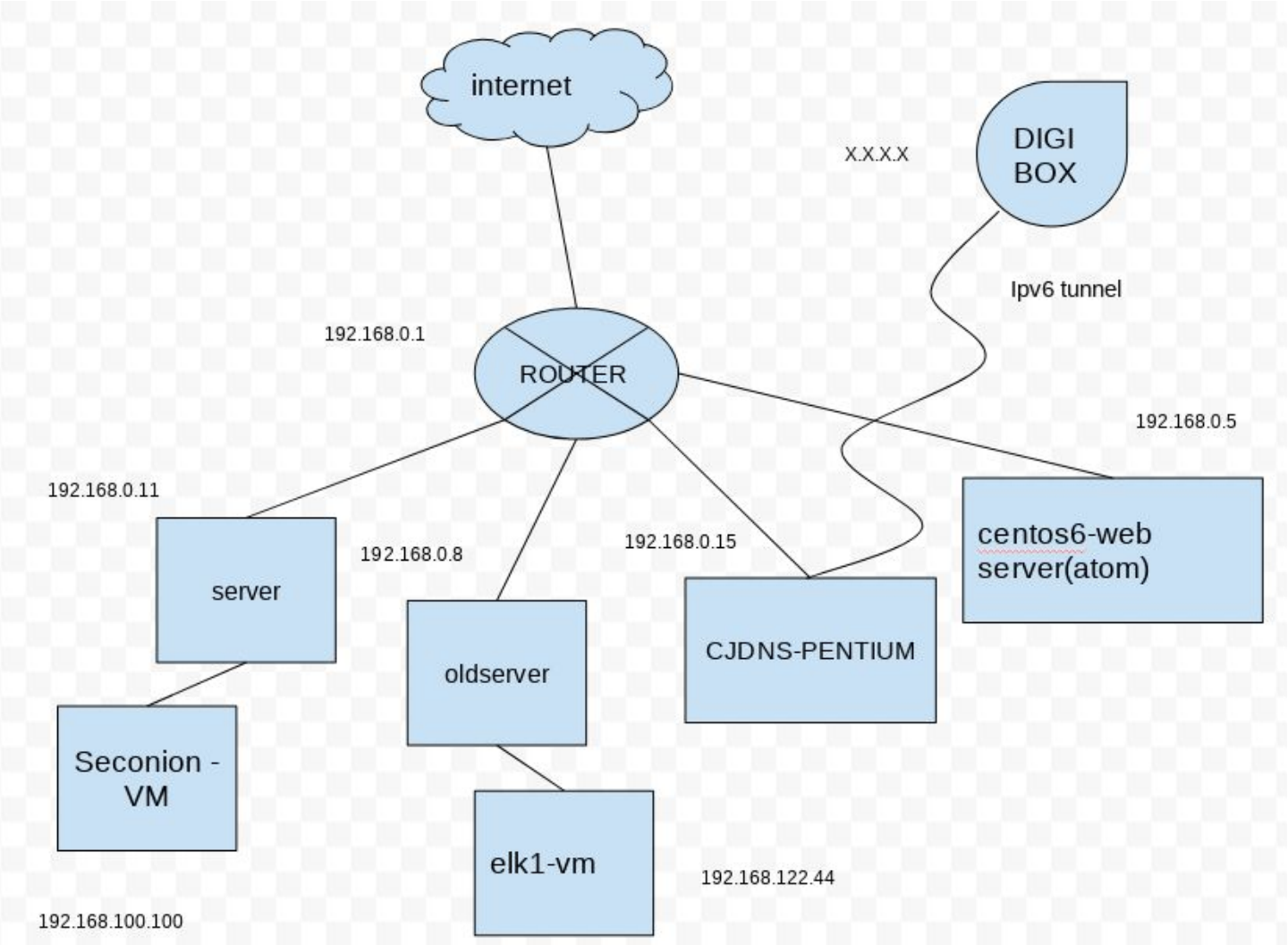
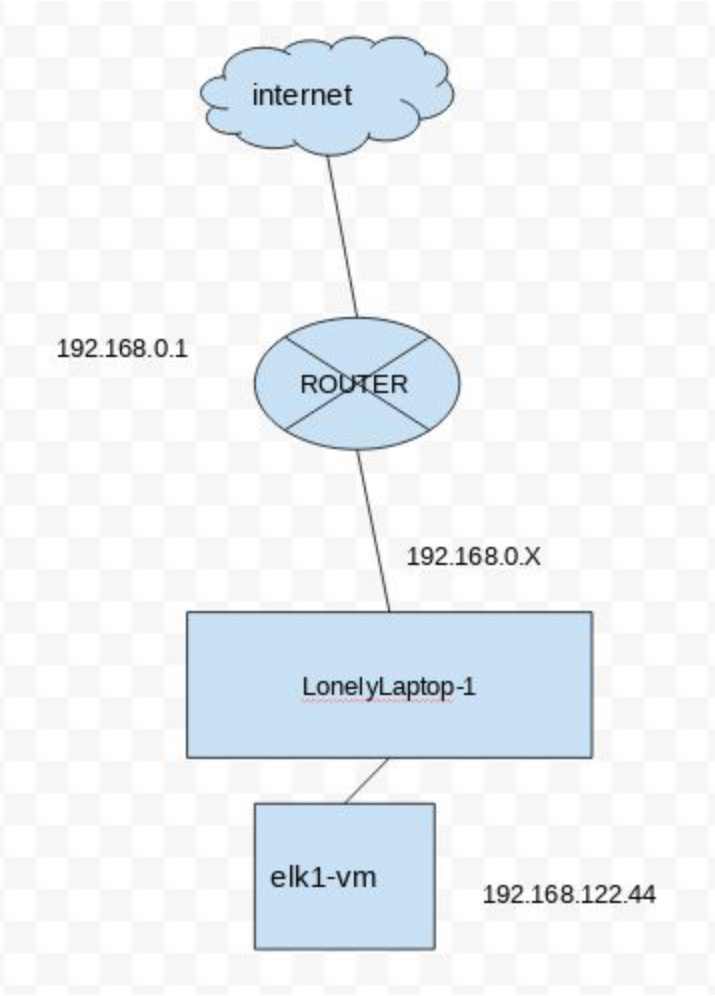
Author: J.Ortiz

Scenario

- **Rationale**
    - Blue team needs data. The more data, and the higher the quality of the data, the better. We need a way to collect this data, aggregate it, and visualize, which is where ELK (Elasticsearch, Logstash, Kibana, hereafter just referred to as elastic) stack comes in. Elastic is preferred for a number of reasons: it's open source, and thus the economic barrier to entry for those wanting to learn it is almost nil (save for the cost of your own home equipment and time).
    - The framework has a ton of use cases; beyond security data aggregation, tons of internal customers can benefit from the data.
    - CentOS (hereafter referred to as just centos) ? : Centos and RHEL are what you're going to encounter in production environments, which is part of the reason this lab is based on that flavor. Furthermore, last time I checked I think most if not all of the infrastructure at AZ02 was centos 7 based, so it was a sort of natural extension to choose that flavor for the home lab.
    - Historically we've been good at pushing cyber security interest via red team stuff, so this is an attempt to bring more balance and bring in more blue.
- **But, security onion and others setup ELK for me automatically, why should I do this?? -** This is true; security onion effortlessly sets up ELK stack and many other tools for you on the fly, but it robs you of the learning opportunity, satisfaction, and pain of setting up and troubleshooting such a framework yourself; think of it like digital legos; what's the point of buying pre-built lego structures? The fun is in building them.
- **But, I only have 1 computer:** I deployed this in a home lab that's moderately sized and probably has more machines than what most individuals will have at their disposal; that being said, you could make the setup more barebones and noob friendly by just running centos 7 on one computer and then using virt-manager to manage the install/setup of the centos 6 machine.
- Below is a diagram of the current setup I have this functioning in



- Server, oldserver, and digibox are all CentOS 7 machines
- Cjdns-pentium, elk1-vm, centos6-web server, are all CentOS 6 machines.
- A stripped down version would look something like the below
    - Note the 192.168.122.44 for the vm; this is the default virtual network created by kvm/libvirt, I recommend leaving this as the default until you're ready to start breaking networking and spending hours fixing shit; but who knows, maybe you'll find it easier to fix that shit than I did at first.



- So in the above we'd have LonelyLpatop-1 as a centos 7 machine.

- **Optional:** You can download a centos 6 ISO and use that to install the VM in virt-manager; In the CLI-only version of this process I didn't use an ISO but instead install the OS from the web (this was mostly due to the fact that some of the copy pasta used for the command line argument was failing miserably; the web install worked first try, so I figured why not?)

Requirements

- A machine running centos 7 and an internet connection
- Some level of comfort at the linux command line.
- Packages
  - libvirt
  - qemu
  - Kvm

    - virt-manager if you want a GUI
    - virt-install
    - These may be installed on your centos 7 (host) image depending what you've done, if not, just yum install them.
        - `$ yum -y install libvirt qemu kvm virt-manager virt-install`
        - `$ yum -y update`
- Time.

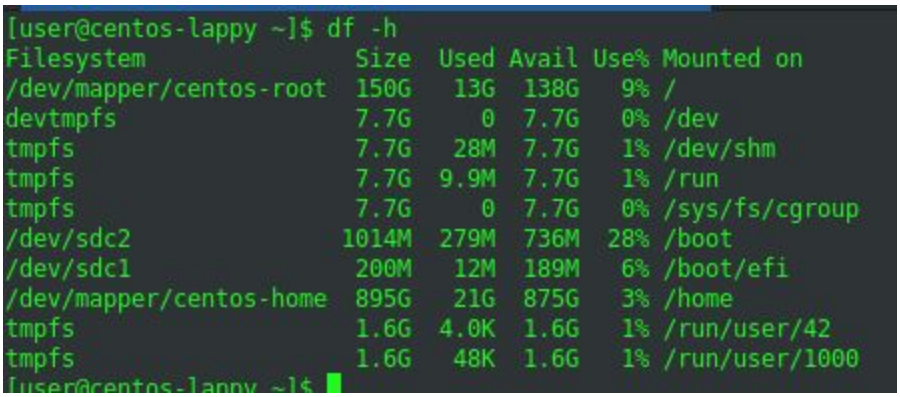
Text Conventions

- Command line arguments will be prepended with "$" and appear in a courier type font; otherwise screen shots should suffice.
    - `$ echo "This is a sample command line"`
- Errors -- in **Appendix A** i'll cover some common errors that were encountered; each entry will be prepended with the word "Error:" this is a label, it is NOT what you'll see on the command line, what follows immediately related to what you see at the command line

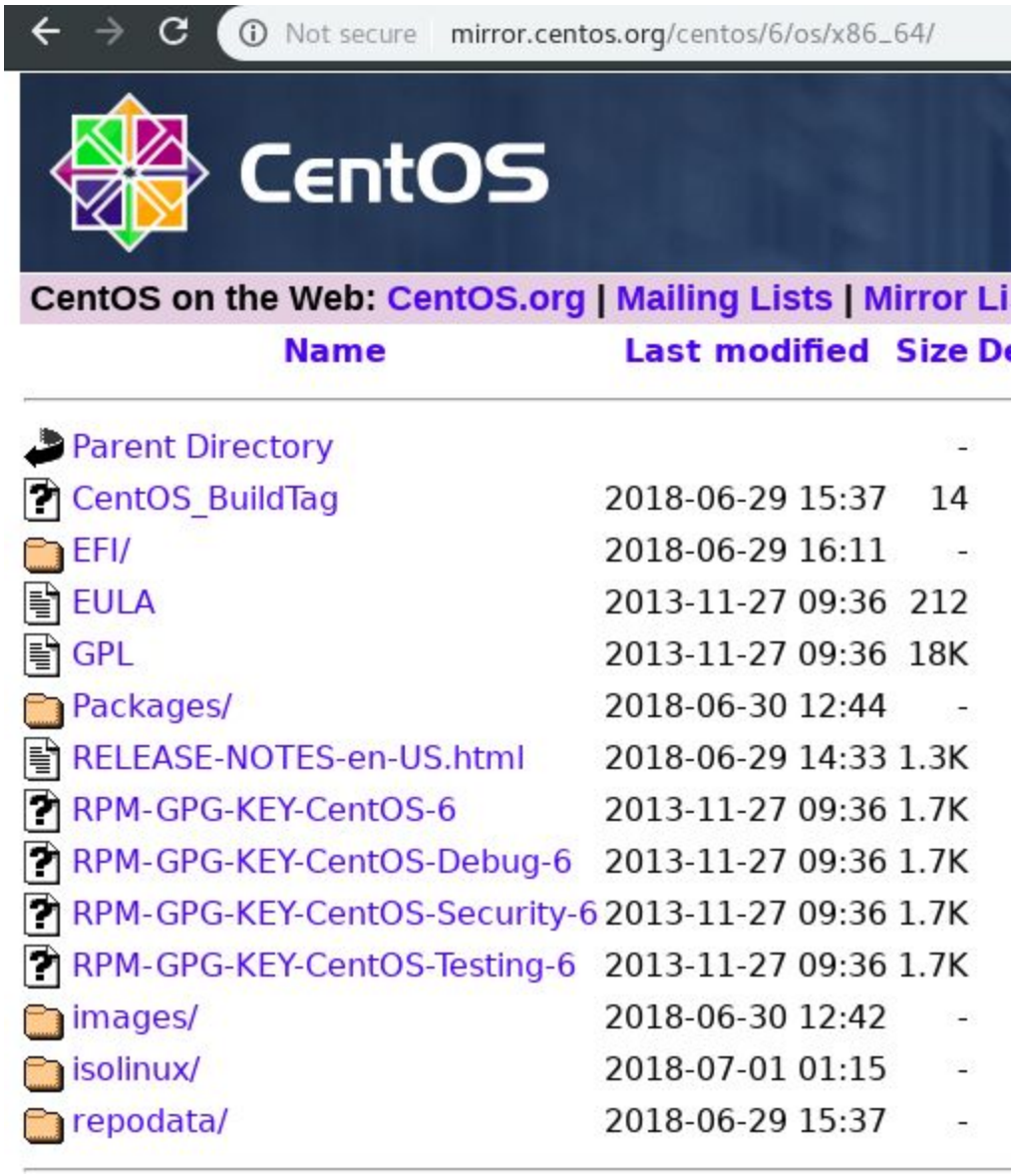
Bombs away

1. First thing we need to do is install our vm from an image directory we can point to on the web. The first command is all on one line with arbitrary line breaks where this editor saw fit; you could beautify it with " \ " breaks, but… meh, let chaos rain.
    a. `$ virt-install -n centos6-elk --description "Test VM with RHEL 6" --os-type=Linux --os-variant=rhel6 --ram=8192 --vcpus=2 --disk path=/var/lib/libvirt/images/centos6-elk.img,bus=virtio,size=50 --graphics none --location 'http://mirror.centos.org/centos/6/os/x86_64/' --extra-args 'console=ttyS0,115200 serial'`
    b. `-n : this flag allows us to specify an arbitrary name for our new vm, in this case i chose centos6-elk; you should practice coming up with a self-identifying, standardized naming scheme, because you'll run into them in industry. Example: cen7-pl02 could designate a Centos 7 Production Linux machine.`
    c. `--description : this is an arbitrary description of your vm`
    d. `--os-type : this is self explanatory above`
    e. `--os-variant=rhel6 : again this is self-explanatory, rhel6 == red hat enterprise linux 6 == centos 6`
    f. `--ram : this allows us to specify the ram in KB; THIS WILL DEPEND ON YOUR SETUP, above I give mine 8GB because I have 8GB to spare because the server has a total of 20GB, 4GB should be sufficient; 2GB is probably pushing it.`

g. --vcpus : allows us to specify the number of virtual CPUs for the VM; I read somewhere that the limit for this number depends on the number of CPUs your metal has; i.e. an 8 core machine can only virtualize an 8 cpu VM and simple reasoning suggests the # of cpus for your vm should be less than what your machine.
h. --disk path= : this specifies where our image for this VM should be stored; above I point it to the default path where libvirt stores VM images;
i. size=50 : this sets the size of the VM's "hard drive" to 50G; this will require you to have at least 50G free on your root partition; can be troublesome because automatic partitioning on centos 7 installs a smaller partition for / and makes the extra space available at /home. Below is an example; root only has 138G after I extended it to be such, whereas the bulk of my 1TB hard drive is available to the /home partition. (This size is optional; you could probably get away with a vm of 30, or even 20GB; after several weeks of logging I've only used about 12GB of storage on my VM's hard drive).



j. --location : this is the location of an iso directory (I forget the exact term). The above url will look like this



k. `--extra-args` : here we specify the console for our VM; this enables us to user virsh console on the VM later so we can access a "console" on the VM without having to ssh to it (i.e. in case we were to break ssh because we suck at networking, which I've done several times).

2. If you ran into errors on the previous step, see Appendix A; the errors covered there will progress in the order encountered; i.e.e in the logical order of the install. Below is a sample paste of the output from completing this command successfully (this starts an in-terminal graphical install, which is pretty cool if you've never seen it done before)

```
[user@centos-lappy ~]$ virt-install -n centos6-elk2 --description "Test VM with RHEL 6"
--os-type=Linux --os-variant=rhel6 --ram=2048 --vcpus=1 --disk
path=/var/lib/libvirt/images/centos6-elk.img,bus=virtio,size=30 --graphics none --location
```

```
'http://mirror.centos.org/centos/6/os/x86_64/' --extra-args 'console=ttyS0,115200 serial'
ERROR    Error: --disk path=/var/lib/libvirt/images/centos6-elk.img,bus=virtio,size=30: Could not
start storage pool: cannot open directory '/var/lib/libvirt/images': Permission denied
[user@centos-lappy ~]$ sudo virt-install -n centos6-elk2 --description "Test VM with RHEL 6"
--os-type=Linux --os-variant=rhel6 --ram=2048 --vcpus=1 --disk
path=/var/lib/libvirt/images/centos6-elk.img,bus=virtio,size=30 --graphics none --location
'http://mirror.centos.org/centos/6/os/x86_64/' --extra-args 'console=ttyS0,115200 serial'

Starting install...
Retrieving file vmlinuz...                                                    | 4.1 MB
00:00:01
Retrieving file initrd.img...                                                 |  39 MB
00:00:08
Allocating 'centos6-elk.img'                                                  |  30 GB
00:00:00
Connected to domain centos6-elk2
Escape character is ^]
Initializing cgroup subsys cpuset
Initializing cgroup subsys cpu
Linux version 2.6.32-754.el6.x86_64 (mockbuild@x86-01.bsys.centos.org) (gcc version 4.4.7 20120313
(Red Hat 4.4.7-23) (GCC) ) #1 SMP Tue Jun 19 21:26:04 UTC 2018
Command line: console=ttyS0,115200 serial method=http://mirror.centos.org/centos/6/os/x86_64/
KERNEL supported cpus:
  Intel GenuineIntel
  AMD AuthenticAMD
  Centaur CentaurHauls
BIOS-provided physical RAM map:
 BIOS-e820: 0000000000000000 - 000000000009fc00 (usable)
 BIOS-e820: 000000000009fc00 - 00000000000a0000 (reserved)
 BIOS-e820: 00000000000f0000 - 0000000000100000 (reserved)
 BIOS-e820: 0000000000100000 - 000000007fff7000 (usable)
 BIOS-e820: 000000007fff7000 - 0000000080000000 (reserved)
 BIOS-e820: 00000000feffc000 - 00000000ff000000 (reserved)
 BIOS-e820: 00000000fffc0000 - 0000000100000000 (reserved)
SMBIOS version 2.4 @ 0xF6210
SMBIOS 2.4 present.
Hypervisor detected: KVM
last_pfn = 0x7fff7 max_arch_pfn = 0x400000000
x86 PAT enabled: cpu 0, old 0x7040600070406, new 0x7010600070106
init_memory_mapping: 0000000000000000-000000007fff7000
RAMDISK: 7d8d8000 - 7ffefc9a
ACPI: Deleted _OSI(Windows 2012)
ACPI: Deleted _OSI(Windows 2013)
ACPI: RSDP 00000000000f61e0 00014 (v00 BOCHS )
ACPI: RSDT 000000007ffffad7 00030 (v01 BOCHS  BXPCRSDT 00000001 BXPC 00000001)
ACPI: FACP 000000007ffff177 00074 (v01 BOCHS  BXPCFACP 00000001 BXPC 00000001)
ACPI: DSDT 000000007fffe040 01137 (v01 BOCHS  BXPCDSDT 00000001 BXPC 00000001)
ACPI: FACS 000000007fffe000 00040
ACPI: SSDT 000000007ffff1eb 00874 (v01 BOCHS  BXPCSSDT 00000001 BXPC 00000001)
ACPI: APIC 000000007ffffa5f 00078 (v01 BOCHS  BXPCAPIC 00000001 BXPC 00000001)
Setting APIC routing to flat.
No NUMA configuration found
Faking a node at 0000000000000000-000000007fff7000
Bootmem setup node 0 0000000000000000-000000007fff7000
  NODE_DATA [000000000000a000 - 000000000003dfff]
  bootmap [000000000003e000 -  000000000004dfff] pages 10
(7 early reservations) ==> bootmem [0000000000 - 007fff7000]
  #0 [0000000000 - 0000001000]   BIOS data page ==> [0000000000 - 0000001000]
  #1 [0000006000 - 0000008000]       TRAMPOLINE ==> [0000006000 - 0000008000]
  #2 [0001000000 - 0002050a64]    TEXT DATA BSS ==> [0001000000 - 0002050a64]
```

```
  #3 [007d8d8000 - 007ffefc9a]          RAMDISK ==> [007d8d8000 - 007ffefc9a]
  #4 [000009fc00 - 0000100000]    BIOS reserved ==> [000009fc00 - 0000100000]
  #5 [0002051000 - 00020510b9]              BRK ==> [0002051000 - 00020510b9]
  #6 [0000008000 - 000000a000]          PGTABLE ==> [0000008000 - 000000a000]
found SMP MP-table at [ffff8800000f6380] f6380
kvm-clock: Using msrs 4b564d01 and 4b564d00
kvm-clock: cpu 0, msr 0:1c3db81, boot clock
Zone PFN ranges:
  DMA      0x00000001 -> 0x00001000
  DMA32    0x00001000 -> 0x00100000
  Normal   0x00100000 -> 0x00100000
Movable zone start PFN for each node
early_node_map[2] active PFN ranges
    0: 0x00000001 -> 0x0000009f
    0: 0x00000100 -> 0x0007fff7
ACPI: PM-Timer IO Port: 0x608
Setting APIC routing to flat.
ACPI: LAPIC (acpi_id[0x00] lapic_id[0x00] enabled)
ACPI: LAPIC_NMI (acpi_id[0xff] dfl dfl lint[0x1])
ACPI: IOAPIC (id[0x00] address[0xfec00000] gsi_base[0])
IOAPIC[0]: apic_id 0, version 17, address 0xfec00000, GSI 0-23
ACPI: INT_SRC_OVR (bus 0 bus_irq 0 global_irq 2 dfl dfl)
ACPI: INT_SRC_OVR (bus 0 bus_irq 5 global_irq 5 high level)
ACPI: INT_SRC_OVR (bus 0 bus_irq 9 global_irq 9 high level)
ACPI: INT_SRC_OVR (bus 0 bus_irq 10 global_irq 10 high level)
ACPI: INT_SRC_OVR (bus 0 bus_irq 11 global_irq 11 high level)
Using ACPI (MADT) for SMP configuration information
SMP: Allowing 1 CPUs, 0 hotplug CPUs
PM: Registered nosave memory: 000000000009f000 - 00000000000a0000
PM: Registered nosave memory: 00000000000a0000 - 00000000000f0000
PM: Registered nosave memory: 00000000000f0000 - 0000000000100000
Allocating PCI resources starting at 80000000 (gap: 80000000:7effc000)
Booting paravirtualized kernel on KVM
NR_CPUS:4096 nr_cpumask_bits:1 nr_cpu_ids:1 nr_node_ids:1
PERCPU: Embedded 33 pages/cpu @ffff880002200000 s104088 r8192 d22888 u2097152
pcpu-alloc: s104088 r8192 d22888 u2097152 alloc=1*2097152
pcpu-alloc: [0] 0
kvm-clock: cpu 0, msr 0:2218b81, primary cpu clock
kvm-stealtime: cpu 0, msr 2212d00
Built 1 zonelists in Node order, mobility grouping on.  Total pages: 516912
Policy zone: DMA32
Kernel command line: console=ttyS0,115200 serial
method=http://mirror.centos.org/centos/6/os/x86_64/
PID hash table entries: 4096 (order: 3, 32768 bytes)
x86/fpu: xstate_offset[2]: 0240, xstate_sizes[2]: 0100
xsave/xrstor: enabled xstate_bv 0x7, cntxt size 0x340
Memory: 2010672k/2097116k available (5520k kernel code, 392k absent, 86052k reserved, 6909k data,
1340k init)
Kernel/User page tables isolation: enabled
Hierarchical RCU implementation.
NR_IRQS:33024 nr_irqs:256
Console: colour *CGA 80x25
console [ttyS0] enabled
allocated 8388608 bytes of page_cgroup
please try 'cgroup_disable=memory' option if you don't want memory cgroups
Detected 2893.430 MHz processor.
Calibrating delay loop (skipped) preset value.. 5786.86 BogoMIPS (lpj=2893430)
pid_max: default: 32768 minimum: 301
Security Framework initialized
SELinux:  Initializing.
```

```
Dentry cache hash table entries: 262144 (order: 9, 2097152 bytes)
Inode-cache hash table entries: 131072 (order: 8, 1048576 bytes)
Mount-cache hash table entries: 256
Initializing cgroup subsys ns
Initializing cgroup subsys cpuacct
Initializing cgroup subsys memory
Initializing cgroup subsys devices
Initializing cgroup subsys freezer
Initializing cgroup subsys net_cls
Initializing cgroup subsys blkio
Initializing cgroup subsys perf_event
Initializing cgroup subsys net_prio
mce: CPU supports 10 MCE banks
Speculative Store Bypass: Vulnerable
FEATURE SPEC_CTRL Present
FEATURE IBPB_SUPPORT Present
Spectre V2 : Mitigation: Full retpoline
alternatives: switching to unfair spinlock
SMP alternatives: switching to UP code
Freeing SMP alternatives: 38k freed
ACPI: Core revision 20090903
ftrace: converting mcount calls to 0f 1f 44 00 00
ftrace: allocating 22051 entries in 87 pages
Enabling x2apic
Enabled x2apic
APIC routing finalized to physical x2apic.
..TIMER: vector=0x30 apic1=0 pin1=2 apic2=-1 pin2=-1
CPU0: Intel Xeon E3-12xx v2 (Ivy Bridge, IBRS) stepping 09
Performance Events: unsupported p6 CPU model 58 no PMU driver, software events only.
NMI watchdog disabled (cpu0): hardware events not enabled
Brought up 1 CPUs
Total of 1 processors activated (5786.86 BogoMIPS).
devtmpfs: initialized
regulator: core version 0.5
NET: Registered protocol family 16
ACPI: bus type pci registered
PCI: Using configuration type 1 for base access
bio: create slab <bio-0> at 0
ACPI: Interpreter enabled
ACPI: (supports S0 S5)
ACPI: Using IOAPIC for interrupt routing
ACPI: No dock devices found.
PCI: Using host bridge windows from ACPI; if necessary, use "pci=nocrs" and report a bug
ACPI: PCI Root Bridge [PCI0] (domain 0000 [bus 00-ff])
pci_root PNP0A03:00: host bridge window [io  0x0000-0x0cf7]
pci_root PNP0A03:00: host bridge window [io  0x0d00-0xffff]
pci_root PNP0A03:00: host bridge window [mem 0x000a0000-0x000bffff]
pci_root PNP0A03:00: host bridge window [mem 0x80000000-0xfebfffff]
PCI host bridge to bus 0000:00
pci_bus 0000:00: root bus resource [io  0x0000-0x0cf7]
pci_bus 0000:00: root bus resource [io  0x0d00-0xffff]
pci_bus 0000:00: root bus resource [mem 0x000a0000-0x000bffff]
pci_bus 0000:00: root bus resource [mem 0x80000000-0xfebfffff]
pci 0000:00:01.3: quirk: [io  0x0600-0x063f] claimed by PIIX4 ACPI
pci 0000:00:01.3: quirk: [io  0x0700-0x070f] claimed by PIIX4 SMB
ACPI: PCI Interrupt Link [LNKA] (IRQs 5 *10 11)
ACPI: PCI Interrupt Link [LNKB] (IRQs 5 *10 11)
ACPI: PCI Interrupt Link [LNKC] (IRQs 5 10 *11)
ACPI: PCI Interrupt Link [LNKD] (IRQs 5 10 *11)
ACPI: PCI Interrupt Link [LNKS] (IRQs *9)
```

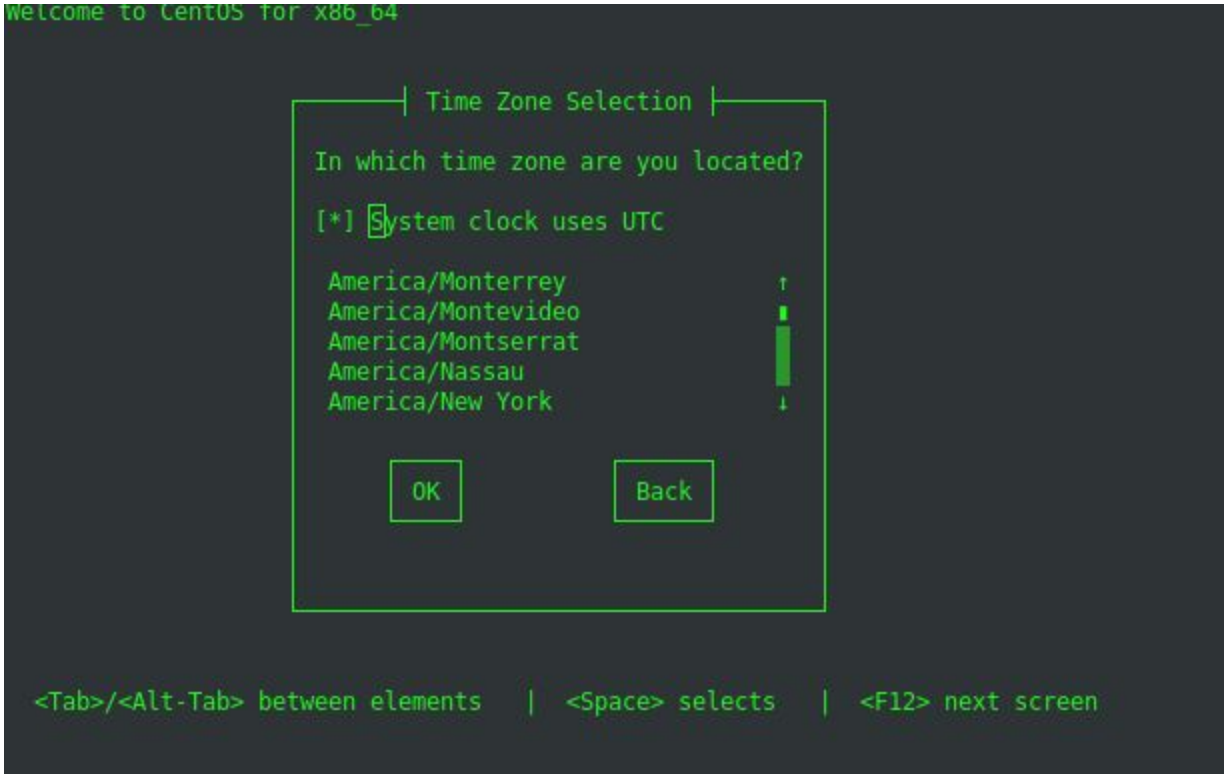
```
vgaarb: loaded
SCSI subsystem initialized
usbcore: registered new interface driver usbfs
usbcore: registered new interface driver hub
usbcore: registered new device driver usb
PCI: Using ACPI for IRQ routing
NetLabel: Initializing
NetLabel:  domain hash size = 128
NetLabel:  protocols = UNLABELED CIPSOv4
NetLabel:  unlabeled traffic allowed by default
Switching to clocksource kvm-clock
pnp: PnP ACPI init
ACPI: bus type pnp registered
pnp: PnP ACPI: found 6 devices
ACPI: ACPI bus type pnp unregistered
NET: Registered protocol family 2
IP route cache hash table entries: 65536 (order: 7, 524288 bytes)
TCP established hash table entries: 262144 (order: 10, 4194304 bytes)
TCP bind hash table entries: 65536 (order: 8, 1048576 bytes)
TCP: Hash tables configured (established 262144 bind 65536)
TCP reno registered
NET: Registered protocol family 1
pci 0000:00:00.0: Limiting direct PCI/PCI transfers
pci 0000:00:01.0: PIIX3: Enabling Passive Release
pci 0000:00:01.0: Activating ISA DMA hang workarounds
ACPI: PCI Interrupt Link [LNKD] enabled at IRQ 11
pci 0000:00:04.0: PCI INT A -> Link[LNKD] -> GSI 11 (level, high) -> IRQ 11
pci 0000:00:04.0: PCI INT A disabled
ACPI: PCI Interrupt Link [LNKA] enabled at IRQ 10
pci 0000:00:04.1: PCI INT B -> Link[LNKA] -> GSI 10 (level, high) -> IRQ 10
pci 0000:00:04.1: PCI INT B disabled
ACPI: PCI Interrupt Link [LNKB] enabled at IRQ 10
pci 0000:00:04.2: PCI INT C -> Link[LNKB] -> GSI 10 (level, high) -> IRQ 10
pci 0000:00:04.2: PCI INT C disabled
ACPI: PCI Interrupt Link [LNKC] enabled at IRQ 11
pci 0000:00:04.7: PCI INT D -> Link[LNKC] -> GSI 11 (level, high) -> IRQ 11
pci 0000:00:04.7: PCI INT D disabled
Trying to unpack rootfs image as initramfs...
Freeing initrd memory: 40031k freed
sha256_ssse3: Using AVX optimized SHA-256 implementation
futex hash table entries: 256 (order: 2, 16384 bytes)
audit: initializing netlink socket (disabled)
type=2000 audit(1544107628.671:1): initialized
HugeTLB registered 2 MB page size, pre-allocated 0 pages
VFS: Disk quotas dquot_6.5.2
Dquot-cache hash table entries: 512 (order 0, 4096 bytes)
msgmni has been set to 4005
ksign: Installing public key data
Loading keyring
- Added public key 283EE91FE7BCC01A
- User ID: CentOS (Kernel Module GPG key)
Block layer SCSI generic (bsg) driver version 0.4 loaded (major 250)
io scheduler noop registered
io scheduler anticipatory registered
io scheduler deadline registered
io scheduler cfq registered (default)
pci_hotplug: PCI Hot Plug PCI Core version: 0.5
pciehp: PCI Express Hot Plug Controller Driver version: 0.4
acpiphp: ACPI Hot Plug PCI Controller Driver version: 0.5
acpiphp: Slot [2] registered
```

```
acpiphp: Slot [3] registered
acpiphp: Slot [5] registered
acpiphp: Slot [6] registered
acpiphp: Slot [7] registered
acpiphp: Slot [8] registered
acpiphp: Slot [9] registered
acpiphp: Slot [10] registered
acpiphp: Slot [11] registered
acpiphp: Slot [12] registered
acpiphp: Slot [13] registered
acpiphp: Slot [14] registered
acpiphp: Slot [15] registered
acpiphp: Slot [16] registered
acpiphp: Slot [17] registered
acpiphp: Slot [18] registered
acpiphp: Slot [19] registered
acpiphp: Slot [20] registered
acpiphp: Slot [21] registered
acpiphp: Slot [22] registered
acpiphp: Slot [23] registered
acpiphp: Slot [24] registered
acpiphp: Slot [25] registered
acpiphp: Slot [26] registered
acpiphp: Slot [27] registered
acpiphp: Slot [28] registered
acpiphp: Slot [29] registered
acpiphp: Slot [30] registered
acpiphp: Slot [31] registered
input: Power Button as /devices/LNXSYSTM:00/LNXPWRBN:00/input/input0
ACPI: Power Button [PWRF]
[Firmware Bug]: No valid trip found
GHES: HEST is not enabled!
Non-volatile memory driver v1.3
Linux agpgart interface v0.103
crash memory driver: version 1.1
Serial: 8250/16550 driver, 4 ports, IRQ sharing enabled
serial8250: ttyS0 at I/O 0x3f8 (irq = 4) is a 16550A
00:05: ttyS0 at I/O 0x3f8 (irq = 4) is a 16550A
brd: module loaded
loop: module loaded
input: Macintosh mouse button emulation as /devices/virtual/input/input1
Fixed MDIO Bus: probed
ehci_hcd: USB 2.0 'Enhanced' Host Controller (EHCI) Driver
ehci_hcd 0000:00:04.7: PCI INT D -> Link[LNKC] -> GSI 11 (level, high) -> IRQ 11
ehci_hcd 0000:00:04.7: EHCI Host Controller
ehci_hcd 0000:00:04.7: new USB bus registered, assigned bus number 1
ehci_hcd 0000:00:04.7: irq 11, io mem 0xfebc1000
ehci_hcd 0000:00:04.7: USB 2.0 started, EHCI 1.00
usb usb1: New USB device found, idVendor=1d6b, idProduct=0002
usb usb1: New USB device strings: Mfr=3, Product=2, SerialNumber=1
usb usb1: Product: EHCI Host Controller
usb usb1: Manufacturer: Linux 2.6.32-754.el6.x86_64 ehci_hcd
usb usb1: SerialNumber: 0000:00:04.7
usb usb1: configuration #1 chosen from 1 choice
hub 1-0:1.0: USB hub found
hub 1-0:1.0: 6 ports detected
ohci_hcd: USB 1.1 'Open' Host Controller (OHCI) Driver
uhci_hcd: USB Universal Host Controller Interface driver
uhci_hcd 0000:00:04.0: PCI INT A -> Link[LNKD] -> GSI 11 (level, high) -> IRQ 11
uhci_hcd 0000:00:04.0: UHCI Host Controller
```

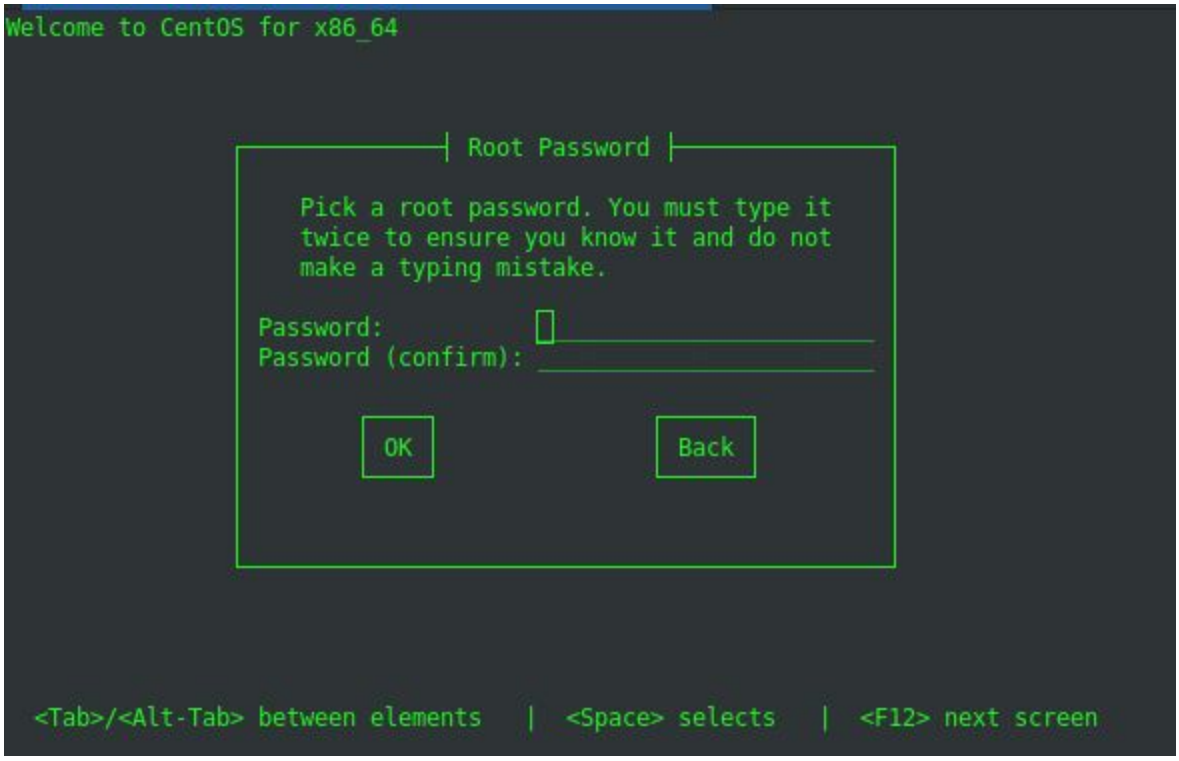
```
uhci_hcd 0000:00:04.0: new USB bus registered, assigned bus number 2
uhci_hcd 0000:00:04.0: irq 11, io base 0x0000c060
usb usb2: New USB device found, idVendor=1d6b, idProduct=0001
usb usb2: New USB device strings: Mfr=3, Product=2, SerialNumber=1
usb usb2: Product: UHCI Host Controller
usb usb2: Manufacturer: Linux 2.6.32-754.el6.x86_64 uhci_hcd
usb usb2: SerialNumber: 0000:00:04.0
usb usb2: configuration #1 chosen from 1 choice
hub 2-0:1.0: USB hub found
hub 2-0:1.0: 2 ports detected
uhci_hcd 0000:00:04.1: PCI INT B -> Link[LNKA] -> GSI 10 (level, high) -> IRQ 10
uhci_hcd 0000:00:04.1: UHCI Host Controller
uhci_hcd 0000:00:04.1: new USB bus registered, assigned bus number 3
uhci_hcd 0000:00:04.1: irq 10, io base 0x0000c080
usb usb3: New USB device found, idVendor=1d6b, idProduct=0001
usb usb3: New USB device strings: Mfr=3, Product=2, SerialNumber=1
usb usb3: Product: UHCI Host Controller
usb usb3: Manufacturer: Linux 2.6.32-754.el6.x86_64 uhci_hcd
usb usb3: SerialNumber: 0000:00:04.1
usb usb3: configuration #1 chosen from 1 choice
hub 3-0:1.0: USB hub found
hub 3-0:1.0: 2 ports detected
uhci_hcd 0000:00:04.2: PCI INT C -> Link[LNKB] -> GSI 10 (level, high) -> IRQ 10
uhci_hcd 0000:00:04.2: UHCI Host Controller
uhci_hcd 0000:00:04.2: new USB bus registered, assigned bus number 4
uhci_hcd 0000:00:04.2: irq 10, io base 0x0000c0a0
usb usb4: New USB device found, idVendor=1d6b, idProduct=0001
Welcome to CentOS for x86_64

                    ┌────────────────┤ Choose a Language ├────────────────┐
                    │                                                     │
                    │ What language would you like to use                 │
                    │ during the installation process?                    │
                    │                                                     │
                    │      Catalan                ↑                       │
                    │      Chinese(Simplified)    ▒                       │
                    │      Chinese(Traditional)   ▮                       │
                    │      Croatian               ▒                       │
                    │      Czech                  ▒                       │
                    │      Danish                 ▒                       │
                    │      Dutch                  ▒                       │
                    │      English                ↓                       │
                    │                                                     │
                    │                 ┌──────┐                            │
                    │                 │ OK   │                            │
                    │                 └──────┘                            │
                    │                                                     │
                    │                                                     │
                    └─────────────────────────────────────────────────────┘

  <Tab>/<Alt-Tab> between elements  | <Space> selects | <F12> next screen
```

a. When prompted for text vs vnc, choose text and you'll be taken to the centos screen; if you get an error about re-initializing see the appendix

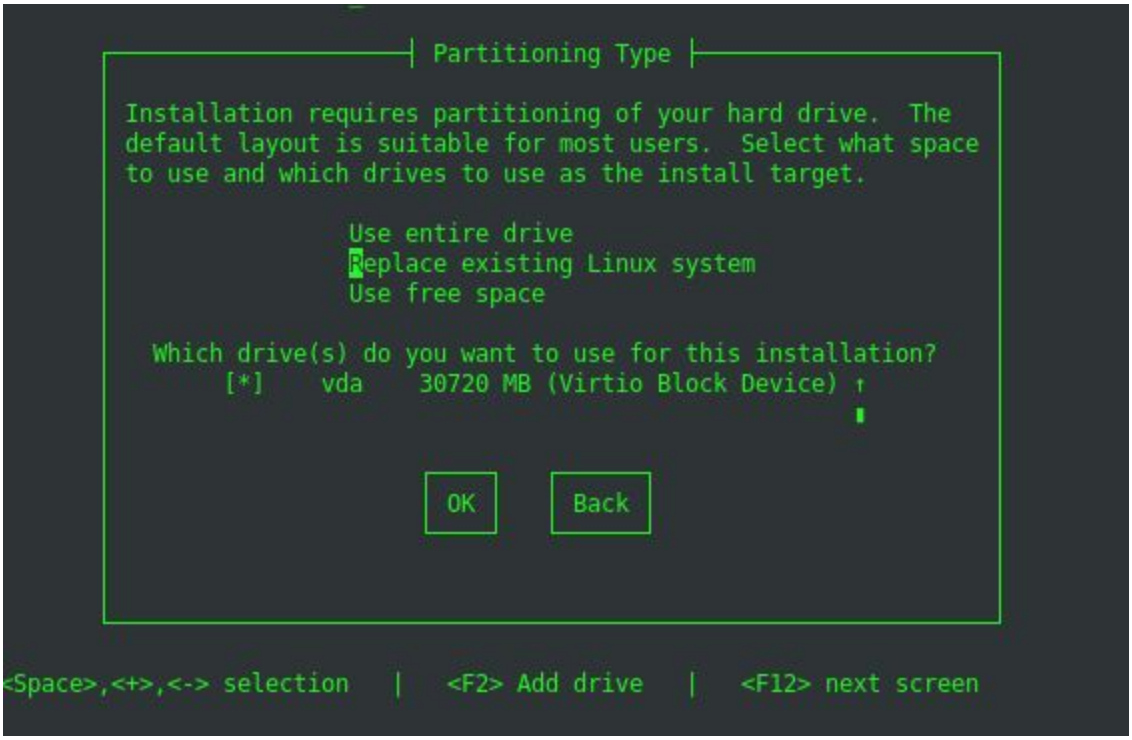
b. Select time zone



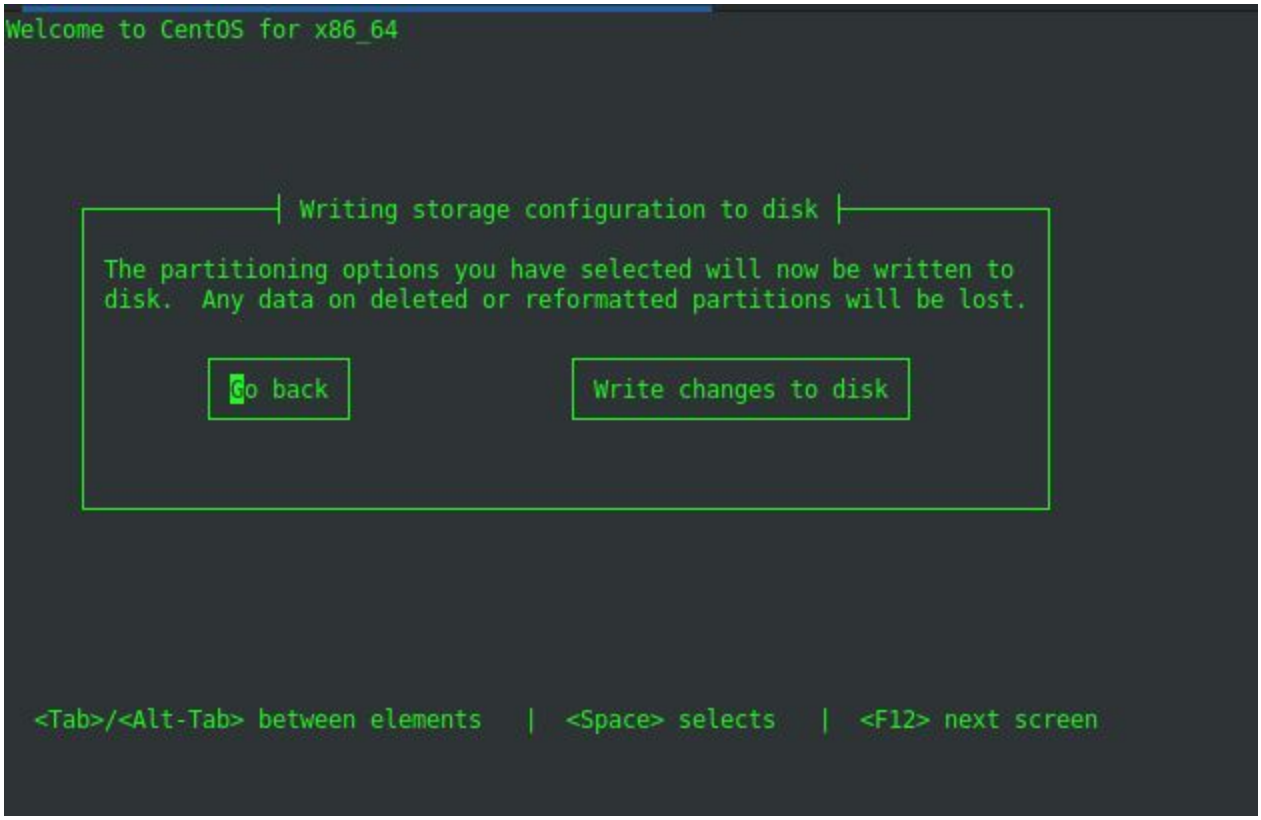
c. Set root password



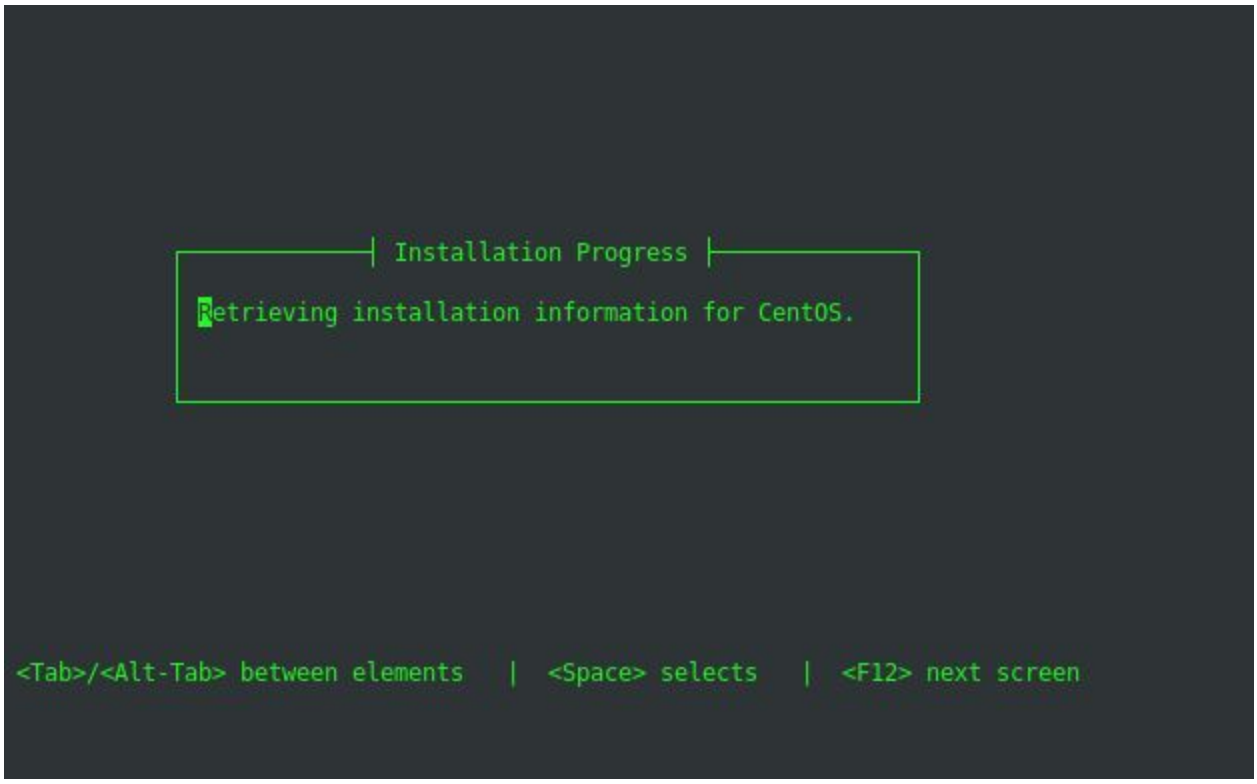
d. For partitioning type select "Replace existing linux system" ; no clear marker, i just make sure the cursor is on it, then hit enter, and tab down to OK



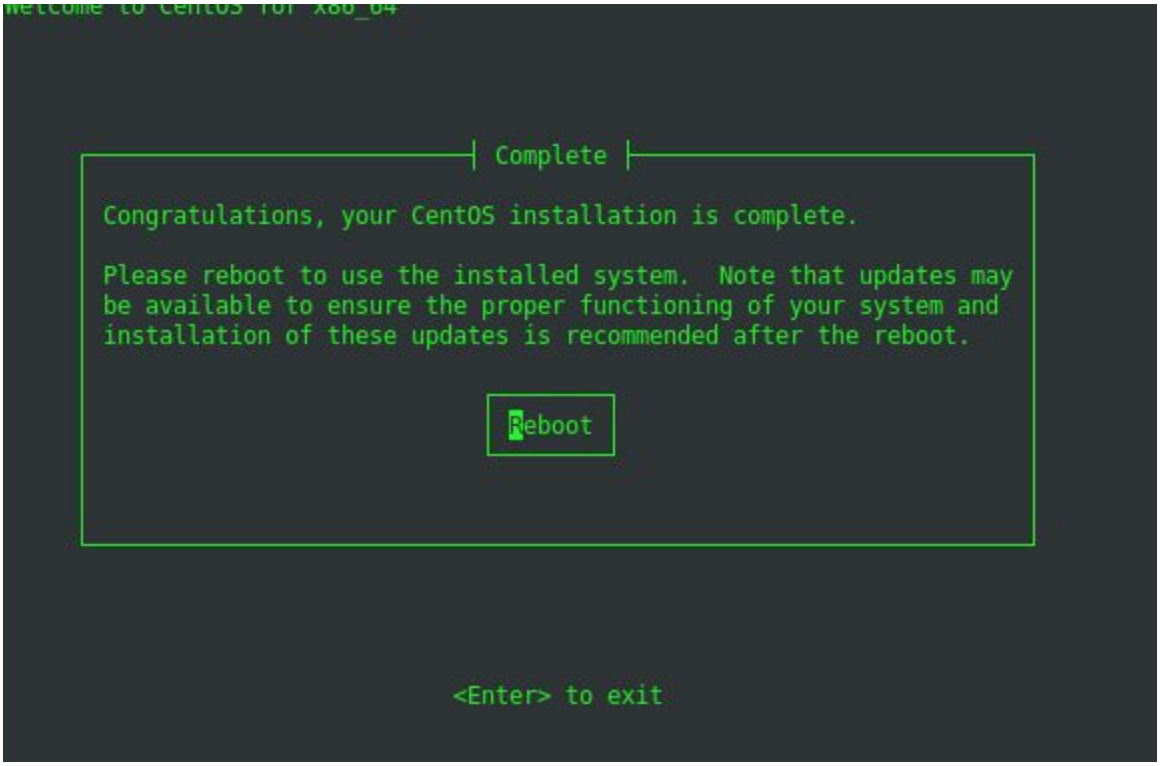
e. Write the changes to disk



f. Then it will go through the install for a bit



g. Once the installation is complete we just have to reboot

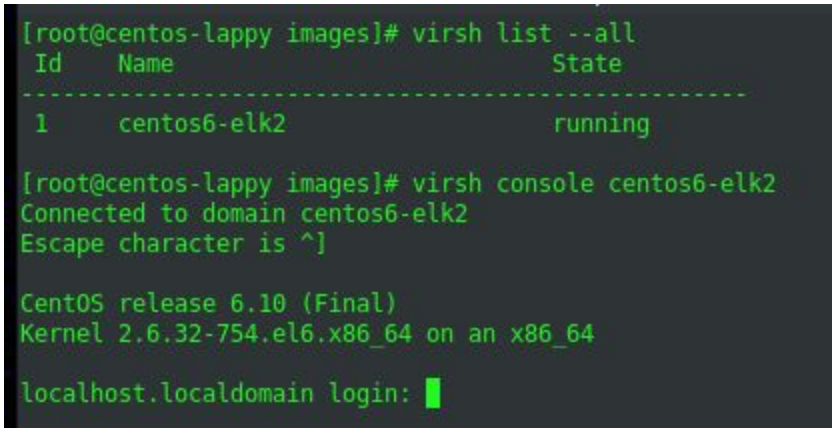


h. If you can catch it you'll see a prompt that says "Escape character is ^]" meaning CTRL-],; this is because we have a console session running to the vm which we can exit at anytime with the above key sequence. This can be confusing to some.

3. Once your VM install is done you can reconnect to it via the virsh console means with the command below and complete the setup of your VM.

a. `$ virsh console vm-name`

i. If you don't see a prompt to login right away just hit enter



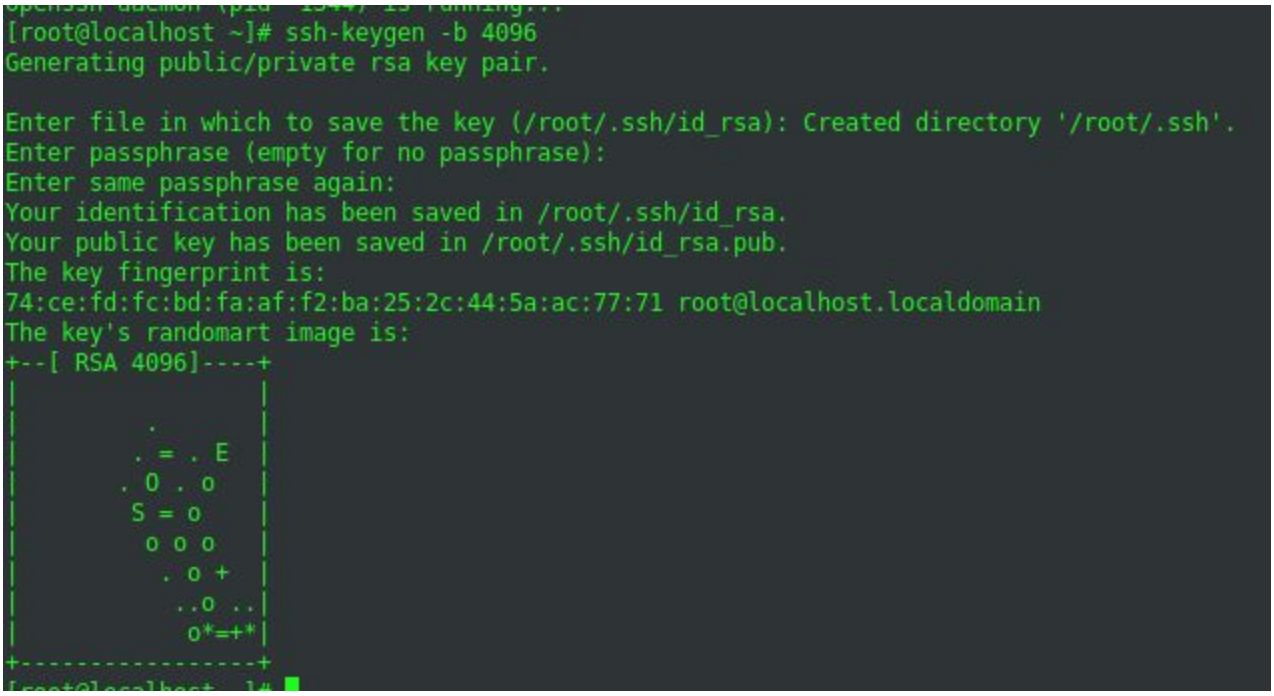
b. The first thing you should do is check if you have an IP address; if everything went well your vm should have been automatically assigned an IP address on the 192.168.122.0/24 subnet, a

virtual network which is automatically created on the host when you start using libvirt. Below is BAD output, i have no IP

i. If your vm isn't already assigned an IP address see Appendix A for some troubleshooting

```
[root@localhost ~]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 52:54:00:06:cc:11 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::5054:ff:fe06:cc11/64 scope link
       valid_lft forever preferred_lft forever
[root@localhost ~]# cat /etc/sysconfig/network
network          networking/      network-scripts/
[root@localhost ~]# cat /etc/sysconfig/network-scripts/ifcfg-
ifcfg-eth0  ifcfg-lo
[root@localhost ~]# cat /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE="eth0"
BOOTPROTO="dhcp"
HWADDR="52:54:00:06:CC:11"
IPV6INIT="yes"
IPV6_AUTOCONF="yes"
NM_CONTROLLED="yes"
ONBOOT="yes"
TYPE="Ethernet"
UUID="85764b05-bc71-4c0f-9ab4-6889d5aba25f"
```

4. Once your VM has an IP we need to allow it to communicate to the world so we can install software on it and navigate to it from our LAN (or at least our host). We'll do some ssh setup
    a. Create an ssh key pair
        i. `$ ssh-keygen -b 4096`



ii. Hit enter a few times to power through and store the pub/priv key pair in the default location.

iii. If you have ssh pub keys you want to store in .ssh/authorized_keys, now is the time to do so. If you don't have networking up and running yet and no connection to the outside world, then you can at least test ssh by putting the pub key you just generated into the authorized_keys file

```
1. $ cat /root/id_rsa.pub >> /root/.ssh/authorized_keys
```

iv. Once your desired pub keys are in the authorized_keys file, disable password authentication in /etc/ssh/sshd_config and reboot the sshd daemon

1. Desired config setting

```
# To disable tunneled clear text passwords, change to no here!
#PasswordAuthentication yes
#PermitEmptyPasswords no
PasswordAuthentication no
```

2. Reboot - recall that this is on the GUEST VM, which is running centos 6 and thus uses service restart, versus systemctl (both work on centos 7).

```
a. $ service restart sshd
```

v. For in depth look at good SSH configs search mozilla or see Sources [2]

b. If your VM can ping the outside net (test by pinging google or something) then run a yum update and yum upgrade on your VM asap because the install uses a minimal image and will thus have limited software (including some command line utilities).

```
 i.   $ yum -y update
ii.   $ yum -y upgrade
iii.  $ yum -y install
```

c. If you're doing an advanced setup and want this VM accessible from LAN, you'll need to see **Appendix B-1** for some firewall rules to forward traffic to the VM; if you're not very good with networking and routing, proceed with caution as it's easy to get this shit wrong, in which case you can spend hours troubleshooting which will become frustrating. At the end of it though you'll probably learn a lot about network troubleshooting, tcpdump, iptables, "ip route add/del" and other bits.

5. Assuming we're all connected and we can ping back and forth from HOST ⟵ ⟶ GUEST and our GUEST vm can access the web, we're ready to start installing.
6. Install java via rpm
   a. Option 1: download the RPM to your HOST (if you have a GUI this may be more noob friendly) in a browser from the following link and transfer it to the vm with scp or something
      i. https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html
   b. Or you can use some wget magic to download the rpm directly on the guest; as of 12/7/2018 this was functional copypasta.

```
c. $ wget --no-cookies --no-check-certificate --header "Cookie:
   gpw_e24=http%3A%2F%2Fwww.oracle.com%2F;
   oraclelicense=accept-securebackup-cookie"
   https://download.oracle.com/otn-pub/java/jdk/8u191-b12/2787e4a523244c269598db4
   e85c51e0c/jdk-8u191-linux-x64.rpm
```

   d. Once you have the rpm on the GUEST vm yum install it

```
 i.   $ yum -y install jdk-8u191-linux-x64.rpm
```

```
[root@elk2 elk-install]# yum -y install jdk-8u191-linux-x64.rpm
Loaded plugins: fastestmirror
Setting up Install Process
Examining jdk-8u191-linux-x64.rpm: 2000:jdk1.8-1.8.0_191-fcs.x86_64
Marking jdk-8u191-linux-x64.rpm to be installed
Loading mirror speeds from cached hostfile
 * base: centos-distro.cavecreek.net
 * extras: centos-distro.cavecreek.net
 * updates: centos-distro.cavecreek.net
base                                                                    | 3.7 kB     00:00
extras                                                                  | 3.4 kB     00:00
updates                                                                 | 3.4 kB     00:00
Resolving Dependencies
--> Running transaction check
---> Package jdk1.8.x86_64 2000:1.8.0_191-fcs will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package           Arch              Version                   Repository      Size
================================================================================
Installing:
 jdk1.8            x86_64            2000:1.8.0_191-fcs        /jdk-8u191-linux-x64  288 M

Transaction Summary
================================================================================
Install       1 Package(s)

Total size: 288 M
Installed size: 288 M
Downloading Packages:
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing : 2000:jdk1.8-1.8.0_191-fcs.x86_64                             1/1
Unpacking JAR files...
        tools.jar...
        plugin.jar...
        javaws.jar...
```

7. Install elasticsearch just the same
    a. From elastic: “We sign all of our packages with the Elasticsearch Signing Key (PGP key D88E42B4, available from https://pgp.mit.edu) with fingerprint: 4609 5ACC 8548 582C 1A26 99A9 D27D 666C D88E 42B4”
    b. First we can import the elastic keys
        i. `$ rpm --import https://artifacts.elastic.co/GPG-KEY-elasticsearch`
    c. Then we can create a repo from scratch in /etc/yum.repos.d/; the repo should be named elasticsearch.repo and the contents should look like below (copy pasta should work with echo)

```
[elasticsearch-6.x]
```

```
name=Elasticsearch repository for 6.x packages
baseurl=https://artifacts.elastic.co/packages/6.x/yum
gpgcheck=1
gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch
enabled=1
autorefresh=1
type=rpm-md
```

d. Then you can run a yum to install elasticsearch. If successful your output should look like that below.

   i. `$ yum -y install elasticsearch.`

```
[root@elk2 elk-install]# echo "[elasticsearch-6.x]
> name=Elasticsearch repository for 6.x packages
> baseurl=https://artifacts.elastic.co/packages/6.x/yum
> gpgcheck=1
> gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch
> enabled=1
> autorefresh=1
> type=rpm-md
> " > /etc/yum.repos.d/elasticsearch.repo
[root@elk2 elk-install]# yum install elasticsearch
Loaded plugins: fastestmirror
Setting up Install Process
Loading mirror speeds from cached hostfile
 * base: centos-distro.cavecreek.net
 * extras: centos-distro.cavecreek.net
 * updates: centos-distro.cavecreek.net
elasticsearch-6.x                                                | 1.3 kB     00:00
elasticsearch-6.x/primary                                        | 133 kB     00:00
elasticsearch-6.x                                                              337/337
Resolving Dependencies
--> Running transaction check
---> Package elasticsearch.noarch 0:6.5.2-1 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package                Arch                Version                Repository                      Size
================================================================================
Installing:
 elasticsearch          noarch              6.5.2-1                elasticsearch-6.x               108 M

Transaction Summary
================================================================================
Install       1 Package(s)

Total download size: 108 M
Installed size: 160 M
Is this ok [y/N]: y
Downloading Packages:
elasticsearch-6.5.2.rpm                  29% [=========                ] 3.1 MB/s |  32 MB     00:24 ETA
```

e. Now that elasticsearch is installed we need to set it to automatically start; since were on our GUEST which is centos 6 we'll use chkconfig (centos 7 users systemctl enable). If the command below is successful you won't see any output.

1. `$ chkconfig --add elasticsearch`

```
[root@elk2 elk-install]# chkconfig --add elasticsearch
[root@elk2 elk-install]#
```

f. Now we can fire up this baby

i. `$ service elasticsearch start`

g. And check that it's working by curling localhost:9200 (elasticsearch's default port)

i. `$ curl localhost:9200`

h. If successful, your output should look like that below.

```
[root@elk2 elk-install]# service elasticsearch start
Starting elasticsearch:                                    [  OK  ]
[root@elk2 elk-install]# curl localhost:9200
curl: (7) couldn't connect to host
[root@elk2 elk-install]# curl 127.0.0.1:9200
{
  "name" : "GrMSZUi",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "jyomKyJ8QRG-5s-M-8Bp5g",
  "version" : {
    "number" : "6.5.2",
    "build_flavor" : "default",
    "build_type" : "rpm",
    "build_hash" : "9434bed",
    "build_date" : "2018-11-29T23:58:20.891072Z",
    "build_snapshot" : false,
    "lucene_version" : "7.5.0",
    "minimum_wire_compatibility_version" : "5.6.0",
    "minimum_index_compatibility_version" : "5.0.0"
  },
  "tagline" : "You Know, for Search"
}
```

i. Next thing we're going to do is modify the config; find the line in /etc/elasticsearch/elasticsearch.yml where network.host is declared; we're going to uncomment it and set it equal to: 0.0.0.0 so we can reach it from our HOST or LAN; when we make this change we also need to explicitly declare transport.host, transport.tcp.port, and http.port, otherwise elasticsearch will break when we try to start it again. Below is how it should look

```
#
# Set the bind address to a specific IP (IPv4 or IPv6):
#
network.host: 0.0.0.0
#
#
#  JMO ADDED BELOW FOR TROUBLESHOOTING ATTEMPT
#  FROM: https://discuss.elastic.co/t/elasticsearch-network-bind-for-public-private/83644/3
#   - Note that with the below network.host 0000 is used
transport.host: localhost
transport.tcp.port: 9300
http.port: 9200
#
```

j. Now we can restart elasticsearch and curl again to test it.
   i. `$ service elasticsearch restart`

k. Now you should be able to curl from HOST to the GUEST vm
   i. In the first command i'm showing my HOST ip 0.3, and my virtual NIC acting as a gateway at 122.1/24; elk2 is in my /etc/hosts file so this is just curling 192.168.122.62:9200 and we're good to go.

```
[user@centos-lappy ~]$ ip a | grep inet | grep 192
    inet 192.168.0.3/24 brd 192.168.0.255 scope global noprefixroute dynamic wlp3s0
    inet 192.168.122.1/24 brd 192.168.122.255 scope global virbr0
[user@centos-lappy ~]$ curl elk2:9200
{
  "name" : "GrMSZUi",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "jyomKyJ8QRG-5s-M-8Bp5g",
  "version" : {
    "number" : "6.5.2",
    "build_flavor" : "default",
    "build_type" : "rpm",
    "build_hash" : "9434bed",
    "build_date" : "2018-11-29T23:58:20.891072Z",
    "build_snapshot" : false,
    "lucene_version" : "7.5.0",
    "minimum_wire_compatibility_version" : "5.6.0",
    "minimum_index_compatibility_version" : "5.0.0"
  },
  "tagline" : "You Know, for Search"
}
[user@centos-lappy ~]$
```

l. That should wrap up most of the config we need to do for elasticsearch.

8. Now we can install Kibana
   a. We've already imported the elastic GPG keys so all we have to do is create /etc/yum.repos.d/kibana.repo with the following contents

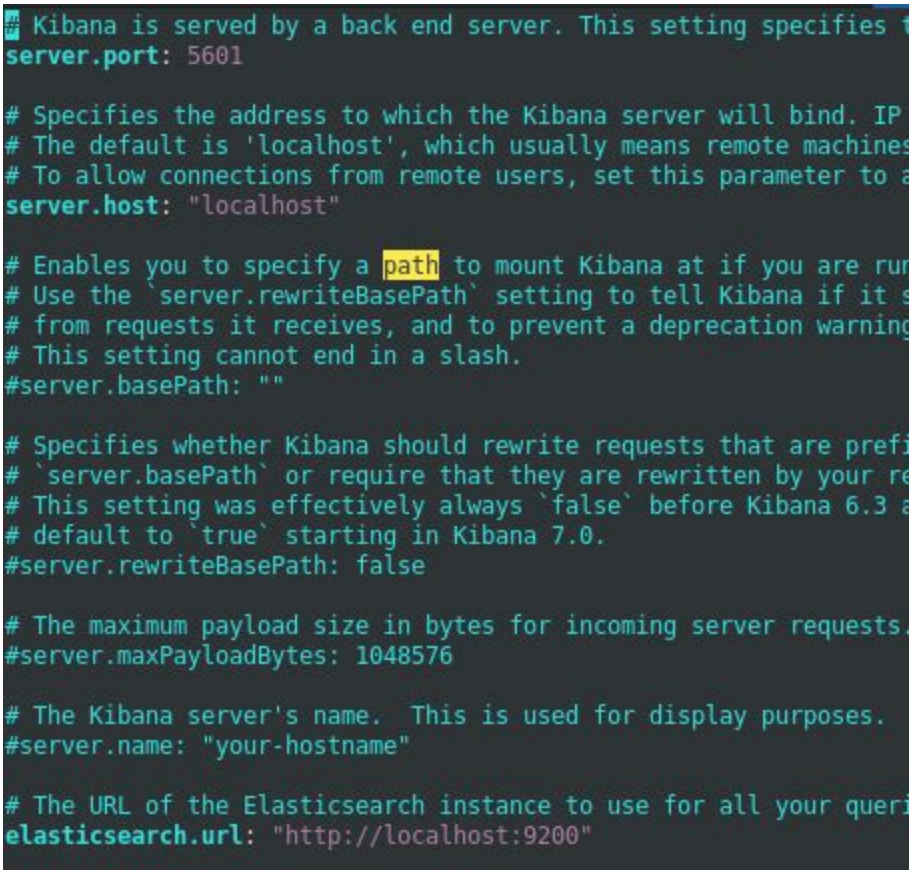
```
[kibana-6.x]
name=Kibana repository for 6.x packages
baseurl=https://artifacts.elastic.co/packages/6.x/yum
gpgcheck=1
gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch
enabled=1
autorefresh=1
type=rpm-md
```

b. Then we can yum it
   i. $ yum -y install kibana

```
[root@elk2 ~]# echo "[kibana-6.x]
> name=Kibana repository for 6.x packages
> baseurl=https://artifacts.elastic.co/packages/6.x/yum
> gpgcheck=1
> gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch
> enabled=1
> autorefresh=1
> type=rpm-md
> " > /etc/yum.repos.d/kibana.repo
[root@elk2 ~]# yum -y install kibana
Loaded plugins: fastestmirror
Setting up Install Process
Loading mirror speeds from cached hostfile
 * base: centos-distro.cavecreek.net
 * extras: centos-distro.cavecreek.net
 * updates: centos-distro.cavecreek.net
kibana-6.x                                                  | 1.3 kB     00:00
kibana-6.x/primary                                          | 133 kB     00:00
kibana-6.x                                                    337/337
Resolving Dependencies
--> Running transaction check
---> Package kibana.x86_64 0:6.5.2-1 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package              Arch                 Version                Repository                          Size
================================================================================
Installing:
 kibana               x86_64               6.5.2-1                elasticsearch-6.x                  200 M

Transaction Summary
================================================================================
Install       1 Package(s)

Total download size: 200 M
Installed size: 480 M
Downloading Packages:
kibana-6.5.2-x86_64.rpm                  68% [=====================          ] 5.6 MB/s | 138 MB     00:11 ETA
```

c. This package installs the config at /etc/kibana/kibana.yml; you'll really only care about having 3 parts in this config uncommented the values we want are below

i. server.port: 5601
ii. server.host: "localhost"
iii. elasticsearch.url: "http://localhost:9200"

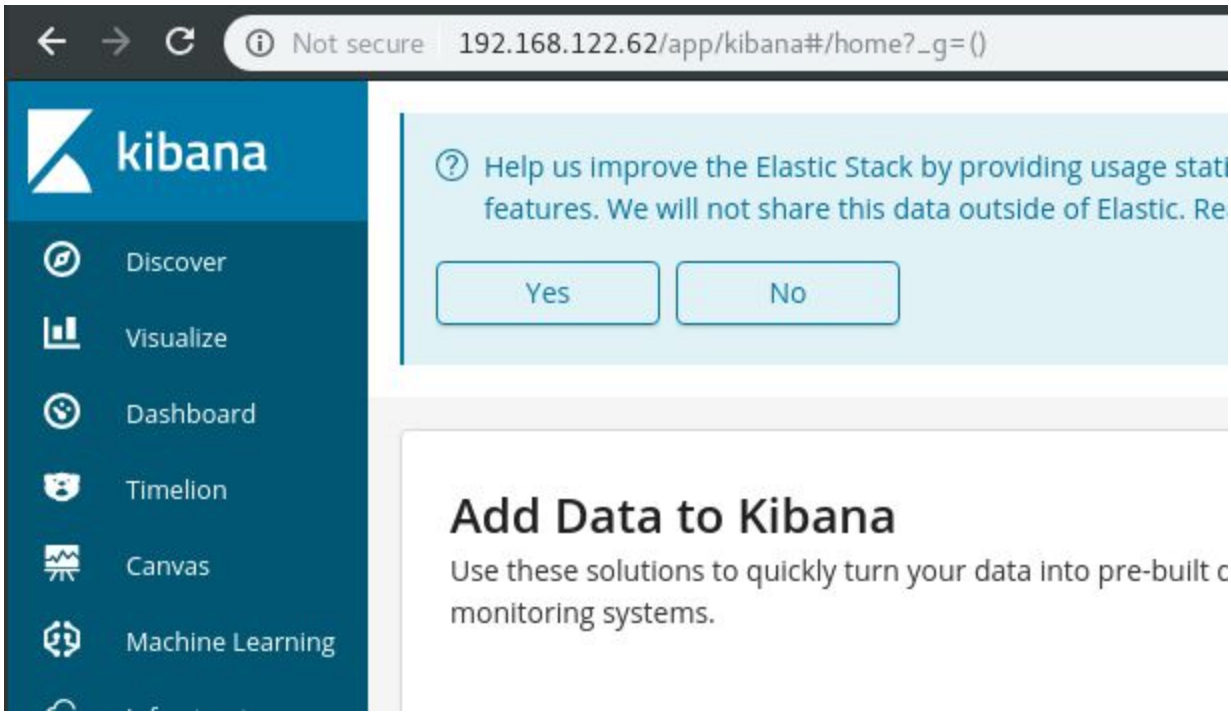


d. Once those 3 sections have been uncommented, the configuration of kibana is essentially done for now.

9. NGINX to serve up Kibana
   a. Nginx isn't available from the default centos repos, we need the epel repo for that so first we install it.
      i. `$ yum -y install epel-release`
   b. Then we need to install 2 packages
      i. `$ yum -y install nginx httpd-tools`
   c. After the packages are installed we'll add the following into /etc/nginx/conf.d/default.conf

```
server {
    listen 80;

    server_name kibana;

    #auth_basic "Restricted Access";
    #auth_basic_user_file /etc/nginx/htpasswd.kibana;
    error_log /var/log/nginx/kibana.error.log;
    access_log /var/log/nginx/kibana.access.log;
    location / {
        proxy_pass http://localhost:5601;
        proxy_http_version 1.1;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header Host $http_host;
    }
}
```

d. The auth_ sections are commented out because we don't care about them for this scope; if you're interested in configuring authentication, go nuts.
e. Now we need to explicitly allow httpd in selinux, otherwise we'll get 502 BAD GATEWAY errors when we try to navigate to our VM's ip in the browser
   i. `$ setsebool -P httpd_can_network_connect true`
f. Now if we start Kibana and start NGINX we should be able to see kibana when we navigate to elk1's ip in a browser
   i. `$ service kibana start`
   ii. `$ service nginx start`
   iii. If you see the below, you're good to go.



g. Feel free to poke around the gui a bit… but it'll be pretty boring without any data.

10. Installing logstash
    a. Logstash is where you'll really end up spending a lot of your time; logstash configurations can allow you to create input / filter / output rules for incoming data that then gets shipped onward to elasticsearch.
    b. As with the previous two, we'll create another repo, this time /etc/yum.repos.d/logstash.repo

```
[logstash-6.x]
name=Elastic repository for 6.x packages
baseurl=https://artifacts.elastic.co/packages/6.x/yum
gpgcheck=1
gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch
enabled=1
autorefresh=1
type=rpm-md
```

    c. Once installed the configuration is in /etc/logstash/conf.d/
        i. You should take some time to research configs in depth; they can get very complex; one way to organize them is numerically into different input / output / filter sections, similar to the structure below

```
[root@elk1 ~]# ll /etc/logstash/conf.d/
total 20
-rw-r--r--. 1 logstash root   64 Nov 17 20:46 0-beats-input.conf
-rw-r--r--. 1 logstash root  277 Nov 22 07:52 1-rysyslog-input.conf
-rw-r--r--  1 logstash root  208 Dec  2 15:43 2-var-log-secure.conf
-rw-r--r--. 1 logstash root 3526 Nov 27 07:45 3-filters-testing.conf
-rw-r--r--. 1 logstash root  744 Nov 27 19:19 4-output-to-elasticsearch-localhost.conf
[root@elk1 ~]#
```

d. We'll cover a basic config that will ingest data from you system's messages log which you can then see in kibana. Below is the sample config
   i. NOTE: IF YOU PASTE THIS CONFIG YOU HAVE TO CHANGE THE QUOTES OTHERWISE THE LOGSTASH CONFIG FILE WILL THROW ERRORS

```
input {
    file {
      path => "/var/log/messages"
      tags => [“local.log.messages"]
    }
}
output {
    if “local.log.messages” in [tags] {
        elasticsearch {
            action => “index”
            hosts => “localhost:9200”
            index => “local-messages-%{+YYYY.MM.dd}”
        }
    }
}
```

e. Save this file in the GUEST path listed above.
f. Now we're going to edit /etc/logstash/logstash.yml and allow the auto-reloading of our config file; this is helpful when making changes on the fly because you won't have to start/stop anything, logstash will just poll the config every X seconds and reload it and you can watch the log messages live in /var/log/logstash/logstash-plain.log. The relevant sections should look like the example below

```
# This can also be triggered manually through the SIGHUP signal
#
config.reload.automatic: true
#
# How often to check if the pipeline configuration has changed (in seconds)
#
config.reload.interval: 3s
#
# Show fully compiled configuration as debug log message
```

g. start logstash (logstash uses initctl to start; annoying but don't forget it)
   i. `$ initctl start logstash`

```
[root@elk2 ~]# initctl start logstash
logstash start/running, process 15945
[root@elk2 ~]#
```

h. Logstash is pretty slow to start and stop so it could be a while; you can attempt to see what it's doing bay using tailing some of it's log files
   i. `$ tail -f /var/log/logstash-stderr.log`
   ii. `$ tail -f /var/log/logstash-stdout.log`

```
[root@elk2 elk-install]# tail -f /var/log/logstash-stdout.log
Sending Logstash logs to /var/log/logstash which is now configured via log4j2.properties
[2018-12-07T20:42:56,400][INFO ][logstash.setting.writabledirectory] Creating directory {:setting=>"path.queue"
 :path=>"/var/lib/logstash/queue"}
[2018-12-07T20:42:56,516][INFO ][logstash.setting.writabledirectory] Creating directory {:setting=>"path.dead_l
tter_queue", :path=>"/var/lib/logstash/dead_letter_queue"}
[2018-12-07T20:42:57,663][INFO ][logstash.runner          ] Starting Logstash {"logstash.version"=>"6.5.2"}
[2018-12-07T20:42:57,725][INFO ][logstash.agent           ] No persistent UUID file found. Generating new UUID
:uuid=>"79a6fef6-534d-4643-8007-ee165da198d8", :path=>"/var/lib/logstash/uuid"}
[2018-12-07T20:43:00,711][ERROR][logstash.agent           ] Failed to execute action {:action=>LogStash::Pipeli
eAction::Create/pipeline_id:main, :exception=>"LogStash::ConfigurationError", :message=>"Expected one of #, \",
', -, [, {, ] at line 4, column 11 (byte 61) after input { \n    file { \n\tpath => \"/var/log/messages\"\n\tta
s => [", :backtrace=>["/usr/share/logstash/logstash-core/lib/logstash/compiler.rb:41:in `compile_imperative'",
/usr/share/logstash/logstash-core/lib/logstash/compiler.rb:49:in `compile_graph'", "/usr/share/logstash/logstas
-core/lib/logstash/compiler.rb:11:in `block in compile_sources'", "org/jruby/RubyArray.java:2486:in `map'", "/u
r/share/logstash/logstash-core/lib/logstash/compiler.rb:10:in `compile_sources'", "org/logstash/execution/Abstr
ctPipelineExt.java:149:in `initialize'", "/usr/share/logstash/logstash-core/lib/logstash/pipeline.rb:22:in `ini
ialize'", "/usr/share/logstash/logstash-core/lib/logstash/pipeline.rb:90:in `initialize'", "/usr/share/logstash
logstash-core/lib/logstash/pipeline_action/create.rb:42:in `block in execute'", "/usr/share/logstash/logstash-c
re/lib/logstash/agent.rb:92:in `block in exclusive'", "org/jruby/ext/thread/Mutex.java:148:in `synchronize'", "
usr/share/logstash/logstash-core/lib/logstash/agent.rb:92:in `exclusive'", "/usr/share/logstash/logstash-core/l
b/logstash/pipeline_action/create.rb:38:in `execute'", "/usr/share/logstash/logstash-core/lib/logstash/agent.rb
317:in `block in converge_state'"]}
[2018-12-07T20:43:01,458][INFO ][logstash.agent           ] Successfully started Logstash API endpoint {:port=>
600}
```

i. One thing you'll notice is that logstash shits out a ton of information; this can be difficult to comb through when trying to troubleshoot but if you look closely at the output above you'll notice that while the logstash API successfully started, my config actually has an error at line 4, column 11
   i. This error was related to the quotes; they weren't encoded correctly when they were pasted and caused errors.

j. When i fix this i start seeing another error
   i. `$ tail -f /var/log/logstash/logstash-plain.log`

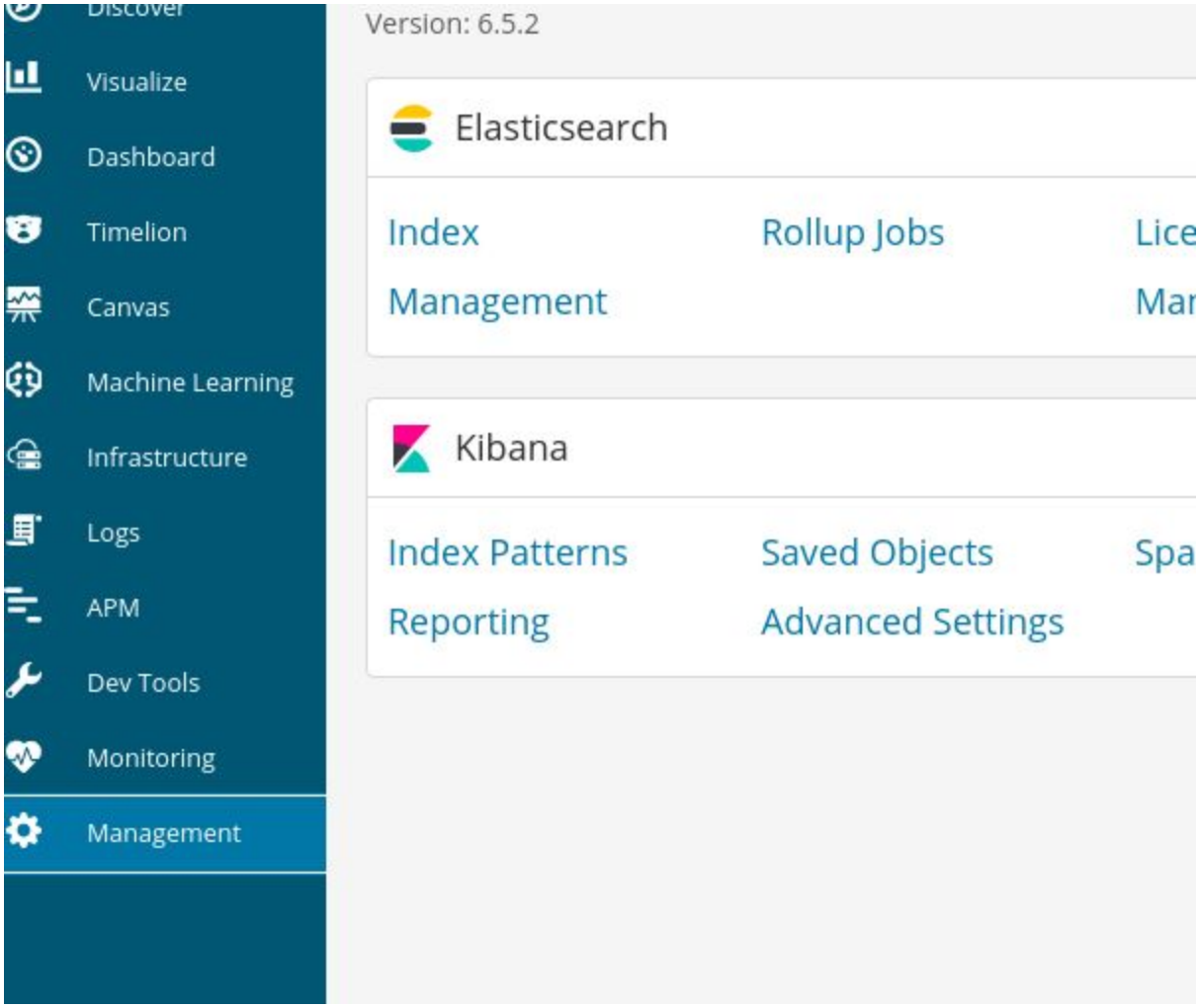
```
[2018-12-07T21:06:22,466][WARN ][filewatch.tailmode.handlers.createinitial] open_file OPEN_WARN_INTERVAL is '300
'
[2018-12-07T21:06:23,469][WARN ][filewatch.tailmode.handlers.createinitial] open_file OPEN_WARN_INTERVAL is '300
'
[2018-12-07T21:06:24,471][WARN ][filewatch.tailmode.handlers.createinitial] open_file OPEN_WARN_INTERVAL is '300
'
```

ii. The above error is related to permissions; i need to modify /var/log/messages to be readable by elasticsearch; as it is it has 600 permissions and it's owned by root which means only root can read it, it needs to have 644 permissions, meaning it can be read by anyone but only written to by root so we'll modify it

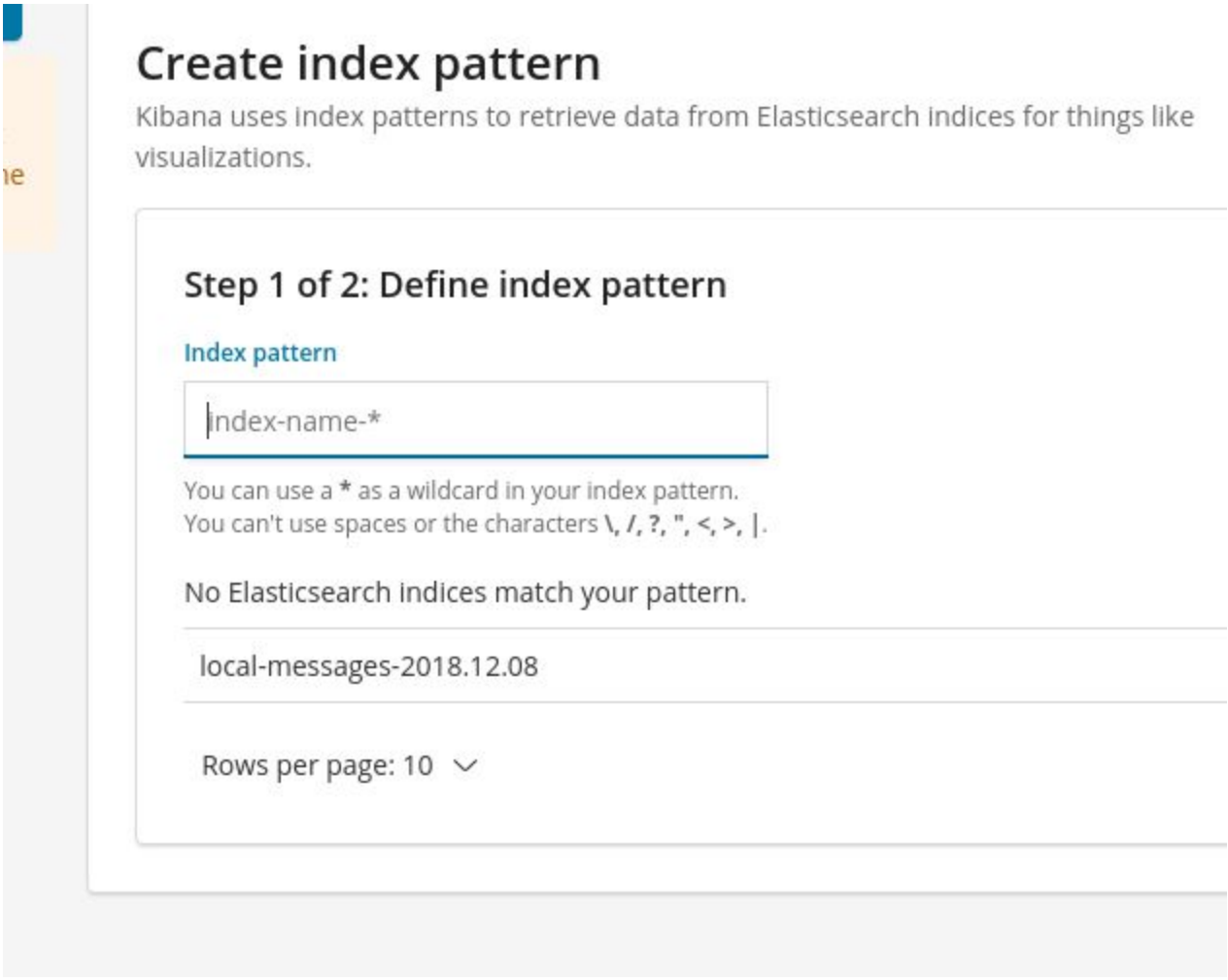
```
1. $ chmod 644 /var/log/messages
```

iii. Now you should check kibana; if you're not seeing data there you may need to restart

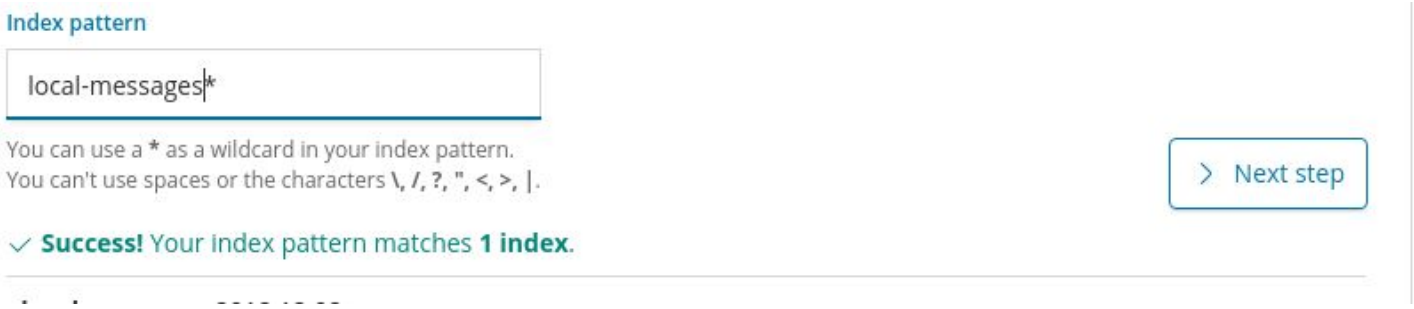
11. Index management
    a. From the Management menu in kibana we can choose Index Patterns



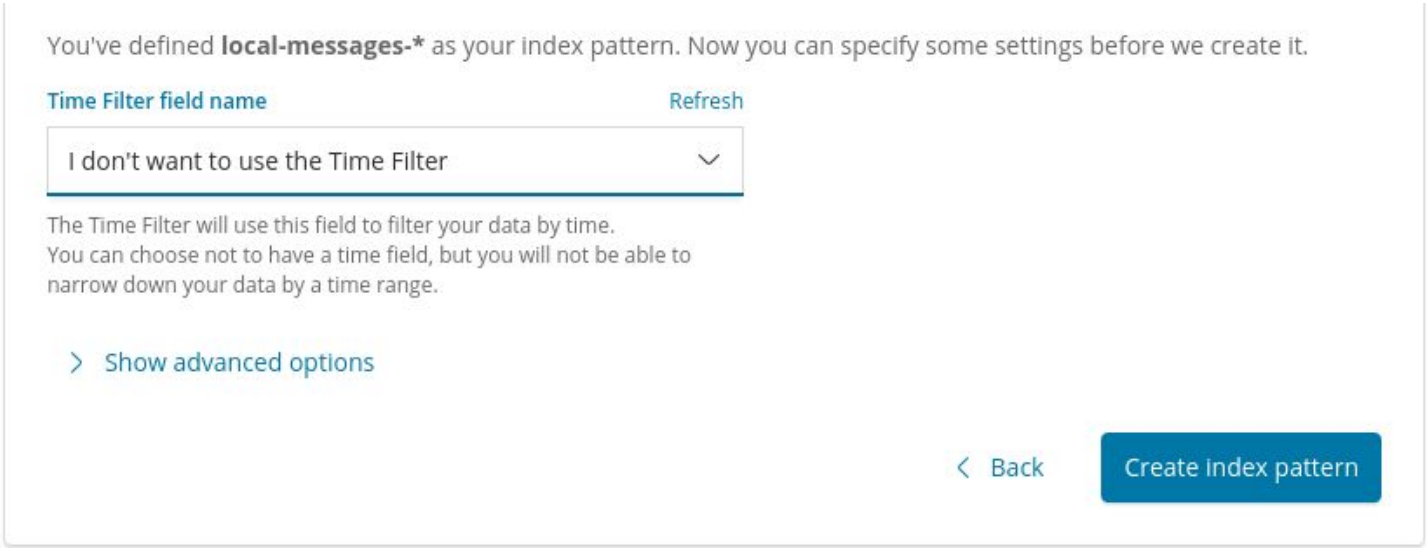
    b. Index patterns are used to retrieve data from elasticsearch for viewing. We create an index pattern by typing in a string that matches index names; the convention is to use a name followed by a string for indices, that way we can order things by name, and then secondly by date..



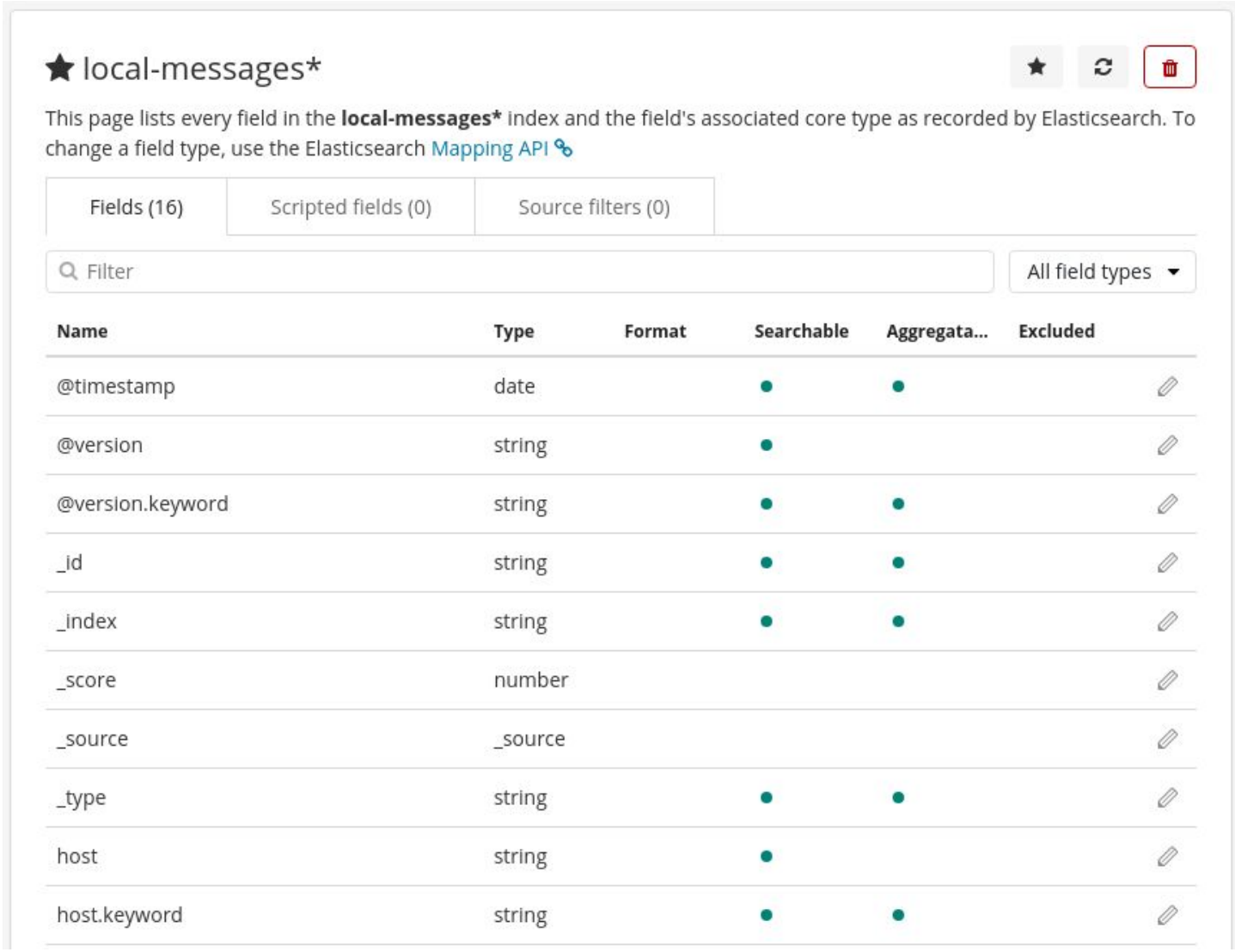
c. Since we only have one index right now, we'll create the pattern "local-messages*" then hit next



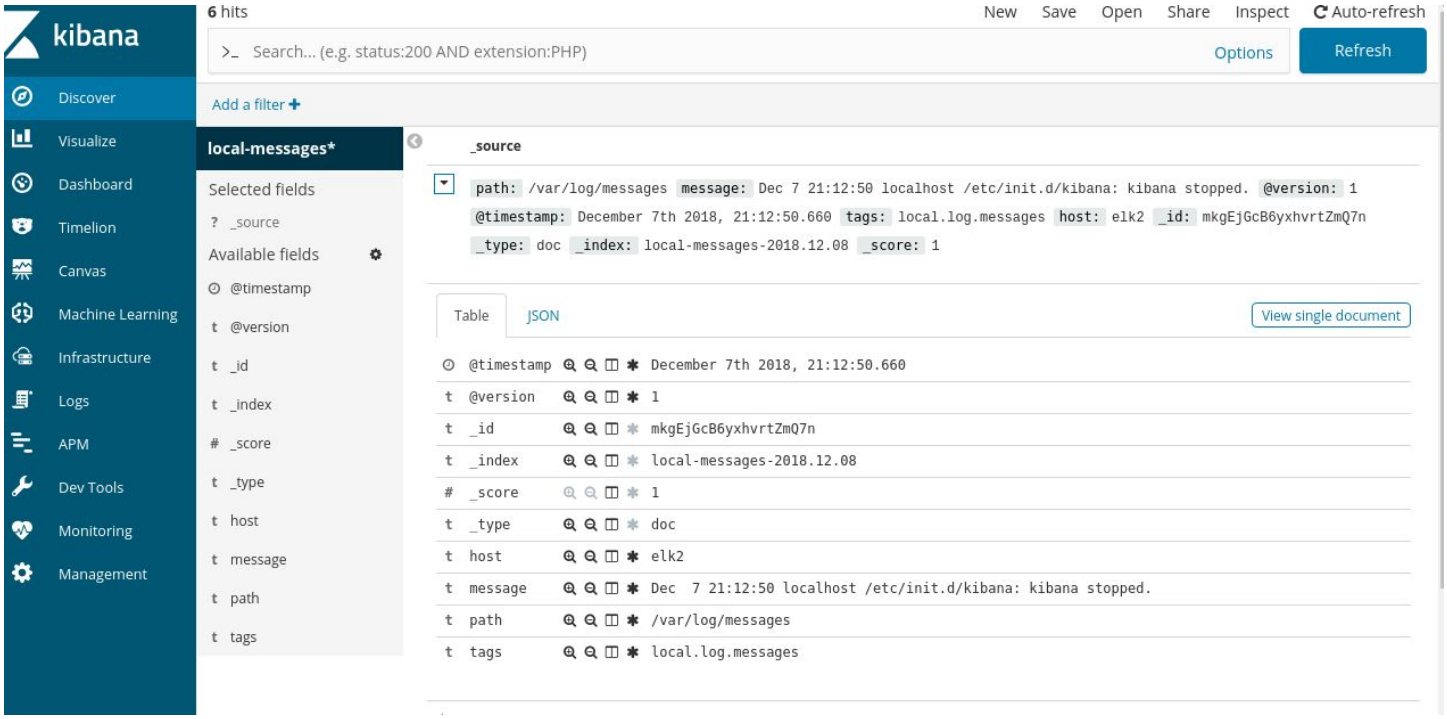
d. This will take you to menu where you can specify the "time filter field name"; it's easy to be confused keeping track of where your time fields are coming from at first; choose "I don't want to use the time filter" for the simplest settings.



e. Now click create index pattern and your pattern will be created.



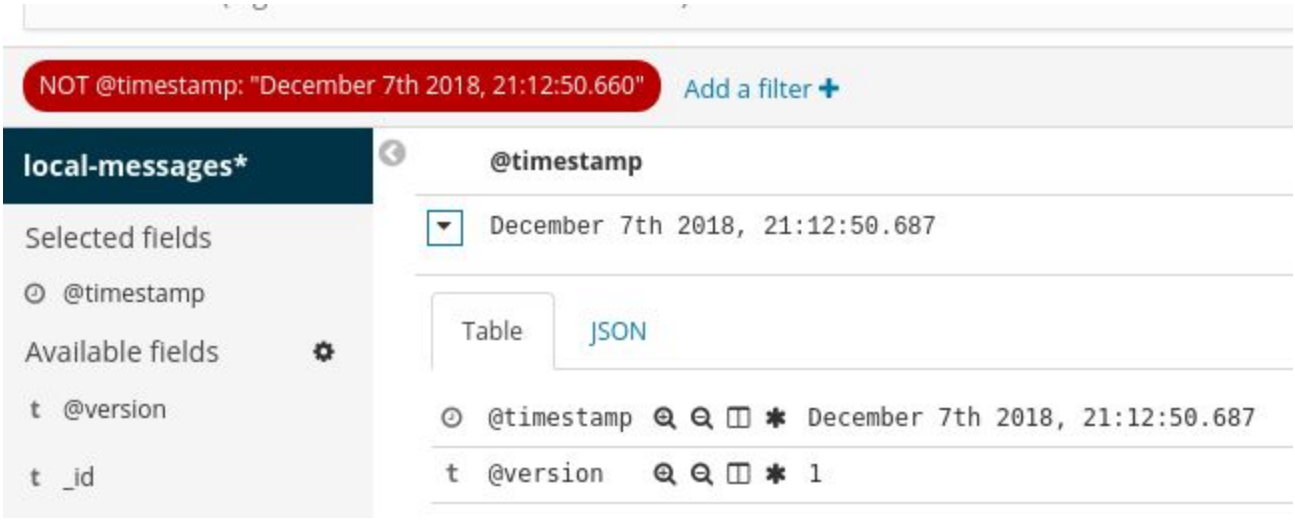
f. You don't really have to do anything with this now; but you can navigate to the discover tab and start looking at your data.



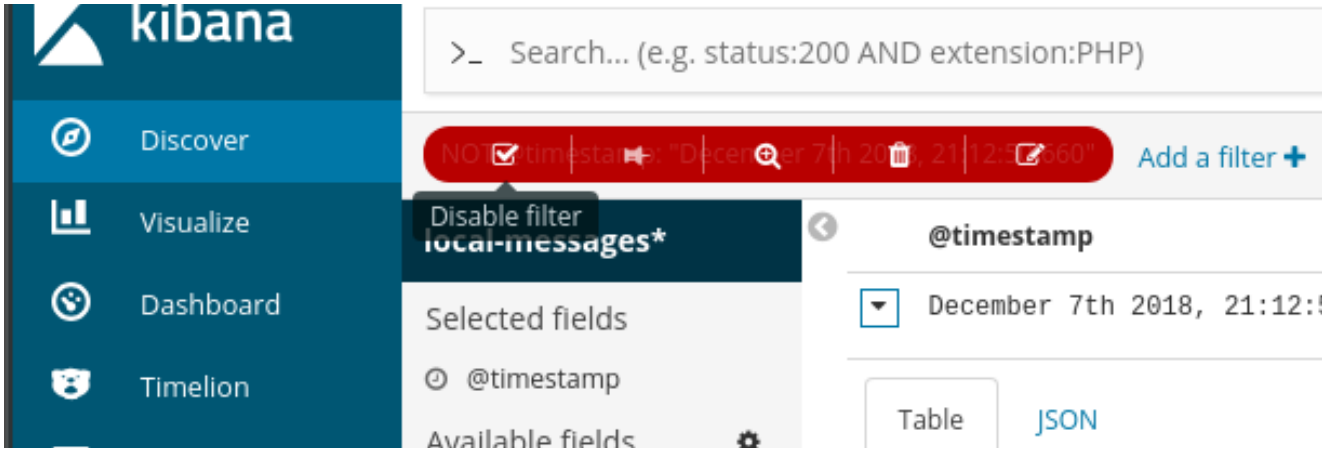
g. Each record you see is called a "document" in elastic terms, you can view this in table mode or in JSON mode by toggling the tabs on an expanded document

h. When viewing document fields in Table mode you'll notice 3 icons

i. These will be come important when you start sifting around through your data; the + sign will FILTER FOR a value, the - will FILTER OUT a given value. The little square on the far right will add a field as a column;

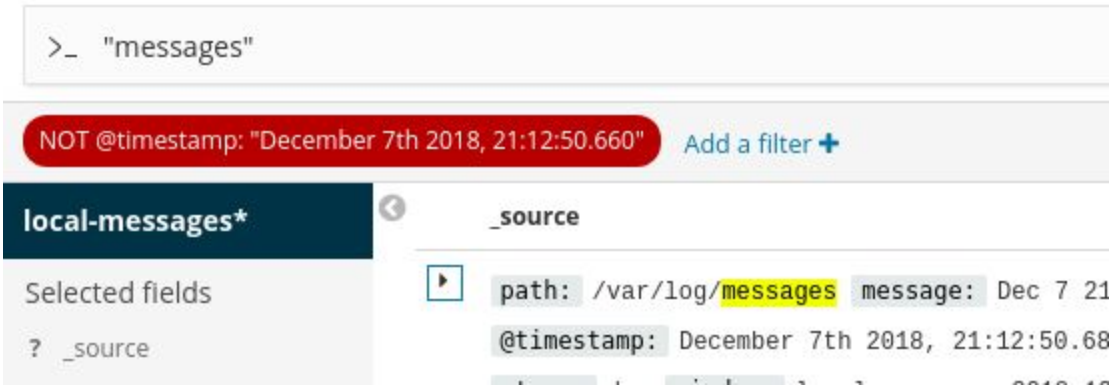


j. Above we've clicked the column button on the timestamp field to add it as a column, and we clicked the minus sign on the timestamp field of the first record we saw, and it added a big red oval at the top so we know we've filtered something out

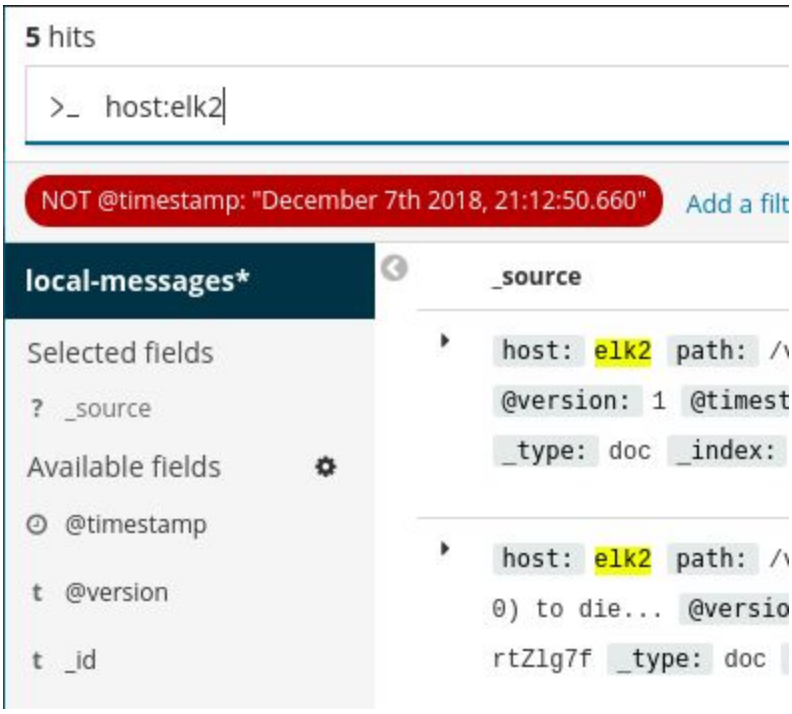


k. If you hover over this field you'll notice there are several options; we can disable the filter, pin it, flip the logic of our filter on it's head, delete it, or edit it.

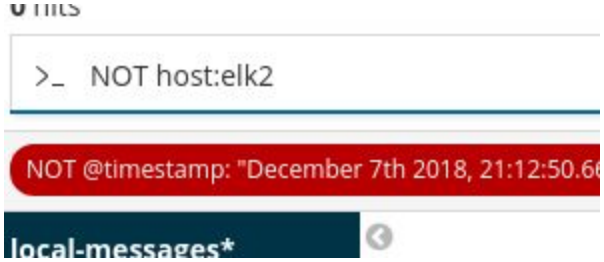
l. In our search box up top we can type in various queries. The most basic is a keyword search



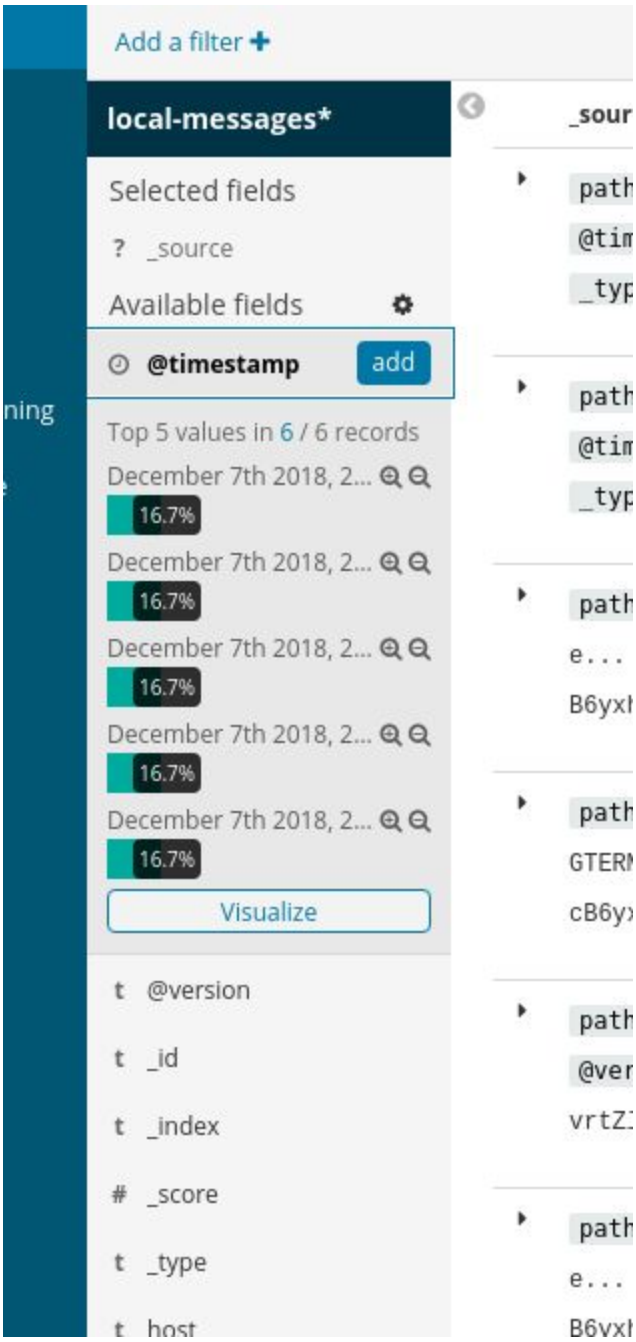
m. Optionally we can search for a specific value of a field, for example we can search for all documents where host == elk2



n. Logical operators must be specified in all caps



o. If we click on the field names in the left hand column, they'll expand and show you the top values of that field; this can be helpful when investigating as it gives you a quick idea of what most of your values are and what you can ignore or what you want to pay more attention to.



12. That pretty much wraps up the setup; wasn't that painful was it?

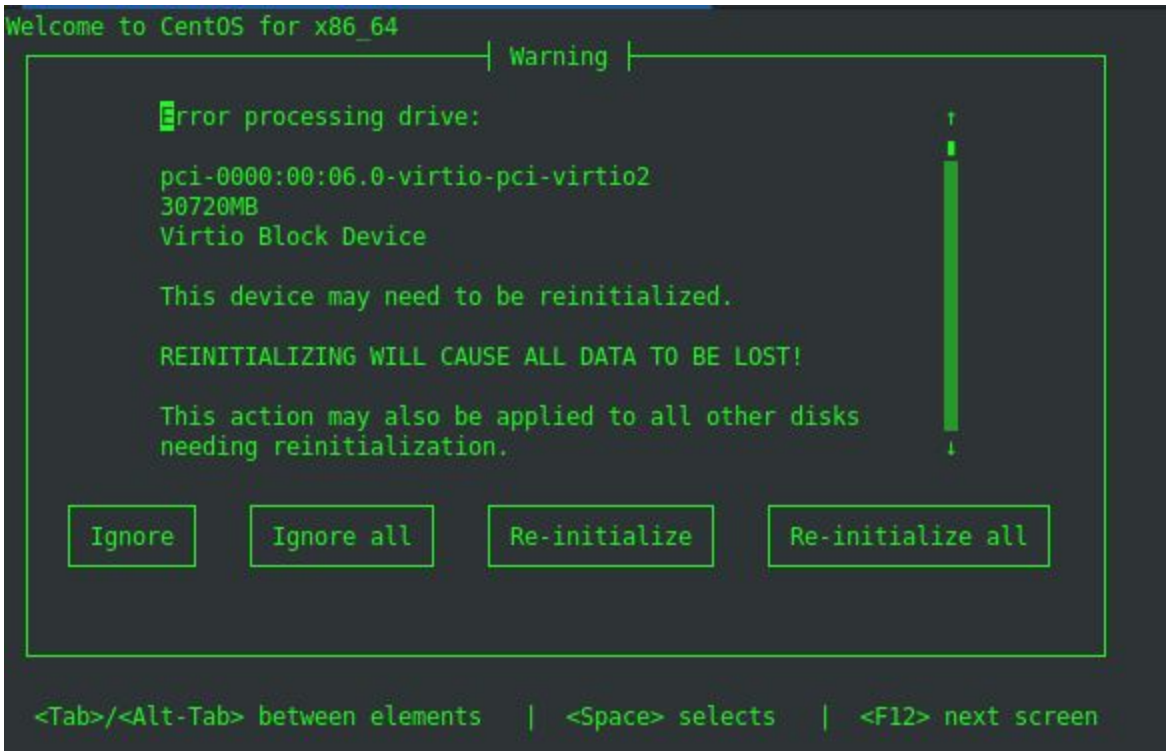
Appendix A: Troubleshooting

1. **Error: bash: virt-install: command not found...**

a. Most of the time when you encounter a "command not found" error from bash it just means you need to install another package, in the screenshot below i use "yum provides" to find out which package I need to install so i have the binary "virt-install"

b. `$ yum provides */AnyMissingBinaryNameHere`

2. Error: Error processing drive: pci XXXXX-virtio-pci-virtio2; deice may need to be reinitialized



a. Just select re-initialize all and ignore

3. ERROR: No IP address assigned to VM automatically

a. This could be any number of factors. The first thing you should do is try to restart some services.

b. First, restart libvirtd on the HOST and then check if you have an IP

i. `$ service libvirtd restart`

ii. `$ virsh console VmNameHere`

iii. `(guest) $ ip a`

1. To see if you were automatically assigned an IP

iv. `(guest) $ service network restart`

1. If everything went well you should be assigned an IP address

```
[root@localhost ~]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qle
    link/ether 52:54:00:06:cc:11 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::5054:ff:fe06:cc11/64 scope link
       valid_lft forever preferred_lft forever
[root@localhost ~]# whoami
root
[root@localhost ~]# pwd
/root
[root@localhost ~]# service network restart
Shutting down interface eth0:  [  OK  ]
Shutting down loopback interface:  [  OK  ]
Bringing up loopback interface:  lo: Disabled Privacy Extensions
[  OK  ]
Bringing up interface eth0:
Determining IP information for eth0... done.
[  OK  ]
[root@localhost ~]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qle
    link/ether 52:54:00:06:cc:11 brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.62/24 brd 192.168.122.255 scope global eth0
    inet6 fe80::5054:ff:fe06:cc11/64 scope link
       valid_lft forever preferred_lft forever
```

v. If not, try restarting networking on the HOST machine and repeat the steps above on the guest.

Appendix B - Extra Bits

1. TODO[Double check these settings] If you want multiple devices on LAN to be able to connect to your VM; i.e. advanced install.
    a. NOTE: If you're doing this you may want to also look into changing the default virtual network IP address, otherwise if you're setting up different VMs on different servers on your LAN then shit will get confusing real quick; this presents an additional learning opportunity but covering it here is beyond the scope of this document.
    b. This requires some additional steps because we have to tell the HOST firewall to do some routing for us..
    c. First we enable port forwarding on the HOST

i. $ sysctl -w net.ipv4.ip_forward=1

d. Then we make this persistent by adding net.ipv4.ip_forward=1 to /etc/sysctl.conf on the HOST
   i. $ echo “net.ipv4.ip_forward=1” >> /etc/sysctl.conf

```
[user@centos-lappy ~]$ sudo sysctl -w net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
[user@centos-lappy ~]$ echo "net.ipv4.ip_forward=1" >> /etc/sysct.conf
```

e. Next we want to tell the firewall to take anything destined for our virtual network 192.168.122.0/24 and accept it
   i. `$ iptables -I FORWARD -m state -d 192.168.122.0/24 --state NEW,RELATED,ESTABLISHED -j ACCEPT`

f. Now we want to forward anything headed from LAN to GUEST VM and accept it
   i. `$ iptables -I FORWARD -s 192.168.0.0/24 -d 192.168.122.0/24 -j ACCEPT`

Sources

1. https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/virtualization_deployment_and_administration_guide/sect-troubleshooting-common_libvirt_errors_and_troubleshooting
2. Mozilla’s recommended SSH configurations
   a. https://infosec.mozilla.org/guidelines/openssh.html