Command line

Be a command line hero.
Configuration tips, tricks, and things I often forget.

    - Unlock the ESXi host account

- Windows
    - Active Directory
    - Windows Management Consoles
    - Windows Install Media

Apple macOS

Things I forget the macOS can do. Because you know, it's a UNIX.

Apple macOS

PowerShell on macOS

Wait. What? PowerShell on macOS? Here's how.

Brew .NET and PowerShell

Don't have homebrew? Get it at brew.sh

```
brew cask install dotnet
```

Update OpenSSL

```
brew install openssl
```

```
ln -s /usr/local/opt/openssl/lib/libcrypto.1.0.0.dylib /usr/local/lib/
```

```
ln -s /usr/local/opt/openssl/lib/libssl.1.0.0.dylib /usr/local/lib/
```

```
brew cask install powershell
```

Install Azure PowerShell modules

To download and install the Azure PowerShell module package use the following PowerShell code:

```
Install-Package -Name Az
```

Test if the installation was successful with the cmdlet

```
Import-Module Az | Login-AzAccount
```

You should be prompted to go to https://microsoft.com/devicelogin to enter the provided code and your Azure admin credentials.

See more about the Az module here.

Linux

Mostly Red Hat and it's clones.

Linux

CentOS 7 GSSAPI module

The GSSPI module has been built as a replacement for the aging mod_auth_kerb. Its aim is to use only GSSAPI calls and be as much as possible agnostic of the actual mechanism used.

Installing packages

```
yum install -y epel-release
yum install -y krb5-workstation krb5-devel krb5-libs mod_auth_gssapi mod_session
```

Prepare a `/etc/krb5.conf` against the AD environment

```
includedir /etc/krb5.conf.d/

[logging]
 default = FILE:/var/log/krb5libs.log
 kdc = FILE:/var/log/krb5kdc.log
 admin_server = FILE:/var/log/kadmind.log

[libdefaults]
  default_realm = SAMPLE.COM
  default_tkt_enctypes = aes256-cts-hmac-sha1-96 aes256-cts aes128-cts-hmac-sha1-96 aes128-cts rc4-hmac
  default_tgs_enctypes = aes256-cts-hmac-sha1-96 aes256-cts aes128-cts-hmac-sha1-96 aes128-cts rc4-hmac
  permitted_enctypes = aes256-cts-hmac-sha1-96 aes256-cts aes128-cts-hmac-sha1-96 caes128-cts rc4-hmac
  forwardable = true
  dns_lookup_realm = false
  dns_lookup_kdc = false

[realms]
  SAMPLE.COM  = {
    kdc = domainc1.sample.com
    default_domain = SAMPLE.COM
}

[domain_realm]
  .SAMPLE.COM = SAMPLE.COM
  SAMPLE.COM = SAMPLE.COM
```

Check the time. Kerberos is extremely time sensitive.

```
ntpdate domainc1.sample.com
```

Test the login

```
kinit Administrator@SAMPLE.COM
klist
kdestroy
```

Linux

CentOS Linux 8 to CentOS Stream

Step 1: Enable CentOS Stream Repo

```
dnf install centos-release-stream
```

Step 2: Set CentOS Stream repo as the default

```
dnf swap centos-{linux,stream}-repos
```

Step 3: Synchronize installed packages to the latest versions

```
dnf distro-sync
```

When complete, reboot and your CentOS 8 Linux is now CentOS Steam.

Good bye CentOS. I will miss you.

Linux

Custom CentOS ISO

There are many reason to create a custom installer. This example was created to add the console output to the 1st serial port to deploy CentOS on a headless device. This process works for CentOS 7 and 8/Stream.

Before you begin, make sure you have genisoimage and syslinux installed

```
yum install -y genisoimage syslinux
```

1. Download .iso from a CentOS mirror. See https://www.centos.org/download/mirrors/ for a list of mirrors nearest to you.

```
wget http://linux.cc.lehigh.edu/centos/7.6.1810/isos/x86_64/CentOS-7-x86_64-Minimal-1810.iso
```

2. Mount .iso to a directory

```
sudo mount -o loop -t iso9660 CentOS-7-x86_64-Minimal-1810.iso centos_iso
```

3. copy its contents to another working directory:

```
cp -rf centos_iso/ centos_customized_iso
```

4. edit centos_customized_iso/isolinux/isolinux.cfg, overwrite its contents with the following. DO NOT alter the LABEL.

```
default linux
prompt 1
timeout 50

label linux
kernel vmlinuz
append initrd=initrd.img inst.stage2=hd:LABEL=CentOS\x207\x20x86_64 console=tty0
console=ttyS0,115200n8

label text
kernel vmlinuz
append initrd=initrd.img text inst.stage2=hd:LABEL=CentOS\x207\x20x86_64 console=tty0
console=ttyS0,115200n8

label check
```

```
kernel vmlinuz
append initrd=initrd.img inst.stage2=hd:LABEL=CentOS\x207\x20x86_64 rd.live.check quiet
```

5. prepare .iso again from centos_customized_iso directory:
`sudo mkisofs -r -V “CentOS 7 x86_64” -cache-inodes -J -l -b isolinux/isolinux.bin -c isolinux/boot.cat -no-emul-boot -boot-load-size 4 -boot-info-table -o custom.iso centos_customized_iso`

6. run isohybrid on the new .iso (otherwise it won’t boot from USB)
`isohybrid custom.iso`

7. Make sure .iso is proper with a format like this, with “file” command:
`file custom.iso`

The output will be similar to this:

`custom.iso: DOS/MBR boot sector ISO 9660 CD-ROM filesystem data (DOS/MBR boot sector) ‘CentOS 7 x86_64’ (bootable); partition 1 : ID=0x17, active, start-CHS (0x0,0,1), end-CHS (0x2ae,63,32), startsector 0, 1406976 sectors`

8. Burn this to USB with dd command. Be sure you are outputting to the correct device. A mistake here will wipe data or your root.
`dd if=custom.iso of=/dev/sdb bs=1MB`

9. Insert this USB to your machine and boot from USB, and in the boot: prompt, type “linux text” as the install command
`boot: linux text`

To further customize and automate the installation, add a Kickstart to the append.

Linux

LVM with cache

Create a logical volume with nvme cache.

```
vgcreate storage /dev/sdb /dev/nvme0n1
lvcreate -L 6T -n lv storage /dev/sdb
lvcreate -L 900G -n lv_cache storage /dev/nvme0n1
lvcreate -L 9G -n lv_meta storage /dev/nvme0n1

lvconvert --type cache-pool --cachemode writethrough --poolmetadata storage/lv_cache_meta storage/lv_cache

lvconvert --type cache --cachepool storage/lv_cache storage/lv
```

Linux

Move RHEL Users

Step 1, on source

Run the following command**s** as root source) system which has users configured

```
ID_minimum=500
for f in /etc/{passwd,group}; do awk -F: -vID=$ID_minimum '$3>=ID && $1!="nfsnobody"' $f |sort -nt: -k3 >
${f#/etc/}.bak; done
while read line; do grep -w "^${line%%:*}" /etc/shadow; done <passwd.bak >shadow.bak
while read line; do grep -w "^${line%%:*}" /etc/gshadow; done <group.bak >gshadow.bak
```

After running the above, 4 new files will be in the current directory (`passwd.bak`, `group.bak`, `shadow.bak`, and `gshadow.bak`). Inspect them and then transfer to the new *destination* system.

Step 2, on destination

Run the following command as root on the destination system in a directory containing the four `.bak` files.

```
for f in {passwd,group,shadow,gshadow}.bak; do cat $f >>/etc/${f%.bak}; done
```

Step 3, on destination

Run the following **final compound command destination system in the same directory as the previous step**

```
for uidgid in $(cut -d: -f3,4 passwd.bak); do
    dir=$(awk -F: /$uidgid/{print\$6} passwd.bak)
    mkdir -vm700 "$dir"; cp -r /etc/skel/.[[:alpha:]]* "$dir"
    chown -R $uidgid "$dir"; ls -ld "$dir"
done
```

This final command will setup home directories for the users.

Linux

NFS on ZFS HA Cluster

Build a high-available dual-controller storage array using open-source technologies.
This solution should be capable of presenting shared storage to NFS client and can be expanded to iSCSI and FCoE.

What is needed:

- 2 nodes with A LOT of memory.
- Fast storage that is directly attached to each node.
- Root access.
- The ZFS filesystem configured via the ZFS on Linux repository.
- Familiarity with basic ZFS operations: zpool/zfs filesystem creation and modification.
- Understanding of network under RHEL Linux.

These steps describe the construction of a two host, single JBOD cluster that manages a single ZFS pool.

On both nodes:

Get EPEL and ZFS repositories

```
yum install -y epel-release
yum install -y http://download.zfsonlinux.org/epel/zfs-release.el7_8.noarch.rpm
yum update -y
yum install -y kernel-devel zfs
systemctl preset zfs-import-cache zfs-import-scan zfs-mount zfs-share zfs-zed zfs.target
systemctl enable zfs-import-scan
systemctl start zfs-import-scan
```

Add some clustering requirements:

```
yum install -y pcs fence-agents-all device-mapper-multipath nfs-utils
touch /etc/multipath.conf
systemctl start multipathd
systemctl enable multipathd
```

Add each node to the /etc/hosts file.

```
echo "172.16.100.142 node1" >> /etc/hosts
echo "172.16.100.144 node2" >> /etc/hosts
```

Set the hacluster password:

```
passwd hacluster wljgnFAW4fgwEGF21
```

Add needed heartbeat files

```
cd /usr/lib/ocf/resource.d/heartbeat/
wget https://raw.githubusercontent.com/clusterapps/stmf-ha/master/heartbeat/ZFS
wget https://github.com/ClusterLabs/resource-agents/raw/master/heartbeat/iSCSITarget
wget https://github.com/ClusterLabs/resource-agents/raw/master/heartbeat/iSCSILogicalUnit
chmod a+x ./ZFS
chmod a+x ./iSCSILogicalUnit
chmod a+x ./iSCSITarget
```

Update the firewall configuration.

```
firewall-cmd --add-service=nfs --permanent
firewall-cmd --add-service=high-availability --permanent
firewall-cmd --permanent --add-service=nfs
firewall-cmd --permanent --add-service=mountd
firewall-cmd --permanent --add-service=rpc-bind
firewall-cmd --reload
```

Enable the services and reboot. This step helps verify all of the services are properly configured on boot.

```
systemctl enable pcsd
systemctl enable corosync
systemctl enable pacemaker
reboot
```

On the Primary

Create the pool. You will need the path to the devices. Get them from:

```
ls -l /dev/disk/by-id/
```

Run zpool create (Your devices will be different!)
This pool has similar qualities to a RAID5+0. You should build what you need.

```
zpool create array1 -o ashift=12 -o autoexpand=on -o autoreplace=on -o cachefile=none \
raidz1 /dev/disk/by-id/scsi-35000c500740a6277 /dev/disk/by-id/scsi-35000c5007411d36b /dev/disk/by-id/scsi-35000c5007411cb1b \
raidz1 /dev/disk/by-id/scsi-35000c50076703a9f /dev/disk/by-id/scsi-35000c50070d3a853 /dev/disk/by-id/scsi-35000c50076701f57 \
raidz1 /dev/disk/by-id/scsi-35000c5007411cfa7 /dev/disk/by-id/scsi-35000c5007411d0bf /dev/disk/by-id/scsi-35000c500740a109f \
log mirror /dev/disk/by-id/scsi-35000c5007670107f /dev/disk/by-id/scsi-35000c5007411d467 spare /dev/disk/by-id/scsi-35000c50076bd0c03
```

Update a few ZFS settings

```
zfs set acltype=posixacl array1
zfs set atime=off array1
zfs set xattr=sa array1
zfs set compression=lz4 array1
```

Authorize Cluster

```
pcs cluster auth node1 node2
pcs cluster setup --start --name NASOne node1 node2
```

Set some cluster properties and ad the resources

```
pcs property set no-quorum-policy=ignore
pcs stonith create fence-array1 fence_scsi pcmk_monitor_action="metadata" pcmk_host_list="node1 node2" \
devices="/dev/mapper/35000c5007670107f,/dev/mapper/35000c5007411d467,/dev/mapper/35000c50076bd0c03" \
meta provides=unfencing --group=group-array1
pcs resource create array1-ip IPaddr2 ip=172.16.100.99 cidr_netmask=24 --group group-array1
pcs resource create array1 ZFS pool="array1" importargs="-d /dev/mapper/" op start timeout="90" op stop timeout="90" --group=group-array1
pcs resource defaults resource-stickiness=100
```

Create and share a ZFS directory.

```
zfs create array1/nfs1
zfs set sharenfs=rw=@172.16.100.0/24,sync,no_root_squash,no_wdelay array1/nfs1
```

Enable and start NFS released services.

```
systemctl enable rpcbind nfs-server
systemctl start rpcbind nfs-server
```

Check the status of the cluster.

```
pcs cluster status
pcs status resources
showmount -e localhost
```

Linux

RHEL Move Printers

To migrate all or most the printers from one system to another, run the following commands on the **old system**:

Copy all the files in `/etc/cups/ppd` to a temporary location.

```
scp /etc/cups/ppd/*.ppd newsys:/tmp/
```

Create a list of all the printers on the old system. Copy file to new system

```
lpstat -v > /tmp/printers.txt
scp /tmp/printers.txt newsys:/tmp/printers.txt
```

Copy the lpoptions file, if you have one, to a temporary location.

```
scp /etc/cups/lpoptions newsys:/tmp/lpoptions
```

On the **new system**:

Create a script to read the printers file and create the devices on the new system.

```
cat /tmp/printers.txt \
| sed -e 's/device for //' -e 's/: / /'  \
| ( while read p u; do
    if [ -e /tmp/${p}.ppd ]; then
      echo lpadmin -p ${p} -P /tmp/${p}.ppd -v ${u} -E
    else
      echo lpadmin -p ${p} -v ${u} -E
    fi
  done) > create-prints.sh
```

Verify the create-printers.sh and then run it.

```
chmod +x ./create-prints.sh
./create-prints.sh
```

Copy the CUPS options.

```
cat /tmp/lpoptions >> /etc/cups/lpoptions
```

Verify printers were created.

```
lpstat -v
```

systemd services

Various systemd services.

Oracle WebLogic

/etc/systemd/system/wls_nodemanager.service

```
[Unit]
Description=WebLogic nodemanager service

[Service]
Type=simple
WorkingDirectory=/u02/oracle/domains/base_domain
ExecStart=//u02/oracle/domains/base_domain/bin/startNodeManager.sh
ExecStop=/u02/oracle/domains/base_domain/bin/stopNodeManager.sh
User=oracle
Group=oinstall
KillMode=process
LimitNOFILE=65535

[Install]
WantedBy=multi-user.target
```

/etc/systemd/system/wls_adminserver.service

```
[Unit]
Description=WebLogic Adminserver service

[Service]
Type=simple
WorkingDirectory=/u02/oracle/domains/base_domain
ExecStart=/u02/oracle/domains/base_domain/startWebLogic.sh
ExecStop=/u02/oracle/domains/base_domain/bin/stopWebLogic.sh
User=oracle
Group=oinstall
```

```
KillMode=process
LimitNOFILE=65535

[Install]
WantedBy=multi-user.target
```

Linux

tmux

Session Control (from the command line)

`tmux`	Start a new session
`tmux new -s <session-name>`	Start a new session with the name chosen
`tmux ls`	List all sessions
`tmux attach -t <target-session>`	Re-attach a detached session
`tmux attach -d -t <target-session>`	Re-attach a detached session (and detach it from elsewhere)
`tmux kill-session -t <target-session>`	Delete session

Pane Control

`Ctrl` `b`, `"`	Split pane horizontally
`Ctrl` `b`, `%`	Split pane vertically
`Ctrl` `b`, `o`	Next pane
`Ctrl` `b`, `;`	Previous pane
`Ctrl` `b`, `q`	Show pane numbers
`Ctrl` `b`, `z`	Toggle pane zoom
`Ctrl` `b`, `!`	Convert pane into a window
`Ctrl` `b`, `x`	Kill current pane
`Ctrl` `b`, `Ctrl` `O`	Swap panes
`Ctrl` `b`, `t`	Display clock
`Ctrl` `b`, `q`	Transpose two letters (delete and paste)
`Ctrl` `b`, `{`	Move to the previous pane
`Ctrl` `b`, `}`	Move to the next pane
`Ctrl` `b`, `Space`	Toggle between pane layouts
`Ctrl` `b`, `↑`	Resize pane (make taller)
`Ctrl` `b`, `↓`	Resize pane (make smaller)
`Ctrl` `b`, `←`	Resize pane (make wider)
`Ctrl` `b`, `→`	Resize pane (make narrower)

Window Control

Ctrl b , c	Create new window
Ctrl b , d	Detach from session
Ctrl b , ,	Rename current window
Ctrl b , &	Close current window
Ctrl b , w	List windows
Ctrl b , p	Previous window
Ctrl b , n	Next window

Copy-Mode (Emacs)

Ctrl b , [	Enter copy mode
Ctrl b , M-<	Bottom of history
Ctrl b , M->	Top of history
Ctrl b , M-m	Back to indentation
Ctrl b , M-w	Copy selection
Ctrl b , M-y	Paste selection
Ctrl b , Ctrl g	Clear selection
Ctrl b , M-R	Cursor to top line
Ctrl b , M-r	Cursor to middle line
Ctrl b , ↑	Cursor Up
Ctrl b , ↓	Cursor Down
Ctrl b , ←	Cursor Left
Ctrl b , →	Cursor Right

Copy-Mode (vi)

Ctrl b , [	Enter copy mode
Ctrl b , G	Bottom of history
Ctrl b , g	Top of history
Ctrl b , Enter	Copy selection
Ctrl b , p	Paste selection
Ctrl b , k	Cursor Up
Ctrl b , j	Cursor Down
Ctrl b , h	Cursor Left
Ctrl b , l	Cursor Right

Linux

FFMPEG on EL 8

```
dnf install -y epel-release
dnf install -y https://download1.rpmfusion.org/free/el/rpmfusion-free-release-8.noarch.rpm
https://download1.rpmfusion.org/nonfree/el/rpmfusion-nonfree-release-8.noarch.rpm -y &&
dnf install http://rpmfind.net/linux/centos/8-stream/PowerTools/x86_64/os/Packages/SDL2-2.0.10-
2.el8.x86_64.rpm &&
dnf install -y ffmpeg ffmpeg-devel
```

Networking

Networking

Azure to Ubiquiti IPSec

Connecting Azure to on-premises.

After you've created your Azure Virtual Network Gateway log in to your router.

```
set vpn ipsec auto-firewall-nat-exclude enable

set vpn ipsec esp-group FOO0 lifetime 3600
set vpn ipsec esp-group FOO0 pfs disable
set vpn ipsec esp-group FOO0 proposal 1 encryption aes256
set vpn ipsec esp-group FOO0 proposal 1 hash sha1

set vpn ipsec ike-group FOO0 key-exchange ikev2
set vpn ipsec ike-group FOO0 lifetime 3600

! REPLACE "1" IF YOU ALREADY HAVE A PROPOSAL USING THIS IDENTIFIER
set vpn ipsec ike-group FOO0 proposal 1 dh-group 2
set vpn ipsec ike-group FOO0 proposal 1 encryption aes256
set vpn ipsec ike-group FOO0 proposal 1 hash sha1

set vpn ipsec site-to-site peer 22.24.48.131 authentication mode pre-shared-secret
set vpn ipsec site-to-site peer 22.24.48.131 authentication pre-shared-secret SXYHVuH5p4vySL3eeKHEut64m
set vpn ipsec site-to-site peer 22.24.48.131 connection-type respond
set vpn ipsec site-to-site peer 22.24.48.131 description IPsecAzure
set vpn ipsec site-to-site peer 22.24.48.131 ike-group FOO0
set vpn ipsec site-to-site peer 22.24.48.131 local-address 17.15.138.22

!REPLACE "vti0" BY ANOTHER VTI INTERFACE ID IF THIS ONE IS ALREADY USED BY YOUR UBIQUITI DEVICE
set vpn ipsec site-to-site peer 22.24.48.131 vti bind vti0
set vpn ipsec site-to-site peer 22.24.48.131 vti esp-group FOO0
set interfaces vti vti0
set protocols static interface-route 192.168.0.0/22 next-hop-interface vti0

set firewall options mss-clamp interface-type vti
```

```
set firewall options mss-clamp mss 1350
set system offload ipsec enable
```

Networking

Ubiquti - USG disable NAT

1. **ssh <adminusername>@<IP of USG LAN>**
2. type '**configure**'
3. type '**show service nat**' #you should see rule 6001, 6002, 6003 by default
4. type '**set service nat rule 6001 disable**' #disables corporate network NAT
5. type '**set service nat rule 6002 disable**' #disables remote user network NAT
6. type '**set service nat rule 6003 disable**' #disables guest network NAT
7. type '**compare**' #just to see if you did things right
8. type '**commit**'
9. type '**save**'
10. type '**mca-ctrl -t dump-cfg > config.gateway.json**'
11. copy this file over to your Unifi controller,

    .'**scp config.gateway.json root@<controller_IP>:/usr/lib/unifi/data/sites/{{site folder}}/config.gateway.json**'

tcpdump

One-Liners: tcpdump

Helpful tcpdump commands.
Add -v to -vvvv to see from some to a lot of information.

General networking

Get CDP or LLDP information.

```
tcpdump -i enp132s0f0 -v -s 1500 -c 1 '(ether[12:2]=0x88cc or ether[20:2]=0x2000)'
```

Watch DHCP traffic

```
tcpdump -i enp1s0f0  port 67 or port 68 -e -n -vv
```

Show VLAN tags

```
tcpdump -i enp1s0f0  -e vlan -nn
```

Watch for 1 host

```
tcpdump -vvvv -n dst host 10.200.200.13 or src host 10.200.200.13
```

Security(ish)

commands the are mostly security and privacy related

Extract key/cert from PFX

1. Extract key from pfx file
   `openssl pkcs12 -in /path/to/file.pfx --nocerts -out /path/to/exported.key`
2. Extract certificate from pfx file
   `openssl pkcs12 -in /path/to/file.pfx -clcerts -nokeys -out /path/to/cert.crt`
3. decrypt private key if desired.
   `openssl rsa -in /path/to/exported.key -out /path/to/decrypted.key`

Security(ish)

Postfix TLS

Configuring Postfix to use TLS on CentOS 7

1. Install all required packages

`yum install cyrus-sasl cyrus-sasl-devel cyrus-sasl-gssapi cyrus-sasl-md5 cyrus-sasl-plain postfix`

1b. Backup default postfix config

`cp /etc/postfix/main.cf /etc/postfix/main.cf_orig`

2. Configure SMTP-AUTH and TLS using postconf

```
/usr/sbin/postconf -e 'smtpd_sasl_local_domain =fqdn.com'
/usr/sbin/postconf -e 'smtpd_sasl_auth_enable = yes'
/usr/sbin/postconf -e 'smtpd_sasl_security_options = noanonymous'
/usr/sbin/postconf -e 'broken_sasl_auth_clients = yes'
/usr/sbin/postconf -e 'smtpd_recipient_restrictions =
permit_mynetworks,permit_sasl_authenticated,reject_unauth_destination'
/usr/sbin/postconf -e 'inet_interfaces = all'
/usr/sbin/postconf -e 'mynetworks = 127.0.0.0/8, 10.0.0.0/8, 192.168.1.0/24, 192.168.100.0/24'
```

3. Set postfix to allow LOGIN and PLAIN logins.

`vim /etc/sasl2/smtpd.conf`

```
pwcheck_method: saslauthd
mech_list: plain login
```

4. Create key for SSL certificate signing request

```
mkdir /etc/postfix/ssl
cd /etc/postfix/ssl/
openssl genrsa -des3 -rand /etc/hosts -out smtpd.key 1024
chmod 600 smtpd.key
```

5. Create the signing request with the key

```
openssl req -new -key smtpd.key -out smtpd.csr
```

6. Create the SSL certificate with the signing request and the key

```
openssl x509 -req -days 3650 -in smtpd.csr -signkey smtpd.key -out smtpd.crt
```

7. Create RSA key

```
openssl rsa -in smtpd.key -out smtpd.key.unencrypted
mv smtpd.key.unencrypted smtpd.key
```

8. Create CA key and cert

```
openssl req -new -x509 -extensions v3_ca -keyout cakey.pem -out cacert.pem -days 3650
```

9. Configure postfix for TLS

```
/usr/sbin/postconf -e 'smtpd_tls_auth_only = no'
/usr/sbin/postconf -e 'smtp_use_tls = yes'
/usr/sbin/postconf -e 'smtpd_use_tls = yes'
/usr/sbin/postconf -e 'smtp_tls_note_starttls_offer = yes'
/usr/sbin/postconf -e 'smtpd_tls_key_file = /etc/postfix/ssl/smtpd.key'
/usr/sbin/postconf -e 'smtpd_tls_cert_file = /etc/postfix/ssl/smtpd.crt'
/usr/sbin/postconf -e 'smtpd_tls_CAfile = /etc/postfix/ssl/cacert.pem'
/usr/sbin/postconf -e 'smtpd_tls_loglevel = 1'
/usr/sbin/postconf -e 'smtpd_tls_received_header = yes'
/usr/sbin/postconf -e 'smtpd_tls_session_cache_timeout = 3600s'
/usr/sbin/postconf -e 'tls_random_source = dev:/dev/urandom'
```

10. Set servers hostname and mydomain in postfix config

```
/usr/sbin/postconf -e 'myhostname = host.yourdomain.com'
/usr/sbin/postconf -e 'mydomain = yourdomain.com'
```

11. Check through the postfix config to verify all of the settings.

```
more /etc/postfix/main.cf
```

12. Stop sendmail and Start postfix, saslauthd

```
systemctl stop sendmail
systemctl restart postfix
systemctl restart saslauthd
```

OpenSSL tricks

Download a site's certificate.

This command will connect to example.com on port 443 using the s_client subcommand and output the site's certificate information in text format using the x509 subcommand. The -text option tells openssl to print the certificate information in human-readable text format, while the -noout option tells it not to output the certificate itself.

You can replace example.com with the hostname or IP address of the site you want to get the certificate for. The < /dev/null part of the command is used to prevent the s_client command from waiting for input.

```
openssl s_client -connect example.com:443 < /dev/null | openssl x509 -text -outform PEM > /path/to/cert.cer
```

VMware

vCenter, ESXi and NSX.

VMware

Unlock the ESXi host account

1. At the console press CTRL+ALT+F2 to get to the ESXi shell. If a login shows up continue with step 3, otherwise continue with step 2.
2. Login to the DCUI (to enable the ESXi Shell if not already done)
3. Login with root and the correct password.
4. Go to Troubleshooting Options
5. Select Enable ESXi Shell
6. Press CTRL+ALT+F1
7. At the ESXi shell login with root and the password
8. Run the following commands to show number of failed attempts:

```
pam_tally2 --user root
```

Run the following command to unlock the root account:

```
pam_tally2 --user root --reset
```

Windows

Yeah, Windoze is here too

Active Directory

To get a list of the FSMO Role holders for a Single Domain.

```
Get-ADDomain | Select-Object DistinguishedName, SchemaMaster, DomainNamingMaster, InfrastructureMaster, PDCEmulator, RIDMaster
```

To get a list of the FSMO Role holders in a Forest.

```
Get-ADForest | Select-Object Name,SchemaMaster, DomainNamingMaster,InfrastructureMaster, PDCEmulator, RIDMasterall
```

To get a nicely formatted list with all the Domain Controllers and who owns which particular role.

```
Get-ADDomainController -Filter * | Select-Object Name, Domain, Forest, OperationMasterRoles | Where-Object {$_.OperationMasterRoles}
```

Create a new AD computer.

```
$ServerName="newsys"
$DC="ads01.example.com"
$OU="OU=SERVERS,DC=example,DC=com"
New-ADComputer -Name $ServerName -SamAccountName $ServerName -Path $OU  -Description "Apache Win Development"  -Server $DC -Enabled $True
```

Unlock an account

```
get-ADUser -Identity <username> -Properties LockedOut
Unlock-ADAccount -Identity <username>
```

Change a password

```
$SecPaswd= ConvertTo-SecureString -String 'kPnguoHTUQ' -AsPlainText -Force
Set-ADAccountPassword -Reset -NewPassword $SecPaswd -Identity cesvcsso01
Set-ADUser -Identity <username> -ChangePasswordAtLogon $false
```

New AD group

```
$GroupName="SvrOps"
$OU="OU=GROUPS,DC=exmple,DC=com"
New-ADGroup $GroupName -Path $OU -GroupCategory Security -GroupScope Global -PassThru -Verbose
```

Windows Management Consoles

CERTMGR.MSC	Certificates snap-in
CERTSRV.MSC	Certification Services
CMD.EXE	Command Prompt
COMPMGMT.MSC	Computer Management
DCPOL.MSC	Domain Controller Security Policy
DEVMGMT.MSC	Device Manager
DFRG.MSC	Disk Defragmenter
DFSGUI.MSC	Distributed File System
DHCPMGMT.MSC	DHCP Manager
DISKMGMT.MSC	Disk Management
DNSMGMT.MSC	DNS Manager
DOMAIN.MSC	Active Directory Domains & Trust
DOMPOL.MSC	Domain Security Policy
DSA.MSC	Active Directory Users & Computers
DSA.MSC /DOMAIN=domainname	
DSA.MSC /SERVER=servername	
DSSITE.MSC	Active Directory Sites & Services
EVENTVWR.MSC	Event Viewer
FSMGMT.MSC	Shared Folders
GPEDIT.MSC	local Group Policy Editor
IAS.MSC	Internet Authentication Service
INETMGR	Internet Information Service (\Windows\system32\inetsrv)
LUSRMGR.MSC	Local Users and Groups
MMC.EXE	Microsoft Management Console
MSINFO32.EXE	Hardware and software configuration information
PERFMON.MSC	Performance Monitor
REGEDIT.EXE	Run Registry Editor
RRASMGMT.MSC	Routing and Remote Access
RSOP.MSC	Resultant Set of Policy
SECPOL.MSC	Local Security Policy
SERVICES.MSC	Service Configuration
TSCC.MSC	Terminal Services
MSTSC	Remote Desktop

Windows

Windows Install Media

`./create_win_usb.sh ~/Downloads/Win10.iso /dev/sdc`

```
#!/bin/bash
# ----------------------------------------------------------------
# Create Bootable Windows 10/11 USB in Linux (Debian/RPM-based)
# Version: 1.0
# Author: LinuxConfig.org
# Date: 23-06-2023
# License: GPL-3.0
# ----------------------------------------------------------------
# Input parameters: Path to the ISO file and USB block device location
ISO_PATH=$1
USB_BLOCK=$2

echo "ISO Path is: $ISO_PATH"
echo "USB block device is: $USB_BLOCK"

# Check for required commands
REQUIRED_COMMANDS=("rsync" "parted" "wipefs" "mkfs.vfat" "mkfs.ntfs" "udisksctl" "sha256sum" "mount"
"umount" "mkdir" "cp" "sync" "mktemp")

echo "Checking for required commands..."
for cmd in "${REQUIRED_COMMANDS[@]}"; do
    if ! command -v $cmd >/dev/null 2>&1; then
        echo "Missing command $cmd. Please install the corresponding package and rerun this script."
        exit 1
    fi
done
echo "All required commands are available."

# 1. Checking ISO file
read -p "Do you want to calculate the ISO file checksum? This could take some time. Press 'y' to continue or any
other key to skip: " response
if [[ $response =~ ^[Yy]$ ]]
then
```

```
    echo "Calculating ISO file checksum..."
    ISO_CHECKSUM=$(sha256sum $ISO_PATH)
    echo "Checksum is: $ISO_CHECKSUM"
    read -p "Please verify the checksum. Press 'y' to continue: " response

    if [[ ! $response =~ ^[Yy]$ ]]
    then
        echo "Aborting due to user response."
        exit 1
    fi
fi

# Warning about formatting and data loss
echo "WARNING: All data on $USB_BLOCK will be lost!"
read -p "Are you sure you want to continue? [y/N]: " confirm
confirm=${confirm:-N}
if [[ $confirm =~ ^[Yy]$ ]]
then
    # 2. Formatting the USB drive
    echo "Formatting USB drive..."
    wipefs -a $USB_BLOCK

    parted $USB_BLOCK mklabel gpt
    parted $USB_BLOCK mkpart BOOT fat32 0% 1GiB
    parted $USB_BLOCK mkpart INSTALL ntfs 1GiB 100%
    parted $USB_BLOCK unit B print

    # Create temporary directories for mounting
    ISO_MOUNT=$(mktemp -d)
    VFAT_MOUNT=$(mktemp -d)
    NTFS_MOUNT=$(mktemp -d)

    # 3. Mounting the ISO
    echo "Mounting ISO..."
    mount $ISO_PATH $ISO_MOUNT

    # 4. Formatting and copying to the USB
    echo "Copying to USB..."
    mkfs.vfat -n BOOT ${USB_BLOCK}1
    mount ${USB_BLOCK}1 $VFAT_MOUNT
```

```
    rsync -r --progress --exclude sources --delete-before $ISO_MOUNT/ $VFAT_MOUNT/

    mkdir -p $VFAT_MOUNT/sources
    cp $ISO_MOUNT/sources/boot.wim $VFAT_MOUNT/sources/

    mkfs.ntfs --quick -L INSTALL ${USB_BLOCK}2
    mount ${USB_BLOCK}2 $NTFS_MOUNT

    rsync -r --progress --delete-before $ISO_MOUNT/ $NTFS_MOUNT/

    # 5. Unmounting and power off
    echo "Unmounting drives and syncing data... This may take a while, do not disconnect your USB drive."
    umount $NTFS_MOUNT
    umount $VFAT_MOUNT
    umount $ISO_MOUNT
    sync

    # Remove temporary mount directories
    rmdir $ISO_MOUNT $VFAT_MOUNT $NTFS_MOUNT

    udisksctl power-off -b $USB_BLOCK

    echo "Done! You can now safely disconnect your USB drive."
else
    echo "Aborted by user."
    exit 1
fi
```