

Axcient

x360Recover - Linux basics

Last published at: April 21, 2021

Overview: Linux primer for x360Recover

Topics in this Linux primer:

- [Linux and Microsoft Windows](#)
- [Linux console](#)
- [Linux shell](#)
- [SSH - SCP - PuTTY](#)
- [Linux file structure](#)
- [Linux wildcards](#)
- [Linux devices](#)
- [Useful Linux commands](#)
- [Linux text editor](#)

[Download this primer as a PDF:](#)

x360Recover is built on the Ubuntu Linux distribution.

Although we've designed x360Recover to be fully configured and controlled via the web user interface, there are some situations where it is preferable or necessary to access the Linux console to perform troubleshooting or tweak configurations.

Many partners have little or no experience with Linux, and that's OK. ([Axcient Support](#) is always available to help.) However, for those partners that just need a little guidance, we've created this basic primer on Linux in general to assist with managing your x360Recover devices.

Linux and Microsoft Windows

Linux has both similarities to and differences from Microsoft Windows.

How is Linux similar to Microsoft Windows?

- Both Linux and Windows are operating systems that provide a platform for launching applications and managing the interaction with users.
- Both Linux and Windows have text and graphical interfaces that behave somewhat similarly.

How is Linux different from Microsoft Windows?

- In Linux, almost everything is case sensitive. This includes usernames, passwords, and file paths.
- With Windows, every disk drive is assigned a drive letter label, which acts as the root path for that device. (i.e. c:\) Under Linux, all devices are referenced from a single directory tree, referred to as the root of the filesystem, and referenced as '/'. Physical devices are mapped into subfolders of

the root directory. For example, disk 0 may be mapped as root (/) and disk 1 may be mapped as the user home folder (/home) etc. We will discuss more about Linux devices later in this primer.

- Unlike Windows, the graphical user interface is an optional component in Linux. Most server and appliance type installations of Linux forgo the graphical user interface. This saves space and reduces security exposure by reducing the total number of installed packages that must be maintained and patched on the system.
 - Unlike Windows, text mode in Linux is a complete, stand-alone user interface, including support for multi-user login and remote access.
 - In Windows, the master user account is 'Administrator'. In Linux, the master user is 'root'.
-

How to interact with the Linux console

x360Recover does not install the graphical user interface of Linux.

All console operations are performed in the Text User Interface (TUI).

There are two ways to interact with the Linux console:

1. Directly on the device using the hardware keyboard
2. Via a remote shell connection using SSH

x360Recover also provides a web-based SSH client on all devices to simplify remote management.

(From the web console of any device add /ssh/ to the end of the web URL to access the SSH shell)

Linux shell

Working with the Linux shell is like using DOS or Windows Command Prompt or PowerShell.

- All operations are performed from a (mostly) text-only interface, with limited graphical display.
- Unlike DOS, however, the Linux Shell **does** utilize a mouse. Linux shell applications that are mouse-aware can take advantage of easy menus. The mouse can then be used to highlight, perform copy/paste operations and complete other tasks.

To review some basic options available in the x360Recover Linux user shell, [read this article](#):

SSH

Secure SHell (SSH) is a utility for establishing secure, encrypted communications with a remote device.

In addition to providing secure remote text console access, SSH can also be used to establish secure communications tunnels for other applications, and secure file copy operations (using SCP). There are many different SSH capable utilities available. One of the most popular for use on Windows is PuTTY, a free utility that supports both Telnet and SSH. (Telnet is a text console utility similar to SSH but without any security or encryption.) PuTTY is available from the **Tools** menu of *ScreenConnect* when connected to a guest partner's system.

SCP

Related to SSH, SCP is a Secure CoPy utility that manipulates files over an SSH tunnel connection to the target system. A common, free Windows version of SCP is WinSCP.

PuTTY

Get putty here: <https://the.earth.li/~sgtatham/putty/latest/x86/putty.exe>

- Run PuTTY and enter the IP address or FQDN of the device you wish to connect to.
- On the first access for a new device, PuTTY will exchange security keys with the device. Click **Yes** to save the new key. If the security keys ever change, you will receive a warning that the device may have been compromised. (This warning is normal if you have just reinstalled a new version of x360Recover on the same device or connect to another device with the same IP address.)
- Once connected, you will receive a login prompt just like at the physical device console.
- Enter the username root and hit enter. Then enter the password and hit enter again.

Linux file structure: Folders

As discussed above, all file paths in Linux proceed from root (/) downward.

There are a number of important directories in the root folder that have specific purposes.

Here is a list of the most relevant folders and their purpose:

/boot

This folder contains the operating system kernel as well as the system boot loader files.

/dev

This folder contains special pointer files that represent physical devices, such as hard drives. (See the [Linux devices section](#) below for more information.)

/etc

The etc folder is where all system configuration files are stored, as well as service management scripts.

/home

The home folder is the default location for user home directories.

/media and /mnt

These folders are used for mounting removable media and other devices.

Generally, automatically-removable media (like USB drives and CDRoms) will be automatically discovered on connection and will be attached in a subfolder of /media. The name is based on the volume label of the

device. `/mnt` is a legacy folder often used for manually mounted filesystems from locally attached storage devices or remote network shares.

/proc

This folder contains dynamically generated files that represent the state of the running system. In some ways, this folder is like the Windows Registry, in that you can read or write values into the files contained here in order to get or set certain system configuration states.

/root

This is the home folder for the root user. Linux architecture requires that the root home folder be in the top level Root (`/`) filesystem folder. It cannot be located on another device and mounted here.

/run

This folder contains lock files generated by running processes. These are commonly used to identify when a specific application or service is already running.

/sys

This folder is similar to `/proc`. `/sys` is a product of more modern Linux kernel architecture and is a replacement for much (but not all) of what is in `/proc`.

/usr

This folder tree is where user applications and utilities reside.

/var

Short for **variable**, the `/var` folder is traditionally where all user and application data are stored.

Linux tab completion feature

One of the most useful features of the Linux shell is tab completion.

- When typing a path or command in the shell, press the `<TAB>` key and your entry will be automatically completed.
- If there are multiple possibilities, it will complete to the nearest unique value and wait for you to enter more text.
- Type a few more characters and press tab again, or press `<TAB>` twice to enumerate a list of all matching possibilities.

Tab completion is also context sensitive.

- For example, if you are typing a command, it will search all folders in the `%path%`, but if you are entering a fully qualified path, it will only search the indicated directory.

Even better, it understands most Linux commands and the context search is command specific.

- For example, if you enter `python <text>` and press `<TAB>`, it will complete the line by searching for Python scripts only (i.e. `*.py`) that are located in the current directory.

Linux wildcards

Linux accepts a powerful set of wildcard options to apply with most commands. This simplifies targeting multiple objects.

Note: Multiple wildcards of the same or different types can be combined within a match pattern to create highly-complex search criteria.

Here is a list of the different classes of wildcard objects:

~ The 'tilde' character when used as a path represents the home folder for the currently logged in user.

? Use a ? to substitute any single character.

Example: `rm file.???` would remove all files whose names start with 'file.' and end with any three characters

* Use the asterisk wildcard to substitute any number of characters, (including none) matching the pattern.

Examples: `ls *.txt` or `ls *something*.txt`

[] Use square brackets to denote a range or set of options.

Example: `m[a,u,o]m` could be man, mum, or mom and `a[0-3]` could be a0, a1, a2, or a3, etc.

{ } Use curly brackets to denote multiple items or wildcard patterns.

Example: `cp {*.doc,*.pdf} ~` would copy anything matching `*.doc` or `*.pdf` to your home folder. (Note the special character `~` is a shortcut for the current user's home folder path)

[!] This is similar to the square brackets above, but the ! indicates a logical NOT, reversing the pattern match to include objects that do NOT match the bracket entry.

Example: `myfile![9]` would match any 'myfileX' where X is not '9'

Note that wildcards are stackable and recursive, so you can build complex match strings like `{*some[a-z]hing.do?,*some[a-z]hing.pd?}`

Pipe command

It is often convenient to send the output of one command to another command for further processing.

This is accomplished using the **pipe command**, represented with [the '|' character](#).

Example: Use **grep** to search the output of a command for matching context.

ls | grep <text> would list the contents of the current directory and send the output to **grep**.

grep will then search for <text> within each line and output to the console only lines containing <text>.

Linux devices

As discussed earlier, the folder **/dev** contains special pointers to access physical and virtual devices detected within the running system.

- Each of these objects is not a standard file, but rather a special link, dynamically generated by the kernel or a running service that acts as a handle to access and identify the device.
- All Linux devices are treated as block devices that can be read and/or written to using their **/dev** object as a handle.
- The format of the files located in **/dev** have specific meanings.

Depending on the device, the first two or three letters indicates the device class. The remaining characters in the device name indicates a specific instance of the device.

Examples of devices:

- Devices that start with **hd** are hard drives.

Specifically, **/hdX** indicates a hard drive connected to a legacy IDE style disk controller. The third letter of the name will be a letter, a-z, (followed by aa-zz if more than 26 devices exist), indicating a specific hard disk instance. (hda is the first disk, hdb is the second, and so on.)

For each hard disk detected, there will be an entry in **dev** (hda, hdb, etc.)

For each hard disk discovered, if filesystem partitions exist on the disk, there will be an entry **hdXn** where **X** is the disk number and **n** is the partition number (such as **hda1**).

Linux device enumeration makes no distinction between MBR and GPT formatted disks. Use **fdisk** or **gdisk** to query partition table information.

- Devices that start with **sd** are hard drives attached to a SCSI interface rather than IDE.

SATA, SAS, and USB attached devices are also bundled into the `sd` device class. Nomenclature for the drive and partition instances is identical to `hd` devices.

- A CDROM drive on an IDE bus is `/dev/cdrX`

If located on SCSI, SATA, SAS, or USB, it's `/dev/srX`, where X is a number starting with 0.

- Modern device mapper disks, which can include hardware or software RAID, CRYPTO, LVM or other types of disk sets are listed as `/dev/dm-X` where X is a number starting at 0.

- Legacy software RAID disks are listed as `/dev/mdX`, where X is a number starting with 0.

- Some third-party RAID controller drivers, notably HP Server RAID controllers, have `/dev` naming conventions specific to the controller family.

For example, HP uses the device **name** `/dev/cciss` to denote the controller root class, and specifies sub-devices as `/dev/cciss/cXdYpZ` where cX is controller X, dY is disk Y on the controller, and pZ is the partition on the disk.

Useful Linux commands

Important notes:

- All Linux shell commands are case sensitive
- (Almost) all Linux shell commands are lower case
- Almost all Linux commands accept wildcards in their parameters
- Most Linux commands allow optional parameter groups. (Enter `rm -r -f` or just `rm -rf`)
- Most Linux commands offer a help option. Enter **command --help**

These common Linux commands are helpful to know:

`cd <path>`

Change Directory. Use this command to change the focus of the shell to another directory. i.e. `cd /tank/admin`. Entering `cd` without a path option will return you to your home directory (i.e. `/root` or `/home/replibit` depending on which user you are logged in on.)

apt-get

Part of the Debian Package Manager. Use **apt-get install <pkg>**. Apt will locate the indicated package and any dependencies using the configured network package repositories, download all required packages, and install them.

cp <src> <dst>

Copy. Copy file(s) to a different directory. Use **-r** to recursively copy subfolders and files.

df

Drive Free. Displays disk total size and disk in-use for all mounted devices. Use **df -h** for human readable output.

dpkg <option> <package>

Debian Package Manager. Installs and uninstalls individual application packages. Use **dpkg -i <pkg>** to install. Use **dpkg -x <pkg>** to remove.

du

Disk Used. Often used with **-h** option to provide 'human readable' output. Displays all files and folders recursively in the current folder and lists their size on disk.

grep <pattern>

grep is a streaming search utility. It only prints out lines of text it receives as input which contain a match for the provided pattern. Most often, *grep* is used with the piped output of another command to search for matching content. Unlike most Linux commands, *grep* does not accept any wildcards in the <pattern>.

grep <pattern> implies *grep *<pattern>**. This means it will match the pattern anywhere in the data being scanned.

Another powerful feature of *grep* is the ability to scan within file CONTENT. To search inside the content of all files and folders in a given path use:

```
grep -rnw <path> -e <pattern>
```

where <path> is the starting location to be traversed recursively downwards and <pattern> is what to search for inside each file. *Grep* will return file names containing the matching pattern in their content.

head/tail <file>

Head and Tail display the first or last few lines of a file, respectively. By default, 10 lines of the file are displayed, but this can be altered using the parameter **-l** Example: **tail -l 20 /var/log/replibit/replibit.log** will display the last 20 lines of the replibit.log file.

ls/l

List. Like *dir* in DOS, this command lists the contents of the current folder. It is equivalent to **ls -l** and outputs the listing in long form. Other useful parameters: **-a** shows all files, including hidden. **-h** displays information in human readable form (xxGB instead of xxxxxx Bytes, etc.) **-r** lists subfolders recursively.

mkdir <path>

Make Directory. This command will create a new folder in the current directory.

mount/umount <path>

Mount and unmount filesystems and devices. Run mount without options to list all mounted filesystems. Use umount -l to unmount a device and force kill any outstanding filesystem locks.

mv <src> <dst>

Move. Identical to cp but deletes the original files in <src> after copying them to the <dst> path

rm <options> <target>

Remove. This command deletes files and folders. If entered with just a target, it will delete matching items in the current directory. Common options are -r to recursively delete objects in the current directory and all subfolders, and -f to force deletion of protected objects (like directories).

ssh

Connect remotely to the shell of another Linux device, such as an appliance or vault.

tar

TapeARchive. This command is similar to WinZIP or other archiving utilities. It bundles together files and/or folders into a single file, and optionally compresses the archive to save space. Originally developed as a simple utility to stream backups to a tape drive, it is by far the most commonly used Linux method for bundling multiple files and folders together for easy transport to another location or system. Use -c to create, and -x to extract. -z indicated that gzip compression should be applied. -f indicates the target archive is a file (default is the console) Example: tar -xzf <file.tgz> would extract file.tgz. Common tar file extensions: *.tgz, *.tar, *.tar.gz. Note that the name is only suggestive of the real format. For example, *file.tar* semantically indicates that the file is not compressed, but it is possible that compression does exist.

vi/vim

Standard Linux text editor. See Using Vi/Vim below.

wget <URL>

Command-line HTTP file retrieve. Saves a target URL file in the current directory. This is a useful way to retrieve a file available for download from a web server.

Example: wget https://my.domain.com/path/file.tgz

Linux text editor: What is vi (or vim)?

The most fundamental tool in Linux is the lowly text editor, and every distribution of Linux requires that a text edit be available for making configuration changes.

- **vi** is the default Linux text editor.

The Linux gods have declared that every Linux distribution, big or small, must have vi installed. Anyone working with Linux should learn at least the basic features of vi. One of the true universal givens of the Linux world: "There shall be vi."

While vi is not pretty, it **is** simple. There are only a few common key-strokes to remember to make basic text file changes with vi.

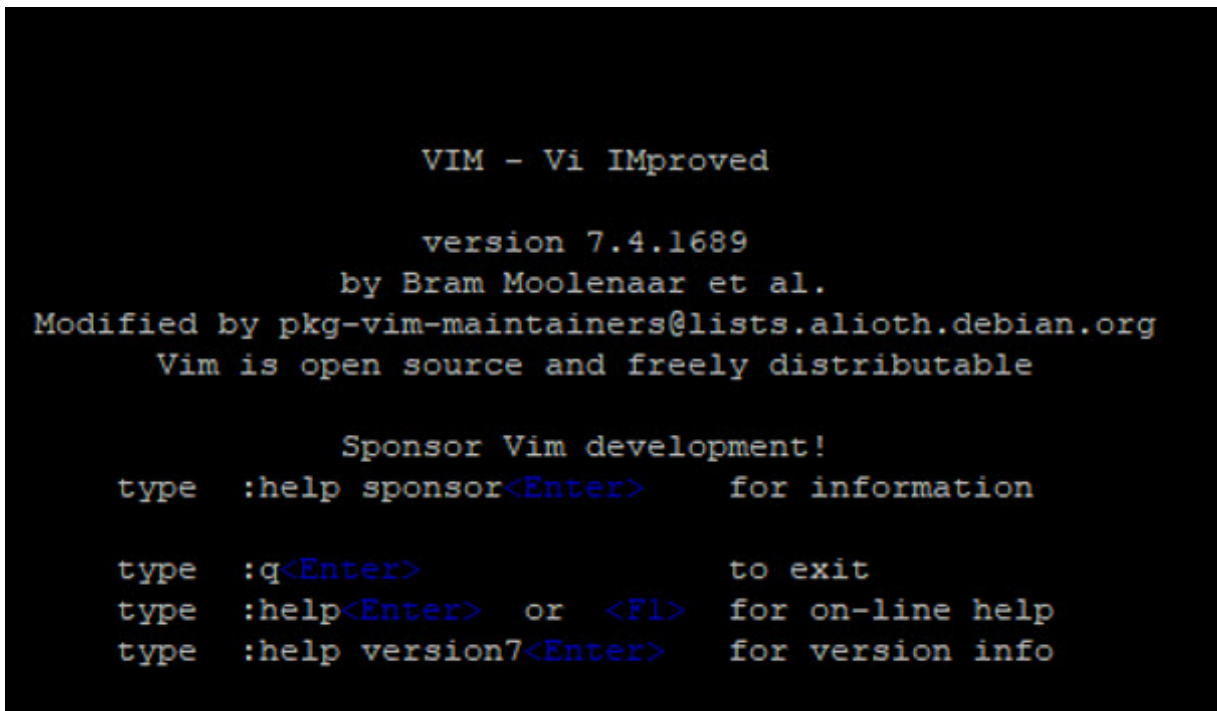
But first, let's talk about the versions of vi.

In modern Linux, when most people talk about vi, what they really mean is vim, or **Vi Improved**.

- The original vi (from the dawn of Linux) has earned the well-deserved nick-name "ViLE."
- How to tell if you are using the original vi: If the cursor keys (or the BACKSPACE key) insert escape codes into the document (rather than moving the cursor), you have the original vi. We suggest you then exit and try specifying *vim* instead.
- Note: If vim is not available, install it with *apt-get install vim*. Most modern Linux distributions install *vim* by default and create a default alias for *vi* that transforms it to *vim*. Be aware that, Debian, unfortunately, installs the original *vi* by default, so use *vim* when editing.

Use vim to edit

To open a document: Type *vi <file>* then **Enter**. This will load the document in read-only mode. Use cursor keys or page-up/down to navigate the file.



```
VIM - Vi IMproved

version 7.4.1689
by Bram Moolenaar et al.
Modified by pkg-vim-maintainers@lists.alioth.debian.org
Vim is open source and freely distributable

Sponsor Vim development!
type  :help sponsor<Enter>    for information

type  :q<Enter>                to exit
type  :help<Enter>  or  <F1>   for on-line help
type  :help version7<Enter>   for version info
```

To make changes to a document: Enter insert mode.

1. Press the letter **I** and you will see **INSERT** appear in the bottom left corner, indicating that you are in insert mode.

2. Press the letter **I** a second time to switch to **REPLACE** mode, which overwrites characters as you type rather than inserting new characters.
3. Press **ESC** to exit **insert** or **replace** mode, after you are done editing.

To search for text within the document (when *not* in **insert** or **replace** mode): Type a forward slash '/' followed by the text you are searching for, then **Enter**.

Use vim command mode

Most of the remaining *vi* commands are accessed via *command* mode.

All vim command mode options start with a colon : followed by one or more letters:

Note: vim command mode options require that you **not be in insert or replace mode**

- To **save** your changes, type **:w** then **Enter**
- To **exit** *vi* when you are done editing, type **:q**
- Combine the save and quit options by entering **:wq**
- To exit without saving changes, type **:q!**
- For help with additional commands, type **:help** then **Enter**. Scroll down with the cursor keys to read the help section. Then, type **:q** to exit help mode

That's it for *vi* basics! Though there are many other options and commands available, these few options are all that is necessary to perform and save simple text file editing.

Use nano to edit

For a more user-friendly text editing experience, x360Recover includes the **nano** editor.

- Use the command **nano <file>** to open a file in nano.

```
root@STX-Nano: ~
GNU nano 2.5.3 New Buffer

^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
```

Nano is a simple text editor, with user-friendly menu commands displayed along the bottom edge of the screen.

Unlike **vi**, the editor in **nano** defaults to read/write mode and behaves more like Windows Notepad or other simple text editors you might be familiar with.

- If editing via ssh terminal, make the shell window wider to reveal more menu commands, or press **<CTRL>-g** to get help on all available commands.
- Menu options are selected by pressing **<CTRL>** (symbolized by '^' on screen) and a command character. For example, press **<CTRL>-o** to save your changes, or **<CTRL>-x** to exit the program.
- If unsaved changes exist when you are exiting, you will be prompted whether to save them or not. Select **Y** to save or **N** to exit without saving

SUPPORT | 720-204-4500 | 800-352-0248

- Contact Axcient Support at <https://partner.axcient.com/login> or call 800-352-0248
- Free certification courses are available in the [Axcient x360Portal](#) under [Training](#)
- To learn more about any of our Axcient products, sign up for a [free one-on-one training](#)
- Subscribe to the [Axcient Status page](#) for a list of status updates and scheduled maintenance