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

Стенд с Vagrant с Rsyslog

Цель домашнего задания

Научится проектировать централизованный сбор логов. Рассмотреть особенности разных платформ для сбора логов.

Описание домашнего задания

1. В Vagrant разворачиваем 2 виртуальные машины web и log
2. на web настраиваем nginx
3. на log настраиваем центральный лог сервер на любой системе на выбор
   - journald;
   - rsyslog;
   - elk.
4. настраиваем аудит, следящий за изменением конфигов nginx

Все критичные логи с web должны собираться и локально и удаленно.
Все логи с nginx должны уходить на удаленный сервер (локально только критичные).
Логи аудита должны также уходить на удаленную систему.

Формат сдачи ДЗ - vagrant + ansible

Дополнительное задание

- развернуть еще машину с elk
- таким образом настроить 2 центральных лог системы elk и какую либо еще;
- в elk должны уходить только логи Nginx;
- во вторую систему все остальное.

Статус "Принято" ставится, если присылаете логи скриншоты без вагранта.

Задание со звездочкой выполняется по желанию.

Введение

Функция системного журналирования (логирование) — это основной источник информации о работе системы и ошибках. В системе Linux почти все действия записываются. Именно эти данные помогают разбираться в проблемах с ОС.

Логи могут храниться как локально, так и пересылаться на удаленную систему. Пересылка логов имеет следующие плюсы:

- Возможность централизованного сбора и анализа логов (все логи со всех устройств прилетают в одно место. Это значительно упростит работу с логами)
- Защита от удаления логов на локальной машине
- Оптимизация места на диске в локальной ОС (логи не будут храниться в ОС, т.к. будут сразу пересылаться в систему сбора логов. Данная функция настраивается отдельно)

В ОС Linux главным файлом локального журналирования является:

- Ubuntu/Debian — /*var*/log/syslog

Логи в ОС можно настроить. Например, указывать больше информации или отключить логирование конкретного компонента.

Помимо логов, в Unix-системах используют аудит. В linux эту функцию выполняет linux audit daemon.

Linux Audit Daemon - это среда, позволяющая проводить аудит событий в системе Linux. Используя мощную систему аудита возможно отслеживать многие типы событий для мониторинга и проверки системы, например:

- доступ к файлам;
- изменение прав на файлы;
- просмотр пользователей, изменивших конкретный файл;
- обнаружение несанкционированных изменений;
- мониторинг системных вызовов и функций;
- обнаружение аномалий, таких как сбои;
- мониторинг набора команд.

Аудит различает 4 вида доступа к файлу:

- r — чтение
- w — запись в файл
- x — выполнение файла
- a — изменение атрибута

Функциональные и нефункциональные требования

- ПК на Unix с 8ГБ 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 2.7 и выше) - https://docs.ansible.com/ansible/latest/installation_guide/intro_installation.html

Инструкция по выполнению домашнего задания

1. Создаём виртуальные машины

Создаём каталог, в котором будут храниться настройки виртуальной машины. В каталоге создаём файл с именем Vagrantfile, добавляем в него следующее содержимое:

```
Vagrant.configure("2") do |config|
  # Base VM OS configuration.
  config.vm.box = ""bento/ubuntu-22.04"
  config.vm.provider :virtualbox do |v|
    v.memory = 1512
    v.cpus = 2
  end

  # Define two VMs with static private IP addresses.
  boxes = [
    { :name => "web",
      :ip => "192.168.56.10",
    },
    { :name => "log",
      :ip => "192.168.56.15",
    }
  ]
  # Provision each of the VMs.
  boxes.each do |opts|
    config.vm.define opts[:name] do |config|
      config.vm.hostname = opts[:name]
      config.vm.network "private_network", ip: opts[:ip]

    end
  end
end
```

Результатом выполнения команды *vagrant up* станут 2 созданные виртуальные машины

Заходим на web-сервер: *vagrant ssh web*

Дальнейшие действия выполняются от пользователя root. Переходим в root пользователя: *sudo -i*

Для правильной работы с логами, нужно, чтобы на всех хостах было настроено одинаковое время.

```
root@web:~# timedatectl
               Local time: Sun 2023-12-17 20:16:41 MSK
           Universal time: Sun 2023-12-17 17:16:41 UTC
                 RTC time: Sun 2023-12-17 17:16:41
                 Time zone: Etc/UTC (MSK, +0300)
System clock synchronized: no
          NTP service: inactive
          RTC in local TZ: no
```

…

Далее проверим, что время и дата указаны правильно: date

```
[root@web ~]# date
Sun Dec 17 20:20:26 MSK 2023
[root@web ~]#
```

Настроить NTP нужно на обоих серверах.

2. Установка nginx на виртуальной машине web

Установим nginx: *apt update && apt install -y nginx*

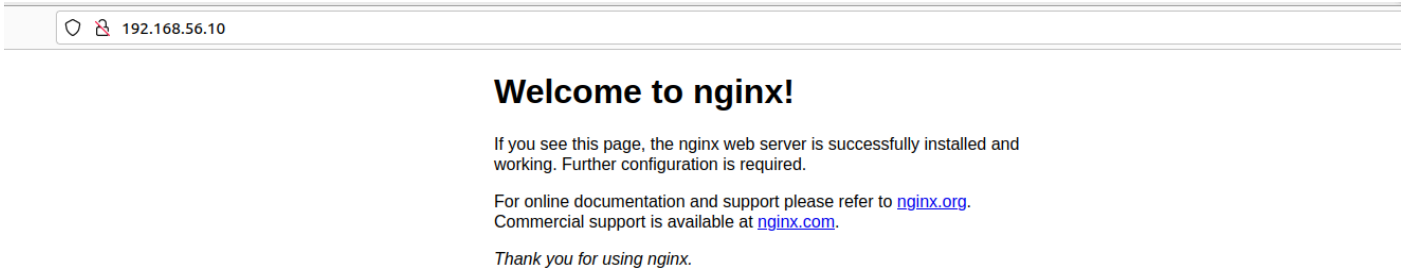
Проверим, что nginx работает корректно:

```
root@web:~# systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
     Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor
preset: enabled)
     Active: active (running) since Sun 2023-12-17 20:20:50 MSK; 47s ago
     Docs: man:nginx(8)
     Process: 2941 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on;
master_process on; (code=exited, status=0/SUCCESS)
     Process: 2942 ExecStart=/usr/sbin/nginx -g daemon on; master_process
on; (code=exited, status=0/SUCCESS)
   Main PID: 3034 (nginx)
     Tasks: 3 (limit: 1586)
     Memory: 4.7M
     CPU: 31ms
     CGroup: /system.slice/nginx.service
           ├─3034 "nginx: master process /usr/sbin/nginx -g daemon on;
master_process on;"
           ├─3036 "nginx: worker process" "" "" "" "" "" "" "" "" "" "" ""
"" "" "" "" "" "" "" "" "" "" "" "" "" "" "" ""
           └─3037 "nginx: worker process" "" "" "" "" "" "" "" "" "" "" ""
"" "" "" "" "" "" "" "" "" "" "" "" "" "" "" ""

Dec 17 20:20:50 web systemd[1]: Starting A high performance web server and a
reverse proxy server...
Dec 17 20:20:50 web systemd[1]: Started A high performance web server and a
reverse proxy server.

root@web:~#
root@web:~# ss -tln | grep 80
LISTEN   0        511              0.0.0.0:80            0.0.0.0:*
LISTEN   0        511                 [::]:80               [::]:*
root@web:~#
```

Также работу nginx можно проверить на хосте. В браузере ввведем в адерсную строку http://192.168.56.10



Видим что nginx запускается корректно.

3. Настройка центрального сервера сбора логов

Откроем еще одно окно терминала и подключаемся по ssh к ВМ log: *vagrant ssh log*
Перейдем в пользователя root: *sudo -i*

rsyslog должен быть установлен по умолчанию в нашей ОС, проверим это:

```
root@log:~# apt list rsyslog
Listing... Done
rsyslog/jammy-updates,jammy-security,now 8.2112.0-2ubuntu2.2 amd64
[installed,automatic]
```

Все настройки Rsyslog хранятся в файле */etc*/rsyslog.conf

Для того, чтобы наш сервер мог принимать логи, нам необходимо внести следующие изменения в файл:

Открываем порт 514 (TCP и UDP):

Находим закомментированные строки:

```
# provides UDP syslog reception
#module(load="imudp")
#input(type="imudp" port="514")

# provides TCP syslog reception
#module(load="imtcp")
#input(type="imtcp" port="514")
```

И приводим их к виду:

```
# provides UDP syslog reception
module(load="imudp")
input(type="imudp" port="514")

# provides TCP syslog reception
module(load="imtcp")
input(type="imtcp" port="514")
```

```
#Add remote logs
$template RemoteLogs,"/var/log/rsyslog/%HOSTNAME%/%PROGRAMNAME%.log"
*.* ?RemoteLogs
& ~
```

В конец файла */etc*/rsyslog.conf добавляем правила приёма сообщений от хостов:

Данные параметры будут отправлять в папку */var*/log/rsyslog логи, которые будут приходить от других серверов. Например, Access-логи nginx от сервера web, будут идти в файл */var*/log/rsyslog/web/nginx_access.log

Далее сохраняем файл и перезапускаем службу rsyslog: *systemctl restart rsyslog*

Если ошибок не допущено, то у нас будут видны открытые порты TCP,UDP 514:

```
root@log:~# ss -tuln
Netid  State      Recv-Q Send-Q     Local Address:Port      Peer Address:Port
Process
udp    UNCONN     0      0          0.0.0.0:514            0.0.0.0:*
udp    UNCONN     0      0          127.0.0.53%lo:53             0.0.0.0:*
udp    UNCONN     0      0          10.0.2.15%eth0:68            0.0.0.0:*
udp    UNCONN     0      0                [::]:514               [::]:*
tcp    LISTEN     0      4096       127.0.0.53%lo:53             0.0.0.0:*
tcp    LISTEN     0      128        0.0.0.0:22             0.0.0.0:*
tcp    LISTEN     0      25         0.0.0.0:514            0.0.0.0:*
tcp    LISTEN     0      128              [::]:22                [::]:*
tcp    LISTEN     0      25               [::]:514               [::]:*
```

Далее настроим отправку логов с web-сервера

Заходим на web сервер: *vagrant ssh web*

Переходим в root пользователя: *sudo -i*

Проверим версию nginx: nginx -v

Версия nginx должна быть 1.7 или выше. В нашем примере используется версия nginx 1.18.

Находим в файле */etc/nginx/nginx.conf раздел с логами и приводим их к следующему виду:*

```
error_log  /var/log/nginx/error.log;
error_log  syslog:server=192.168.56.15:514,tag=nginx_error;
access_log syslog:server=192.168.56.15:514,tag=nginx_access,severity=info combined;
```

Для Access-логов указываем удаленный сервер и уровень логов, которые нужно отправлять. Для error_log добавляем удаленный сервер. *Если требуется чтобы логи хранились локально и отправлялись на удаленный сервер, требуется указать 2 строки.*

Tag нужен для того, чтобы логи записывались в разные файлы.

По умолчанию, error-логи отправляют логи, которые имеют severity: error, crit, alert и emerg. Если требуется хранить или пересылать логи с другим severity, то это также можно указать в настройках nginx.

```
root@web:~# nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
```

Далее проверяем, что конфигурация nginx указана правильно: *nginx -t*

Далее перезапускаем nginx: *systemctl restart nginx*

Попробуем несколько раз зайти по адресу http://192.168.56.10

Далее заходим на log-сервер и смотрим информацию об nginx:

- *cat /var/log/rsyslog/web/nginx_access.log*
- *cat /var/log/rsyslog/web/nginx_error.log*

Поскольку наше приложение работает без ошибок, файл *nginx_error.log* не будет создан. Чтобы сгенерировать ошибку, можно переместить файл веб-страницы, который открывает nginx -

mv /var/www/html/index.nginx-debian.html /var/www/

После этого мы получим 403 ошибку.

Видим, что логи отправляются корректно.

```
root@log:~# cat /var/log/rsyslog/web/nginx_error.log
Dec 17 20:36:15 web nginx_error: 2023/12/17 20:36:15 [error] 3132#3132: *2
directory index of "/var/www/html/" is forbidden, client: 192.168.56.1,
server: _, request: "GET / HTTP/1.1", host: "192.168.56.10"
Dec 17 20:36:16 web nginx_error: 2023/12/17 20:36:16 [error] 3132#3132: *2
directory index of "/var/www/html/" is forbidden, client: 192.168.56.1,
server: _, request: "GET / HTTP/1.1", host: "192.168.56.10"

root@log:~# cat /var/log/rsyslog/web/nginx_access.log
Dec 17 20:34:40 web nginx_access: 192.168.56.1 - - [17/Dec/2023:20:34:40
+0300] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (X11; Ubuntu; Linux x86_64;
rv:109.0) Gecko/20100101 Firefox/116.0"
Dec 17 20:34:40 web nginx_access: message repeated 2 times: [ 192.168.56.1 -
- [17/Dec/2023:20:34:40 +0300] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (X11;
Ubuntu; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/116.0"]
Dec 17 20:34:41 web nginx_access: 192.168.56.1 - - [17/Dec/2023:20:34:41
+0300] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (X11; Ubuntu; Linux x86_64;
rv:109.0) Gecko/20100101 Firefox/116.0"
Dec 17 20:34:41 web nginx_access: message repeated 4 times: [ 192.168.56.1 -
- [17/Dec/2023:20:34:41 +0300] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (X11;
Ubuntu; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/116.0"]
Dec 17 20:34:42 web nginx_access: 192.168.56.1 - - [17/Dec/2023:20:34:42
+0300] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (X11; Ubuntu; Linux x86_64;
rv:109.0) Gecko/20100101 Firefox/116.0"
```

4. Создать третью виртуальную машину, настроить rsyslog на отправку всех логов на центральный сервер логов.

Пример настройки rsyslog для отправки логов взять из лекционного материала.

Критерии оценивания

Статус "Принято" ставится при выполнении следующих условий:
1. Ссылка на репозиторий GitHub.
2. Логи и скриншоты без Vagrantfile

Помимо базового задания **рекомендуется** сделать данное задание следующим образом:
3. Все настройки сделать с помощью Ansible и добавить запуск Ansible playbook из Vagrantfile.
Для запуска playbook по команде *vagrant up* достаточно добавить следующую конструкцию в раздел Boxes:

```
boxes.each do |opts|
    config.vm.define opts[:name] do |config|
      config.vm.hostname = opts[:name]
      config.vm.network "private_network", ip: opts[:ip]

      if opts[:name] == boxes.last[:name]
        config.vm.provision "ansible" do |ansible|
          ansible.playbook = "ansible/provision.yml"
          ansible.inventory_path = "ansible/hosts"
          ansible.host_key_checking = "false"
          ansible.limit = "all"
        end
      end

    end
  end
```

4. Предоставить Vagrantfile

Опционально для выполнения:
* развернуть еще машину с elk
- таким образом настроить 2 центральных лог системы elk и какую либо еще;
- в elk должны уходить только логи nginx;
- во вторую систему все остальное.

Рекомендуемые источники

- Статья «Настройка rsyslog для хранения логов на удаленном сервере» - https://www.dmosk.ru/miniinstruktions.php?mini=rsyslog
- Статья «Elasticsearch + Kibana + Logstash на Ubuntu» - https://www.dmosk.ru/instruktions.php?object=elk-ubuntu#clients-install-ubuntu
- Статья «Директивы в nginx» - https://nginx.org/ru/docs/ngx_core_module.html#error_log