Методическое пособие по выполнению домашнего задания по курсу «Администратор Linux. Professional»

Vagrant-стенд с DNS

Цель домашнего задания

Создать домашнюю сетевую лабораторию. Изучить основы DNS, научиться работать с технологией Split-DNS в Linux-based системах

Описание домашнего задания

1. взять стенд https://github.com/erlong15/vagrant-bind
   - добавить еще один сервер client2
   - завести в зоне dns.lab имена:
     - web1 - смотрит на клиент1
     - web2 смотрит на клиент2
   - завести еще одну зону newdns.lab
   - завести в ней запись
     - www - смотрит на обоих клиентов

2. настроить split-dns
   - клиент1 - видит обе зоны, но в зоне dns.lab только web1
   - клиент2 видит только dns.lab

Дополнительное задание
* настроить все без выключения selinux

Формат сдачи ДЗ - vagrant + ansible

Введение

DNS(Domain Name System, Служба доменных имён) - это распределенная система, для получения информации о доменах. DNS используется для сопоставления IP-адресов и доменных имён.

Сопостовления IP-адресов и DNS-имён бывают двух видов:

- Прямое (DNS-bмя в IP-адрес)
- Обратное (IP-адрес в DNS-имя)

Доменная структура DNS представляет собой древовидную иерархию, состоящую из узлов, зон, доменов, поддоменов и т.д. «Вершиной» доменной структуры является корневая зона. Корневая (root) зона обозначается точкой. Далее следуют домены первого уровня (.com, ,ru, .org и т. д.) и т д.

В DNS встречаются понятия зон и доменов:

- Зона — это любая часть дерева системы доменных имён, размещаемая как единое целое на некотором DNS-сервере.
- Домен – определенный узел, *включающий в себя все подчинённые узлы.*

Давайте разберем основное отличие зоны от домена. Возьмём для примера ресурс otus.ru — это может быть сразу и зона и домен, однако, при использовании зоны otus.ru мы можем сделать отдельную зону mail.otus.ru, которая будет управляться не нами. В случае домена так сделать нельзя...

FQDN (Fully Qualified Domain Name) - полностью указанное доменное имя, т.е. от корневого домена. Ключевой идентификатор FQDN - точка в конце имени. Максимальный размер FQDN — 255 байт, с ограничением в 63 байта на каждое имя домена. *Пример FQDN: mail.otus.ru.*

Вся информация о DNS-ресурсах хранится в ресурсных записях. Записи хранят следующие атрибуты:

- Имя (NAME) - доменное имя, к которому привязана или которому принадлежит данная ресурсная область, либо IP-адрес. При отсутствии данного поля, запись ресурса наследуется от предыдущей записи.
- TTL (время жизни в кэше) - после указанного времени запись удаляется, данное поле может не указываться в индивидуальных записях ресурсов, но тогда оно должно быть указано в начале файла зоны и будет наследоваться всеми записями.
- Класс (CLASS) - определяет тип сети (в 99% используется IN - интернет)
- Тип (TYPE) - тип записи, синтаксис и назначение записи
- Значение (DATA)

Типы рекурсивных записей:

- A (Address record) - отображают имя хоста (доменное имя) на адрес IPv4
- AAAA - отображает доменное имя на адрес IPv6
- CNAME (Canonical name record/псевдоним) - привязка алиаса к существующему доменному имени
- MX (mail exchange) - указывает хосты для отправки почты, адресованной домену. При этом поле NAME указывает домен назначения, а поле DATA приоритет и доменное имя хоста, ответственного за приём почты. Данные вводятся через пробел
- NS (name server) - указывает на DNS-сервер, обслуживающий данный домен.
- PTR (pointer) - Отображает IP-адрес в доменное имя
- SOA (Start of Authority/начальная запись зоны) - описывает основные начальные настройки зоны.
- SRV (server selection) — указывает на сервера, обеспечивающие работу тех или иных служб в данном домене (например Jabber и Active Directory).

Для работы с DNS (как клиенту) в linux используют утилиты dig, host и nslookup

Также в Linux есть следующие реализации DNS-серверов:

- bind
- powerdns (умеет хранить зоны в БД)
- unbound (реализация bind)
- dnsmasq
- и т д.

Split DNS (split-horizon или split-brain) — это конфигурация, позволяющая отдавать разные записи зон DNS в зависимости от подсети источника запроса. Данную функцию можно реализовать как с помощью одного DNS-сервера, так и с помощью нескольких DNS-серверов…

Функциональные и нефункциональные требования

- ПК на Unix с 8 ГБ ОЗУ или виртуальная машина с включенной Nested Virtualization.

Предварительно установленное и настроенное следующее ПО:

- Hashicorp Vagrant (https://www.vagrantup.com/downloads)
- Oracle VirtualBox (https://www.virtualbox.org/wiki/Linux_Downloads).
- Ansible (версия 2.8 и выше) - https://docs.ansible.com/ansible/latest/installation_guide/intro_installation.html
- Любой редактор кода, например Visual Studio Code, Atom и т.д.

Инструкция по выполнению домашнего задания

Все дальнейшие действия были проверены при использовании Vagrant 2.2.19, VirtualBox v6.1.32_Ubuntu r149290. В качестве ОС на хостах установлена CentOS 7. Серьёзные отступления от этой конфигурации могут потребовать адаптации с вашей стороны.

1. Работа со стендом и настройка DNS

Скачаем себе стенд https://github.com/erlong15/vagrant-bind, перейдём в скаченный каталог и изучим содержимое файлов:

```
➜ git clone https://github.com/erlong15/vagrant-bind.git
➜ cd vagrant-bind
➜ vagrant-bind ls -l
total 12
drwxrwxr-x 2 alex alex 4096 мар 22 18:03 provisioning
-rw-rw-r-- 1 alex alex  414 мар 22 18:03 README.md
-rw-rw-r-- 1 alex alex  820 мар 22 18:03 Vagrantfile
```

Мы увидем файл Vagrantfile. Откроем его в любом, удобном для вас текстовом редакторе и добавим необходимую ВМ:

```
Vagrant.configure(2) do |config|
  config.vm.box = "centos/7"

  config.vm.provision "ansible" do |ansible|
    ansible.verbose = "vvv"
    ansible.playbook = "provisioning/playbook.yml"
    ansible.become = "true"
  end


  config.vm.provider "virtualbox" do |v|
          v.memory = 256
```

```
  end

  config.vm.define "ns01" do |ns01|
    ns01.vm.network "private_network", ip: "192.168.50.10", virtualbox__intnet: "dns"
    ns01.vm.hostname = "ns01"
  end

  config.vm.define "ns02" do |ns02|
    ns02.vm.network "private_network", ip: "192.168.50.11", virtualbox__intnet: "dns"
    ns02.vm.hostname = "ns02"
  end

  config.vm.define "client" do |client|
    client.vm.network "private_network", ip: "192.168.50.15", virtualbox__intnet: "dns"
    client.vm.hostname = "client"
  end

  config.vm.define "client2" do |client2|
    client2.vm.network "private_network", ip: "192.168.50.16", virtualbox__intnet: "dns"
    client2.vm.hostname = "client2"
  end

end
```

Vagranfile описывает создание 4 виртуальных машин на CentOS 7, каждой машине будет выделено по 256 МБ ОЗУ. В начале файла есть модуль, который отвечает за настройку ВМ с помощью Ansible.

Жирным выделены следующие фрагменты:

- Параметр *ansible.sudo = "true"* рекомендуется замененить на *ansible.become = "true"*, так как *ansible.sudo* скоро перестанет использоваться…
- Добавлено описание виртуальной машины client2.

После внесения изменений, можно попробовать развернуть наши ВМ, для этого нужно воспользоваться командой: *vagrant up*

После того, как у нас получилось добавить виртуальную машину client2, давайте подробнее разберем остальные файлы. Для этого перейдём в каталог provisoning: *cd provisioning*

Рассмотрим требуемые нам файлы:

- playbook.yml — это Ansible-playbook, в котором содержатся инструкции по настройке нашего стенда
- client-motd — файл, содержимое которого будет появляться перед пользователем, который подключился по SSH
- named.ddns.lab и named.dns.lab — файлы описания зон ddns.lab и dns.lab соответсвенно
- master-named.conf и slave-named.conf — конфигурационные файлы, в которых хранятся настройки DNS-сервера
- client-resolv.conf и servers-resolv.conf — файлы, в которых содержатся IP-адреса DNS-серверов

Рассмотрим содержимое файла playbook.yml:

```
---
- hosts: all
  become: yes
```

```
  tasks:
#Установка пакетов bind, bind-utils и ntp
  - name: install packages
    yum: name={{ item }} state=latest
    with_items:
      - bind
      - bind-utils
      - ntp
#Копирование файла named.zonetransfer.key на хосты с правами 0644
#Владелец файла — root, група файла — named
  - name: copy transferkey to all servers and the client
    copy: src=named.zonetransfer.key dest=/etc/named.zonetransfer.key owner=root group=named mode=0644

#Настройка хоста ns01
- hosts: ns01
  become: yes
  tasks:
#Копирование конфигурации DNS-сервера
  - name: copy named.conf
    copy: src=master-named.conf dest=/etc/named.conf owner=root group=named mode=0640

#Копирование файлов с настроками зоны.
#Будут скопированы все файлы, в имя которых начинается на «named.d»
  - name: copy zones
    copy: src={{ item }} dest=/etc/named/ owner=root group=named mode=0660
    with_fileglob:
      - named.d*

#Копирование файла resolv.conf
  - name: copy resolv.conf to the servers
    copy: src=servers-resolv.conf dest=/etc/resolv.conf owner=root group=root mode=0644

#Изменение прав каталога /etc/named
#Права 670, владелец — root, группа — named
  - name: set /etc/named permissions
    file: path=/etc/named owner=root group=named mode=0670

#Перезапуск службы Named и добавление её в автозагрузку
  - name: ensure named is running and enabled
    service: name=named state=restarted enabled=yes

- hosts: ns02
  become: yes
  tasks:
  - name: copy named.conf
    copy: src=slave-named.conf dest=/etc/named.conf owner=root group=named mode=0640

  - name: copy resolv.conf to the servers
    copy: src=servers-resolv.conf dest=/etc/resolv.conf owner=root group=root mode=0644

  - name: set /etc/named permissions
    file: path=/etc/named owner=root group=named mode=0670

  - name: ensure named is running and enabled
    service: name=named state=restarted enabled=yes

- hosts: client
  become: yes
  tasks:
```

```
  - name: copy resolv.conf to the client
    copy: src=client-resolv.conf dest=/etc/resolv.conf owner=root group=root mode=0644

#Копирование конфигруационного файла rndc
  - name: copy rndc conf file
    copy: src=rndc.conf dest=/home/vagrant/rndc.conf owner=vagrant group=vagrant mode=0644
#Настройка сообщения при входе на сервер
  - name: copy motd to the client
    copy: src=client-motd dest=/etc/motd owner=root group=root mode=0644
```

Так как мы добавили ещё одну виртуальную машину (client2), нам потребуется её настроить. Так как настройки будут совпадать с ВМ client, то мы можем просто добавить хост в модуль по настройке клиента:

```
- hosts: client,client2
  become: yes
  tasks:
  - name: copy resolv.conf to the client
    copy: src=client-resolv.conf dest=/etc/resolv.conf owner=root group=root mode=0644

#Копирование конфигруационного файла rndc
  - name: copy rndc conf file
    copy: src=rndc.conf dest=/home/vagrant/rndc.conf owner=vagrant group=vagrant mode=0644
#Настройка сообщения при входе на сервер
  - name: copy motd to the client
    copy: src=client-motd dest=/etc/motd owner=root group=root mode=0644
```

Заметки по Ansible-playbook:

1) В начале нашего плейбука есть модуль yum, в новых версиях Ansible, можно сразу указывать пакеты в разделе name:

```
- name: install packages
    yum:
      name:
        - bind
        - bind-utils
        - ntp
        - vim
      state: latest
      update_cache: true
```

Если продолжать использовать устаревший формат описания пакетов в yum Ansible будет выдавать предупреждения о том, что в следующих версиях данная функция будет недоступна. На текущий момент можно использовать оба варианта модуля yum.

2) Для нормальной работы DNS-серверов, на них должно быть настроено одинаковое время. Для того, чтобы на всех серверах было одинаковое время, нам потребуется настроить NTP.

В модуле установки пакетов мы видим, что планируется установить утилититу ntp, однако, в CentOS по умолчанию уже есть NTP-клиент Chrony. Обычно он всегда включен и добавлен в автозагрузку. Проверить работу службы можно командой: systemctl status chronyd

Если мы планируем установить утилиту ntp, тогда после установки ntp нам потребуется:

- Остановить службу chronyd: *systemctl stop chronyd*
- Удалить службу chronyd из автозагрузки: *systemctl disable chronyd*
- Включить службу ntpd: *systemctl start ntpd*
- Добавить службу ntpd в автозагрузку: *systemctl enable ntpd*

Пример данной настройки в Ansible (YAML-формат):

```
 - name: stop and disable chronyd
   service:
     name: chronyd
     state: stopped
     enabled: false

 - name: start and enable ntpd
   service:
     name: ntpd
     state: started
     enabled: true
```

Альтернативный вариант — не устанавливать ntp и просто запустить службу chronyd: systemctl start chronyd

Пример данной настройки в Ansible (YAML формат):

```
 - name: install packages
   yum:
     name:
       - bind
       - bind-utils
       - vim
     state: latest
     update_cache: true

 - name: start chronyd
   service:
     name: chronyd
     state: restarted
     enabled: true
```

3) Перед выполнением следующих заданий, нужно обратить внимание, на каком адресе и порту работают наши DNS-сервера.

Проверить это можно двумя способами:

- Посмотреть с помощью команды SS: *ss -tulpn*

```
[root@ns01 ~]# ss -ulpn
State      Recv-Q Send-Q                 Local Address:Port                    Peer Address:Port
UNCONN     0      0                          *:111                                *:*
users:(("rpcbind",pid=338,fd=6))
UNCONN     0      0                          *:930                                *:*
users:(("rpcbind",pid=338,fd=7))
UNCONN     0      0                  192.168.50.10:53                                 *:*
users:(("named",pid=30971,fd=512))
```

```
UNCONN     0      0                  127.0.0.1:323                      *:*
users:(("chronyd",pid=341,fd=5))
UNCONN     0      0                    *:68                             *:*
users:(("dhclient",pid=2449,fd=6))
UNCONN     0      0                  [::]:111                           [::]:*
users:(("rpcbind",pid=338,fd=9))
UNCONN     0      0                  [::]:930                           [::]:*
users:(("rpcbind",pid=338,fd=10))
UNCONN     0      0                  [::1]:53                           [::]:*
users:(("named",pid=30971,fd=513))
UNCONN     0      0                  [::1]:323                          [::]:*
users:(("chronyd",pid=341,fd=6))
[root@ns01 ~]#
```

- Посмотреть информацию в настройках DNS-сервера (/etc/named.conf)
  - ✔ На хосте ns01:

```
// network
   listen-on port 53 { 192.168.50.10; };
   listen-on-v6 port 53 { ::1; };
```

  - ✔ На хосте ns02

```
                // network
                    listen-on port 53 { 192.168.50.11; };
                    listen-on-v6 port 53 { ::1; };
```

Исходя из данной информации, нам нужно подкорректировать файл /etc/resolv.conf для DNS-серверов: на хосте ns01 указать nameserver 192.168.50.10, а на хосте ns02 — 192.168.50.11

В Ansible для этого можно воспользоваться шаблоном с Jinja. Изменим имя файла servers-resolv.conf на servers-resolv.conf.j2 и укажем там следующие условия:

```
domain dns.lab
search dns.lab
#Если имя сервера ns02, то указываем nameserver 192.168.50.11
{% if ansible_hostname == 'ns02' %}
nameserver 192.168.50.11
{% endif %}
#Если имя сервера ns01, то указываем nameserver 192.168.50.10
{% if ansible_hostname == 'ns01' %}
nameserver 192.168.50.10
{% endif %}
```

После внесение изменений в файл, внесём измения в ansible-playbook:
Используем вместо модуля *copy* модуль template:

```
- name: copy resolv.conf to the servers
  template: src=servers-resolv.conf.j2 dest=/etc/resolv.conf owner=root group=root mode=0644
```

Или используя YAML-формат:

```
  - name: copy resolv.conf to the servers
    template:
      src: servers-resolv.conf.j2
```

```
    dest: /etc/resolv.conf
    owner: root
    group: root
    mode: 0644
```

Добавление имён в зону dns.lab

Давайте проверим, что зона dns.lab уже существует на DNS-серверах:

Фрагмент файла */etc/named.conf* на сервере ns01:

```
// Имя зоны
zone "dns.lab" {
    type master;
    // Тем, у кого есть ключ zonetransfer.key можно получать копию файла зоны
    allow-transfer { key "zonetransfer.key"; };
    // Файл с настройками зоны
    file "/etc/named/named.dns.lab";
};
```

Похожий фрагмент файла */etc/named.conf* находится на slave-сервере ns02:

```
// Имя зоны
zone "dns.lab" {
    type slave;
    // Адрес мастера, куда будет обращаться slave-сервер
    masters { 192.168.50.10; };
};
```

Также на хосте ns01 мы видим файл /etc/named/named.dns.lab с настройкой зоны:

```
$TTL 3600
; описание зоны dns.lab.
$ORIGIN dns.lab.
@               IN      SOA     ns01.dns.lab. root.dns.lab. (
                                2711201407 ; serial
                                3600       ; refresh (1 hour)
                                600        ; retry (10 minutes)
                                86400      ; expire (1 day)
                                600        ; minimum (10 minutes)
                        )

                IN      NS      ns01.dns.lab.
                IN      NS      ns02.dns.lab.

; DNS Servers
ns01            IN      A       192.168.50.10
ns02            IN      A       192.168.50.11
```

Именно в этот файл нам потребуется добавить имена. Допишем в конец файла следующие строки:

```
;Web
web1        IN    A     192.168.50.15
web2        IN    A     192.168.50.16
```

Если изменения внесены вручную, то для применения настроек нужно:

- Перезапустить службу named: *systemctl restart named*
- Изменить значение Serial (добавить +1 к числу 2711201407), изменение значения serial укажет slave-серверам на то, что были внесены изменения и что им надо обновить свои файлы с зонами.

После внесения изменений, выполним проверку с клиента:

```
[vagrant@client ~]$ dig @192.168.50.10 web1.dns.lab

; <<>> DiG 9.11.4-P2-RedHat-9.11.4-26.P2.el7_9.9 <<>> @192.168.50.10 web1.dns.lab
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 49207
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 3

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;web1.dns.lab.                  IN      A

;; ANSWER SECTION:
web1.dns.lab.           3600    IN      A       192.168.50.15

;; AUTHORITY SECTION:
dns.lab.                3600    IN      NS      ns01.dns.lab.
dns.lab.                3600    IN      NS      ns02.dns.lab.

;; ADDITIONAL SECTION:
ns01.dns.lab.           3600    IN      A       192.168.50.10
ns02.dns.lab.           3600    IN      A       192.168.50.11

;; Query time: 0 msec
;; SERVER: 192.168.50.10#53(192.168.50.10)
;; WHEN: Sun Mar 27 00:37:28 UTC 2022
;; MSG SIZE  rcvd: 127

[vagrant@client ~]$ dig @192.168.50.11 web2.dns.lab

; <<>> DiG 9.11.4-P2-RedHat-9.11.4-26.P2.el7_9.9 <<>> @192.168.50.11 web2.dns.lab
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 36834
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 3

;; OPT PSEUDOSECTION:
```

```
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;web2.dns.lab.                  IN      A

;; ANSWER SECTION:
web2.dns.lab.           3600    IN      A       192.168.50.16

;; AUTHORITY SECTION:
dns.lab.                3600    IN      NS      ns01.dns.lab.
dns.lab.                3600    IN      NS      ns02.dns.lab.

;; ADDITIONAL SECTION:
ns01.dns.lab.           3600    IN      A       192.168.50.10
ns02.dns.lab.           3600    IN      A       192.168.50.11

;; Query time: 2 msec
;; SERVER: 192.168.50.11#53(192.168.50.11)
;; WHEN: Sun Mar 27 00:37:28 UTC 2022
;; MSG SIZE  rcvd: 127

[vagrant@client ~]$
```

В примерах мы обратились к разным DNS-серверам с разными запросами

Добавление имён в зону dns.lab с помощью Ansible

В существующем Ansible-playbook менять ничего не потребуется. Нам потребуется добавить стоки с новыми именами в файл named.dns.lab и снова запустить Ansible.

Модули, которые внесут данные изменения (переписаны в YAML формате):

```
#Настройка хоста ns01
- hosts: ns01
  become: yes
  tasks:
#Копирование конфигурации DNS-сервера
  - name: copy named.conf
    copy:
      src: master-named.conf
      dest: /etc/named.conf
      owner: root
      group: named
      mode: 0640

#Копирование файлов с настроками зоны.
#Будут скопированы все файлы, в имя которых начинается на «named.d»
  - name: copy zones
    copy:
      src: "{{ item }}"
      dest: /etc/named/
      owner: root
```

```
    group: named
    mode: 0660
  with_items:
    - named.ddns.lab
    - named.dns.lab
    - named.dns.lab.client
    - named.dns.lab.rev
    - named.newdns.lab

#Перезапуск службы Named и добавление её в автозагрузку
 - name: ensure named is running and enabled
   service:
     name: named
     state: restarted
     enabled: yes
```

Создание новой зоны и добавление в неё записей

Для того, чтобы прописать на DNS-серверах новую зону нам потребуется:

- На хосте *ns01* добавить зону в файл */etc/named.conf:*

```
        // lab's newdns zone
zone "newdns.lab" {
    type master;
    allow-transfer { key "zonetransfer.key"; };
    allow-update { key "zonetransfer.key"; };
    file "/etc/named/named.newdns.lab";
};
```

- *На хосте ns02 также добавить зону и указать с какого сервера запрашивать информацию об этой зоне (фрагмент файла /etc/named.conf):*

```
// lab's newdns zone
zone "newdns.lab" {
    type slave;
    masters { 192.168.50.10; };
    file "/etc/named/named.newdns.lab";
};
```

- На хосте ns01 создадим файл */etc/named/named.newdns.lab*

```
        vi /etc/named/named.newdns.lab

$TTL 3600
$ORIGIN newdns.lab.
@           IN    SOA    ns01.dns.lab. root.dns.lab. (
                         2711201007 ; serial
                         3600       ; refresh (1 hour)
                         600        ; retry (10 minutes)
                         86400      ; expire (1 day)
                         600        ; minimum (10 minutes)
                        )

                IN    NS     ns01.dns.lab.
                IN    NS     ns02.dns.lab.
```

```
; DNS Servers
ns01        IN    A     192.168.50.10
ns02        IN    A     192.168.50.11

;WWW
www         IN    A     192.168.50.15
www         IN    A     192.168.50.16
```

В конце этого файла добавим записи www. У файла должны быть права 660, владелец — root, группа — named.

После внесения данных изменений, изменяем значение serial *(добавлем +1 к значению 2711201007)* и перезапускаем named: *systemctl restart named*

Создание новой зоны и добавление в неё записей с помощью Ansible

Для создания зоны и добавления в неё записей, добавляем зону в файл */etc/named.conf* на хостах ns01 и ns02, а также создаем файл named.newdns.lab, который далее отправим на сервер ns01.

Добавим в модуль copy наш файл named.newdns.lab:

```
- name: copy zones
  copy: src={{ item }} dest=/etc/named/ owner=root group=named mode=0660
  with_fileglob:
    - named.d*
    - named.newdns.lab
```

Соответственно файл named.newdns.lab будет скопирован на хост ns01 по адресу /etc/named/named.newdns.lab

Остальную часть playbook можно оставить без изменений.

2. Настройка Split-DNS

У нас уже есть прописанные зоны dns.lab и newdns.lab. Однако по заданию client1 должен видеть запись web1.dns.lab и не видеть запись web2.dns.lab. Client2 может видеть обе записи из домена dns.lab, но не должен видеть записи домена newdns.lab Осуществить данные настройки нам поможет технология Split-DNS.

Для настройки Split-DNS нужно:
1) Создать дополнительный файл зоны dns.lab, в котором будет прописана только одна запись: *vim /etc/named/named.dns.lab.client*

```
$TTL 3600
$ORIGIN dns.lab.
@            IN    SOA    ns01.dns.lab. root.dns.lab. (
                          2711201407 ; serial
                          3600       ; refresh (1 hour)
                          600        ; retry (10 minutes)
                          86400      ; expire (1 day)
```

```
                        600       ; minimum (10 minutes)
                    )

                IN      NS      ns01.dns.lab.
                IN      NS      ns02.dns.lab.

; DNS Servers
ns01            IN      A       192.168.50.10
ns02            IN      A       192.168.50.11

;Web
web1            IN      A       192.168.50.15
```

Имя файла может отличаться от указанной зоны. У файла должны быть права 660, владелец — root, группа — named.

2) Внести изменения в файл /etc/named.conf на хостах ns01 и ns02

Прежде всего нужно сделать access листы для хостов client и client2. Сначала сгенерируем ключи для хостов client и client2, для этого на хосте ns01 запустим утилиту tsig-keygen (ключ может генериться 5 минут и более):

```
[root@ns01 ~]# tsig-keygen
key "tsig-key" {
        algorithm hmac-sha256;
        secret "IxAIDfcewAtWxE5NnD54HBwXvX4EFThcW1o0DqL15oI=";
};
[root@ns01 ~]#
```

После генерации, мы увидим ключ (secret) и алгоритм с помощью которого он был сгенерирован. Оба этих параметра нам потребуются в access листе.

Если нам потребуется, использовать другой алгоритм, то мы можем его указать как аргумент к команде, например: *tsig-keygen -a hmac-md5*

Всего нам потребуется 2 таких ключа. После их генерации добавим блок с access листами в конец файла /etc/named.conf

```
#Описание ключа для хоста client
key "client-key" {
   algorithm hmac-sha256;
   secret "lQg171Ht4mdGYcjjYKhI9gSc1fhoxzHZB+h2NMtyZWY=";
};
#Описание ключа для хоста client2
key "client2-key" {
   algorithm hmac-sha256;
   secret "m7r7SpZ9KBcA4kOl1JHQQnUilIpQA1IJ9xkBHwdRAHc=";
};
#Описание access-листов
acl client { !key client2-key; key client-key; 192.168.50.15; };
acl client2 { !key client-key; key client2-key; 192.168.50.16; };
```

В данном блоке access листов мы выделяем 2 блока:

- client имеет адрес 192.168.50.15, использует client-key и не использует client2-key
- client2 имеет адрес 192ю168.50.16, использует clinet2-key и не использует client-key

Описание ключей и access листов будет одинаковое для master и slave сервера.

Далее нужно создать файл с настройками зоны dns.lab для client, для этого на мастер сервере создаём файл */etc/named/named.dns.lab.client* и добавляем в него следующее содержимое:

```
$TTL 3600
$ORIGIN dns.lab.
@               IN      SOA     ns01.dns.lab. root.dns.lab. (
                                2711201407 ; serial
                                3600       ; refresh (1 hour)
                                600        ; retry (10 minutes)
                                86400      ; expire (1 day)
                                600        ; minimum (10 minutes)
                        )

                IN      NS      ns01.dns.lab.
                IN      NS      ns02.dns.lab.

; DNS Servers
ns01            IN      A       192.168.50.10
ns02            IN      A       192.168.50.11

;Web
web1            IN      A       192.168.50.15
```

Это почти скопированный файл зоны dns.lab, в конце которого удалена строка с записью web2. Имя зоны надо оставить такой же — dns.lab

Теперь можно внести правки в /*etc*/named.conf

Технология Split-DNS реализуется с помощью описания представлений (view), для каждого отдельного acl. В каждое представление (view) добавляются только те зоны, которые разрешено видеть хостам, адреса которых указаны в access листе.

Все ранее описанные зоны должны быть перенесены в модули view. Вне view зон быть недолжно, зона any должна всегда находиться в самом низу.

После применения всех вышеуказанных правил на хосте ns01 мы получим следующее содержимое файла */etc*/named.conf

```
options {

    // На каком порту и IP-адресе будет работать служба
        listen-on port 53 { 192.168.50.10; };
        listen-on-v6 port 53 { ::1; };
```

```
    // Указание каталогов с конфигурационными файлами
            directory         "/var/named";
            dump-file         "/var/named/data/cache_dump.db";
            statistics-file "/var/named/data/named_stats.txt";
            memstatistics-file "/var/named/data/named_mem_stats.txt";

    // Указание настроек DNS-сервера
    // Разрешаем серверу быть рекурсивным
            recursion yes;
    // Указываем сети, которым разрешено отправлять запросы серверу
            allow-query     { any; };
    // Каким сетям можно передавать настройки о зоне
    allow-transfer { any; };

    // dnssec
            dnssec-enable yes;
            dnssec-validation yes;

    // others
            bindkeys-file "/etc/named.iscdlv.key";
            managed-keys-directory "/var/named/dynamic";
            pid-file "/run/named/named.pid";
            session-keyfile "/run/named/session.key";
};

logging {
        channel default_debug {
                file "data/named.run";
                severity dynamic;
        };
};

// RNDC Control for client
key "rndc-key" {
    algorithm hmac-md5;
    secret "GrtiE9kz16GK+OKKU/qJvQ==";
};
controls {
        inet 192.168.50.10 allow { 192.168.50.15; 192.168.50.16; } keys { "rndc-key"; };
};

key "client-key" {
    algorithm hmac-sha256;
    secret "lQg171Ht4mdGYcjjYKhI9gSc1fhoxzHZB+h2NMtyZWY=";
};
key "client2-key" {
    algorithm hmac-sha256;
    secret "m7r7SpZ9KBcA4kOl1JHQQnUiIlpQA1IJ9xkBHwdRAHc=";
};

// ZONE TRANSFER WITH TSIG
include "/etc/named.zonetransfer.key";

server 192.168.50.11 {
```

```
    keys { "zonetransfer.key"; };
};
// Указание Access листов
acl client { !key client2-key; key client-key; 192.168.50.15; };
acl client2 { !key client-key; key client2-key; 192.168.50.16; };
// Настройка первого view
view "client" {
    // Кому из клиентов разрешено подключаться, нужно указать имя access-листа
    match-clients { client; };

    // Описание зоны dns.lab для client
    zone "dns.lab" {
        // Тип сервера — мастер
        type master;
        // Добавляем ссылку на файл зоны, который создали в прошлом пункте
        file "/etc/named/named.dns.lab.client";
        // Адрес хостов, которым будет отправлена информация об изменении зоны
        also-notify { 192.168.50.11 key client-key; };
    };

    // newdns.lab zone
    zone "newdns.lab" {
        type master;
        file "/etc/named/named.newdns.lab";
        also-notify { 192.168.50.11 key client-key; };
    };
};

// Описание view для client2
view "client2" {
    match-clients { client2; };

    // dns.lab zone
    zone "dns.lab" {
        type master;
        file "/etc/named/named.dns.lab";
        also-notify { 192.168.50.11 key client2-key; };
    };

    // dns.lab zone reverse
    zone "50.168.192.in-addr.arpa" {
        type master;
        file "/etc/named/named.dns.lab.rev";
        also-notify { 192.168.50.11 key client2-key; };
    };
};

// Зона any, указана в файле самой последней
view "default" {
    match-clients { any; };

    // root zone
    zone "." IN {
        type hint;
```

```
        file "named.ca";
    };

    // zones like localhost
    include "/etc/named.rfc1912.zones";
    // root DNSKEY
    include "/etc/named.root.key";

    // dns.lab zone
    zone "dns.lab" {
        type master;
        allow-transfer { key "zonetransfer.key"; };
        file "/etc/named/named.dns.lab";
    };

    // dns.lab zone reverse
    zone "50.168.192.in-addr.arpa" {
        type master;
        allow-transfer { key "zonetransfer.key"; };
        file "/etc/named/named.dns.lab.rev";
    };

    // ddns.lab zone
    zone "ddns.lab" {
        type master;
        allow-transfer { key "zonetransfer.key"; };
        allow-update { key "zonetransfer.key"; };
        file "/etc/named/named.ddns.lab";
    };

    // newdns.lab zone
    zone "newdns.lab" {
        type master;
        allow-transfer { key "zonetransfer.key"; };
        file "/etc/named/named.newdns.lab";
    };
};
```

Далее внесем изменения в файл */etc*/named.conf на сервере ns02. Файл будет похож на файл, лежащий на ns01, только в настройках будет указание забирать информацию с сервера ns01:

```
options {

    // network
            listen-on port 53 { 192.168.50.11; };
            listen-on-v6 port 53 { ::1; };

    // data
            directory       "/var/named";
            dump-file       "/var/named/data/cache_dump.db";
            statistics-file "/var/named/data/named_stats.txt";
```

```
        memstatistics-file "/var/named/data/named_mem_stats.txt";

    // server
        recursion yes;
        allow-query     { any; };
    allow-transfer { any; };

    // dnssec
        dnssec-enable yes;
        dnssec-validation yes;

    // others
        bindkeys-file "/etc/named.iscdlv.key";
        managed-keys-directory "/var/named/dynamic";
        pid-file "/run/named/named.pid";
        session-keyfile "/run/named/session.key";
};

logging {
      channel default_debug {
            file "data/named.run";
            severity dynamic;
      };
};

// RNDC Control for client
key "rndc-key" {
    algorithm hmac-md5;
    secret "GrtiE9kz16GK+OKKU/qJvQ==";
};
controls {
      inet 192.168.50.11 allow { 192.168.50.15; 192.168.50.16; } keys { "rndc-key"; };
};

key "client-key" {
    algorithm hmac-sha256;
    secret "lQg171Ht4mdGYcjjYKhl9gSc1fhoxzHZB+h2NMtyZWY=";
};
key "client2-key" {
    algorithm hmac-sha256;
    secret "m7r7SpZ9KBcA4kOl1JHQQnUiIlpQA1IJ9xkBHwdRAHc=";
};


i
```

Так как файлы с конфигурациями получаются достаточно большими — возрастает вероятность сделать ошибку. При их правке можно воспользоваться утилитой *niamed-checkconf.* Она укажет в каких строчках есть ошибки. *Использование данной утилиты рекомендуется после изменения настроек на DNS-сервере.*

После внесения данных изменений можно перезапустить (по очереди) службу named на серверах ns01 и ns02.

Далее, нужно будет проверить работу Split-DNS с хостов client и client2. Для проверки можно использовать утилиту ping:

Проверка на client:

```
[root@client ~]# ping www.newdns.lab
PING www.newdns.lab (192.168.50.15) 56(84) bytes of data.
64 bytes from client (192.168.50.15): icmp_seq=1 ttl=64 time=0.014 ms
64 bytes from client (192.168.50.15): icmp_seq=2 ttl=64 time=0.066 ms
^C
--- www.newdns.lab ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 0.014/0.040/0.066/0.026 ms
[root@client ~]# ping web1.dns.lab
PING web1.dns.lab (192.168.50.15) 56(84) bytes of data.
64 bytes from client (192.168.50.15): icmp_seq=1 ttl=64 time=0.015 ms
64 bytes from client (192.168.50.15): icmp_seq=2 ttl=64 time=0.068 ms
^C
--- web1.dns.lab ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1005ms
rtt min/avg/max/mdev = 0.015/0.041/0.068/0.027 ms
[root@client ~]# ping web2.dns.lab
ping: web2.dns.lab: Name or service not known
[root@client ~]# ^C
[root@client ~]#
```

На хосте мы видим, что client видит обе зоны (dns.lab и newdns.lab), однако информацию о хосте web2.dns.lab он получить не может.

Проверка на client2:

```
[root@client2 ~]# ping www.newdns.lab
ping: www.newdns.lab: Name or service not known
[root@client2 ~]#
[root@client2 ~]# ping web1.dns.lab
PING web1.dns.lab (192.168.50.15) 56(84) bytes of data.
64 bytes from 192.168.50.15 (192.168.50.15): icmp_seq=1 ttl=64 time=0.809 ms
^C
--- web1.dns.lab ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.809/0.809/0.809/0.000 ms
[root@client2 ~]# ping web2.dns.lab
PING web2.dns.lab (192.168.50.16) 56(84) bytes of data.
64 bytes from client2 (192.168.50.16): icmp_seq=1 ttl=64 time=0.037 ms
64 bytes from client2 (192.168.50.16): icmp_seq=2 ttl=64 time=0.065 ms
^C
--- web2.dns.lab ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1038ms
rtt min/avg/max/mdev = 0.037/0.051/0.065/0.014 ms
[root@client2 ~]#
```

Тут мы понимаем, что client2 видит всю зону dns.lab и не видит зону newdns.lab

Для того, чтобы проверить что master и slave сервера отдают одинаковую информацию, в файле /etc/resolv.conf можно удалить на время nameserver 192.168.50.10 и попробовать выполнить все те же проверки. Результат должен быть идентичный.

Настройка Split-DNS с помощью Ansible

В существующем Ansible-playbook менять ничего не потребуется. Нам потребуется изменить содержимое файлов master-named.conf и slave-named.conf, а также добавить файл named.dns.lab.client. Содержимое всех файлов представлено в прошлом пункте.

Модули, которые внесут данные изменения (переписаны в YAML формате):

```
# Настройка хоста ns01
- hosts: ns01
  become: yes
  tasks:
  # Копирования файла named.conf
  - name: copy named.conf
    copy:
      src: master-named.conf
      dest: /etc/named.conf
      owner: root
      group: named
      mode: 0640

  #Копирование файлов с настройками зоны
  - name: copy zones
    copy:
      src: "{{ item }}"
      dest: /etc/named/
      owner: root
      group: named
      mode: 0660
    with_items:
      - named.ddns.lab
      - named.dns.lab
      - named.dns.lab.client
      - named.dns.lab.rev
      - named.newdns.lab

  #Копирование файла resolv.conf на хосты
  - name: copy resolv.conf to the servers
    template:
      src: servers-resolv.conf.j2
      dest: /etc/resolv.conf
      owner: root
      group: root
      mode: 0644

  # Настройка прав для каталога /etc/named
```

```
  - name: set /etc/named permissions
    file:
      path: /etc/named
      owner: root
      group: named
      mode: 0670

  #Перезапуск и добавление в автозагрузку службы named
  - name: ensure named is running and enabled
    service:
      name: named
      state: restarted
      enabled: yes

- hosts: ns02
  become: yes
  tasks:
  - name: copy named.conf
    copy:
      src: slave-named.conf
      dest: /etc/named.conf
      owner: root
      group: named
      mode: 0640

  - name: copy resolv.conf to the servers
    template:
      src: servers-resolv.conf.j2
      dest: /etc/resolv.conf
      owner: root
      group: root
      mode: 0644

  - name: set /etc/named permissions
    file:
      path: /etc/named
      owner: root
      group: named
      mode: 0670

  - name: ensure named is running and enabled
    service:
      name: named
      state: restarted
      enabled: yes
```

Критерии оценивания

Статус «Принято» ставится при выполнении следующих условий:

1. Ссылка на репозиторий GitHub.
2. Vagrantfile, который будет разворачивать виртуальные машины
3. Настройка виртуальных машин происходит с помощью Ansible.
4. Документация по каждому заданию:
Создайте файл README.md и снабдите его следующей информацией:

- название выполняемого задания;
- текст задания;
- особенности проектирования и реализации решения,
- заметки, если считаете, что имеет смысл их зафиксировать в репозитории.

Дополнительное задание выполняется по желанию

Рекомендуемые источники

- Статья о DNS (общая информация) - https://ru.wikipedia.org/wiki/DNS
- Статья «DNS сервер BIND (теория)» - https://habr.com/ru/post/137587/
- Статья «Настройка Split DNS на одном сервере Bind» - https://www.dmosk.ru/miniinstruktions.php?mini=split-dns-bind
- Статья «Split DNS: заставим BIND работать на два фронта!» - http://samag.ru/archive/article/771
- Статья «DNS BIND Zone Transfers and Updates» - https://www.zytrax.com/books/dns/ch7/xfer.html#:~:text=also%2Dnotify%20Statement%20(Pre%20BIND9.9)&text=also%2Dnotify%20defines%20a%20list,NS%20records%20for%20the%20zone