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

**Vagrant-стенд c PAM**

**Цель домашнего задания**

Научиться создавать пользователей и добавлять им ограничения

**Описание домашнего задания**

1. Запретить всем пользователям кроме группы admin логин в выходные (суббота и воскресенье), без учета праздников

\* дать конкретному пользователю права работать с докером и возможность перезапускать докер сервис

**Введение**

Почти все операционные системы Linux — многопользовательские. Администратор Linux должен уметь создать и настраивать пользователей.

В Linux есть 3 группы пользователей:

* Администраторы — привилегированные пользователи с полным доступом к системе. По умолчанию в ОС есть такой пользователь — root
* Локальные пользователи — их учетные записи создает администратор, их права ограничены. Администраторы могут изменять права локальных пользователей
* Системные пользователи — учетный записи, которые создаются системой для внутренних процессов и служб. Например пользователь — nginx

У каждого пользователя есть свой уникальный идентификатор — UID.

Чтобы упростить процесс настройки прав для новых пользователей, их объединяют в группы. Каждая группа имеет свой набор прав и ограничений. Любой пользователь, создаваемый или добавляемый в такую группу, автоматически их наследует. Если при добавлении пользователя для него не указать группу, то у него будет своя, индивидуальная группа — с именем пользователя. Один пользователь может одновременно входить в несколько групп.

Информацию о каждом пользователе сервера можно посмотреть в файле /etc/passwd

Для более точных настроек пользователей можно использовать подключаемые модули аутентификации (PAM)

PAM (Pluggable Authentication Modules - подключаемые модули аутентификации) — набор библиотек, которые позволяют интегрировать различные методы аутентификации в виде единого API.

PAM решает следующие задачи:

* Аутентификация — процесс подтверждения пользователем своей подлинности. Например: ввод логина и пароля, ssh-ключ и т д.
* Авторизация — процесс наделения пользователя правами
* Отчетность — запись информации о произошедших событиях

PAM может быть реализован несколькими способами:

* Модуль pam\_time — настройка доступа для пользователя с учетом времени
* Модуль pam\_exec — настройка доступа для пользователей с помощью скриптов
* И т.д.

**Формат сдачи:**

Vagrantfile + ansible

**Критерии оценивания**

Статус «Принято» ставится при выполнении следующих условий:

1. Ссылка на репозиторий GitHub.

2. Vagrantfile, который будет разворачивать виртуальную машину

3. Документация по каждому заданию:

Создайте файл README.md и снабдите его следующей информацией:

- название выполняемого задания;

- текст задания;

- описание команд и их вывод;

- особенности проектирования и реализации решения,

- заметки, если считаете, что имеет смысл их зафиксировать в репозитории.

**Функциональные и нефункциональные требования**

* ПК на Unix c 8ГБ ОЗУ или виртуальная машина с включенной Nested Virtualization.
* Созданный аккаунт на GitHub - <https://github.com/>
* Если Вы находитесь в России, для корректной работы Вам может потребоваться VPN.

**Предварительно установленное и настроенное следующее ПО:**

* Hashicorp Vagrant (<https://www.vagrantup.com/downloads>)
* Oracle VirtualBox (<https://www.virtualbox.org/wiki/Linux_Downloads>).
* 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 v6.1.26 r145957. В качестве ОС на хостах установлена Ubuntu 22.04. Серьёзные отступления от этой конфигурации могут потребовать адаптации с вашей стороны.

Создадим Vagrantfile, в котором будут указаны параметры наших ВМ:

MACHINES = {

:"pam" => {

:box\_name => "ubuntu/jammy64",

:cpus => 2,

:memory => 1024,

:ip => "192.168.57.10",

}

}

Vagrant.configure("2") do |config|

MACHINES.each do |boxname, boxconfig|

config.vm.synced\_folder ".", "/vagrant", disabled: true

config.vm.network "private\_network", ip: boxconfig[:ip]

config.vm.define boxname do |box|

box.vm.box = boxconfig[:box\_name]

box.vm.box\_version = boxconfig[:box\_version]

box.vm.host\_name = boxname.to\_s

box.vm.provider "virtualbox" do |v|

v.memory = boxconfig[:memory]

v.cpus = boxconfig[:cpus]

end

box.vm.provision "shell", inline: <<-SHELL

sed -i 's/^PasswordAuthentication.\*$/PasswordAuthentication yes/' /etc/ssh/sshd\_config

systemctl restart sshd.service

SHELL

end

end

end

После создания Vagrantfile запустим нашу ВМ командой *vagrant up*. Будет создана одна виртуальная машина.

Уточним некоторые моменты из Vagrantfile:

* Так как в данной лабораторной работе нам предстоит подключаться к нашей ВМ в неё добавлен дополнительный сетевой интерфейс:

...

# Указываем IP-адрес для ВМ

:ip => "192.168.57.10",

...

# Добавляем сетевой интерфейс

config.vm.network "private\_network", ip: boxconfig[:ip]

...

* Для удобства, в параметрах SSH разрешена аутентификация пользователя по паролю:

box.vm.provision "shell", inline: <<-SHELL

#Разрешаем подключение пользователей по SSH с использованием пароля

sed -i 's/^PasswordAuthentication.\*$/PasswordAuthentication yes/' /etc/ssh/sshd\_config

#Перезапуск службы SSHD

systemctl restart sshd.service

SHELL

Настройка запрета для всех пользователей (кроме группы Admin) логина в выходные дни (Праздники не учитываются)

1. Подключаемся к нашей созданной ВМ: vagrant ssh

2. Переходим в root-пользователя: sudo -i

3. Создаём пользователя otusadm и otus: sudo useradd otusadm && sudo useradd otus

4. Создаём пользователям пароли: echo "Otus2022!" | sudo passwd --stdin otusadm && echo "Otus2022!" | sudo passwd --stdin otus

*Для примера мы указываем одинаковые пароли для пользователя otus и otusadm*

5. Создаём группу admin: sudo groupadd -f admin

6. Добавляем пользователей vagrant,root и otusadm в группу admin:

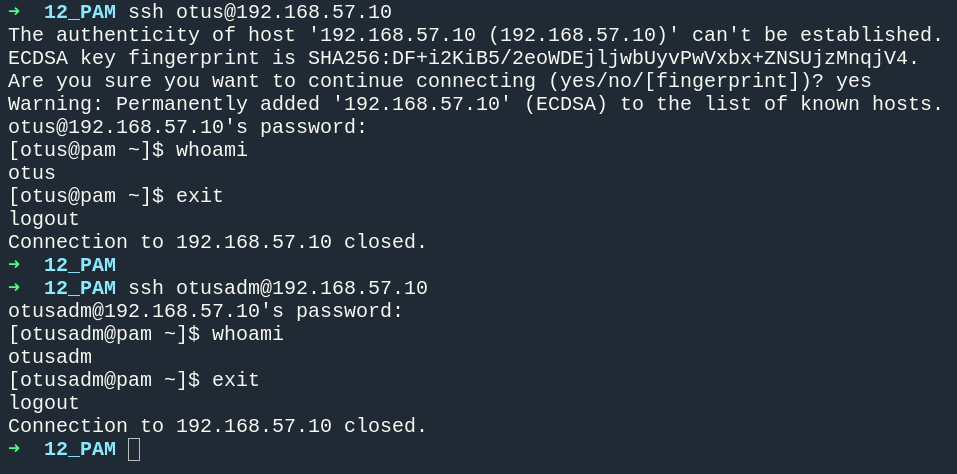
usermod otusadm -a -G admin && usermod root -a -G admin && usermod vagrant -a -G admin

*Обратите внимание, что мы просто добавили пользователя otusadm в группу admin. Это не делает пользователя otusadm администратором.*

После создания пользователей, нужно проверить, что они могут подключаться по SSH к нашей ВМ. Для этого пытаемся подключиться с хостовой машины:

*ssh otus@192.168.57.10*

*Далее вводим наш созданный пароль.*



*Если всё настроено правильно, на этом моменте мы сможем подключиться по SSH под пользователем otus и otusadm*

Далее настроим правило, по которому все пользователи кроме тех, что указаны в группе admin не смогут подключаться в выходные дни:

7. Проверим, что пользователи root, vagrant и otusadm есть в группе admin:

[root@pam ~]# cat /etc/group | grep admin

printadmin:x:994:

admin:x:1003:otusadm,root,vagrant

[root@pam ~]#

*Информация о группах и пользователях в них хранится в файле /etc/group, пользователи указываются через запятую.*

Выберем метод PAM-аутентификации, так как у нас используется только ограничение по времени, то было бы логично использовать метод pam\_time, однако, данный метод не работает с локальными группами пользователей, и, получается, что использование данного метода добавит нам большое количество однообразных строк с разными пользователями. В текущей ситуации лучше написать небольшой скрипт контроля и использовать модуль pam\_exec

8. Создадим файл-скрипт /usr/local/bin/login.sh

*vim /usr/local/bin/login.sh*

*#!/bin/bash*

#Первое условие: если день недели суббота или воскресенье

if [ $(date +%a) = "Sat" ] || [ $(date +%a) = "Sun" ]; then

#Второе условие: входит ли пользователь в группу admin

if getent group admin | grep -qw "$PAM\_USER"; then

#Если пользователь входит в группу admin, то он может подключиться

exit 0

else

#Иначе ошибка (не сможет подключиться)

exit 1

fi

#Если день не выходной, то подключиться может любой пользователь

else

exit 0

*fi*

*В скрипте подписаны все условия. Скрипт работает по принципу:*

*Если сегодня суббота или воскресенье, то нужно проверить, входит ли пользователь в группу admin, если не входит — то подключение запрещено. При любых других вариантах подключение разрешено.*

9. Добавим права на исполнение файла: chmod +x /usr/local/bin/login.sh

10. Укажем в файле /etc/pam.d/sshd модуль pam\_exec и наш скрипт:

vim /etc/pam.d/sshd

#%PAM-1.0

auth substack password-auth

auth include postlogin

auth required pam\_exec.so debug /usr/local/bin/login.sh

account required dad

account required pam\_nologin.so

account include password-auth

password include password-auth

# pam\_selinux.so close should be the first session rule

session required pam\_selinux.so close

session required pam\_loginuid.so

# pam\_selinux.so open should only be followed by sessions to be executed in the user context

session required pam\_selinux.so open env\_params

session required pam\_namespace.so

session optional pam\_keyinit.so force revoke

session optional pam\_motd.so

session include password-auth

session include postlogin

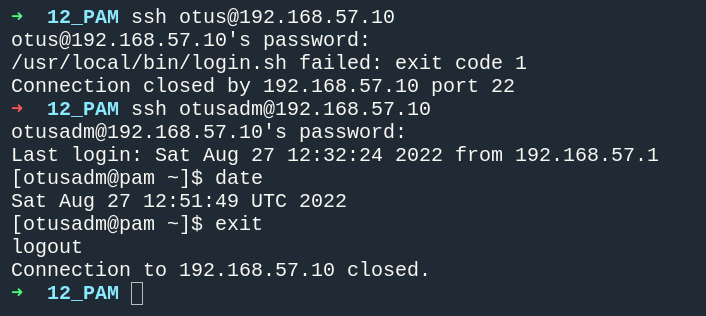
На этом настройка завершена, нужно только проверить, что скрипт отрабатывает корректно.

Если Вы выполняете данную работу в выходные, то можно сразу попробовать подключиться к нашей ВМ. Если нет, тогда можно руками поменять время в нашей ОС, например установить 27 августа 2022 года (Суббота):

sudo date 082712302022.00

и попробовать подключиться.

Если настройки выполнены правильно, то при логине пользователя otus у Вас должна появиться ошибка. Пользователь otusadm должен подключаться без проблем:



**Рекомендуемые источники**

* [Статья об управлении пользователей](https://firstvds.ru/technology/linux-user-management#:~:text=%D0%92%20Linux%20%D1%81%D1%83%D1%89%D0%B5%D1%81%D1%82%D0%B2%D1%83%D1%8E%D1%82%20%D1%82%D1%80%D0%B8%20%D1%82%D0%B8%D0%BF%D0%B0,%D0%9B%D0%BE%D0%BA%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5%20%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D0%B8%20%E2%80%94%20%D0%BD%D0%B5%D0%BF%D1%80%D0%B8%D0%B2%D0%B8%D0%BB%D0%B5%D0%B3%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D1%8B%D0%B5%20%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D0%B8)