The Unix Shell

Permissions
shell
shell  pwd, mkdir, cp, ...
shell  

```
pwd, mkdir, cp, ...
```
shell

pwd, mkdir, cp, ...

*, |

Permissions

Introduction
shell

(pwd, mkdir, cp, ...)

*, |

Who can see what?
shell

pwd, mkdir, cp, ...

*, |

Who can see what?

change
Who can see what?
change
run
Simplified version of Unix permissions
Simplified version of Unix permissions

Windows uses similar concepts...
Simplified version of Unix permissions
Windows uses similar concepts...
...but there is no exact translation between the two
Permissions

Introduction

user
Has unique *user name* and *user ID*
Has unique *user name* and *user ID*

User name is text: "imhotep", "larry", "vlad", ...
Has unique *user name* and *user ID*

User name is text: "imhotep", "larry", "vlad", ...

User ID is numeric (easier for computer to store)
Has unique *group name* and *group ID*
Has unique *group name* and *group ID*

User can belongs to zero or more groups
User can belong to zero or more groups.

List is usually stored in /etc/group
Permissions

Introduction

user | group | all
Permissions

Introduction

Everyone else

- user
- group
- all
Has user and group IDs
<table>
<thead>
<tr>
<th>user</th>
<th>group</th>
<th>all</th>
</tr>
</thead>
<tbody>
<tr>
<td>read</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>user</td>
<td>group</td>
</tr>
<tr>
<td>-------</td>
<td>------</td>
<td>-------</td>
</tr>
<tr>
<td>read</td>
<td></td>
<td></td>
</tr>
<tr>
<td>write</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>user</td>
<td>group</td>
</tr>
<tr>
<td>-------</td>
<td>------</td>
<td>-------</td>
</tr>
<tr>
<td>read</td>
<td></td>
<td></td>
</tr>
<tr>
<td>write</td>
<td></td>
<td></td>
</tr>
<tr>
<td>execute</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>user</td>
<td>group</td>
</tr>
<tr>
<td>--------</td>
<td>------</td>
<td>-------</td>
</tr>
<tr>
<td>read</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>write</td>
<td>✔</td>
<td>✗</td>
</tr>
<tr>
<td>execute</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>
File's owner can read and write it

<table>
<thead>
<tr>
<th></th>
<th>user</th>
<th>group</th>
<th>all</th>
</tr>
</thead>
<tbody>
<tr>
<td>read</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>write</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>execute</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>
File's owner can read and write it

Others in group can read

<table>
<thead>
<tr>
<th>perm</th>
<th>user</th>
<th>group</th>
<th>all</th>
</tr>
</thead>
<tbody>
<tr>
<td>read</td>
<td>✔</td>
<td>✔</td>
<td>✗</td>
</tr>
<tr>
<td>write</td>
<td>✔</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>execute</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>
File's can read and write it
Others in group can read

That's all

<table>
<thead>
<tr>
<th></th>
<th>user</th>
<th>group</th>
<th>all</th>
</tr>
</thead>
<tbody>
<tr>
<td>read</td>
<td>✔</td>
<td>✔</td>
<td>✗</td>
</tr>
<tr>
<td>write</td>
<td>✔</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>execute</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>
$ cd labs
$ ls
  safety.txt  setup  waiver.txt
$
```
$ cd labs
$ ls
safety.txt    setup      waiver.txt
$ ls -F
safety.txt    setup*     waiver.txt
$ 
```
```bash
$ cd labs
$ ls
safety.txt  setup  waiver.txt
$ ls -F
safety.txt  setup*  waiver.txt
$
```

means "executable"
```bash
$ cd labs
$ ls

safety.txt    setup     waiver.txt

$ ls -F

safety.txt    setup*    waiver.txt

$ ls -l

-rw-rw-r-- 1 vlad bio  1158  2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988  2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312  2010-07-11 08:23 waiver.txt

$
$ cd labs
$ ls
safety.txt    setup      waiver.txt
$ ls -F
safety.txt    setup*     waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio  1158  2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988  2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312  2010-07-11 08:23 waiver.txt
$
$ cd labs
$ ls
safety.txt setup waiver.txt

$ ls -F
safety.txt setup* waiver.txt

$ ls -l
-rw-rw-r-- 1 vlad bio 1158 2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988 2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio 2312 2010-07-11 08:23 waiver.txt
$
$ cd labs
$ ls
safety.txt setup waiver.txt
$ ls -F
safety.txt setup* waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio 1158 2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988 2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio 2312 2010-07-11 08:23 waiver.txt
$
$ cd labs
$ ls
safety.txt    setup      waiver.txt
$ ls -F
safety.txt    setup*     waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio 1158 2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988 2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio 2312 2010-07-11 08:23 waiver.txt
$
```bash
$ cd labs
$ ls
safety.txt    setup      waiver.txt
$ ls -F
safety.txt    setup*     waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio  1158  2010-07-11 08:22 safety.txt
-rw-r-xr-x 1 vlad bio 31988  2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312  2010-07-11 08:23 waiver.txt
$ 
user owner
```
$ cd labs
$ ls
safety.txt    setup      waiver.txt
$ ls -F
safety.txt    setup*     waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio  1158  2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio  31988  2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312  2010-07-11 08:23 waiver.txt
$

don't care (for now)
$ cd labs
$ ls
safety.txt    setup      waiver.txt
$ ls -F
safety.txt    setup*     waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio 1158 2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988 2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio 2312 2010-07-11 08:23 waiver.txt
$
$ cd labs
$ ls

safety.txt    setup      waiver.txt

$ ls -F

safety.txt    setup*     waiver.txt

$ ls -l

-rw-rw-r--  1  vlad  bio  1158  2010-07-11 08:22 safety.txt
-rwxr-xr-x  1  vlad  bio  31988  2010-07-23 20:04 setup
-rw-rw-r--  1  vlad  bio  2312  2010-07-11 08:23 waiver.txt

$
$ cd labs
$ ls
safety.txt  setup  waiver.txt
$ ls -F
safety.txt  setup*  waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio  1158  2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988  2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312  2010-07-11 08:23 waiver.txt
$ 

-rwxr-xr-x

file type
$ cd labs
$ ls
safety.txt    setup      waiver.txt
$ ls -F
safety.txt    setup*     waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio  1158  2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988  2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312  2010-07-11 08:23 waiver.txt
$
$ cd labs
$ ls

safety.txt    setup    waiver.txt

$ ls -F

safety.txt    setup*    waiver.txt

$ ls -l

-rw-rw-r-- 1 vlad bio  1158  2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988  2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312  2010-07-11 08:23 waiver.txt

$
$ cd labs

$ ls

safety.txt    setup      waiver.txt

$ ls -F

safety.txt    setup*     waiver.txt

$ ls -l

-rw-rw-r-- 1 vlad bio 1158 2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988 2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio 2312 2010-07-11 08:23 waiver.txt

$  

-rwxr-xr-x

↑

user owner permissions
$ cd labs
$ ls
safety.txt   setup      waiver.txt
$ ls -F
safety.txt   setup*     waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio 1158 2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988 2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio 2312 2010-07-11 08:23 waiver.txt
$ 

-rwxr-xr-x

↑

group owner permissions
$ cd labs
$ ls
safety.txt  setup  waiver.txt
$ ls -F
safety.txt  setup*  waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio 1158 2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988 2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio 2312 2010-07-11 08:23 waiver.txt
$

-rwxr-xr-x

everyone else's permissions
$ ls -a -l

```
drwxr-xr-x 1 vlad bio   0 2010-08-14 09:55 .
drwxr-xr-x 1 vlad bio  8192 2010-08-27 23:11 ..
-rw-rw-r-- 1 vlad bio  1158 2010-07-11 08:22 safety.txt
-rw-rw-r-- 1 vlad bio  31988 2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312 2010-07-11 08:23 waiver.txt
```

$
$ ls -a -l

-rw-rw-r-- 1 vlad bio 1158 2010-07-11 08:22 safety.txt
-rw-rw-r-- 1 vlad bio 2312 2010-07-11 08:23 waiver.txt
$
```
$ ls -a -l

-drwxr-xr-x 1 vlad bio  0 2010-08-14 09:55 .
-drwxr-xr-x 1 vlad bio  8192 2010-08-27 23:11 ..
-rw-rw-r-- 1 vlad bio  1158 2010-07-11 08:22 safety.txt
-rw-rw-r-- 1 vlad bio  31988 2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312 2010-07-11 08:23 waiver.txt
```

What does "execute" mean for directories?
What does "execute" mean for directories?

Gives the right to *traverse*

the directory
What does "execute" mean for directories?

Gives the right to *traverse* the directory.
What does "execute" mean for directories?

Gives the right to *traverse* the directory.

Diagram:
- Root directory
  - vlad
    - venus
    - notes
  - mars
    - notes
  - pluto
    - notes
What does "execute" mean for directories?

Gives the right to *traverse*

the directory

![Diagram of directory structure](image)
What does "execute" mean for directories?

Gives the right to *traverse* the directory
What does "execute" mean for directories?

Gives the right to *traverse*
the directory

```
$ ls venus venus/notes
```
What does "execute" mean for directories?

Gives the right to *traverse* the directory

```
$ ls venus venus/notes ✓
```

```
vlad

venus
  notes
  ...

mars
  notes
  ...

pluto
  notes
  ...
```
What does "execute" mean for directories?

Gives the right to *traverse* the directory

```
$ ls venus venus/notes  
```

```
vlad
  └── venus
      ├── notes
      │     └── ...
      └── mars
          └── notes
              └── ...
      └── pluto
          └── notes
              └── ...
```
What does "execute" mean for directories?

Gives the right to *traverse* the directory

```
$ ls venus venus/notes  ✔
$ ls mars mars/notes   ✔
```
What does "execute" mean for directories?

Gives the right to **traverse**

the directory

```
$ ls venus venus/notes ✔
$ ls mars mars/notes ✔
```
What does "execute" mean for directories?

Gives the right to *traverse* the directory

```
$ ls venus venus/notes ✓
$ ls mars mars/notes ✓
$ ls pluto ✗
```

**Permissions**

**Introduction**

<table>
<thead>
<tr>
<th>Directory</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>venus</td>
<td>notes</td>
</tr>
<tr>
<td>mars</td>
<td>notes</td>
</tr>
<tr>
<td>pluto</td>
<td>notes</td>
</tr>
</tbody>
</table>

```
...         ...
...         ...
...         ...
```
What does "execute" mean for directories?

Gives the right to traverse the directory

$ ls venus venus/notes ✓
$ ls mars mars/notes ✓
$ ls pluto ❌
$ ls pluto/notes

---

Permissions

Introduction
What does "execute" mean for directories?

Gives the right to *traverse* the directory

- $ ls venus venus/notes ✔
- $ ls mars mars/notes ✔
- $ ls pluto ✗
- $ ls pluto/notes ✔
Change permission with `chmod` (change mode)
Change permission with \texttt{chmod} (change mode)

\begin{verbatim}
$ ls -l final.grd
-rw-rw-rwx 1 vlad bio 4215 2010-08-29 22:30 final.grd
\end{verbatim}
Change permission with `chmod` *(change mode)*

$ ls -l final.grd

```
-rw-rw-rw- 1 vlad bio 4215 2010-08-29 22:30 final.grd
```

↑

Everyone can read it
Change permission with \texttt{chmod (change mode)}

\$ \texttt{ls -l final.grd}

\texttt{-rwxrwxrwx 1 vlad bio 4215 2010-08-29 22:30 final.grd}

\textcolor{red}{\uparrow}

Everyone can read it
Modify it
Change permission with `chmod` (change mode)

$ ls -l final.grd
-rwxrwxrwx 1 vlad bio 4215 2010-08-29 22:30 final.grd

↑
Everyone can read it
Modify it
Try to run it (which probably doesn't make sense)
Change permission with \texttt{chmod} (\texttt{change mode})

\begin{verbatim}
$ ls -l final.grd
-rwxrwxrwx 1 vlad bio 4215 2010-08-29 22:30 final.grd
$ chmod u=rw final.grd
$ 
\end{verbatim}
Change permission with chmod (change mode)

$ ls -l final.grd
-rw-rw-rwx 1 vlad bio 4215 2010-08-29 22:30 final.grd

$ chmod u=rw final.grd

User (u) has read-write (rw)
Change permission with `chmod` (change mode)

```
$ ls -l final.grd
-rwxrwxrwx 1 vlad bio  4215  2010-08-29 22:30 final.grd
$ chmod u=rw final.grd
$ ls -l final.grd
-rw-rwxrwx 1 vlad bio  4215  2010-08-30 08:19 final.grd
$ 
```
Change permission with `chmod` (*change mode*)

$ ls -l final.grd
-rwxrwxrwx 1 vlad bio  4215  2010-08-29 22:30 final.grd
$ chmod u=rw final.grd
$ ls -l final.grd
-rw-rw-rwx 1 vlad bio  4215  2010-08-30 08:19 final.grd
$ chmod g=r final.grd; ls -l final.grd
-rw-r--rwx 1 vlad bio  4215  2010-08-30 08:19 final.grd
$
Change permission with `chmod` *(change mode)*

```bash
$ ls -l final.grd
-rwxrwxrwx 1 vlad bio 4215 2010-08-29 22:30 final.grd
$ chmod u=rw final.grd
$ ls -l final.grd
-rw-rwxrwx 1 vlad bio 4215 2010-08-30 08:19 final.grd
$ chmod g=r final.grd; ls -l final.grd
-rw-r--rw- 1 vlad bio 4215 2010-08-30 08:19 final.grd
$ 

Use `;` to put multiple commands on a single line
Change permission with `chmod` (change mode)

$ ls -l final.grd
-rwxrwxrwx 1 vlad bio  4215  2010-08-29 22:30 final.grd

$ chmod u=rw final.grd
$ ls -l final.grd
-rw-rwxrwx 1 vlad bio  4215  2010-08-30 08:19 final.grd

$ chmod g=r final.grd; ls -l final.grd
-rw-r--rw- 1 vlad bio  4215  2010-08-30 08:19 final.grd

$ chmod a= final.grd; ls -l final.grd
-rw-r----- 1 vlad bio  4215  2010-08-30 08:20 final.grd
Change permission with chmod (change mode)

$ ls -l final.grd
-rwxrwxrwx 1 vlad bio  4215  2010-08-29 22:30 final.grd
$ chmod u=rw final.grd
$ ls -l final.grd
-rw-rwxrwx 1 vlad bio  4215  2010-08-30 08:19 final.grd
$ chmod g=r final.grd; ls -l final.grd
-rw-r--rwx 1 vlad bio  4215  2010-08-30 08:19 final.grd
$ chmod a= final.grd; ls -l final.grd
-rw-r---- 1 vlad bio  4215  2010-08-30 08:20 final.grd

No permissions at all
Again, things are different on Windows
Again, things are different on Windows

Permissions defined by Access Control Lists (ACLs)
Again, things are different on Windows

Permissions defined by Access Control Lists (ACLs)

A list of (who, what) pairs
Again, things are different on Windows
Permissions defined by Access Control Lists (ACLs)
A list of (who, what) pairs
More flexible...
Again, things are different on Windows
Permissions defined by Access Control Lists (ACLs)
A list of (who, what) pairs
More flexible...
...but more complex to administer and understand
Again, things are different on Windows
Permissions defined by Access Control Lists (ACLs)
A list of (who, what) pairs
More flexible...
...but more complex to administer and understand
Some flavors of Unix provide ACLs, but hardly anyone uses them
Create your own commands
Create your own commands

$ cat > smallest
Create your own commands

$ cat > smallest

No input file specified, so read from keyboard
Create your own commands

$ cat > smallest

Send output to a file called smallest
Create your own commands

```bash
$ cat > smallest
wc -l *.pdb | sort | head -1
```
Create your own commands

```bash
$ cat > smallest
wc -l *.pdb | sort | head -1
^D
$
```
Create your own commands

$ cat > smallest
wc -l *.pdb | sort | head -1
^D
$

Ctrl-D means "end of input" in Unix
Create your own commands

```
$ cat > smallest
wc -l *.pdb | sort | head -1
^D
$
```

Ctrl-D means "end of input" in Unix
Ctrl-Z does the same thing in Windows
Create your own commands

```
$ cat > smallest
wc -l *.pdb | sort | head -1
^D
$ chmod u+x smallest
$
```
Create your own commands

```
$ cat > smallest
wc -l *.pdb | sort | head -1
^D
$ chmod u+x smallest
$
```

Give the user owner permission to run this file
Create your own commands

$ cat > smallest

wc -l *.pdb | sort | head -1

^D

$ chmod u+x smallest

$ ./smallest
Create your own commands

$ cat > smallest
wc -l *.pdb | sort | head -1
^D
$ chmod u+x smallest
$ ./smallest

Put ./ at the front to be sure of running the smallest that it's this directory
Create your own commands

```
$ cat > smallest
wc -l *.pdb | sort | head -1
^D
$ chmod u+x smallest
$ ./smallest
   9 methane.pdb
$
```
Create your own commands

$ cat > smallest
wc -l *.pdb | sort | head -1
^D
$ chmod u+x smallest
$ ./smallest
  9  methane.pdb
$

Try doing *that* with a desktop full of GUIs