SCL System Administration

Part 2 (update 3)

5. Package Management

zypper, rpm

Manage Software in Packages

- **Packages** (residing in **Repositories**) are the basis for Software Distribution.
- **Package installers** are designed for software installation and removal
- **Packaging Systems** use a **dependency model** to ensure the proper libraries and configuration files are used with the software installation
- Package installers run scripts to create and verify proper software functionality
- **Patches** to various OS Software use the dependency model
- **Repository Index Service (RIS)** - has a list of other repositories indexed via this list.

zypper Package Manager

- **zypper (8) -** Commandline Package manager based on RPM (replaced that of YaST) to install, remove and query packages.
- Works with YaST, RPM-MD Software Repositories and plain directories with .rpm files Uses **libzypp**, a software management engine
- Manage Repositories, Queries (patterns), Updating, Package Locks, Verify dependencies, Suggest Packages based on new HW or SW, xml output, non-interactive mode for scripts, translate local paths as dir: URI
- command-not-found (**cnf**) for command (e.g. **repquota**) suggests package list and running **sudo zypper install quota**
- Provides a commandline shell to type commands directly via:
  $ **zypper sh[ell]**
- See $ **man zypper**; **zypper help [*command*]** or **zypper** Command Cheat Sheet: **<https://en.opensuse.org/images/1/17/Zypper-cheat-sheet-1.pdf>**
- **Reference: <https://en.opensuse.org/Portal:Zypper>**

zypper Commands

$ zypper [*{shortcut} or Command*]	Purpose
{ar} addrepo *uri*	add a repository to the working set
{dup} dist-upgrade	Updates to current distribution release
{if} info *packages*	Displays information about packages
{in} install *packages*	Downloads and installs packages
{lu} list-updates	Lists all updated packages in repository
{mr} modifyrepo *uri*	Modify repository properties
{ref} refresh	Update local cache's repos. metadata
{rm} remove *packages*	Uninstall packages
{lr} repos	List repositories in current working set
{se} search string	Search packages with matching names
{sh} shell	Start an interactive **zypper** session
{up} update	Install updated versions of **all** packages

zypper Exercise U2

1. Type **sudo zypper se nmap** to search for **nmap** package
2. Type **sudo zypper info nmap** for more information about **nmap**. To see what will be installed with it: **sudo zypper info --recommends \ nmap**; For dependencies, **sudo zypper info --requires nmap**
3. Type **sudo zypper in nmap** to install the nmap package.
4. Type **sudo zypper se -t pattern | fmt > zpatterns; less zpatterns** to see all the patterns
5. Get more information about the **fips** (functionality) pattern: Type **sudo zypper info -t pattern fips**
6. Install the fips pattern: Type: **sudo zypper in -t pattern fips**
7. Get more information about the fips pattern: Type **sudo zypper info -t pattern fips**
8. Prevent nmap upgrade Type: **zypper addlock nmap**
   Verify lock has been set Type: **zypper ll**
   Remove the lock again Type: **zypper rl nmap; zypper ll**

RPM Package Manager

- **rpm (8)** Older, lower-level Package Manager that installs, verifies and queries the status of packages
- **rpmbuild (8)** exclusively builds packages
- **rpm** can be considered several commands with the same name.
  Install (rpm -i) **Remove** (rpm -e) **Update** (rpm -U)
  Query (rpm -q) **Search** (rpm -qa | grep 'pattern')
  Verify (rpm -V)
- See man page: <rpm.org/max-rpm-snapshot/rpm.8.html>,
  See <lpar.ath0.com/2009/11/20/rpm-cheat-sheet/>
- See 20 Effective rpm commandlines: < tecmint.com/20-practical-examples-of-rpm-commands-in-linux/ >

rpm Exercise U2

You need to find the configuration file for a specific binary file. Use **vsftpd** as an example to see what can be done here.

1. You need the exact name of the binary to query. Type **sudo which vsftpd** Output should be **/usr/sbin/vsftpd**
2. So which RPM is file from? Type: **sudo rpm -qf /usr/sbin/vsftpd** **#** Output shows **vsftpd** is the RPM name.
3. Read the package description. Type: **sudo rpm -qi vsftpd**
4. Get a list of files in the package. Type: **sudo rpm -ql vsftpd**
5. Which files are used for its configuration Type: **sudo rpm -qc vsftpd**
6. If you have to read more documentation about this binary, type **sudo rpm -qd vsftpd**

6. root (superuser) Tasks

su sudo set[ug]id

Superuser (root) Behavior

- Superuser has UID = 0 and called **root**.
- If your UID = 0, you are superuser, no matter what your user name is. Have just one UID=0 on your system or one per administrator?
- **root** violates all linux filesystem permission rules
- Powers are magnified, so are errors; Your actions may directly affect other users and programs on your system
- Each SA to do normal work as unprivileged user; do SA work by becoming root (via su or sudo) for traceability.
- It is not safe (and may not be allowed) to log in as root over a network
- linux-lwsr:/ # # Beware! You are Superuser!

Configuring booting u2

- **Computer Startup Sequence**
  - Reading the boot loader (**GRUB2**) from master boot record
  - Loading/Initializing the Kernel
  - Detecting/Configuring Devices (**initd** or **systemd***)
  - Creation of Kernel Processes
  - Administrator Intervention (Single User mode)
  - SLES 12.x: Starting System: (startup/shutdown scripts in /usr/sbin/rccifs --> /etc/init.d and /usr/sbin/service)
  - SLES 11.x: Starting System: (startup/shutdown scripts in /etc/init.d

  * **initd** used in SLES 11.x, **systemd** used in SLES 12.x

Boot Process - (SLES 12) Configuring GRUB2 U2

- via **YaST - Select System --> Bootloader** to access Boot Code Options (generic parameters), (specific) Kernel Parameters, and Bootloader Options (to decide how the boot loader menu is displayed)
- Boot Code Options:
  - Location (MBR or root partition-default)
  - Distributor (SLES 12 (RC3))
- Kernel Parameters: For applying permanent changes. Apply to default and failsafe kernels.
  - VGA mode: Unspecified or Text Only (no GUI)
  - To see boot progress, **delete splash=silent quiet showopts**
  - Which console to use: graphical (directly attached) or serial (remote console over a modem)

Boot Process - (SLES 12) Configuring GRUB2 (2)

- Boot Loader Options:
  - Set a time-out (seconds) to let administrator intervene
  - Probe Foreign OS implies Dual Boot usually with Windows
  - **Show** (on Server) / Hide (on Users Desktop) **Boot Menu**
  - Default Boot (Drop down list)
  - Protect boot loader with a password at boot prompt
- Configuring GRUB2 (for **SLES 12**) manually:
  - Review Configuration file: /etc/default/grub # SLES 12 only
  - Make changes and run stage 2:
    # grub2-mkconfig > /boot/grub2/grub.cfg
    Do **not** change grub.cfg file **directly**. It will **break** the configuration for GRUB2. /etc/grub.d templates also used.
- **SLES 11.x** grub configuration files: **/boot/grub/menu.lst**
  /boot/grub/device.map /etc/grub.conf /etc/sysconfig/bootloader

Boot Process - (SLES 12) Troubleshooting u2

- Use GRUB2 Boot Menu Selections:
  -Boot with default settings,
  -Advanced Options (limited HW Support),
  -Start bootloader (readonly Btrfs snapshots)
- Snapshots created when YaST makes modifications
- Enter GRUB2 password, type **e** to see menu editor (= **/boot/grub2/grub.cfg** in memory)
- Start the server in emergency mode:
  at line: **linux /boot/vmlinuz- ...** append at the end:
  systemd.unit=emergency.target [SLES 12.x]
  -Use **<Ctrl-X> (or F10, if configured)** to boot
- Once Emergency boot starts, In single user mode, login as root use **journalctl -xb** for system logs and **systemctl reboot|default**

SLES 12.x systemd Daemon u1

- 1st process started (no longer SLES 11.x **/sbin/init**)
- Starts everything else at appropriate run-level
- Starts/Stops **units**: service, socket, (auto)mount, target, snapshot, swap, path, slice, scope.
  # systemctl -t help
- Unit Locations: Default: /usr/lib/systemd/system; Custom: /etc/systemd/system
  # **less /usr/lib/systemd/system/sshd.service**
- Startup/Shutdown Initialization script format in **/etc/init.d/rc[35].d/[SK]{nn}scriptname**
- See **initd, systemd** comparison article**: <tecmint.com/systemd-replaces-init-in-linux/>**

systemd Daemon Definitions (2)

- **Unit (file)**: Encodes information about various services, targets, snapshots... in Unit, Service and Install Sections.
  Target: A Unit configuration file whose suffix = target. Groups units, organizes dependencies and represents synchronization points during system startup.
  Want: defines a dependency between at least 2 targets
  Slice: Hierarchical management of resources of a group of processes
  Seat: the set of hardware available at a work station
  Session: When a user is logged on at a specific seat
  - Only one session can be active per seat
  - Default seat is seat0

  Hardware: Those physical items assigned to seats.
  [replaces ConsoleKit]

systemd Daemon (3)

- **Start/Stop A Service:**
  # systemctl start sshd.service
  # systemctl stop sshd.service
  # systemctl status sshd.service
- **Enable/Disable A Service:**
  # systemctl enable sshd.service
  # systemctl disable sshd.service
- **Reboot/Shutdown the System:**
  # systemctl reboot
  # systemctl halt
  # systemctl poweroff
- **Change runlevels**
  # systemctl rescue # to 1 or S
  # systemctl default # to default runlevel

systemd Daemon (4)

- **Review System Logs:**
  # journalctl -u sshd
  # journalctl -x -u sshd ### more verbose
  # journalctl -f -u sshd ### =tail -f /var/log/sshd.log
  # journalctl ### View every log message
- **Review Kernel Messages: # journalctl -k [-f]**
- **View 2 Daemons' activities (logical Or; latest first)**
  # journalctl -r _SYSTEMD_UNIT=avahi-daemon.service + _SYSTEMD_UNIT=dbus-daemon.service
- **List status of a service: # systemctl -l status cron.service**
- **List dependencies: # systemctl list-dependencies**
- **List unit files states: # systemctl list-unit-files**
- **Run systemd on remote hosts: # systemctl -H user@host**

Run levels - System Targets (who -r) u1

Run-level	Equivalent systemd target
0	poweroff.target [system halt]
1, S	rescue.target [single user mode]
2, 3, 4	multi-user.target [Commandline]
5	graphical.target [GUI]
6	reboot.target

systemd References

- <suse.com/documentation/sles-12/book_sle_admin/data/cha_systemd.html> <suse.com/documentation/sles-12/pdfdoc/book_sle_admin/book_sle_admin.pdf>
- <freedesktop.org/wiki/Software/systemd> <www.freedesktop.org/wiki/Software/systemd/FrequentlyAskedQuestions/> <freedesktop.org/wiki/Software/systemd/Debugging>
- <linux.com/learn/tutorials/788613-understanding and-using-systemd>
- <en.wikipedia.org/wiki/Systemd>
- <linoxide.com/linux-command/systemd-vs-sysvinit-cheatsheet/>

systemd Exercise For SLES 12.x u1

You'll install the **vsftpd** service, start it and enable it.

0. Show the contents of **/etc/issue** to see SLES version.

1. Type **zypper in vsftpd** (This installs the **vsftpd** service or indicates that it is already installed.)

2. Type: **cd /etc/systemd/system/multi-user.target.wants** to change to the wants directory for the **multi-user.target** .

3. Type: **ls** to see which **wants** are there. There are no wants for the **vsftpd** service file.

4. Type: **systemctl start vsftpd ; system status vsftpd** to insure it was started.

5. Type: **systemctl enable vsftpd** to be sure the service will start automatically upon reboot. Use **ls** again in the multi-user.target.wants directory. Notice the symlink for vsftpd exists.

Configuring logins

- **systemd-logind** is a daemon login manager (based on systemd-logind.service)
- Keeps track of users and sessions, and their processes and states.
- Provides policy based access for users to operations of shutdown or sleep
- Multi-seat, Session switch mgmt.
- Device access management for users
- Automatic spawning of text logins (gettys) on virtual console activation
- See **logind.conf (5)** and **loginctl (1)** for configuration information

7. Managing SLES

autoyast you rpm

Repositories and Meta Package Handlers

- Programs create specific functionality and rely on libraries to make it work.
- **Dependencies** are the code needed by the author but not written by him/her
- **Software package management** keeps track of **packages (software + dependencies)**
- A **Meta Package Handler** uses **Repositories** to install software
- A **Repository** is an installation source with a collection of packages
- Servers use multiple Repositories and use a regularly downloaded index of available packages
- In SUSE, a **Service** manages repositories, called the Repository Index Service (**RIS**) and provides a master index of all Repositories

YaST Online Update

- **YaST Online Update (YOU):**
  o Subscription based for SLES (1 or 3 year)
  o Contains software security, relevant patches, fixes and/or enhancements
- Commandline: # **yast2 online_update**
- **Gnome Update Applet**
  o icon is in notification area
  o **<Alt-F2>** and command: **gpk-update-viewer**

AutoYaST u1

- System For installing one or more SUSE Linux systems automatically and without user (admin) intervention.
- Performed using an **AutoYaST profile** with installation and configuration data (via **AutoYaST** configuration interface)
- Single System: [user input] --> [Install proposal] --> [Install]
- Multiple Systems: share same environment, similar hardware, doing similar tasks - via an **AutoYaST** profile, installed in parallel.
- o **Preparation**: target system info --> directives of profile --> YaST sensible data file
  o **Installation**: Installs base system
  o **Configuration**: **YaST2** and user defined post-install scripts run
- See also: <users.suse.com/~ug/autoyast_doc/index.html>

RPM Package Manager

- Main commands: **rpm** and **rpmbuild**
- **RPM database** queried by users, admins, developers
- *.rpm archives are binary; contain programs+meta info.
- **rpm** 5 Modes: **Installing**, **uninstalling/updating**, **rebuilding** db, **querying** databases or archives, **integrity checking and signing** packages
- $ **rpm --checksig package-3.2.1.rpm** # for pedigree
  $ **rpm -i package.rpm** # install a package
  $ **rpm {-U|-F} package.rpm** # upgrade/freshen package
  $ **rpm -e package.rpm** # uninstall (erase) a package

Querying RPM Database

- $ **rpm -q iptables**
  iptables-1.4.21-3.1.2.x86_64
  $ **rpm -qR iptables # list dependencies**
  $ **rpm -ql iptables # list package contents**
  $ **rpm -q -i iptables # show information about**

Managing Remote Access

PuTTY, VNC, ssh

Using VNC

- Lets you control a remote computer having a GUI from any OS GUI Desktop
- SUSE supports One-time sessions as well as persistent sessions in between logouts/shutdowns.
- Client to Server Connection uses IPaddress:PortNo, where 5900 is default but may be higher
- See <tightvnc.com/doc/win/TightVNC_for_Windows-Installation_and_Getting_Started.pdf>

Observation

When cryptography is outlawed, bayl bhgynjf jvyy unir cevinpl

Using PuTTY

- **PuTTY** is an open-source suite of tools that do secure terminal emulation, copying (**PSCP**) and file transfer (**PSFTP**). It also does key-generation (**PuTTYgen**) and authentication (**Pageant**)
- Cryptographic signatures and checksums are available for all the tools in the suite.
- **PuTTY** (Client only) supports and connects to any Unix/Linux Server running **ssh**
- **See <chiark.greenend.org.uk/~sgtatham/putty/faq.html>**

Remote Desktop Connection

- An application client for the Remote Desktop Service (RDS) that lets you remotely log into a networked computer running the terminal services server.
- The remote system's graphical user interface is shown in a window.
- See: <https://en.wikipedia.org/wiki/Remote_Desktop_Services>

openSSH Secure Shell

- A client-server based network protocol for initiating encrypted text based shell sessions remotely and securely
- Supports logins, remote command execution, copying, file transfer, and tunneling, forwarding TCP ports and X11 Connections.
- Uses a variety of authentication methods including passwords, public/private keys
- Requires stringent directory and file permissions to work (properly).

SSH features

- login to a shell on a remote host
- execute a single command on a remote host
- providing passwordless authentication to a remote server to streamline scripting
- Secure file transfer & secure copy even of a file on one remote server to that of another [data compression-decompression used for transfers]
- Securely mounting a directory on a remote server as a file system on a local computer using SSHFS

ssh Program Suite (1)

- **ssh** secure terminal emulation
- **scp** secure copies across servers
- **sftp** secure file transfer
- **ssh-add** loads private keys into **ssh-agent** process
- **ssh-agent** daemon used to automate client key authentications
- **ssh-keysign** ssh helper program for host-based authentication

ssh Program Suite (2)

- **ssh-keyscan** Gather public keys
- **ssh-keygen** Generates public/private key pairs for DSA/RSA Authentication (including host keys)
- **sftp-server** Secure ftp Server Subsystem
- **sshd** ssh server daemon

ssh References

- Barrett, D., Silverman, R., Byrnes, R. (2005). SSH, The Secure Shell: The Definitive Guide Second Edition. Cambridge: O'Reilly.

- Lucas, M.D. (2012). SSH Mastery: OpenSSH, PuTTY, Tunnels and Keys. CreateSpace Publishing

- OpenSSH Links:

  http://www.openssh.org/manual.html
  http://sial.org/howto/openssh/
  http://www.openssh.com/faq.html

ssh Configurations

- **/etc/ssh/sshd_config** configures openssh server (root only). Settings can't be overridden.
- **~/.ssh/config**, **/etc/ssh/ssh_config**, configures openssh client [your settings trump systems settings, but commandline options trump yours]
- Format: **Keyword value(s)**
  - default values are commented

SLES 12.x ssh Exercise u1

- To enable and test ssh connectivity
  Enable:
  1. type: **sudo systemctl status sshd.service**
  If the Active: line says active (running) skip to 3.
  2. If **sshd** not running, type:
  sudo systemctl start sshd.service and also:
  sudo systemctl enable sshd.service for rebooting
  Test:
  3. Open a new terminal window and type:
  ssh root@localhost # also type in the password;
  if you cannot successfully log in, replace root with an ordinary user.

SLES 12.x ssh Exercise (2) u3

- **Securing the SSH Server**
  1. Open a root shell on your server and create 2 accounts with simple passwords:
  /usr/sbin/useradd -m -d /home/jean jean; passwd jean
  /usr/sbin/useradd -m -d /home/john john; passwd john
  2. Back up the **sshd_config** file:
  cd /etc/ssh; cp -p sshd_config sshd_config.orig
  3. Edit this file: type: **vim sshd_config**
  4. Change the Port parameter to 4443 and deny root logins (under #Port 22, insert **Port 4443**; under #PermitRootLogin, insert **PermitRootLogin no**)
  5. On last line, insert **AllowUsers jean**, then save and quit (**:wq**)
  6. Restart sshd to have it read the modified sshd_config via **systemctl restart sshd.service**
  7. test: **ssh -p 4443 root@localhost** #verify that access denied
  ssh -p 4443 john@localhost #verify that access denied
  ssh -p 4443 jean@localhost #verify successful login

SLES 11.x ssh Exercise u1

- To enable and test ssh connectivity
  Enable:
  1. type: **ps -ef | grep -v grep | grep sshd**
  If sshd shows up on the right side, skip to 3.
  2. If **sshd** not running, type:
  sudo /etc/init.d/sshd start and also:

  Test:
  3. Open a new terminal window and type:
  ssh root@localhost # also type in the password;
  if you cannot successfully log in, replace root with an ordinary user.

SLES 11.x ssh Exercise (2) u3

- **Securing the SSH Server**
  1. Open a root shell on your server and create 2 accounts with simple passwords:
  /usr/sbin/useradd -m -d /home/jean jean; passwd jean
  /usr/sbin/useradd -m -d /home/john john; passwd john
  2. Back up the **sshd_config** file:
  cd /etc/ssh; cp -p sshd_config sshd_config.orig
  3. Edit this file: type: **vim sshd_config**
  4. Change the Port parameter to 4443 and deny root logins (under #Port 22, insert **Port 4443**; under #PermitRootLogin, insert **PermitRootLogin no**)
  5. On last line, insert **AllowUsers jean**, then save and quit (**:wq**)
  6. Restart sshd to have it read the modified **sshd_config** via
  sudo /etc/init.d/sshd restart
  7. test: **ssh -p 4443 root@localhost** #verify that access denied
  ssh -p 4443 john@localhost #verify that access denied
  ssh -p 4443 jean@localhost #verify successful login

ssh Elegances

- Remote Printing:
- $ cat my.file.to.print.txt |
  ssh userid@my.remote.printhost
  'lpr -Pwhich_printer'
- Disk Space shortage, phase 1: backup
- $ tar cvf - source_directory |
  ssh userid@remote_host 'cat > my-tar-file.tar'

sftp Builtin Commands

- Only **put** and **get** are available, but wildcards may be used.
- **put** and **get** can use **-P** option to preserve permissions and timestamps
- type **progress** to show the transfer progress meter
- See: <computerhope.com/unix/sftp.htm>
- Run: $ sftp [options] [userid@]host # Interactive
  $ sftp [userid@]host[:file ...] #Retrieve files automatically
  $ sftp [userid@]host[:dir[/]] # remote cd to dir
  $ sftp -b batchfile [user@]host # automated batch no pw

scp Features

- Copies can be between:
  - local host and remote host
  - two different remote hosts
- scp options allow:
  - attribute preservation (-p),
  - recursion (copy entire directories) (-r)
  - compression (-C)
- Run using:

  $ **scp [options] [[user@]host1:]file1 [...] **
  [[user@]host2:]file2

scp and Key Authentication

- See <arkaye.com/unix/PSESD60807/scp+pkeyauth.html>
  For setting up public/private key authentication between two servers and using ssh-agent

Observation

We all learn by experience, but some of us have to go to summer school.

Verifying Network Connectivity

ping traceroute netstat netcat

Troubleshooting Process

- Document current situation
- Check internally then externally
- Make one change at a time, test it, back out if bad results
- Problems may be transient
- Communicate with affected parties
- Use 7 layers of the network to isolate problem bottom to top: Physical, Data Link, Network, Transport, Session, Presentation, Application

General Checklist

- Is it plugged in?
- Have physical connectivity and a link light?
- Do your ARP tables show other hosts
- **Is there a firewall on your local machine?**
- **Is there a firewall anywhere between you and your destination?**
- If firewalls involved, do they pass ping packets and responses?
- Can you ping the local host address (127.0.0.1)?
- Can you ping other local hosts by IP address?
- Is DNS working properly?
- Can you ping other hosts by hostname?
- Can you ping hosts on another network?
- Do high level services such as web and ssh work?

ping Command

- Sends an ICMP ECHO_REQUEST packet to a target host and waits to see if the host answers back. It measures round trip time and packet loss
- **ping** traffic affects Routing tables, physical networks and gateways, DNS but can be blocked by firewalls.
- If possible, turn off any intervening firewalls temporarily to help debugging.
- Form: $ **ping [-c count] [-p size] <IP address|hostname>**
- Example: $ **ping -c 5 8.8.8.8 # Google DNS Server**

traceroute Command

- Shows the sequence of gateways through which an IP packet travels to reach its destination
- Packets have minimal times to live, triggering time exceeded messages from routers along the way.
- Form: **traceroute [options] <IP address|hostname> [packet size]**
- Example: # **traceroute 8.8.8.8**
- See <kb.pert.geant.net/PERTKB/TracerouteLikeTools> and <traceroute.org>

netstat Command

- Displays network connections for TCP (incoming and outgoing), routing tables, network interfaces, network protocol statistics. [going away for Linux]
- For Interface Configuration info, run: **netstat -i** (**ifconfig -a** # similar)
- For Network Connections, run: **netstat -a**
- Used to debug higher level problems; verify that servers are set up correctly, helps show TCP miscommunication states (SYN_SENT -> bad server; SYN_WAIT -> high connection requests)
- For viewing Listening Ports, run: **netstat -lp** # also use **lsof**
- For viewing Routing Tables, run: **netstat -r # Flags: U=up,G=gateway, H=host**
- For viewing network counters, run: **netstat -s # Alt. ss -s**

Checking NIC

- Use **ip addr** to check the current state of a Network Interface Card.
- Output shows what is up and what address is assigned to it.
- Possible fix: check it is plugged in and then use:
  wicked ifup [interfacename] # Network Management Utility

Checking Routing

- If NIC ok and ping doesn't, check Routing Tables:
- 1. Use **ip route show** # show config including default gateway, else set it
- 2. Use **iptables -L** # if lots of output, blocking firewall may exist
  - Turn off iptables: **systemctl -stop iptables; iptables -L # if command works, it's firewall**
  - Verify correct firewall configurtation
  - Turn on fixed firewall and run:
  service iptables start
- 3. Is it a default Gateway problem? run: # **traceroute 8.8.8.8**

Checking DNS

- **dig (1)** (Domain Information Groper] DNS lookup utility
- Features: Performs a DNS lookup, Finds a Host Address, IP address, Mail Exchange (MX), CNAME, name server records
- Form: **dig [options] [server{IP|name}] [name] [type]**
- Examples: $ **dig www.suse.com** # compare with:
  $ **dig xym.niwt.bfj** # non-existent server
- Record types: **A** IPv4 address; **CNAME** (Canonical name record=alias); **MX** EMail Server hostname; **NS** DNS Name Server; **PTR** Pointer to CNAME;

netcat Command U2

- **netcat (1)** A service for reading from and writing to network connections using TCP or UDP
- Selected Features:
  - Network debugging tool
  - Shows Outbound or Inbound TCP or UDP to or from any ports
  - Transfers files
  - Port listening
- See for examples: <https://en.wikipedia.org/wiki/Netcat>

References

- **Linux Network Commands** <tldp.org/LDP/GNU-Linux-Tools-Summary/html/c8319.htm>
- **Linux Network Statistics Tools** <cyberciti.biz/faq/network-statistics-tools-rhel-centos-debian-linux/>
- **20 Commandline Tools to Monitor Linux Performance**: <tecmint.com/command-line-tools-to-monitor-linux-performance/>