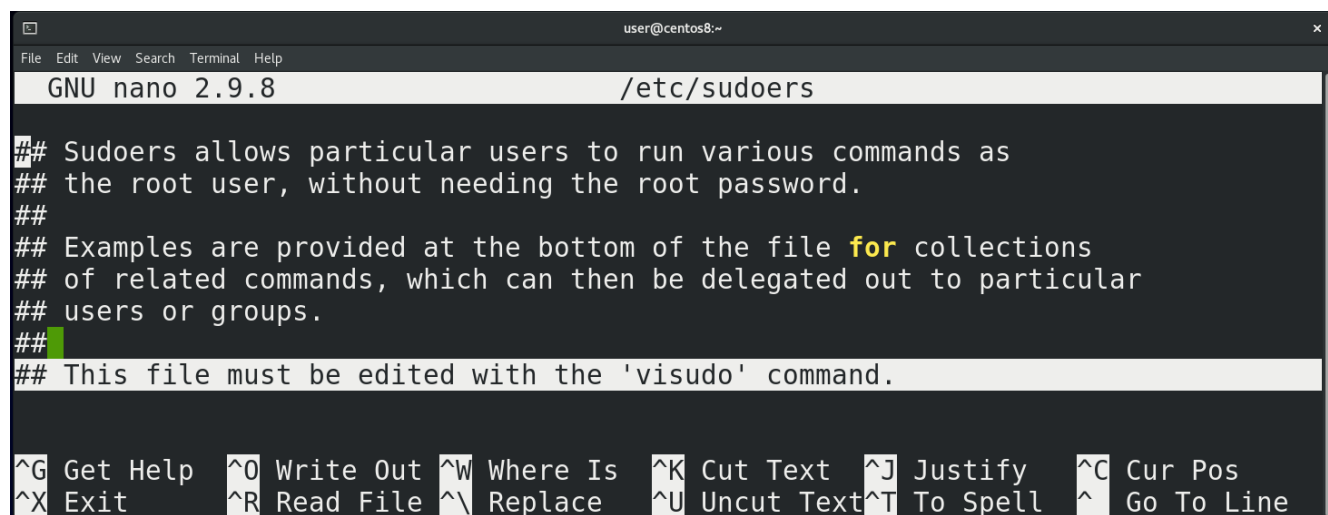


У su есть один нюанс – нужно знать пароли других пользователей, если, конечно, вы не root. Когда вы один админ в системе - это не проблема. Но представьте, что есть несколько пользователей или администраторов. Давать друг другу пароли плохая идея, потому что безопасность строится на полном недоверии друг к другу, включая других администраторов. Другой администратор может умышленно вас подставить, неопытный хелпдеск может сделать ошибку от вашего имени, завтра могут уволить кого-то, а он от вашего имени может положить все сервера. Ну и если не говорить о недоверии, когда несколько админов работают от имени одного пользователя, то кто-то может что-то неправильно настроить, и потом не вспомнит или не сознается. Таким образом в коллективе формируется недоверие и негативная атмосфера. Учётные записи придумали не просто так, в жизни бывает всякое, поэтому всем будет легче работать от своего имени. Ну и во вторых, когда работает несколько пользователей, допустим если есть хелпдеск и вы хотите дать ему возможность только создавать пользователей, то вроде ему и нужны какие-то права суперпользователя, но и давать пароль от рута не хочется.

Для решения таких проблем есть утилита sudo. Вы уже замечали её пару раз – с помощью неё мы устанавливали какие-то программы и создавали пользователей – потому что у нашего пользователя не было соответствующих прав. А если писать перед командой sudo – например, sudo useradd user2 – то команда useradd запускается от имени суперпользователя. Но, естественно, не любой пользователь может написать sudo и выполнить любую команду – у нашего user это работает, потому что при установке системы мы поставили галочку “сделать пользователя администратором”. Как это работает – давайте разбираться.

Вкратце, sudo позволяет запускать какие-то команды от имени какого-то пользователя. А в файле sudoers пишутся те самые политики – кому, где и что. Из того, что мы уже видели – наш пользователь может запускать команды от имени суперпользователя. Давайте для начала найдём, где это написано.

A screenshot of a terminal window titled 'user@centos8:~'. The terminal shows the GNU nano 2.9.8 editor editing the file /etc/sudoers. The file content includes comments about sudoers allowing users to run commands as the root user without the root password, examples of delegation, and a note that the file must be edited with the 'visudo' command. The bottom of the screen shows a status bar with various keyboard shortcuts like ^G Get Help, ^O Write Out, ^W Where Is, ^K Cut Text, ^J Justify, ^C Cur Pos, ^X Exit, ^R Read File, ^\ Replace, ^U Uncut Text, ^T To Spell, and ^_ Go To Line.

```
user@centos8:~  
GNU nano 2.9.8 /etc/sudoers  
## Sudoers allows particular users to run various commands as  
## the root user, without needing the root password.  
##  
## Examples are provided at the bottom of the file for collections  
## of related commands, which can then be delegated out to particular  
## users or groups.  
##  
## This file must be edited with the 'visudo' command.  
  
^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text  ^J Justify    ^C Cur Pos  
^X Exit      ^R Read File  ^\ Replace   ^U Uncut Text ^T To Spell   ^_ Go To Line
```

У sudo есть файл настроек - /etc/sudoers - nano /etc/sudoers. Наш пользователь его читать не может, но так как у нас есть права суперпользователя, можем написать sudo nano /etc/passwd, введём свой пароль и файл откроется. В начале у нас есть строчка – этот файл нужно редактировать с помощью команды visudo. Всё дело в том, что при запуске sudo программа каждый раз считывает этот файл, и, если здесь будет синтаксическая ошибка, sudo просто перестанет работать. А если у нас нет пароля от рута – то придётся повозиться, чтобы восстановить работу.

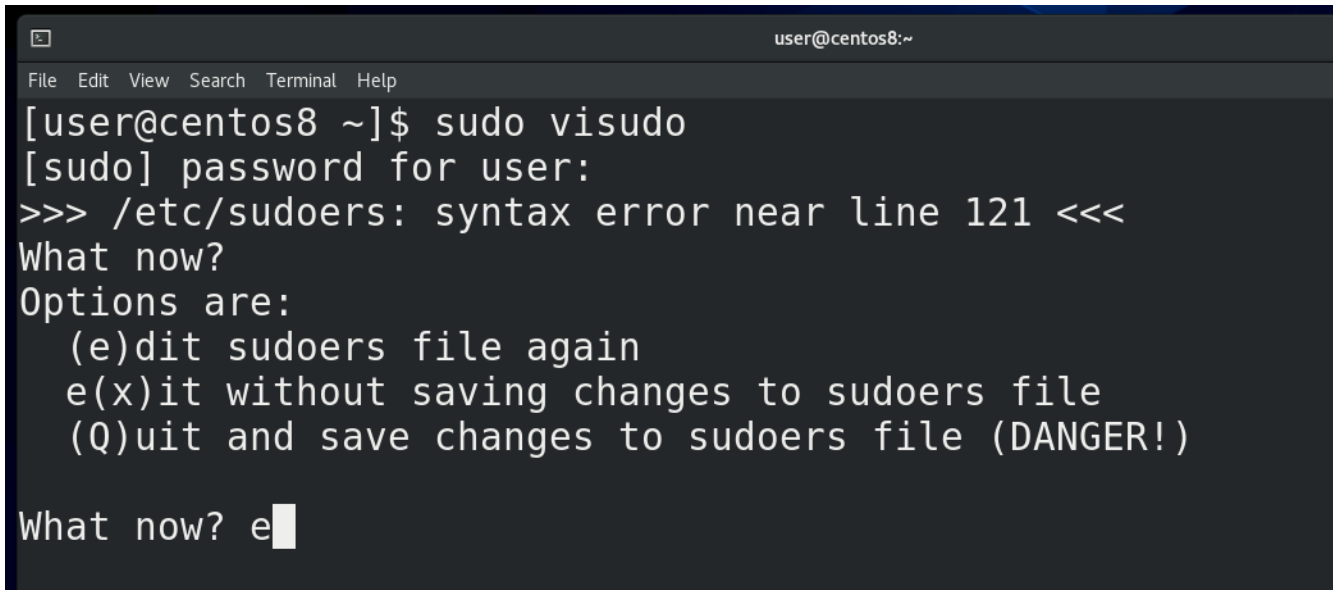
```
user@centos8:~  
File Edit View Search Terminal Help  
GNU nano 2.9.8 /etc/sudoers  
  
asdasdasd  
  
^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify  
^X Exit        ^R Read File   ^\ Replace     ^U Uncut Text  ^T To Spell  
  
user@centos8:~  
File Edit View Search Terminal Help  
[user@centos8 ~]$ sudo nano /etc/sudoers  
>>> /etc/sudoers: syntax error near line 123 <<<  
sudo: parse error in /etc/sudoers near line 123  
sudo: no valid sudoers sources found, quitting  
sudo: unable to initialize policy plugin  
[user@centos8 ~]$
```

Давайте просто покажу – напишу в файле рандомные буквы, сохраню и выйду. Теперь, при попытке заново открыть файл с помощью sudo, у меня ничего не откроется - sudo скажет, что не смог прочесть такую-то строчку и просто откажется работать. К счастью, у меня есть пароль от рута, который я ставил при установке системы – я могу просто написать su, ввести пароль рута, а потом просто зайти и удалить лишнюю строчку — nano /etc/sudoers. Теперь sudo опять будет работать.

```
user@centos8:~  
File Edit View Search Terminal Help  
GNU nano 2.9.8 /home/user/.bashrc  
  
# User specific aliases and functions  
export EDITOR=nano  
  
user@centos8:~  
File Edit View Search Terminal Help  
GNU nano 2.9.8 /etc/sudoers  
  
Defaults    env_reset  
Defaults    env_keep = "EDITOR COLORS DISPLAY HOSTNAME HISTSIZE KDEDIR LS_$  
Defaults    env_keep += "MAIL PS1 PS2 QDIR USERNAME LANG LC_ADDRESS LC_CTY$  
  
^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify  
^X Exit        ^R Read File   ^\ Replace     ^U Uncut Text  ^T To Spell
```

Попробуем прислушаться к совету и запустить команду visudo - sudo visudo. Visudo открывает /etc/sudoers в дефолтном редакторе – vim. Если же я хочу, чтобы всё работало с nano, мне нужно будет кое-что поменять. Как правило, программы по умолчанию задаются с помощью переменных окружения. Как это делать мы уже знаем – зайдём в ~/.bashrc - nano ~/.bashrc - и

напишем – `export EDITOR=nano`. Далее нам нужно зайти в `sudoers` – `sudo nano /etc/sudoers`, найти строчку с `env_keep`. Выше можно заметить `env_reset` – которая сбрасывает все переменные, чтобы сделать окружение с `sudo` минимальным. Так вот, в `env_keep` добавляем `EDITOR`, чтобы `sudoers` не игнорировал нашу переменную, сохраняем, закроем эмулятор и заново запустим, чтобы `bash` перечитал настройки `~/.bashrc`. И теперь, при запуске `sudo visudo`, файл откроется в `nano`.



```
user@centos8:~  
File Edit View Search Terminal Help  
[user@centos8 ~]$ sudo visudo  
[sudo] password for user:  
>>> /etc/sudoers: syntax error near line 121 <<<  
What now?  
Options are:  
  (e)dit sudoers file again  
  e(x)it without saving changes to sudoers file  
  (Q)uit and save changes to sudoers file (DANGER!)  
What now? e
```

Обратите внимание - наверху указано не `/etc/sudoers`, а `/etc/sudoers.tmp`. Это означает, что когда мы запускаем `visudo`, у нас `sudoers` копируется во временный файл. Если здесь сделать ошибку, например, написать рандомные буквы, сохранить и попытаться выйти, то `visudo` сначала проверит, всё ли нормально с файлом, заметит синтаксическую ошибку и спросит нас – что делать? Если нажать `enter`, можно увидеть варианты – `e` чтобы вернуться к редактированию, `x` – чтобы выйти не сохраняя, `Q` – чтобы сохранить, но мы видим предупреждение, что это опасно. Можно вернуться, исправить ошибку, сохранить и выйти. При сохранении, `sudoers.tmp` заменит оригинальный файл `sudoers`. Таким образом `visudo` защищает нас от синтаксических ошибок. Но следует запомнить, что изменения вступают в силу только после сохранения и выхода из `visudo`.

```
user@centos8:~  
File Edit View Search Terminal Help  
GNU nano 2.9.8 /etc/sudoers.tmp  
## Allow root to run any commands anywhere  
root ALL=(ALL) ALL  
  
## Allows members of the 'sys' group to run networking, software,  
## service management apps and more.  
# %sys ALL = NETWORKING, SOFTWARE, SERVICES, STORAGE, DELEGATING, PROCESSES  
  
## Allows people in group wheel to run all commands  
%wheel ALL=(ALL) ALL  
  
## Same thing without a password  
# %wheel ALL=(ALL) NOPASSWD: ALL
```

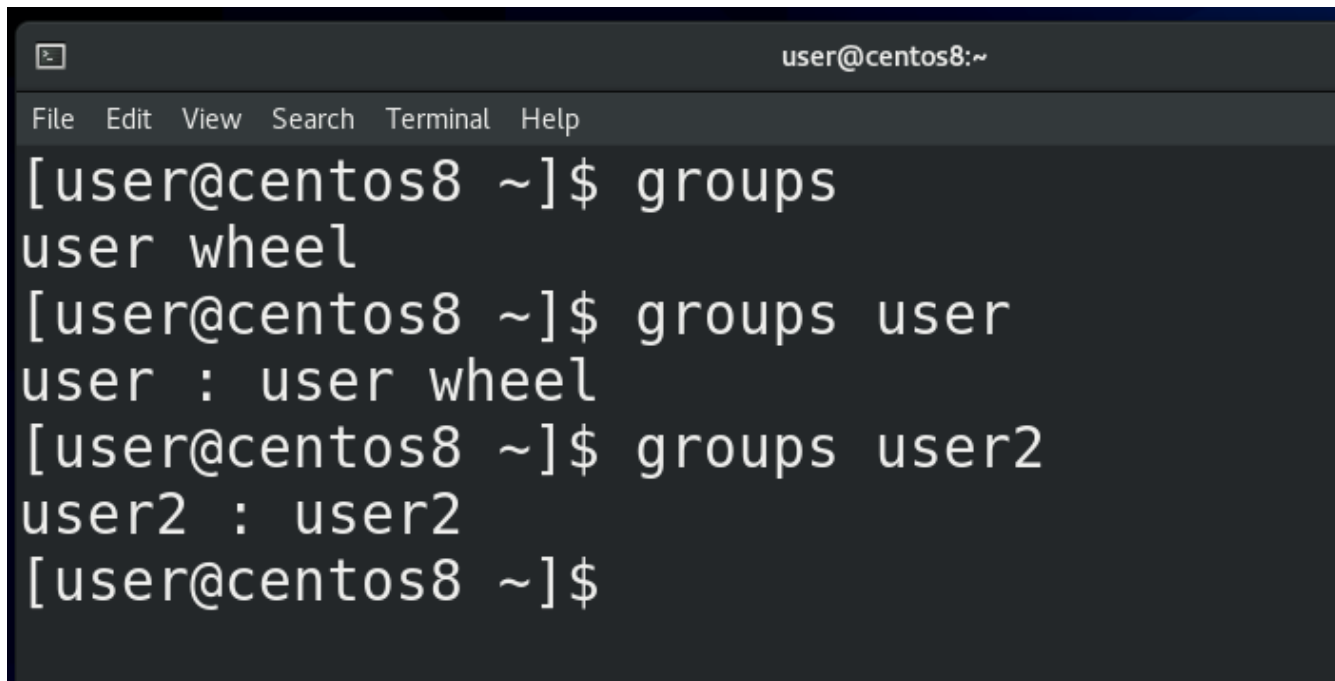
Спускаемся вниз, где у нас начинаются политики. Например, root ALL=(ALL) ALL. Первое – это пользователь, для которого написана политика. Если начинается со знака процент, то речь о группе пользователей. Не то, чтобы руту были нужны sudo права – он и так может делать всё что угодно. Но без этой строчки - если root почему-то запустит sudo, например, если это написано в скрипте, или кто-то скопировал команды с интернета – sudo выдаст ошибку и команда не сработает. Поэтому, на всякий случай, есть эта строчка.

Теперь, что означает ALL=(ALL) ALL. Первая ALL - это на каком компьютере. Если первое значение ALL или совпадает с именем компьютера, которое можно узнать с помощью команды hostname, то sudo будет работать с этой строчкой. Если же тут написано что-то другое – то sudo проигнорирует эту строчку. Если у вас несколько компьютеров, вы можете написать один файл sudoers и закинуть его на все компьютеры. И sudo на каждой машинке будет считывать только строчки, где хост равен ALL или совпадает с хостнеймом машинки.

```
user@centos8:~  
File Edit View Search Terminal Help  
GNU nano 2.9.8 /etc/sudoers.tmp  
## Allow root to run any commands anywhere  
root ALL=(user2) ALL  
  
## Allows members of the 'sys' group to run networking, software,  
## service management apps and more.  
# %sys ALL = NETWORKING, SOFTWARE, SERVICES, STORAGE, DELEGATING,
```

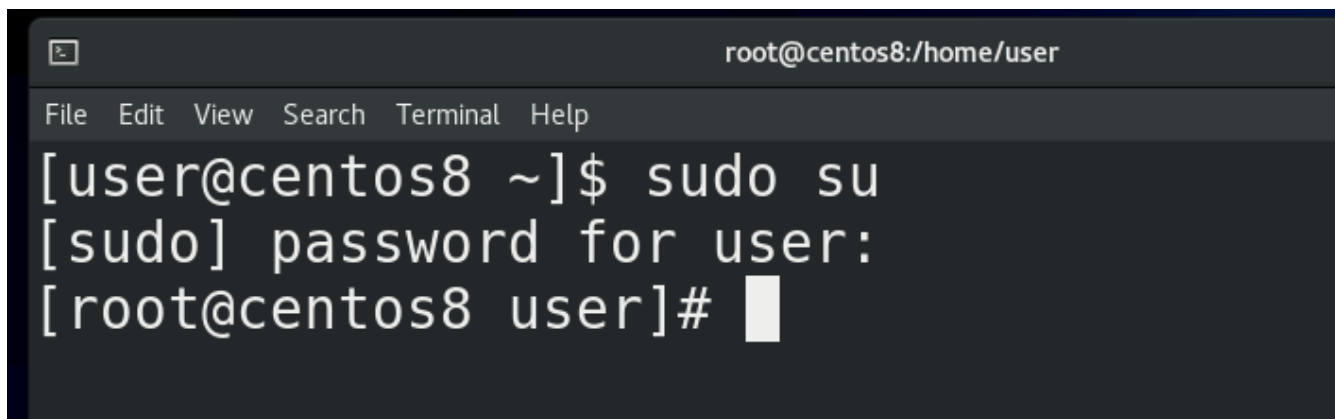
Вторая ALL – от имени какого пользователя, ALL значит от всех, то есть можно запустить команду от имени любого пользователя. По умолчанию, если явно не указывать пользователя, то команда запустится от имени root. А так, здесь можно написать конкретного пользователя, допустим user2. В таком случае пользователь root сможет запускать команды с помощью sudo только от имени user2. Также тут можно указать не только пользователя, а также группу, или просто группу. Чуть дальше будут примеры.

И третья ALL – это команды. В случае ALL можно запустить любую команду, а так, здесь можно прописать только те команды, которые вы хотите разрешить. Или наоборот, если вы хотите какие-то команды разрешить и какие-то запретить. Перед запрещёнными командами ставится восклицательный знак.

A terminal window titled 'user@centos8:~' with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the following commands and output:

```
[user@centos8 ~]$ groups
user wheel
[user@centos8 ~]$ groups user
user : user wheel
[user@centos8 ~]$ groups user2
user2 : user2
[user@centos8 ~]$
```

Так вот, почему наш пользователь user может запускать команды с помощью sudo? Пока что в этом файле всего 2 политики – для пользователя root и для группы wheel, всё остальное – закомментированные примеры. У группы wheel тоже все права – ALL ALL ALL. Чтобы узнать, в каких группах состоит наш пользователь, можно выполнить команду groups от его имени, либо указать имя пользователя — groups; groups user; groups user2. Как видно, наш пользователь user состоит в группе wheel – именно поэтому у него есть все права на систему. В других дистрибутивах вместо wheel может быть группа с именем sudo – но суть одна и та же.

A terminal window titled 'root@centos8:/home/user' with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the following commands and output:

```
[user@centos8 ~]$ sudo su
[sudo] password for user:
[root@centos8 user]#
```

Прежде чем пойдём дальше, давайте я покажу частный случай использования sudo – sudo su. Если выполнить просто команду su, то мы меняем текущего пользователя на root пользователя, правда для этого нам нужно знать пароль этого рут пользователя. Команда sudo позволяет нам выполнить команду от имени суперпользователя, а мы знаем, что если выполнять

команду `su` от имени `root` пользователя, то пароль нам указывать не нужно. А значит выполнив `sudo su` и введя пароль своего юзера, мы просто от имени суперпользователя запустим команду `su`, а так как ему не нужно вводить пароль, мы просто станем рутом. Таким образом, не зная пароля рута, мы можем стать рут пользователем.

```
user@centos8:/home/user
File Edit View Search Terminal Help
[user@centos8 ~]$ echo $PATH
/home/user/.local/bin:/home/user/bin:/home/user/.local/bin:/home
/user/bin:/usr/local/bin:/usr/local/sbin:/usr/bin:/usr/sbin
[user@centos8 ~]$ mkdir ~/.local/bin
[user@centos8 ~]$ ln -s /bin/su ~/.local/bin/htop
[user@centos8 ~]$ htop
Password:
[root@centos8 user]#
```

Учитывая, что `sudo` – это механизм, который позволяет повысить свои привилегии, этим активно пользуются злоумышленники. Давайте, для примера, посмотрим один из теоретических способов с помощью переменной `PATH`. У меня в переменной `PATH` - `echo $PATH` - есть путь `/home/user/.local/bin`. Сейчас этой директории нет, но я могу её создать - `mkdir ~/.local/bin` - и сделать такую обманку – создать символическую ссылку `ln -s /bin/su ~/.local/bin/htop`. Теперь когда я пишу `htop`, `bash` смотрит переменную `PATH`, видит в начале `~/.local/bin`, находит там программу `htop` и запускает, а на самом деле это программа `su`. И представьте ситуацию, если мы разрешим пользователю запускать только команду `htop` с правами суперпользователя, допустим, чтобы понижать `nice`ness, а этот пользователь напишет `sudo htop` – на деле выполнится `sudo su` – и он введя свой пароль станет рут пользователем.

```
user@centos8:~
File Edit View Search Terminal Help
GNU nano 2.9.8 /etc/sudoers.tmp

#
# Defaults    env_keep += "HOME"

Defaults    secure_path = /sbin:/bin:/usr/sbin:/usr/bin

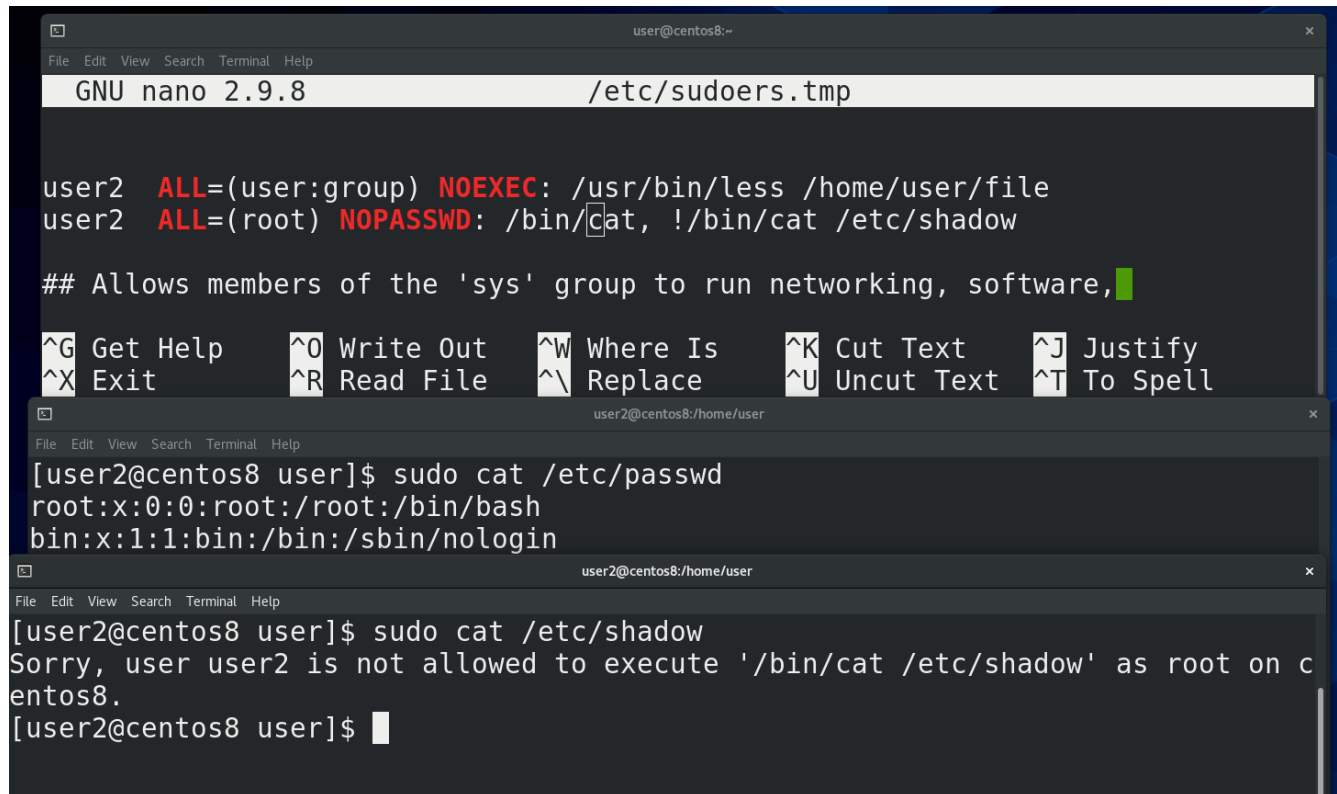
## Next comes the main part: which users can run what software $

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify
^X Exit ^R Read File ^\ Replace ^U Uncut Tex ^T To Spell
```

Естественно, этой очень банальный способ и его невозможно применить – во первых, `sudo` требует, чтобы в `sudoers` был полный путь к программам. То есть, если в `sudo` явно не указан путь `/home/user/~.local/bin/htop` – то ничего не сработает. Во вторых – в `sudoers` есть настройка `secure_path`. Когда пользователь будет писать `sudo command`, то `sudo` будет смотреть

только в директории, указанные в этой настройке. Кстати, насчёт полного пути – его всегда можно легко найти с помощью команды `which` – допустим, `which htop`, `which ls`, `which rm`.

Но нужно понимать, что при `ALL ALL ALL` у пользователя будет полный доступ, а если вы ограничиваете пользователя определёнными командами, следует быть предельно осторожным с выбором команд, потому что очень часто можно повысить привилегии неочевидным способом. Допустим, давая кому-то право создавать пользователей, он может создать пользователя с группой `wheel`, и через него стать рутом. Поэтому, прежде чем давать какому-то пользователю права на какую-то программу, следует очень внимательно изучить, что эта программа позволяет сделать.



```
user@centos8:~  
File Edit View Search Terminal Help  
GNU nano 2.9.8 /etc/sudoers.tmp  
  
user2 ALL=(user:group) NOEXEC: /usr/bin/less /home/user/file  
user2 ALL=(root) NOPASSWD: /bin/cat, !/bin/cat /etc/shadow  
  
## Allows members of the 'sys' group to run networking, software,  
  
^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text     ^J Justify  
^X Exit        ^R Read File   ^\ Replace      ^U Uncut Text   ^T To Spell  
  
user2@centos8:/home/user  
File Edit View Search Terminal Help  
[user2@centos8 user]$ sudo cat /etc/passwd  
root:x:0:0:root:/root:/bin/bash  
bin:x:1:1:bin:/bin:/sbin/nologin  
  
user2@centos8:/home/user  
File Edit View Search Terminal Help  
[user2@centos8 user]$ sudo cat /etc/shadow  
Sorry, user user2 is not allowed to execute '/bin/cat /etc/shadow' as root on c  
entos8.  
[user2@centos8 user]$
```

Допустим, тот же `less` или `vi` позволяют запускать команды, и для них есть специальный ключ, позволяющий предотвратить выполнение сторонних команд - `NOEXEC`. Кроме `NOEXEC` есть ещё пара ключей, один из примечательных - `NOPASSWD` – позволяет запускать указанные команды без ввода пароля. Также, чтобы постоянно не вводить `sudo`, можно предварительно написать `sudo -s`.

В некоторых случаях люди используют `sudo` чтобы предоставить доступ к редактированию каких-то файлов. И хотя лучше в таких случаях использовать права на файлы, ситуации бывают разные, поэтому у `sudo` есть относительно безопасный способ редактирования файлов с помощью `sudoedit`.


```
user@centos8:~  
File Edit View Search Terminal Help  
GNU nano 2.9.8 /etc/sudoers.tmp  
  
user2 ALL=(user:group) NOEXEC: /usr/bin/less /home/user/file  
user2 ALL=(root) NOPASSWD: /usr/bin/sudoedit /etc/passwd  
  
## Allows members of the 'sys' group to run networking, software,  
[ Wrote 123 lines ]  
  
^G Get Help      ^O Write Out     ^W Where Is      ^K Cut Text      ^J Justify  
^X Exit          ^R Read File     ^\ Replace       ^U Uncut Text    ^T To Spell
```

Для примера, давайте предоставлю пользователю user2 право редактировать файл /etc/passwd с помощью nano. Он вроде бы и не может открыть командой другой файл, но я могу изнутри nano открыть другой файл с помощью ^R, допустим, тот же sudoers, изменить его и сохранить как /etc/sudoers, тем самым обеспечив себе все права. Если же в sudoers я пропишу sudoedit, то предыдущая схема не будет работать, потому что принцип работы немного другой. sudoedit копирует нужный мне файл во временный файл, я редактирую временный файл, а когда сохраняю – то sudoedit заменяет нужный файл той копией, которую я изменил. Почти как с visudo.

```
user@centos8:~  
File Edit View Search Terminal Help  
[user@centos8 ~]$ sudo grep Alias /etc/sudoers  
## Host Aliases  
# Host_Alias      FILESERVERS = fs1, fs2  
# Host_Alias      MAILSERVERS = smtp, smtp2  
## User Aliases  
# User_Alias ADMIN = jsmith, mikem  
## Command Aliases  
# Cmnd_Alias NETWORKING = /sbin/route, /sbin/ifconfig, /bin/ping, /sbin/dhclient, /usr/bin/net, /sbin/iptables, /usr/bin/rfcomm, /usr/bin/wvdial, /sbin/iwconfig, /sbin/mii-tool  
# Cmnd_Alias SOFTWARE = /bin/rpm, /usr/bin/up2date, /usr/bin/yum  
# Cmnd_Alias SERVICES = /sbin/service, /sbin/chkconfig, /usr/bin/systemctl start, /usr/bin/systemctl stop, /usr/bin/systemctl reload, /usr/bin/systemctl restart, /usr/bin/systemctl status, /usr/bin/systemctl enable, /usr/bin/systemctl disable
```

Когда у вас много пользователей, много разных команд и компьютеров, для облегчения прописывания политик можно использовать алиасы, где можно перечислить несколько значений через запятую. В файле уже даны примеры использования. Политика состоит из 4 столбиков: User – это собственно пользователь или группа, к которой применяется политика, поэтому алиас называется User_Alias. Во втором столбике хостнеймы, поэтому Host_Alias. Дальше у нас столбик, где указывается от чьего имени разрешено запускать – называется Runas – собственно, Runas_Alias и последний столбик – команды, поэтому здесь Cmnd_Alias.


```
user@centos8:~$ sudo grep sudo /var/log/secure
[sudo] password for user:
Jan  1 19:16:28 centos8 sudo[4513]:    user : TTY=pts/0 ; PWD=/home/user ; USER=root ; COMMAND=/bin/su
Jan  1 19:16:28 centos8 sudo[4513]: pam_systemd(sudo:session): Cannot create session: Already running in a session or user slice
Jan  1 19:16:28 centos8 sudo[4513]: pam_unix(sudo:session): session opened for user root by (uid=0)
Jan  1 19:18:28 centos8 sudo[4513]: pam_unix(sudo:session): session closed for user root
Jan  1 19:29:26 centos8 sudo[4852]:    user : TTY=pts/0 ; PWD=/home/user ; USER=root ; COMMAND=/bin/su
```

Также бывает полезно узнать, кто какие команды вводил с помощью sudo. Все действия sudo логирует и их можно посмотреть в файле /var/log/secure - `sudo grep sudo /var/log/secure`. Также в плане логирования, у sudo есть инструмент, называемый `sudoreplay`. Я его разбирать не буду, это вам такое задание – настроить и проверить его работу. Если будут какие-то сложности – спрашивайте в комментариях.

```
user2@centos8:/home/user$ sudo visudo -f /etc/sudoers.d/test
[user@centos8 ~]$ sudo cat /etc/sudoers.d/test
user2 centos8=(user) /bin/ls
[user@centos8 ~]$ su user2
Password:
[user2@centos8 user]$ sudo passwd user
[sudo] password for user2:
Sorry, user user2 is not allowed to execute '/bin/passwd user' as root on centos8.
[user2@centos8 user]$ sudo ls /home/user
[sudo] password for user2:
Sorry, user user2 is not allowed to execute '/bin/ls /home/user' as root on centos8.
[user2@centos8 user]$ sudo -u user ls /home/user
[sudo] password for user2:
Desktop  dir3      Downloads file      Music  Pictures Templates
dir1     Documents errors   filelist output  Public  Videos
[user2@centos8 user]$
```

Последняя строчка файла говорит нам, что sudoers будет читать настройки из всех файлов, расположенных внутри директории /etc/sudoers.d . Для этого нам понадобится указать для visudo файл – `sudo visudo -f /etc/sudoers.d/test` . Давайте пропишем здесь правило для пользователя user2, допустим, чтобы он мог от имени пользователя user запускать команду `ls - user2 centos8=(user) /bin/ls` . Сохраним, выйдем и проверим – `su user2, sudo passwd user, ls /home/user/, sudo -u user ls /home/user`.

Подводя итоги, sudo – инструмент, который позволяет дать определённым пользователям определённые права. Но это делает sudo очень опасным инструментом, которым активно пользуются злоумышленники. Поэтому нужно запомнить – без острой необходимости

пользователей в `sudo` прописывать не стоит, если речь идёт о правах суперпользователя. Не стоит строить правила на основе запретов – разрешить всё, а потом запретить опасные команды. Это заведомо проигрышный вариант – есть огромное количество способов обойти запреты. Давайте доступ только на необходимые команды, заранее проанализируйте, посмотрите в интернете, а насколько опасно то, что вы разрешаете. В дальнейшем вы научитесь писать скрипты – так вот, вместо самих команд пишете скрипты, которые выполняют строго заданные функции и указываете в `sudoers` эти скрипты вместо команд. Ну и у `sudo` ещё много всяких настроек, которые я не рассмотрел – но для основ этого будет достаточно.