The C-ALE (Cloud & Container Apprentice Linux Engineer) is a series of seminars held at existing conferences covering topics which are fundamental to a Linux professional in the Linux Cloud and Container field of computing.

This seminar will spend equal time on lecture and hands on labs at the end of each seminar which allow you to practice the material you’ve learned.

This material makes the assumption that you have minimal experience with using Linux in general, and a basic understanding of general industry terms. The assumption is also made that you have access to your own computers upon which to practice this material.

More information can be found at [https://cm.ossessentials.org/](https://cm.ossessentials.org/)

This material is licensed under [CC-BY SA4](https://creativecommons.org/licenses/by-sa/4.0/)
Contents

1 Preliminaries 1
  1.1 The Speaker ........................................... 2
  1.2 The Command Line ..................................... 3
  1.3 sudo ................................................... 4
  1.4 Linux Distributions and Desktops .................. 5

2 Networking Basics 9
  2.1 Understanding The Packet ............................. 10
  2.2 Interface Settings ................................... 15
  2.3 Monitoring Network Traffic ......................... 22
  2.4 Using Name Services .................................. 28
  2.5 Labs .................................................... 32
Chapter 1

Preliminaries

1.1 The Speaker .................................................. 2
1.2 The Command Line .......................................... 3
1.3 sudo ................................................................. 4
1.4 Linux Distributions and Desktops ......................... 5
1.1 The Speaker

- Course developer and trainer for The Linux Foundation
  - LFD459/LFD259
  - LFS458/LFS258
  - LFS452/LFS252
  - LFS462
  - LFS243
- Contact:
  - Email: Tim@Serewicz.com
  - Twitter: @TimSerewicz
1.2. The Command Line

Needed Tools

- We need to use some tools before we learn about them!
- Otherwise it will be difficult to accomplish even simple tasks
- We will introduce some subjects briefly here, without much explanation. Topics to research!
1.3 sudo

**sudo and root Privilege**

- Some tasks require **root** (administrator) privilege, including:
  - Changes to system configuration that affect all users
  - Installing new software for all users
  - Accessing certain hardware, such as when **mounting** hard disk partitions
- Can become root user through use of **su** command, but this requires knowing the root password
- Better to use **sudo** because:
  - Requires only normal user password
  - Privileges can be restricted to only those strictly necessary
  - Better logging on the system of such activities
- Will do a laboratory exercise to ensure your system is properly set up to use **sudo**
1.4 Linux Distributions and Desktops

Linux Distributions

- There are many available Linux distributions (See http://lwn.net/Distributions)
- This course is distribution-flexible
- Three main families we work with:
  - Red Hat Enterprise Linux 7 (RHEL 7) (And close cousins CentOS, Fedora, Scientific Linux)
  - Ubuntu
    A derivative of Debian (Close cousin Mint Linux)
  - openSUSE
    Closely related to SLES (SUSE Linux Enterprise Server or just SUSE)
- The exact distribution version is not critical; latest release and previous one or two releases should be fine.
Keeping up to Date

- The entire **Linux** ecosystem is constantly changing
- **Linux** is not a mono-culture, but is very diverse
- Each distribution has its own strategy for updating, upgrading and issuing new releases
- Not everything will work exactly right on all distribution versions at all times
- To make sure everything worked perfectly, on all distributions and versions, at all times, we would have to freeze content, which would rapidly go stale
- It is better to stay current than freeze content.
- Figuring out these changes and how to adapt is part of the **Linux** experience!
There are two major Desktop environments widely available in Linux: GNOME and KDE.

Most Linux distributions offer a choice between the two.

However, most have a preferred default, which is usually GNOME.

Each desktop manager also has different versions; e.g., both GNOME 2 and GNOME 3 are widely used.

This means we can not give precise instructions that will work in every GUI, but it is usually easy to figure out.

Other desktops are also used; e.g., XFCE which is far less resource-intensive but also very comfortable and robust.
Chapter 2

Networking Basics

2.1 Understanding The Packet .................................................. 10
2.2 Interface Settings ............................................................. 15
2.3 Monitoring Network Traffic ................................................. 22
2.4 Using Name Services ......................................................... 28
2.5 Labs ................................................................................. 32
## 2.1 Understanding The Packet

### What Are We Sending

- Payload
- Encapsulation
- Header Sections
- TCP/IP and OSI Models
Encapsulation

- Variable size payload
- Adding parameters for wide range of recipients
- Unencapsulated by recipient network stack
- Flexible and open
Figure 2.1: Header Sections
Figure 2.2: **Network Models Sections**
IPv4 and IPv6 Addresses

- `ip addr show`

  1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000  
     link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00  
     inet 127.0.0.1/8 scope host lo  
       valid_lft forever preferred_lft forever  
     inet6 ::1/128 scope host  
       valid_lft forever preferred_lft forever  
  2: wlp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP  
     group default qlen 1000  
     link/ether 94:65:9c:7b:cd:a7 brd ff:ff:ff:ff:ff:ff  
     inet 192.168.200.175/24 brd 192.168.200.255 scope global dynamic  
       noprefixroute wlp1s0  
       valid_lft 82501sec preferred_lft 82501sec  
     inet6 fe80::54df:86ab:b128:11ef/64 scope link noprefixroute  
       valid_lft forever preferred_lft forever
2.2 Interface Settings

Configure the Interface

- `ip` command
- Edit flat files
- Configuration tools
CHAPTER 2. NETWORKING BASICS

The ip Command

$ ip addr help
Usage: ip address {add|change|replace} IFADDR dev IFNAME [ LIFETIME ]
        [ CONFFLAG-LIST ]
        ip address del IFADDR dev IFNAME [mngtmpaddr]
ip address {save|flush} [ dev IFNAME ] [ scope SCOPE-ID ]
        [ to PREFIX ] [ FLAG-LIST ] [ label LABEL ] [up]
ip address [ show [ dev IFNAME ] [ scope SCOPE-ID ] [ master DEVICE ]
        [ type TYPE ] [ to PREFIX ] [ FLAG-LIST ]
        [ label LABEL ] [up] [ vrf NAME ] ]
ip address {showdump|restore}
IFADDR := PREFIX | ADDR peer PREFIX
        [ broadcast ADDR ] [ anycast ADDR ]
        [ label IFNAME ] [ scope SCOPE-ID ]
SCOPE-ID := [ host | link | global | NUMBER ]
FLAG-LIST := [ FLAG-LIST ] FLAG
FLAG := [ permanent | dynamic | secondary | primary |
[-]tentative | [-]deprecated | [-]dadfailed | temporary |
CONFFLAG-LIST ]
CONFFLAG-LIST := [ CONFFLAG-LIST ] CONFFLAG
CONFFLAG := [ home | nodad | mngtmpaddr | noprefixroute | autojoin ]
LIFETIME := [ valid_lft LFT ] [ preferred_lft LFT ]
LFT := forever | SECONDS
TYPE := { vlan | veth | vcan | vxcan | dummy | ifb | macvlan | macvtap |
bridge | bond | ipoib | ip6tnl | ipip | sit | vxlan | lowpan |
gre | gretap | erspan | ip6gre | ip6gretap | ip6erspan | vti |
nlmon | can | bond_slave | ipvlan | geneve | bridge_slave |
hsr | macsec
Persistant configuration - Debian/Ubuntu

- `/etc/network/interfaces`
- `/etc/network/interfaces.d/*`

```plaintext
auto ens3
iface ens3 inet static
    address 192.168.4.12
    netmask 255.255.255.0
    gateway 192.168.4.1
    dns-nameservers 8.8.8.8
```
2.2. INTERFACE SETTINGS

- `/etc/sysconfig/network-scripts/ifcfg-eth0`

  ```
  DEVICE=ens4
  BOOTPROTO=none
  ONBOOT=yes
  PREFIX=24
  IPADDR=192.168.2.4
  ```
Network Manager

- nm-connection-editor
- nmtui
- nmcli

$ nmcli connection show WorkNet
connection.id: BWOceansidePalms
connection.uuid: 91733f17-40c3-4c91-b030-cde380715e54
connection.stable-id: --
connection.type: 802-11-wireless
...

$ nmcli con mod WorkNet +ipv4.dns 4.2.2.2
2.2. INTERFACE SETTINGS

systemd-networkd

- Attempt to standardize networking tools
- Lighter weight than NetworkManager
- Uses text files

$ vim /etc/systemd/network/30-wireless.network

[Match]
Name=wlpl0s0

[Network]
Address=192.168.8.3/24
Gateway=192.168.8.1
2.3 Monitoring Network Traffic

Where Did It Go?

- ping
- ss
- tracepath
- tcpdump
- Wireshark
2.3. MONITORING NETWORK TRAFFIC

The ss Command

- Dump socket statistics
- Wide range of options and output

$ ss -o state established
$ ss -s
$ ss -a
• Replacement for **traceroute**
• Doesn’t require super-user
• UDP traffic to avoid ICMP issues

```
$ tracepath linux.com
 1?: [LOCALHOST] pmtu 1500
 1: gatewaylogin.info 3.294ms
 1: gatewaylogin.info 3.068ms
  ... 
```
2.3. MONITORING NETWORK TRAFFIC

- Capture all traffic on an interface
- Requires super-user
- Many options and filters

```sh
$ sudo tcpdump -i wlp1s0 -A
```

```
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wlp1s0, link-type EN10MB (Ethernet), capture size 262144
01:35:13.439644 ARP, Request who-has 10.71.5.78 tell gatewaylogin.info ..........E.).
G........
G.N..................
01:35:13.442330 IP Txs-Dell.60252 > dns.google.domain: 10558+ PTR?
78.5.71.10.in-addr.arpa. (41)
```
CHAPTER 2. NETWORKING BASICS

ping

- Uses ICMP echo response
- Reports round-trip time
- Often blocked by firewalls

$ ping -c3 linux.com
PING linux.com (151.101.1.5) 56(84) bytes of data.
64 bytes from 151.101.1.5 (151.101.1.5): icmp_seq=1 ttl=252 time=14.5 ms
64 bytes from 151.101.1.5 (151.101.1.5): icmp_seq=2 ttl=252 time=19.2 ms
64 bytes from 151.101.1.5 (151.101.1.5): icmp_seq=3 ttl=252 time=18.0 ms

--- linux.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 14.541/17.300/19.267/2.015 ms
2.3. MONITORING NETWORK TRAFFIC

- Requires super-user to capture all traffic
- Lots of filters
- Dig deep into every packet

Figure 2.3: Capture Traffic with Wireshark
2.4 Using Name Services

---

Who Remembers Numbers

- The `/etc/hosts` file
- Domain Name Service
- Dynamic Host Configuration Protocol
2.4. USING NAME SERVICES

/etc/hosts

- Local text file
- IP FQDN alias

```
$ cat /etc/hosts
127.0.0.1 localhost
127.0.1.1 sys43.example.com. sys43 loghost
```
Domain Name Service

- Distributed name control
- **BIND** most popular software
- References `/etc/resolv.conf` for nameserver
- Check with `nslookup`, `dig`, and `host`
Dynamic Host Configuration Protocol

- Asks for network configuration from others
- Access leased for period of time
- Use `dhclient` request to ask again
  
  ```bash
  $ dhclient
  ```
2.5 Labs

Exercise 2.1: Networking Basics

Adding A New IP Address

We will begin by documenting our network configuration. There are several tools your distribution could use

1. Open a terminal or a PuTTY session. Use the `ip` command to view your interfaces and addresses. Save it to a file for later reference.

   $ ip a
   1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
     link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
     inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
     inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
   2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc mq state UP group default qlen 1000
     link/ether 06:04:be:c0:bf:7c brd ff:ff:ff:ff:ff:ff
     inet 172.31.21.106/20 brd 172.31.31.255 scope global dynamic eth0
       valid_lft 2225sec preferred_lft 2225sec
     inet6 fe80::404:beff:fec0:bf7c/64 scope link
       valid_lft forever preferred_lft forever

   $ ip a > ~/before.txt

2. Determine if `systemd-networkd` is running. As there are many possible tools, it may or may not be running. Following shows an example of it not running. If your system is using the daemon and it is active, find your config files, probably under `/etc/systemd/network/`. Review the contents.
2.5. LABS

```
$ systemctl status systemd-networkd
systemd-networkd.service - Network Service
   Loaded: loaded (/lib/systemd/system/systemd-networkd.service; disabled; vendor)
   Active: inactive (dead)
     Docs: man:systemd-networkd.service(8)
```

3. Determine if Network Manager is configuring your network. Again, your system may be different. Once you find a profile in use run the command again to view the details.

```
$ nmcli connection show
NAME     UUID                    TYPE     DE
WorkNet  235d15bb-8d60-4889-86a2-c5787c46dea8 wifi  wl
...

$ nmcli con show WorkNet
connection.id:          WorkNet
connection.uuid:        235d15bb-8d60-4889-86a2-c5787c46dea8
connection.stable-id:   --
connection.type:        802-11-wireless
connection.interface-name: --
connection.autoconnect: yes
connection.autoconnect-priority: 0
...
```

4. View flat file configuration. Your file name may vary depending on your distribution. View the file contents as you find them. Note the prompts in the commands indicate which distribution you may find the file.

```
ubuntu$ ls -l /etc/network/interfaces

ubuntu$ ls -l /etc/network/interfaces.d/

redhat$ ls -l /etc/sysconfig/network-scripts/
```
5. Use the `ip` command to add a new IP Address to your primary interface. Your interface name may be different. Typically the interface listed after `lo` in the `ip` output is your primary. Add the 10.1.2.3 address. We will need to use the `sudo` command.

```bash
$ sudo ip addr add 10.1.2.3 dev wlp1s0
```

6. Use `ping` to test connection to the new address.

```bash
$ ping -c3 10.1.2.3
PING 10.1.2.3 (10.1.2.3) 56(84) bytes of data.
64 bytes from 10.1.2.3: icmp_seq=1 ttl=64 time=0.049 ms
64 bytes from 10.1.2.3: icmp_seq=2 ttl=64 time=0.155 ms
64 bytes from 10.1.2.3: icmp_seq=3 ttl=64 time=0.116 ms

--- 10.1.2.3 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2037ms
rtt min/avg/max/mdev = 0.049/0.106/0.155/0.045 ms
```

7. View the `ip` details of the interface. Verify the new IP address has been added. Note it is not an alias, but an equal IP as the previous.

```bash
$ sudo ip addr show wlp1s0
2: wlp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 900 qdisc mq state UP
group default qlen 1000
 link/ether 94:65:9c:7b:cd:a7 brd ff:ff:ff:ff:ff:ff
 inet 172.30.9.192/21 brd 172.30.15.255 scope global dynamic
        noprefixroute wlp1s0
        valid_lft 18306sec preferred_lft 18306sec
 inet 10.1.2.3/32 scope global wlp1s0
        valid_lft forever preferred_lft forever
```

8. View the route information with `ip route`. Note your information will not match the exact output as your network is different. Look for a route for your new IP adress. It may not be there.

```bash
$ ip route
default via 172.30.8.1 dev wlp1s0 proto dhcp metric 600
```
9. Add a new route to 192.0.2.1 via 10.1.2.3 and verify it it is shown in the `ip route` output. Your interface may be different. View the new entry to the route table.

```
$ sudo ip route add 192.0.2.1 via 10.1.2.3 dev wlp1s0
```

```
$ sudo ip route
default via 172.30.8.1 dev wlp1s0 proto dhcp metric 600
192.0.2.1 via 10.1.2.3 dev wlp1s0
```

10. Use the `ifconfig -a` command to view your network settings. Do you see the new, working IP Address?

```
$ sudo ifconfig -a

wlp1s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 900
    inet 172.30.9.192  netmask 255.255.248.0  broadcast 172.30.15.255
    ether 94:65:9c:7b:cd:a7  txqueuelen 1000  (Ethernet)
    RX packets 217530  bytes 154308545 (154.3 MB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 87257  bytes 22436095 (22.4 MB)
    TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0
```

(So, no. It’s not there.)

11. Remove the IP address. After it has been removed, check to see if the routes have been removed as well. If not, remove it.
$ sudo ip addr del 10.1.2.3/32 dev wlp1s0

$sudo ip route
default via 172.30.8.1 dev wlp1s0 proto dhcp metric 600
10.0.3.0/24 dev lxcbr0 proto kernel scope link src 10.0.3.1 linkdown
10.1.2.3 dev wlp1s0 proto kernel scope link src 10.1.2.3 metric 600
169.254.0.0/16 dev virbr1 scope link metric 1000 linkdown
172.17.0.0/16 dev docker0 proto kernel scope link src 172.17.0.1 linkdown
172.30.8.0/21 dev wlp1s0 proto kernel scope link src 172.30.9.192 metric 600
192.0.2.1 via 10.1.2.3 dev wlp1s0
192.168.99.0/24 dev virbr1 proto kernel scope link src 192.168.99.1 linkdown
192.168.123.0/24 dev virbr0 proto kernel scope link src 192.168.123.1 linkdown

$sudo ip route del 10.1.2.3 dev wlp1s0

Changing Parameters

In this section we will change the MTU and view the differences in performance. You can test between two VMs or instances, or between yourself and another in class. For ease of use we will test against ourselves. Be aware that other network traffic can skew results. You may want to test multiple times. Also, some wifi drivers have a lower limit allowed, cloud providers like AWS and GCE set their MTU much higher automatically. Your interface particulars may be different.

1. Verify the MTU of your primary interface is 1500.

   $ sudo ip addr show wlp1s0
   2: wlp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq
      state UP group default qlen 1000
      link/ether 94:65:9c:7b:cd:a7 brd ff:ff:ff:ff:ff:ff
      inet 172.30.9.192/21 brd 172.30.15.255 scope global dynamic
      noprefixroute wlp1s0
valid_lft 40986sec preferred_lft 40986sec

2. Install the Flexible IO, or fio, package. We will use this command to generate a known amount of traffic. Ubuntu users can use the apt-get command, and Red Hat users can use the yum or dnf command.

```
ubuntu$ sudo apt-get update ; sudo apt-get -y install fio
redhat$ sudo yum -y install fio
```

3. Use fio to test network transfer. Copy and paste the summary information to a temporary file or notepad. If you are not familiar with fio you can find several other example files in the same directory. The command will run for about two minutes, with a summary at the end. Go slow the first time and view all the summary.

```
$ fio /usr/share/doc/fio/examples/netio.fio
....
Run status group 0 (all jobs):
  WRITE: bw=5750Mibit/s (6029Mbit/s), 5750Mibit/s-5750Mibit/s (6029Mbit/s-6029Mbit/s), io=100GiB (107GB), run=142471-142471msec
```

4. Change the MTU and run the test again. Note the performance change as the MTU Changes. Set it as low as the driver allows, such as 500, and as high as the driver allows, such as 9000. When testing to a remote node via a switch you may notice an even greater amount of difference in performance.

```
$ sudo ip link set dev wlp1s0 mtu 500
```

**Testing Name Services**

In this section we will test that we can resolve IP addresses, both forward and reverse lookups.

1. Use the host command to view the IP information for Linux.com.
$ host Linux.com
Linux.com has address 151.101.65.5
Linux.com has address 151.101.129.5
Linux.com has address 151.101.193.5
Linux.com has address 151.101.1.5
Linux.com mail is handled by 10 smtp2.linuxfoundation.org.
Linux.com mail is handled by 10 smtp1.linuxfoundation.org.

2. Compare the output with what you can see using the `nslookup` command.

$ nslookup Linux.com
Server: 127.0.0.53
Address: 127.0.0.53#53

Non-authoritative answer:
Name: Linux.com
Address: 151.101.1.5
Name: Linux.com
Address: 151.101.193.5
Name: Linux.com
Address: 151.101.129.5
Name: Linux.com
Address: 151.101.65.5

3. Use the `dig` command to view all the DNS information about Linux.com.

$ dig Linux.com

; <<<> DiG 9.11.3-1ubuntu1.8-Ubuntu <<<> Linux.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 15396
;; flags: qr rd ra; QUERY: 1, ANSWER: 4, AUTHORITY: 0, ADDITIONAL: 1
4. View the name servers (NS) and mail server (MX) for the Linux.com network. The output to the commands is not shown below. You should see a couple entries for each.

   $ $ dig -t NS Linux.com
   ....

   $ dig -t MX Linux.com
   ....

5. Choose an IP from the output and use **dig** to resolve the name using the IP address. Note the PTR record does not show a name.

   $ dig -X 151.101.193.5
   ; <<>> DiG 9.11.3-1ubuntu1.8-Ubuntu <<>> -x 151.101.193.5
   ;; global options: +cmd
   ;; Got answer:
   ;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 31490
6. Check a DNS server for Google. Find the name it resolves to:

$ dig -x 4.2.2.2
.....
2.2.2.4.in-addr.arpa. 60 IN PTR b.resolvers.Level3.net.
.....

7. Check your primary interface IP and determine if a name is known. Check your route gateway as well. By now you should be able to find the IP and use **dig**, with being shown the command. A blank prompt is shown if you want to write in your own command.

$ 

### Monitoring Traffic

In this section we will use commands to view the network traffic.

1. Use **tcpdump** to view traffic on your `lo` interface. Use **Ctrl-c** to stop the command. Then view traffic on your primary interface. Use the **ip** command to find your primary interface.
$ sudo tcpdump -i wlp1s0
....
05:13:19.469554 IP 12.235.16.3.54146 > ip-172-31-21-106.us-west-2.compute.internal.ssh: Flags [P.], seq 1333:1369, ack 2051348, win 2140, options [nop,nop,TS val 3815739841 ecr 397169965], length 36
....

2. Use a filter to narrow down traffic to only port 80, if there is any.

$ sudo tcpdump tcp port 80 -i eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
^C
0 packets captured
0 packets received by filter
0 packets dropped by kernel

3. Use `tcpdump` to save traffic to a `pcap` file. Let the command run for a moment, then interrupt with `Ctrl-c`. If asked, answer yes to any questions.

$ sudo tcpdump -i eth0 -w traffic.out.pcap

4. Install `Wireshark`.

   `ubuntu$ sudo apt-get install -y wireshark`

   `redhat$ sudo yum -y install wireshark`

5. Use `sudo` to start `wireshark`. Be aware you may need to use the `ssh -X` command or other configuration in PuTTY for the display to export back to your local system, if connecting to a remote node. Select any interface to begin collecting packets.

   $ sudo wireshark
CHAPTER 2. NETWORKING BASICS

Figure 2.4: Capture Traffic on Any Interface

Welcome to Wireshark

Capture

...using this filter: Enter a capture filter ...

Any interface is shown.

Learn

User's Guide · Wiki · Questions and Answers · Mailing Lists

You are running Wireshark 2.6.8 (Git v2.6.8 packaged as 2.6.8-1~ubuntu18.04.0).

Figure 2.4: Capture Traffic on Any Interface
6. The red square allows you to stop gathering packets. View some of the display filters by typing in the box, or selecting the Expression drop down.

7. After you stop collecting traffic you can go under **File** then **Open** and open the `traffic.out.pcap` file you created using `tcpdump`. Select a packet and explore the packet.
Figure 2.5: View A PCAP File
8. Experiment with **wireshark** as time allows. Work through the menus at the top and see the various tools and features available.