        86400      ; Retry
                        2419200    ; Expire
                        604800 )   ; Negative Cache TTL
;
@         IN      NS       ns1.zsme.local.
200       IN      PTR      ns1.zsme.local.
150       IN      PTR      xp.zsme.local.
1_        IN      PTR      router.zsme.local.
```

tak jak wcześniej pisałem – jeżeli zadeklarowalibyśmy strefę wyszukiwania wstecz dla sieci 10.0.0/8:

```
zone "10.in-addr.arpa" {
    #typ strefy: master | slave
    type master;
    file "/etc/bind/db.10";
};
```

dla takiej strefy dodając rekordy musielibyśmy podać 3 brakujące oktety, czyli przykładowy rekord: **adres 10.0.0.100** pokazuje na nazwę **pc1.zsme.local** wyglądał by następująco:

```
100.0.0 IN PTR pc1.zsme.local. ; 10.0.0.100
```

Znaczenie poszczególnych parametrów:

- **Czas ważności**- każdy rekord może mieć określony indywidualny czas ważności (TTL) czyli okres przez jaki systemy rozwiązywania nazw (pomocnicze serwery DNS, usługi cache DNS systemów operacyjnych) powinny przechowywać o nim dane.

| | | | | | CodeHL 1.7 [Command Prompt] |
|---|-----|------|----|---|---|
| 1 | pc1 | 1800 | IN | A | 192.168.0.100 ; Czas ważności rekordu A |

- **\$TTL** - parametr globalny określający czas życia dla wszystkich rekordów (pod warunkiem, że nie mają określonego swojego indywidualnego czasu). Parametr ten musi znajdować się na początku linii (w pierwszej kolumnie, nie może być poprzedzony nawet spacją!). Dla każdego rekordu musi być zdefiniowany parametr TTL (globalny lub indywidualny),
- **IN** - typ sieci (IN - Internet),
- **SOA** - definicja głównego rekordu (ang. start of authority), który określa:
 - **ns1.zsme.local.** - nazwę głównego serwera **DNS** dla tej strefy,
 - **root.zsme.local.** - adres email osoby odpowiedzialnej za administrację strefą (**root@zsme.local** - znak @ zastępowany jest kropką),
 - **serial** - numer seryjny rekordu **SOA**, po każdej aktualizacji powinien być zwiększany. Dostępny zakres: **1 - 4294967295**, liczba typu unsigned int,
 - **refresh** - czas kiedy serwery dodatkowe (**slave**) powinny odświeżyć informacje o strefie z serwera głównego (**master**),
 - **retry** - czas po jakim serwery **slave** ponowią próbę odświeżenia informacji jeżeli wcześniejsza próba połączenia z serwerem **master** się nie powiodła,
 - **expire** - czas po jakim serwery **slave** przestaną traktować dane o strefie jako autorytatywne (pewne),
 - **negative cache TTL** - czas przez jaki powinna być przechowywana informacja, że dany rekord (cała strefa) nie istnieje (max 10800 sekund).

Reprezentacja czasu w usłudze bind9:

Parametry czasu dla rekordów domyślnie podawane są w sekundach - 32 bitowe liczby ze znakiem. Możliwe jest użycie innych jednostek takich jak:

- **s** - sekundy
- **m** - minuty
- **h** - godziny
- **d** - dni
- **w** - tygodnie

Wielkość liter nie ma znaczenia.

| Przykład rekordu SOA | | | | | CodeHL 1.7 [Command Prompt] |
|----------------------|---|-----|----|--------|-----------------------------------|
| 1 | @ | 20H | IN | SOA | ns1.zsme.local. root.zsme.local (|
| 2 | | | | 1234 | ; serial |
| 3 | | | | 10800s | ; refresh = 3 godziny |
| 4 | | | | 15M | ; retry = 15 minut |
| 5 | | | | 2w12h | ; expiry = 2 tygodnie i 12 godzin |
| 6 | | | | 2h30M | ; neg. cache = 2 i pół godziny |
| 7 | | | |) | |

10. Restart usługi i testy

Po skonfigurowaniu stref restartujemy nasz serwer DNS:

service bind9 restart

Możemy sprawdzić czy usługa (daemon) działa poprawnie używając polecenia:

service bind9 status

```
root@jkubuntu:/etc/bind# service bind9 status
• bind9.service - BIND Domain Name Server
  Loaded: loaded (/lib/systemd/system/bind9.service; enabled; vendor preset: enabled)
  Active: active (running) since Tue 2020-11-03 16:17:45 UTC; 1min 42s ago
  Docs: man:named(8)
  Process: 2085 ExecStop=/usr/sbin/rndc stop (code=exited, status=0/SUCCESS)
  Main PID: 2089 (named)
  Tasks: 4 (limit: 1108)
  CGroup: /system.slice/bind9.service
          └─2089 /usr/sbin/named -f -u bind

lis 03 16:17:45 jkubuntu named[2089]: managed-keys-zone: loaded serial 9
lis 03 16:17:45 jkubuntu named[2089]: zone 0.in-addr.arpa/IN: loaded serial 1
lis 03 16:17:45 jkubuntu named[2089]: zone 127.in-addr.arpa/IN: loaded serial 1
lis 03 16:17:45 jkubuntu named[2089]: zone 1.168.192.in-addr.arpa/IN: loaded serial 1
lis 03 16:17:45 jkubuntu named[2089]: zone 255.in-addr.arpa/IN: loaded serial 1
lis 03 16:17:45 jkubuntu named[2089]: zone zsme.local/IN: loaded serial 2
lis 03 16:17:45 jkubuntu named[2089]: zone localhost/IN: loaded serial 2
lis 03 16:17:45 jkubuntu named[2089]: all zones loaded
lis 03 16:17:45 jkubuntu named[2089]: running
lis 03 16:17:45 jkubuntu named[2089]: managed-keys-zone: Key 20326 for zone . acceptance timer compl
lines 1-20/20 (END)
```

Ustawmy teraz żeby nasz Linux korzystał z właśnie przez nas skonfigurowanego serwera DNS. W tym celu musimy edytować plik **/etc/resolv.conf** i wpisać w nim adres naszego serwera DNS:

nano /etc/resolv.conf

```
GNU nano 2.9.3 /etc/resolv.conf
# This file is managed by man:systemd-resolved(8). Do not edit.
#
# This is a dynamic resolv.conf file for connecting local clients to the
# internal DNS stub resolver of systemd-resolved. This file lists all
# configured search domains.
#
# Run "systemd-resolve --status" to see details about the uplink DNS servers
# currently in use.
#
# Third party programs must not access this file directly, but only through the
# symlink at /etc/resolv.conf. To manage man:resolv.conf(5) in a different way,
# replace this symlink by a static file or a different symlink.
#
# See man:systemd-resolved.service(8) for details about the supported modes of
# operation for /etc/resolv.conf.

nameserver 192.168.1.200
options edns0

[ Read 18 lines ]
^G Get Help  ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos    M-U Undo
^X Exit      ^R Read File  ^\ Replace   ^U Uncut Text ^T To Spell   ^_ Go To Line  M-E Redo
```

Do testów użyjemy narzędzia **nslookup** – jest ono zarówno w Linuxie jak i w Windowsie. Jako że ustawiliśmy, żeby naszym DNS był nasz lokalny serwer DNS, to wystarczy wpisać polecenie nslookup a po uruchomieniu programu wpisać nazwę domenową, o którą chcemy odpytać serwer DNS:

```
root@jkubuntu:/etc/bind# nslookup
> zsme.local
Server:          192.168.1.200
Address:         192.168.1.200#53

Name:   zsme.local
Address: 192.168.1.200
>
```

Widzimy że serwerem który odpytujemy jest nasz komputer i zna naszą domenę zsme.local

Zapytajmy się o nasze rekordy, które konfigurowaliśmy:

```

root@jkubuntu:/etc/bind# nslookup
> ns1.zsme.local
Server:      192.168.1.200
Address:     192.168.1.200#53

Name:   ns1.zsme.local
Address: 192.168.1.200
> xp.zsme.local
Server:      192.168.1.200
Address:     192.168.1.200#53

Name:   xp.zsme.local
Address: 192.168.1.150
> komp1.zsme.local
Server:      192.168.1.200
Address:     192.168.1.200#53

komp1.zsme.local      canonical name = xp.zsme.local.
Name:   xp.zsme.local
Address: 192.168.1.150
> router.zsme.local
Server:      192.168.1.200
Address:     192.168.1.200#53

Name:   router.zsme.local
Address: 192.168.1.1
>

```

Widzimy, że wszystkie rekordy które ustawiliśmy w strefie wyszukiwania do przodu działają poprawnie.

Spróbujmy teraz wykonać zapytania wstecz, czyli wpisujemy adres IP a serwer powinien nam odpowiedzieć nazwą domenową:

```

zsme@jkubuntu:~$ nslookup
> 192.168.1.200
200.1.168.192.in-addr.arpa      name = ns1.zsme.local.
> 192.168.1.1
1.1.168.192.in-addr.arpa      name = router.zsme.local.
> 192.168.1.150
150.1.168.192.in-addr.arpa    name = xp.zsme.local.
>

```

Serwer też poprawnie odpowiada na zapytania.

przetestujmy, czy serwet będzie potrafił przekazać zapytanie do forwardera (zdefiniowanego przez nas serwera DNS: 8.8.8.8)

```

zsme@jkubuntu:~$ nslookup
> interia.pl
Server:      192.168.1.200
Address:     192.168.1.200#53

Non-authoritative answer:
Name:   interia.pl
Address: 217.74.65.23
> -

```

Jak widać, serwer odpytany o domenę, której przecież nie ma skonfigurowanej, przekazuje zapytanie do forwardera a ten je rozwiązuje i zwraca odpowiedź.

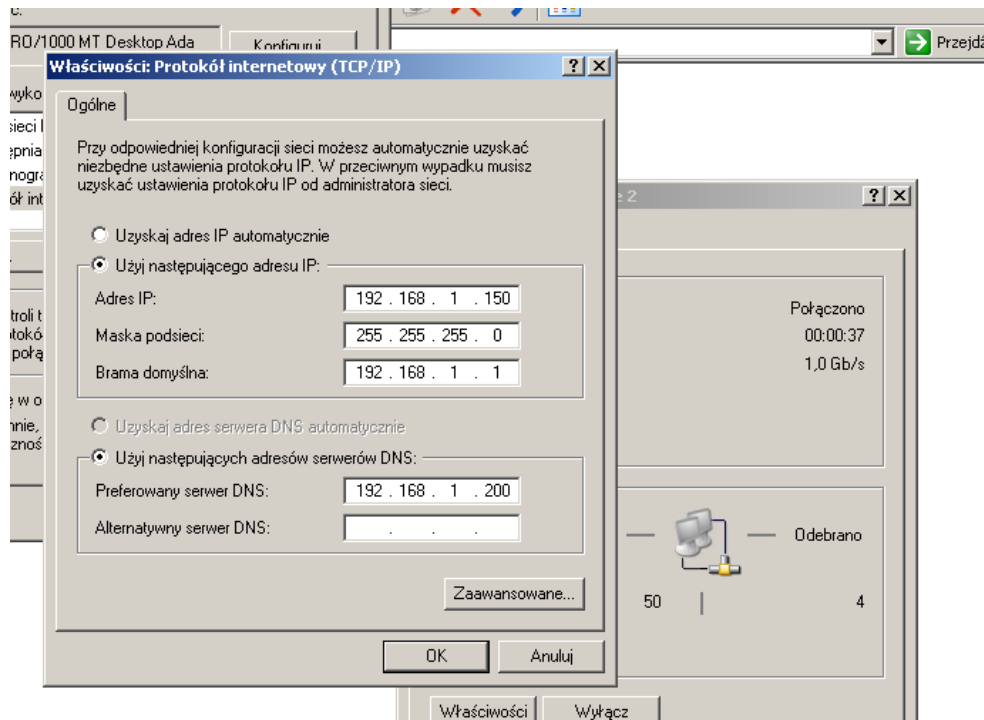
Odpalmy teraz maszynę testową z XP, skonfigurujemy ją do pracy w tej samej sieci z pasującą adresacją:

karta sieciowa: zmostkowana

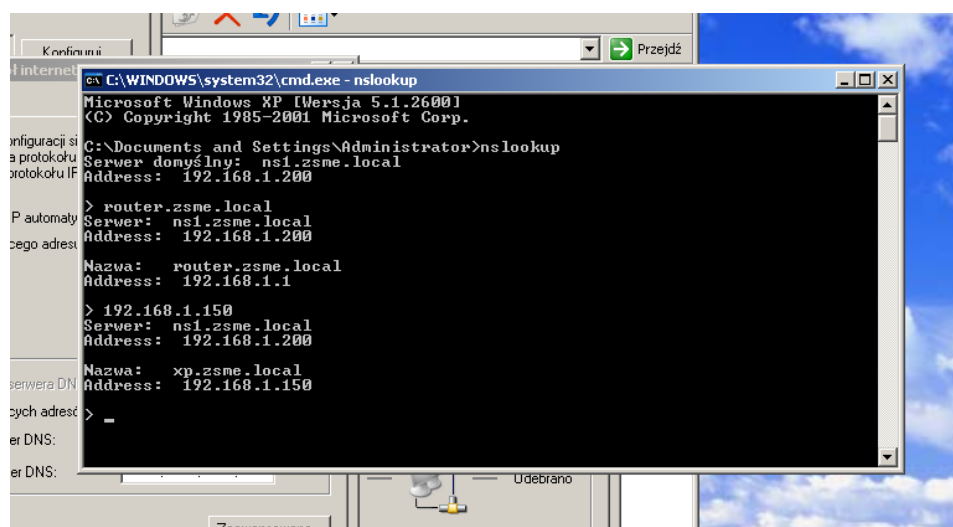
IP:192.168.1.150/24

B:192.168.1.1

DNS:192.168.1.200 (adres naszego Ubuntu Server z uruchomionym serwerem DNS)



Użyjmy też narzędzia nslookup i odpytajmy serwer o parę rekordów:



Jak widać, działa poprawnie. Oczywiście możemy też po prostu puścić ping na nazwę domenową

```
C:\WINDOWS\system32\cmd.exe
C:\Documents and Settings\Administrator>ping ns1.zsme.local
Badanie ns1.zsme.local [192.168.1.200] z użyciem 32 bajtów danych:
Odpowiedź z 192.168.1.200: bajtów=32 czas<1 ms TTL=64
Odpowiedź z 192.168.1.200: bajtów=32 czas<1 ms TTL=64
Odpowiedź z 192.168.1.200: bajtów=32 czas<1 ms TTL=64
Odpowiedź z 192.168.1.200: bajtów=32 czas<1 ms TTL=64
Statystyka badania ping dla 192.168.1.200:
Pakiety: Wysłane = 4, Odebrane = 4, Utracone = 0 (0% straty),
Szacunkowy czas błędzenia pakietów w milisekundach:
Minimum = 0 ms, Maksimum = 0 ms, Czas średni = 0 ms
C:\Documents and Settings\Administrator>_
```

11. NSLOOKUP – dodatek

W momencie, gdy chcemy żeby nslookup nie odpytywał z domyślnego serwera DNS (tego który jest skonfigurowany w systemie) to pod systemem Windows, po uruchomieniu nslookup musimy wpisać polecenie:

server 8.8.8.8 (serwer będzie odpytywał DNS googla zamiast naszego)

```
C:\WINDOWS\system32\cmd.exe - nslookup
C:\Documents and Settings\Administrator>nslookup
Serwer domyślny: ns1.zsme.local
Address: 192.168.1.200

> server 8.8.8.8
Serwer domyślny: dns.google
Address: 8.8.8.8

> onet.pl
Server: dns.google
Address: 8.8.8.8

Nieautorytatywna odpowiedź:
Nazwa: onet.pl
Address: 213.180.141.140

> _
```

W Linuxie musimy uruchomić nslookup z podaniem adresu serwera który ma odpytywać, zamiast domyślnego:

nslookup – 8.8.8.8

```
zsm@jkubuntu:~$ nslookup - 8.8.8.8
> onet.pl
Server:      8.8.8.8
Address:     8.8.8.8#53

Non-authoritative answer:
Name:   onet.pl
Address: 213.180.141.140
>
```

Czyszczenie pamięci podręcznej serwera DNS:

rndc flush

Czyszczenie pamięci podręcznej DNS w Windows:

ipconfig /flushdns