System and Network security

Saurabh Barjatiya

[[2012-04-18 Wed]] and [[2012-04-20 Fri]]

Contents

Note: Commands and configuration files mentioned below are with respect to Cent-OS / Fedora / Red-hat based systems. Same content should apply to other distributions with minor changes in command options, location of configuration files, etc.

1 Securing own system

This section contains information on tools and techniques that can help in securing system from its own users. It has been further divided into various

sub-sections to categorize large number of tools that help in securing systems.

1.1 Preventing changes

One of the objectives of securing is to prevent unauthorized changes. In this sub-section tools that help in preventing unauthorized changes are discussed. Information on `sudo` which helps in providing granular access to users is also mentioned.

1.1.1 Permissions and Ownership

One of the simple ways of preventing changes in Discretionary Access Control (DAC) systems such as Linux is by creating users and groups and setting proper file permissions. In Linux each file or process is owned by a user and a group, and permissions on file can be configured to specifically allow / deny access to it by user owner, group owner or others.

There are three types of permissions for each file / directory:

- Read (`r`, `4`)
- Write (`w`, `2`)
- Execute (`x`, `1`)

Each of these three permissions are applied to three different categories of users:

- User owner (`u`)
- Group owner (`g`)
- Others (`o`)

Various commands that can be used in manipulating permissions and ownership are:

- `ls -l` can be used to check present ownership and permissions on various files and directories
- `chmod` can be used to change permissions. `chmod` command is very flexible as supports permissions supplied in numeric as well as alphabetic format.
- `chown` can be used to change current user owner and group owner of a file or directory.

- `chgrp` can be used to change just group owner without modifying user owner
- `usermod` can be used to modify primary and set of secondary groups of a given user

Linux man pages can be used to learn about each of above mentioned commands. Detailed description of Linux file permissions is available at

- http://www.zzee.com/solutions/linux-permissions.shtml
- http://www.tuxfiles.org/linuxhelp/filepermissions.html
- http://www.tuxfiles.org/linuxhelp/fileowner.html

1.1.2 Sudo access

Many times there are more than one users and administrators of a system. Keeping track of user and administrator activities in multi-user systems is not easy. To complicate things further sometimes users need to run commands like mount, umount, ping (!), which require root permissions for them to work. In most cases these problems are resolved by making root owner of given program and giving sticky execute permissions to given file. But sometimes it is desired that only specific user can run given command and not all the users of the system. When such fine-grained access control requirements are present, `sudo` can be used to get desired results.

Most Linux distributions come with `sudo` installed. Administrators can use `visudo` command to edit `/etc/sudoers` file (which should not be edited using editors directly) and give permissions to specific users to run specific commands. Users who are granted permission for specific command can login as themselves and use `sudo` command to get work done.

Since `sudo` allows administrators to give permissions to normal users to run command as root. It can be (*and is often*) used as root-kit to gain root access by malicious users on compromised systems. Hence even if `sudo` is not used, it is important to understand how it works and monitor `/etc/sudoers` file for changes, to prevent unwanted access grants to users, enabling them to preform operations as `root`.

- Understanding sudo syntax

  Lines in sudo configuration uses following format:

```
<username> or <group> <machine> = [(<user>)] [NOPASSWD] :<command1> [, <command2>
```

For example:

```
saurabh barjatiya = (root) NOPASSWD: /usr/sbin/wireshark, /usr/sbin/tcpdump
```

will allow user '=saurabh=' to run '=/usr/sbin/wireshark=' and '=/usr/sbin/tcpdump=' commands on machine '=barjatiya=' without supplying password as '=root=' user.

- Common sudo usage scenarios

  - Allow specific user to run all commands as root

    Sometimes we may want admin user to be able to run all commands as root. That can be obtained using

    ```
    <username> ALL=(ALL) NOPASSWD:ALL
    ```

    by replacing <username> with desired username.
  - Allow specific user to run given command / script as root

    The most common usage scenario of sudo is to allow specific user to run given command or set of commands are root

    ```
    <username> ALL= NOPASSWD:<script1>, <command2>
    ```

    Note that we specify machine name as ALL as we do not want to restrict the permissions based on specific machine. Machine based restriction would be useful in networked environment where same `/etc/sudoers` file is used on many machines using some kind of configuration management tool.
    Detailed configuration of sudo can be learned from
    * http://ubuntuforums.org/showthread.php?t=1132821

1.1.3 SELinux

- Introduction to SELinux

  SELinux provides Mandatory Access Control (MAC) based security to Linux systems. SELinux assigns a security context to each file and process. Activities performed by a process are allowed or denied based on configured policy and assigned context. SELinux also supports concept of **sensitivity** and **category**.

  - Example 1 - Apache with SELinux

    For example apache executable (`/usr/sbin/httpd`) on most systems is assigned SELinux context with type `httpd_exec_t`. Due to this whenever apache binary file is executed running process gets context with type `httpd_t`. Now default SELinux policy in most cases has list of ports (`http_port_t`) that a `httpd_t` process is allowed to listen to (TCP 80, 443, 488, 8008, 8009, 8443). If we change apache configuration in `/etc/httpd/conf/httpd.conf` file and change listening port from 80 (`Listen 80`) to 90 (`Listen 90`) and try to start apache as root user then it will fail. Note that in DAC root user never gets 'access denied' or 'permission denied' messages and can do virtually whatever he / she desires. But with SELinux root user cannot start apache on port 90 because of various SELinux contexts and policies.

    root user can modify SELinux contexts and policies to get work done. For example root user can add port 90 to list of ports (`http_port_t`) that apache is allowed to listen to. But even such changes can be prohibited by policy to not take effect until system reboot. Thus unwanted changes in policy would require system reboot which will get noticed and may even require manual intervention to enter passwords (in case encrypted file-systems or bios passwords) are in use, making system very secure.

    A bigger advantage of SELinux is that even if apache is compromised and due to some vulnerability attacker is able to get shell access through apache. Attacker will not be able to use privilege escalation and get root privileges as attackers role will remain to be `httpd_t` and will not change to something else like `unconfined_t` which can be used by root users.

- Example 2 - Categories and Sensitivity

  Consider example of small organization with accountants and security guards as two different categories of people. If accountants and security guards are both provided login accounts on same system then it cannot be guaranteed that accountants cannot share information with security guards and vice-versa in DAC based systems. In DAC user owner can always make any owned resource world readable and world writable (777) for everyone to read and write.

  With SELinux we can assign login roles to all accountants and security guards such that their files automatically get categorized. Then policy can be setup so that accountants cannot access files owned / created by security guards with category of security guards and vice-versa. This rule will get enforced even if someone uses `su -` and tries to convert current login to root login.

  In the same organization there can be security guards at different level of security clearance. That is there can be guards at lower level, than manager level and then some top level manager guards. We may not want information entered by top level manager guards to be accessible by security guards at lower level. At the same time we would want all the information accessible to lower level guard to be accessible by manager level guards and top level guards. This can again be achieved using SELinux by assigning proper **sensitivity** to each document and user role. Support for sensitivity makes SELinux very useful for defense organizations where there is clear hierarchy and concrete access rules.

- SELinux modes

  Learning and using SELinux was fairly complex (and probably to some extent still is), and hence most distributions give ways of running SELinux which are comfortable for users and system administrators. Few common ways of running SELinux are:

  - Permissive mode

    In permissive mode SELinux allows all actions if they are acceptable as per DAC rules but logs or alerts in case an action is

supposed to be denied as per configured policy. Thus all the commands and programs work as they would work without having SELinux with minor difference of logs / alerts being generated on SELinux policy violations.

- Targeted mode

  In Targeted mode only specific programs usually server processes like `apache`, `ssh`, etc. which accept network connections or serve users are run with SELinux security and policy constraints. All the other programs like gnome-terminal or text editors run without any SELinux context. Thus servers are secured with SELinux and normal programs, editors and tools are not affected by SELinux in any way.

- Disabled mode

  SELinux can also be completely disabled. To disable SELinux edit `/etc/sysconfig/selinux` file and change `SELINUX=enforcing` to `SELINUX=disabled`. SELinux can be disabled temporarily (provided configured SELinux policy permits so) by using `setenforce 0` command. It can again be enabled later on without requiring reboot using `setenforce 1`. We can use `getenforce` command to check current SELinux mode.

  As a rule of thumb if you ever get access denied messages when doing something as root, it is always best to check whether the error messages are due to SELinux. A quick (but risky) way of checking it is by disabling SELinux temporarily and try the action again. If action succeeds after disabling SELinux and again fails after enabling SELinux then we can be sure that SELinux is causing things to fail and fix contexts and policies to make things work.

- Conclusion

This was just basic introduction to SELinux. It may not be beneficial for small or medium sized organizations to spend efforts in learning SELinux as it is fairly complex and documentation on SELinux is limited. But it can definitely provide great security to large organizations with many users working with different sensitivity of data in different

branches. It is also definitely useful for defense sectors like police, army etc.

To learn SELinux (not for mere mortals) one can use following links:

- http://www.fosteringlinux.com/category/articles/selinux/
- http://www.vantosh.com/publications/LK2010-SELinux-Tutorial.pdf
- http://www.securitytube.net/video/991

1.1.4 Auto-logout

If system is used remotely and it is desired that idle users get logged out automatically then we can create script `/etc/profile.d/autologout.sh` with following contents:

```
#!/bin/bash

TMOUT=900
readonly TMOUT
export TMOUT
```

Here value 900 is in seconds (equivalent to 15 minutes) and can be modified as per use case.

1.2 Keeping file safe

1.2.1 Versioning

- Subversion

  Sample SVN commands:

  - To create new SVN repository

  ```
  svnadmin create <project_name>
  ```

  - To checkout repository in current folder for working

  ```
  svn checkout file:///<full_path_to_project_folder>
  ```

  - To add files to repository

```
svn add <files>
```

  - To commit changes done to files which are part of repository

```
svn commit -m "<message>"
```

  - To check status of files with respect to last update or commit

```
svn status
```

  - To see commit logs of particular file

```
svn log <filename>
```

  - To update current working copy with latest revision of files from repository

```
svn update
```

  - To get help on svn

```
svn help [<topic>]
```

To learn SVN properly in detail use http://maverick.inria.fr/~Xavier.Decoret/resources/svn/index.h

- git

Sample git commands:

  - To set name and email address in global git config file

```
git config --global user.name "Saurabh Barjatiya"
git config --global user.email "saurabh@sbarjatiya.in"
```

  - To initialize a new folder as git repository

```
git init
```

  - To specify which modified files should be part of next index. This would add specified names to index which would be referred during next commit.

`git add`

- To commit current changes to repository

`git commit -m "<commit message>"`

- To check status of repository. It lists files which are part of index and would be updated on next commit, files which are modified since last commit or checkout but not added to index and files which are not being tracked. It also lists which branch we are working on and unresolved conflicts.

`git status`

- To list difference in file with respect to given commit or tag

`git diff`

- To lists all the commits done so far along with commit messages.

`git log`

- To lists all the branches available for the current project

`git branch`

- To creates a new branch with current state of repository

`git branch <new branch name>`

- To checkout repository files with commit name, branch name or tag

`git checkout {<commit> | <branch> | <tag>}`

- To tag current commit with name so that we can later use this for checkout

`git tag`

- To merge other branch with current branch. If merge results into conflict then we are notified about it. All conflicts have to be resolved before next commit.

`git merge <branch>`

To learn git properly and in-depth use http://book.git-scm.com/

1.2.2 Backup

- `rsync`

  Common rsync options are:

  - `-v` (verbose)
  - `-t` (preserve timestamps)
  - `-r` (recursive)
  - `-p` (preserve permissions)
  - `-a` (archive -rlptgoD)
  - `-z` (compress)
  - `-progress`
  - `-inplace`
  - `-delete`
  - `-partial`

  To understand various options of `rsync` refer to rsync man page (`man rsync`).

- `dd`

  Command dd options are:

  - `if=` (input file)
  - `of=` (output file)
  - `count=` (number of blocks to copy)
  - `bs=` (block size)
  - `skip=` (number of blocks to skip from beginning)

  To understand various options of `dd` refer to dd man page (`man dd`)

  Most common usage of `dd` is to clone entire system to setup lab machines with exact same configuration. Or to take bit-level backup after incident for evidence purposes. It can also be used to write zeros over partition tables to make file system / data recovery very hard, in very short time.

1.2.3 Raid

- Types of raid

  Link - http://en.wikipedia.org/wiki/RAID

- Software raid (`mdadm`)

  Basic mdadm commands:

  – Check status of current raid arrays

```
more /proc/mdstat
```

  – Create a new raid raid 1 array with two members

```
mdadm -C /dev/md3 -n2 -l1 /dev/xvdc1 /dev/xvdd1
```

  – Add given device to existing array

```
mdadm -a /dev/md3 /dev/xxdd1
```

  – Stop given array (No reason to do this)

```
mdadm -S /dev/md3
```

  – Store information of running auto-detected arrays in start-up file.

```
mdadm --detail -sv > /etc/mdadm.conf
```

  – Auto detect raid arrays (Very useful when booting on raid system with Live CDs or rescue mode)

```
mdadm --auto-detect
```

  There are myths that software RAID is very slow and unusable. I have been practically using software RAID on servers since more than two years now and have never faced performance and reliability problems. The only error possible is that sometimes GRUB is not installed on all disks which are part of RAID array and hence system may not boot

when one of the disk fails. This can be easily resolved by booting into rescue mode and enabling bootable flag of all RAID partitions which are part of GRUB array. Then grub has to be re-installed on array using something like `grub-install /dev/md0`. Note that this happens when system with failed disk is rebooted. There is no effect on running system due to failed disk. Hence failure of disk will never lead to data loss or unannounced down time. Once we are aware that disk is failed we can announce downtime of few minutes and reboot the system to replace the failed disk.

Fedora-16 seems to have serious problems with raid. Usage of raid is recommended with server class OS like Cent-OS

- Hardware raid

Whenever available can be preferred over software array. Note that some motherboards display RAID options even when there is no hardware RAID. In such systems same Linux style software raid is loaded on motherboard ROM and part of installed RAM is used for RAID. Such fake hardware arrays are very slow and do not provide flexibility and power of actual software array implemented in Linux. Hence care should be taken to ensure that actual hardware RAID controller is present when using hardware RAID.

1.2.4 Encryption

- Encrypting files (`gpg`, `kgpg`)

  - To generate new keys

```
gpg --gen-key
```

  - To encrypt a file using public key (Note: To keep files safe encrypt using your own public key)

```
gpg -r <name> -e <file_to_encrypt>
```

  - To decrypt file encrypted with our public key using our private key

```
gpg --output <file_name> -d <encrypted_file>
```

- To edit key, for example to change pass-phrase

```
gpg --edit-key <name>
```

This takes to a edit-key menu where one of the options (`passwd`) is for changing key pass-phrase. One can use `help` command to see list of all possible commands.

- To encrypt using symmetric key (using password)

```
gpg -c <file_to_encrypt>
```

- To decrypt file encrypted using symmetric key (password)

```
gpg --output <file_name> -d <encrypted_file>
```

- To sign a file

```
gpg -b -s <file_to_sign>
```

- To verify signature on a file

```
gpg --verify <signature> <file that was signed>
```

- To list all keys

```
gpg --list-keys
```

- To list all public keys

```
gpg --list-public-keys
```

- To list all secret keys

```
gpg --list-secret-keys
```

Note that all the above operations can be done very easily using `kgpg` GUI too. The commands are listed so that operations can be scripted / automated. GPG also works using `cmd` in windows equally well and with exact same command line options.

- Encrypted file systems (`encfs`)

  `encfs` can be installed using `yum installed fuse-encfs` provide rpmforge, epel etc. repositories are properly configured. Then we can use

  ```
  encfs <full_path_of_encrypted_dir> <full_path_of_work_dir>
  ```

  When the command is used for first time we can configure password and choose how to encrypt. Default security options for encryption are good enough. All future mounts of encrypted dir would require same password. There is a `.encfs<n>` file in each encrypted dir that is used to store information on how to decrypt.

  To change password used for storing `.encfs<n>` file we can use

  ```
  encfsctl <raw_directory>
  ```

  Note that by default only root user is allowed to access contents of work dir when a encrypted file system is mounted. To allow non-root user to access work dir we must use `-public` option while mounting encrypted raw dir to work dir.

  Sometimes encfs may give error on fully updated older versions of CentOS while initializing new directory for encryption due to Boost C++ library errors. In such cases one can use older `.encfs5` file from existing encrypted folders with same / different password to achieve encryption.

  Encfs is available for windows too. For setup instructions refer to Installation notes at end of lecture notes. Syntax of usage of `encfs` in Windows using `cmd` is exactly same as that in Linux.

1.3 Monitoring what is going on

1.3.1 Tools

- Monitoring resource usage by various processes (`top`, `htop`, `atop`)

If users face performance and response time problems with a system then they can use `top` to get list of processes running on system arranged in descending order of their CPU usage. In `top` output we can look at top three four most CPU consuming processes and kill / stop them if they are not necessary.

Sometimes system maintenance processes like `makewhatis`, `prelink`, `updatedb` etc. which are run periodically to maintain system in good health and update various databases consume lot of CPU. We should avoid killing these processes to improve response time as these processes help in proper functioning of the system.

If users are not sure what top CPU using processes are for and their purpose, they should first search using Internet search engines to determine the purpose of these processes before deciding to terminate them.

- Memory usage

  If users are facing memory usage problems (confirmed by `free -m` command) then they can press 'F' followed by 'n' to sort the processes in decreasing order of memory usage and determine which processes are using most memory and restart / stop those processes.

  In order to see whether lack of memory is causing problems for system performance one can use `free -m` command and look at usage of swap usage. If more than 100 MB of swap is being used on a desktop system which has not been running for more than a week then system is most likely suffering from memory related issues.

- Heavy hard-disk I/O

  If some process uses considerable hard-disk then it will slow down complete system. To determine whether hard-disk usage is causing system to run slow, one can look at `wa` percentage shown by top in first few lines. Whenever this percentage is very high (more than 30%) continuously for long duration then system is suffering from high disk I/O problem.

  Once it is determined that I/O is problem usually the process consuming most I/O will also get listed as among most CPU consuming processes in top output. But to get detailed per-process

disk usage one can use `atop` command, which can give per process disk usage when 'd' key is pressed after running `atop`.

- Too many processes

  One of the parameters in measuring system performance is load average. `top` shows load average in its first line. We can also find load average with help of `uptime` and `w` commands. Load average in approximate sense indicates how much each process has to wait before it gets its turn to execute on one of the CPUs. If this value is close to 1.00 or worse greater than 1.00 then system is facing problem of having too many processes.

  Three values of load average are maintained by system: 1 min, 5 min and 15 min. Hence if there is some persistent load / process problem on a system then its load average of last 15 min would be near 1.00 indicating serious problem.

  If command line based `top` is not easy to use then one can also try its GUI counterpart `gnome=system=monitor` which also provides similar information. `htop` which is more colorful can be used in case GUI is not available.

- Monitoring open files (`lsof`)

  `lsof` command has to executed with superuser / root privileges. It gives list of all open file descriptors by all processes running on system. This information is useful in checking which files are being used by a given process. We can check files opened by suspected processes and stop them in case they seem to be opening files which they are not supposed to.

- Monitoring listening ports, sockets and established connections (`netstat`)

  `netstat` gives two very important set of information related to network and current system:

  - List of ports on which some program is listening for incoming connections along with name of program for each particular port
  - List of established connections for each program including source IP, source port, destination IP and destination port information.

To get detailed information on which process is listening on given port and gets its PID we via netstat, we need to run netstat with superuser / root privileges. We can run netstat without root privileges but in that case we will only get information of open ports and established connections but not about which process is responsible for given connection / port.

Once we have network status information, we can determine whether a particular process should be listening on given port or not. We can also verify whether a given application requires a given network connection to some remote host or not and act accordingly.

- Looking at running connections / network data (`tcptrack`, `iptraf`)

  We can use `tcptrack` command to monitor running tcp connections and bandwidth used by them. This command allows sorting of connections in descending order of bandwidth usage and hence helps in pin-pointing which tcp connections are using most bandwidth.

  We can use `iptraf` to get interface usage statistics. We can also use various other modes of `iptraf` to understand which machines in LAN are communicating with current machine most. We can also filter traffic of particular port / IP and see its usage.

- Details of users logged in on system and what they are doing (`w`)

  It is important to see if there are other users logged on our system and what they are doing. `w` commands help in listing other users logged on our system, username with which they have logged in and their IP addresses. It also tells when they logged in and since how long they have been idle.

- Process information (`ps`)

  We can get important information about any process using `ps` command. Major ways of using `ps` are described below:

  Filtering: * `ps -C <process_name>` :: To get information about process with given name * `ps -p <pid>` :: To get information about process with given Pid. * `ps -u <username> -U <username>` :: Select process of specific user * `ps aux` :: Select all processes

Output control: * `ps -f` :: Full format listing * `ps -F` :: Extra full format listing * `ps -o pid,command` :: To get only specific columns of information

Refer to `man ps` for detailed information on various uses of `ps` command.

- Looking at login logs (`last`, `lastb` and `lastlog`)

  `last` command lists last successful login on current system for this month and duration of each login. We can find details of older login with older versions of `wtmp` file (`wtmp.1`, `wtmp.2`, etc.) present in `/var/log` folder.

  `lastb` command lists last unsuccessful login attempts for some user from remote machine or using GUI / console on same machine. This information is useful in determining if someone has tried to login on our system with various different passwords (brute-force / dictionary attack).

  `lastlog` command lists time of last login for each user on system. This helps in finding out inactive users who have not logged in since many months, so that their accounts can be locked, deactivated or even deleted.

- Blocking brute-force login attempts (`denyhosts`)

  Any system on Internet with port 22 (SSH) gets targeted for brute-force login attempt within few hours after coming on-line. There are two very useful ways of disabling SSH brute-force attacks on public servers: * `denyhosts` * `port knocking`

  We will look at `port knocking` when we study `iptables` firewall. Denyhosts is set of script which periodically monitors login logs `/var/log/secure*` and blocks IP addresses from where there are too many bad login attempts by entering their details in `/etc/hosts.deny` file.

  Denyhosts is configured via `denyhosts.cfg` file. Following values can be changed from default to suit ones need:

```
PURGE_DENY = 12w
PURGE_THRESHOLD = 2
BLOCK_SERVICE = ALL
```

```
DENY_THRESHOLD_ROOT = 10
ADMIN_EMAIL = barjatiya.saurabh@gmail.com
SYSLOG_REPORT = YES
AGE_RESET_VALID = 2d
AGE_RESET_ROOT = 2d
RESET_ON_SUCCESS = YES
DAEMON_SLEEP = 120S
(Uncomment) 'SYNC_SERVER = ...' line
SYNC_DOWNLOAD_THRESHOLD = 10
```

We can also white-list our own IP addresses so that we do not get blocked no matter how many times we enter wrong password. For this we need to edit file `allowed-hosts` which can be created in denyhosts data directory (usually `/usr/share/denyhosts/data`). Entries should be made with one IP per line so that all the mentioned IPs do not get blocked no matter how many bad attempts they make.

Please note that removing blocked IP is not easy as just removing entry from `/etc/hosts.deny` will not work, because denyhosts will re-add the entry on next periodic run. To really remove blocked entry we have to either white list it or remove its bad login attempts from all log files present in denyhosts data directory.

- `service` and `chkconfig`

Many servers and processes are provided in form of services so that they automatically start at system boot and continue to run in background without getting affected by user login / logout events. The start-up / stop commands for each of these scripts are mentioned in executable shell scripts kept in `/etc/init.d` folder. To configure which services will start and which will stop in given run-level, symbolic links of these script files are created in `/etc/rc.d/rc<n>.d` folders.

To help with easy configuration of these services to commands: `service` and `chkconfig` are provided on many Linux distributions. With help of `service` command we can start, stop, restart, etc. any service that is available on system. With help of `chkconfig` command we can enable / disable automatic starting of particular service on given run-level.

To keep system safe and to make it boot faster, it is best to disable all unwanted services from system. One can use following command as

root use to find services that will automatically start in run-levels 3 to 5 on current system

```
chkconfig --list | grep '[3-5]:on'
```

One can then find out details of each service using search engines or description mentioned in `/etc/init.d/<service>` files and choose to stop them from running on start-up using

```
chkconfig <service> off
```

To stop/start any service just for current boot we can use: service <service-name> {start | stop | restart}

- Information about typical services

  In order to utilize knowledge of service / chkconfig properly, it is required to know purpose of many services that get installed with Cent-OS full installation. Many important services and their description are mentioned below to help users judge which services are critical and which can be stopped to improve system performance.
  The list is prepared using Cent-OS 5.8. Cent-OS 6.2 (latest at time of writing) has introduced many new services. One can always refer to /etc/init.d/<service> file to understand purpose of particular service. One can also use `system-config-services` *GUI to configure services.*

S. No	Service Name	Description
1	NetworkManager	This service is used so that NetworkManager applet can be used to configure network without requiring root privileges. This also ends up providing easy and intuitive GUI interface for network configuration in status bar. For servers we can disable NetworkManager and use `NM_CONTROLLED=no` for interfaces that we want to be initialized on system boot, without requiring user login.
2	acpid	This service helps with ACPI events. ACPI helps software to control power to various components. In simplest case ACPI helps computer to power itself off after shutdown.
3	anacron	There are two cron services anacron and crond (vixie cron). Both services together help in running scripts periodically (daily, weekly, monthly, etc.) to keep system in good state. anacron is useful for desktops which are not always on and may remain powered off during configured cron run time for few services. Anacron understands that particular service should be run regularly at given intervals and tries to achieve that even if system is powered off for majority of time during day.
4	atd	atd stands for at daemon. It is used by at command to run spe commands at specific date / time. Like cron is used to run scri / programs at specific intervals. at can be used to run some command on specific date / time once.
5	atop	If `atop` is installed and used then this service monitors various parameters like CPU usage, RAM usage, disk usage by various processes so that same can be displayed when atop command i
6	auditd	auditd service helps in collecting audit logs in format which ca be searched using `ausearch` etc. audit related commands. If th service is not run then audit logs get sent to syslog service.
7	autofs	This service helps in automounting filesystem on use. This is v useful in central login setups where all users home directory is on shared NFS server. In such cases autofs can be used to automatically mount user home folder during login and automa umount it when it is not used for long time. This helps in keep network file-system usage efficient as they get mounted only wh required and get unmounted when work is over. autofs configu is done with help of `/etc/auto.master` and many other suppo configuration files.
8	avahi-daemon 23	This helps in Zeroconf service discovery. In lay mans terms it helps in initializing system when there is no static IP configure and DHCP etc. are also not available using Zeroconf discovery protocol which uses multicast for service discovery. I would recommend to disable this service from start-up for security rea and Zeroconf service discovery works based on trust and should be used in untrusted environment
9	bluetooth	This service can be used for bluetooth mouse / keyboard. Unle

- Hard-disk usage (`df -h`, `quota`)

  We can use `df -h` command to see hard-disk usage of various partitions. It should be noted that failing of services due to no space available on critical partitions like / or /var is very common for servers that are not managed properly. Logwatch reports always contain `df -h` output to keep system administrators updated on usage of various partitions.

  When users do not seem to use disk space fairly or properly disk quotas can be implemented to ensure that system does not fails / other users do not get affected due to mis-use of disk space by few users.

  – Enabling disk quotas

    Steps:

    * To enable disk quotas add `usrquota` option in `/etc/fstab` file for partitions where quota is desired
    * Then use `mount -o remount <partition>` command to remount partition with quotas enabled.
    * Then use `quotacheck -vcf -F vfsv0 <partition>` command to create new quota file for given partition. Note that this command will take long time if there is already considerable data on current partition.
    * Then use `quotaon -av` to enable quota check / monitoring on all filesystems mounted with `usrquota` option.
    * We also have option of `grpquota` to have group based quotas.
    * Various soft limits and hard limits for files and blocks can be set using `edquota -u <username>` command after disk quotas have been set properly.
    * We can use `repquota -a` to see current quota usage of various users.
    * If quota calculations look very incorrect then fresh calculations can be done using following steps
      · Disable quotas using `quotaoff -av`
      · Recheck quota on read-only file-systems using `quotacheck -vbf -F vfsv0 <mount_point>` otherwise recheck quota on file systems in use using `quotacheck -vbfm -F vfsv0 <mount_point>`

· Enable quota again using `quotaon -av` command when quotacheck command is completed.

- Information about hard-disk (`smartctl`, `hdparm`)

  Most modern drivers are SMART (Self-Monitoring, Analysis and Reporting Technology) capable and hence report errors and problems by performing self checks without requiring user intervention. To get best results from SMART capable drives we should leave `smartd` service enabled on start-up and read root users mail frequently.

  – Finding information about hard-disk

    We can use

    ```
    hdparm -I <drive_device>
    ```

    hdparm -tT <$\text{drive}_{\text{device}}$>to find considerable useful information about hard-disk

  – Finding drive speed

    We can use

    ```
    hdparm -tT <drive_device>
    ```

    to get drive speed. **This is not ideal way for finding performance of RAID arrays.**

  – Finding drive SMART related information

    To find all drive SMART related information use:

    ```
    smartctl --all <device_name>
    ```

    This gives considerable important information like status of last self test, power on hours, drive temperature etc.

  – Configuring hard-drive for self test

    We can start complete hard-disk surface self test using

    ```
    smartctl -t long <device_name>
    ```

We can check result of self test when test is complete using

```
smartctl -l selftest <device_name>
```

- Checking connected hardware (`lspci`, `lsusb`)

  We can use `lspci` command to see various PCI devices LAN cards, sound cards, etc. connected to system. We can use `lspci -t` command to see hierarchy of how these devices are connected to each other.

  `lsusb` can be used to see list of various USB device connected to system. We can use `lsusb -v` to get very detailed information about all USB devices connected to system.

  If we are worried about external devices being connected to system without permission then we can have scripts monitor `lsub` and `lspci` output periodically to detect and notify about new hardware being connected to systems.

  Now a days there is considerable security threat due to very small spy pen drives and USB wifi devices. In such cases just disabling USB at BIOS level may not be enough specially if USB printer, keyboard, mouse etc. are going to be used. Hence having custom scripts to monitor additional USB and PCI devices is required.

1.3.2 Remote monitoring

- SNMP

  SNMP protocol helps in remote monitoring and management of devices. With help of SNMP we can configure various parameters and also get values (monitor) of large number of parameters like CPU usage, network usage, disk usage, running connections, packet sent, etc.

  We can configure SNMP-server on Linux so that it starts responding to SNMP queries and can be remotely monitored via SNMP. Steps to configure SNMP server on Linux are:

  – Ensure that packages 'net-snmp' and 'net-snmp-utils' are installed.
  – Start snmpd service and enable it on start-up using 'service snmpd start' and 'chkconfig snmpd on'

- Use command 'snmpwalk -v 1 -c public localhost IP-MIB::ipAdEntIfIndex' to query snmp server and list set of IP addresses assigned to it. Replace localhost with server IP and public with community name, if required. If server is already configured then interface list will get printed and we do not need to modify configuration. If nothing is printed then we can make following changes in configuration file '/etc/snmp/snmpd.conf':
  * Find ' com2sec notConfigUser default public' and replace with
    - `com2sec local localhost public`
    - `com2sec mynetwork 10.0.0.0/8 public`
    - `com2sec mynetwork 172.16.0.0/12 public`
    - `com2sec mynetwork 192.168.0.0/16 public`
  * Find 'group notConfigGroup v1 notConfigUser' and 'group notConfigGroup v2c notConfigUser' and replace with
    - `group MyRWGroup v1 local`
    - `group MyRWGroup v2c local`
    - `group MyRWGroup usm local`
    - `group MyROGroup v1 mynetwork`
    - `group MyROGroup v2c mynetwork`
    - `group MyROGroup usm mynetwork`
  * Find 'view systemview included system' and replace with
    - `view all included .1`
  * Find 'access notConfigGroup "" any noauth exact systemview none none' and replace with
    - `access MyROGroup "" any noauth exact all none none`
    - `access MyRWGroup "" any noauth exact all all none`
  * Find 'syslocation Unknown (edit /etc/snmp/snmpd.conf)' and 'syscontact Root (configure /etc/snmp/snmp.local.conf)' and replace with
    - `syslocation Lab320, 10.3.3.230, VM`
    - `syscontact Saurabh Barjatiya <barjatiya.saurabh@gmail.com>`
- Do '=service snmpd reload='
- Again check using '=snmpwalk -v 1 -c public localhost IP-MIB::ipAdEntIfIndex=' and verify that IP address are getting listed. Replace localhost with server IP and public with community name, if required.

- We can also use '=snmpwalk -v 1 -c public localhost .1 | less=' to see entire list of parameters that can be queried. WARNING: The list can be very large.
- Block incoming packets on UDP ports 161, 162 from unknown hosts which should not be able to query snmp-server installed on the host.

- MRTG

  After configuring Linux machine as SNMP server, it is often desired to periodically poll the server for interesting parameters and generate graph of values obtained for various time intervals. MRTG is one such tool which can query SNMP servers and draw graphs for various interesting parameters.

  - Using MRTG to monitor network interfaces

    Steps to configure basic MRTG based monitoring server are:

    * Use command '=snmpwalk -v 1 -c <community> <host-name> IP-MIB::ipAdEntIfIndex=' to query SNMP server and list set of IP addresses assigned to host / router. If it does not lists anything then we need to configure SNMP server for hosts as mentioned before in these notes.
    * Ensure that package '=mrtg=' is installed.
    * For each host to be monitored follow these steps:
      · Create host specific document root using '=mkdir /var/www/html/<IP or name>='
      · Create configuration file using

```
cfgmaker --global 'WordDir: /var/www/html/<IP or name>' --output /etc/mrtg/<I
```

    * Create index page for MRTG configuration

```
indexmaker --output=/var/www/html/<IP or name>/index.html /etc/mrtg/<IP or na
```

    * Copy mrtg png files to document root using '=cp /var/www/mrtg/* /var/www/html/<IP or name>/='
    * Run mrtg few times using:
      · `env LANG=C /usr/bin/mrtg /etc/mrtg/<IP or name>.cfg`
      · ignore errors shown during first few runs

* Use '=service httpd start=' to start web server if it is not already running.
* Test pages by visiting '=http://<MRTG server IP>/<Monitored host IP or name>/index.html='
* If everything is fine add following cron entry using '=crontab -e=' command

```
*/5 * * * * env LANG=C /usr/bin/mrtg /etc/mrtg/<IP or name>.cfg -logging /var
```

* Optionally restrict to MRTG graphs using apache '=.htaccess=' configuration. Create file '=/var/www/html/<IP or NAME>/.htaccess=' with following lines

```
AuthName "MRTG Graphs/Html restricted access"
AuthType Basic
AuthUserFile /var/members/.htpasswd
require user mrtgadmin
```

* To create '=/var/members/.htpasswd=' file use: (one time only)

```
htpasswd -c /var/members/.htpasswd mrtgadmin
```

* In case server is not asking password then add following to '=/etc/httpd/conf/httpd.conf=' and then do '=service httpd reload='

```
<Directory "/var/www/html/<IP or Name>">
Options All
AllowOverride All
</Directory>
```

– Using MRTG to monitor host CPU, disk, memory etc.

To monitor various parameters like CPU, disk, memory etc. with MRTG use following configuration file:

```
#
# File: /etc/mrtg/server-info.cfg
#
# Configuration file for non bandwidth server statistics
#
#
```

```
# Define global options
#
LoadMIBs: /usr/share/snmp/mibs/UCD-SNMP-MIB.txt,/usr/share/snmp/mibs/TCP-MIB.
workdir: /var/www/html/127.0.0.1-stats/
#
# CPU Monitoring
# (Scaled so that the sum of all three values doesn't exceed 100)
#
Target[server.cpu]:ssCpuRawUser.0&ssCpuRawUser.0:public@localhost + ssCpuRawS
Title[server.cpu]: Server CPU Load
PageTop[server.cpu]: <h1>CPU Load - System, User and Nice Processes</h1>
MaxBytes[server.cpu]: 100
ShortLegend[server.cpu]: %
YLegend[server.cpu]: CPU Utilization
Legend1[server.cpu]: Current CPU percentage load
LegendI[server.cpu]: Used
LegendO[server.cpu]:
Options[server.cpu]: growright,nopercent
Unscaled[server.cpu]: ymwd
#
# Memory Monitoring (Total Versus Available Memory)
#
Target[server.memory]: memAvailReal.0&memTotalReal.0:public@localhost
Title[server.memory]: Free Memory
PageTop[server.memory]: <h1>Free Memory</h1>
MaxBytes[server.memory]: 100000000000
ShortLegend[server.memory]: B
YLegend[server.memory]: Bytes
LegendI[server.memory]: Free
LegendO[server.memory]: Total
Legend1[server.memory]: Free memory, not including swap, in bytes
Legend2[server.memory]: Total memory
Options[server.memory]: gauge,growright,nopercent
kMG[server.memory]: k,M,G,T,P,X
#
# Memory Monitoring (Percentage usage)
#
Title[server.mempercent]: Percentage Free Memory
PageTop[server.mempercent]: <h1>Percentage Free Memory</h1>
Target[server.mempercent]: ( memAvailReal.0&memAvailReal.0:public@localhost )
```

```
options[server.mempercent]: growright,gauge,transparent,nopercent
Unscaled[server.mempercent]: ymwd
MaxBytes[server.mempercent]: 100
YLegend[server.mempercent]: Memory %
ShortLegend[server.mempercent]: Percent
LegendI[server.mempercent]: Free
LegendO[server.mempercent]: Free
Legend1[server.mempercent]: Percentage Free Memory
Legend2[server.mempercent]: Percentage Free Memory
#
# New TCP Connection Monitoring (per minute)
#
Target[server.newconns]: tcpPassiveOpens.0&tcpActiveOpens.0:public@localhost
Title[server.newconns]: Newly Created TCP Connections
PageTop[server.newconns]: <h1>New TCP Connections</h1>
MaxBytes[server.newconns]: 10000000000
ShortLegend[server.newconns]: c/s
YLegend[server.newconns]: Conns / Min
LegendI[server.newconns]: In
LegendO[server.newconns]: Out
Legend1[server.newconns]: New inbound connections
Legend2[server.newconns]: New outbound connections
Options[server.newconns]: growright,nopercent,perminute
#
# Established TCP Connections
#
Target[server.estabcons]: tcpCurrEstab.0&tcpCurrEstab.0:public@localhost
Title[server.estabcons]: Currently Established TCP Connections
PageTop[server.estabcons]: <h1>Established TCP Connections</h1>
MaxBytes[server.estabcons]: 10000000000
ShortLegend[server.estabcons]:
YLegend[server.estabcons]: Connections
LegendI[server.estabcons]: In
LegendO[server.estabcons]:
Legend1[server.estabcons]: Established connections
Legend2[server.estabcons]:
Options[server.estabcons]: growright,nopercent,gauge
#
# Disk Usage Monitoring
#
```

```
#Target[server.disk]: dskPercent.1&dskPercent.2:public@localhost
#Title[server.disk]: Disk Partition Usage
#PageTop[server.disk]: <h1>Disk Partition Usage /home and /var</h1>
#MaxBytes[server.disk]: 100
#ShortLegend[server.disk]: %
#YLegend[server.disk]: Utilization
#LegendI[server.disk]: /home
#LegendO[server.disk]: /var
#Options[server.disk]: gauge,growright,nopercent
#Unscaled[server.disk]: ymwd
```

Note:

* Replace localhost with 'server IP' and 'public' with community name. Also change workdir appropriately.
* Use command '=env LANG=C /usr/bin/mrtg /etc/mrtg/127.0.0.1-stats.cfg=' to collect statistics. Same command with '=–logging=' option can be used in crontab.
* Create index using '=indexmaker –output=/var/www/html/127.0.0.1-stats/index.html /etc/mrtg/mrtg.cfg /etc/mrtg/127.0.0.1-stats.cfg=' command.

- Nagios

Nagios is network monitoring tool which can monitor status of various devices using ping and status of various TCP based services like SMTP, SSH, HTTP, FTP etc. by trying to connect to remote machine at given port. If nagios does not receives ping replies or is not able to connect to remote machine at given port then it reports the same on nagios web console and sends emails about same to administrators (if configured to do so).

One of the very useful features of nagios is support for parent configuration. With help of parent we can define which other nodes depend upon parent and failure of child nodes is not reported if parent itself has failed. This helps in monitoring switches / network devices without worrying about getting too many failure alerts / emails when parent device fails.

– Basic nagios server configuration

We can use following steps to configure basic nagios server:

* Configure proper contact information in '=/etc/nagios/objects/contacts.cfg' file - Edit file '`/etc/httpd/conf.d/nagios.conf`' and allow administrators machine to access nagios web interface. Note location of `htpasswd.users` file mentioned in this configuration file.
* Use '=htpasswd -c /etc/nagios/htpasswd.users nagiosadmin=' to configure nagiosadmin password
* Use '=nagios -v /etc/nagios/nagios.cfg=' to check nagios configuration
* Use '=service nagios start=' to start nagios. Optionally enable it on start-up using '=chkconfig='.
* Open 'http://<server-ip>/nagios' and login using 'nagiosadmin' and configured password

– Configuring service checks

Commands for checking common services are provided by nagios.
If we want we can add custom service checks to file '=/etc/nagios/objects/commands.cfg='.
Steps for configuring service check are:

* Create folder for particular type of checks. For example hosts for checking hosts, switches for checking switches, etc.
* Edit file `/etc/nagios/nagios.cfg` and add one line for each folder created for configuration files. For example add `cfg_dir=/etc/nagios/servers` if you have created `/etc/nagios/servers` folder to store configuration files that will be used to monitor servers.
* Create a `.cfg` in configuration folder for each host for easy management. In .cfg file use following configuration for defining a host

```
define host{
        use                     linux-server  ; Name of template
        host_name               students      ; Short name of server, will be
        alias                   students.iiit.ac.in ; Long name of server, wi
        address                 students.iiit.ac.in ; FQDN or IP address of h
        }
```

* Add host created in previous step to hostgroup configured in '=localhost.cfg=' file. After adding few hosts hostgroup configuration may look like:

```
define hostgroup{
        hostgroup_name  linux-servers ; The name of the hostgroup
        alias           Linux Servers ; Long name of the group
        members         localhost,labpc,students,ss1  ; Comma separated list
        }
```

* Configure service checks starting with the simplest ping check. The service check configuration can be put in host specific configuration file for easy management. Ping check configuration lines are

```
define service{
        use                             generic-service         ; Name of ser
        host_name                       students
        service_description             PING
        check_command                   check_ping!100.0,20%!500.0,60%
        notifications_enabled           1
        }
```

* Verify that we have not broken nagios configuration using '=nagios -v=' periodically. Also do '=service nagios reload=' for changes to take effect and appear on web interface.
* After ping check has been configured, we can configure SSH service check using:

```
define service{
        use                             generic-service         ; Name of ser
        host_name                       ss1
        service_description             SSH
        check_command                   check_ssh
        notifications_enabled           1
        }
```

* Configuration for HTTP service check is:

```
define service{
        use                             generic-service         ; Name of ser
        host_name                       ss1
        service_description             HTTP
        check_command                   check_http
        notifications_enabled           1
        }
```

* Configuration for FTP service check is:

```
define service{
        use             generic-service         ; Inherit default values from
        host_name               ss1
        service_description     FTP
        check_command   check_ftp
        }
```

* Configuration for SMTP service check is:

```
define service{
        use             generic-service         ; Inherit default values from
        host_name               students
        service_description     SMTP
        check_command   check_smtp
        }
```

* Configuration for IMAP service check is:

```
define service{
        use             generic-service         ; Inherit default values from
        host_name               students
        service_description     IMAP
        check_command   check_imap
        }
```

* Configuration for POP3 service check is:

```
define service{
        use             generic-service         ; Inherit default values from
        host_name               students
        service_description     POP3
        check_command   check_pop
        }
```

– Configuring parents

To configure parent just one line in host definition stating parents of the host is enough. Example host configuration with parent value defined is:

```
define host{
        use             generic-switch          ; Inherit default values from
        host_name       10.4.4.251              ; The name we're giving to th
        alias           Linksys SGE 2000 switch kept in GH ground floor, This
        address         10.4.4.251              ; IP address of the switch
        hostgroups      switches,linksys-switches               ; Host groups
        parents         172.16.30.1
        }
```

1.3.3 Capturing packets

- Capturing packets at command line (`tcpdump`)

  - About `tcpdump`

    `tcpdump` is a very powerful command-line based packet capturing tool. It can be used to display information about captured packets on screen or capture packets in pcap file format for later analysis. tcpdump has very small footprint and can be used to capture packets even when there is heavy network I/O. 'tcpdump' accepts filters in kernel filter format so that only the packets we are interested in get captured very efficiently. tcpdump uses `libpcap` for its packet capture operations.

  - Useful command line options

    Some very useful command line options supported by tcpdump are:

Option	Description
-c	Stop capturing after count number of packets have been captured
-C	Do not store more then this many MBs (1,000,000 bytes) in single capture file. When file size exceeds it starts new file with suffixes .1, .2 etc. added between consecutive files.
-D	List interfaces available for capturing along with description.
-i	interface name or number (as shown by -D) on which packets should be captured. Note by default it will choose lowest numbered interface which is up and not a loop-back device.
-n	Do not convert IP address to host-names via reverse DNS look-up. This option is very important if we are capturing packets on heavy traffic links to avoid too many DNS look-ups which may affect packet capture or generate significant DNS traffic.
-nn	If performance is issue or port numbers are preferred over service names then we can use -nn to avoid converting of port numbers to service names, like 22 to ssh or 80 to http. This does not generates any additional traffic as mostly file /etc/services would be used to convert port numbers to service names, but can require some small processing.
-p	Do not put interface in promiscuous mode. Note that interface can already be in promiscuous mode and in that case tcpdump would end up capturing packets meant for other hosts in hub like networks.
-r <file>	Read packets from given file and not from live interface.
-s <size>	Capture only first specified number of bytes of each packets. This is useful if we are interested only in protocol (TCP/IP/UDP, etc.) headers and not in application payload. To capture entire packet the size or snaplen can be specified as '0'
-v	Generate verbose output. We can use -vv or -vvv to increase verbosity level
-q	Generate quieter (lesser) output
-w <file>	Write output to file.
-A	Print information contained in packets in ASCII format
-x	Print information contained in packets in hexadecimal format.

Note:

* A pseudo-interface named all is also shown among with other interfaces. But capturing on any interface has limitation that it can be done only in non-promiscuous mode. We cant cap-

ture packets on any interface in promiscuous mode.

  * If we want to capture packets only meant for current host then we can use filter 'ether host <host-mac-address> or ether broadcast'. This would work even if interface is in promiscuous mode.
  * We can specify filename as '-' to -r or -w options so that input is taken from stdio or output is written to stdout.

– `tcpdump` filter format

We can specify filters (conditions which must be satisfied) for the packets to be captured. Various filter options are:

Expression	Meaning
host <ip-address>	Only packets to or from the specified IP address are captured
src host <ip-address>	Only packets with matching source IP address are captured
dst host <ip-address>	Only packets with matching destination IP address are captured
port <number>	Only packets with source/destination TCP/UDP port specified as argument are captured.
src port <number>	Only packets with source TCP/UDP port specified as argument are captured.
dst port <number>	Only packets with destination TCP/UDP port specified as argument are captured.
<protocol>	Only packets of mentioned protocol will be captured. Accepted protocol names are ip, arp, rarp, tcp, udp, wlan, ip6 and ether.
and, or, not	We can combine multiple expressions with and, or, not
ether host <mac-address>	Allow only with matching source or destination mac address.
ether src <mac-address>	Capture packets only with specified source mac address
ether dst <mac-address>	Capture packets only with specified destination mac address
gateway <host>	Packet was sent or received via host as gateway. Note for this the information about host's MAC address must be present in /etc/ethers' file. Also host must be either resolvable by DNS or its IP information should be mentioned in '/etc/hosts' file.
net <network-number>	Captures packets only when source/destination IP belongs to given network.
net <net> mask <netmask>	Packet matches network with specified netmask specified in dotted decimal format
net <net>/<len>	Packet matches network with specified netmask using bit-mask length notation.
portrange <port1>-<port2>	Port number lies within given range.
src portrange <port1>-<port2>	Source port lies within given range
dst portrange <port1>-<port2>	Capture if destination port lies within given range
less <length>	Capture if packet length is less then specified length

greater <length>	Capture if packet length is greater then specified length
ether broadcast	Capture if ethernet broadcast packet
ip broadcast	Capture if IP broadcast packet
ether multicast	Capture if ethernet multicast packet
ip multicast	Capture if IP multicast packet
vlan <vlan-id>	Capture if the packet is an IEEE 802.1Q VLAN packet. If <vlan_{id}> is specified, only true if the packet has the specified vlan_{id}. Note that the first vlan keyword encountered in expression changes the decoding offsets for the remainder of expression on the assumption that the packet is a VLAN packet. The vlan [vlan_{id}] expression may be used more than once, to filter on VLAN hierarchies. Each use of that expression increments the filter offsets by 4. Read man page to understand this option properly

Note:

* We can combine protocol (ip, tcp, etc.), direction (src, dst) and port in single expressions like 'tcp dst port 80'
* There is also very powerful indexing operation to access byte at specific location in packet which then can be compared using <, <, >=, <=, =, !=, &, | etc. C language operators with other byte or decimal constant. Complete information on this can be found in man page.

Additional information (including above mentioned information) can be found at tcpdump man page

- Capturing and analyzing packets using GUI (`wireshark`)

`wireshark` is packet capturing and analysis tool with GUI interface. Wireshark is very powerful network forensic tool which understands many diverse protocols which are used over Internet. Wireshark also provides statistics, Protocol specific details, display filters, option for TCP reassembly, etc. making it very useful and unique. Wireshark same as tcpdump uses `libpcap` / `dumpcap` and hence accepts same set

of capture filters as tcpdump. Wireshark also provides option for more sophisticated display filters.

One needs administrator or root privileges to run `wireshark`. It is very intuitive to use and one can learn considerable usage of `wireshark` by playing around with it for few hours. `wireshark` is also available for Windows.

1.4 Logging actions

1.4.1 Auditing

A detailed report on auditing is hosted at http://www.sbarjatiya.in/website/courses/2010/monsoon/pro
Interested users can download the report and use it to learn how to use Linux auditing.

1.4.2 Command history

We can create file named `/etc/profile.d/history.sh` with following contents:

```
#!/bin/sh

HISTTIMEFORMAT="%y %m %d %T "
HISTSIZE=100000
HISTFILESIZE=100000
export HISTTIMEFORMAT HISTSIZE HISTFILESIZE
```

to ensure that all commands entered by all users get logged with date and time. This also ensure that last 100000 (practically all) commands get logged and not just last 1000. If root access is not available then the same configuration can be done in `.bash_rc` file of user account so that at least history of given user commands is stored with date and time information

1.4.3 Remote logging

- **TODO** `rsyslog`

One of the very important tools for reliable information after system is compromised is logs. But if system is compromised we can't depend on log files as they could have been altered or deleted. Hence it is

important to store log of all important servers on some different (probably central) log server which is extremely hardened and very tightly controlled. This log server can then be used to analyze causes and sources of break-in.

To send logs to remote machine or to receive logs from remote machine we can use rsyslog. rsyslog configuration can be learned from http://rsyslog.com/doc/rsyslog_conf.html One can also try to use from many samples provided at http://wiki.rsyslog.com/index.php/Configuration_Samples after browsing through basic rsyslog configuration documentation.

1.5 Analyzing logs

1.5.1 Periodic log reports (`logwatch`)

Script `/etc/crond.daily/0logwatch` runs daily due to cron and sends email report of log analysis. This report contains important information like

- Access denied or File not found error messages sent by web server
- Hacking attempts detected on web server
- Disk space analysis
- Commands run as sudo
- Mail server statistics if mail service is configured

Since these reports are sent by email proper configuration of local SMTP server (like `sendmail`) is required so that this emails reach administrator without any problem. If email system or aliases are not configured properly all these reports go to root@localhost and are rarely read. Both syslog and logwatch are highly configurable systems and can be used to log and report interested events periodically. Hence proper use of logwatch can help in detecting unwanted changes without relying on administrators to manually go through log files regularly to detect anomalies.

- Increasing detail of received logwatch logs

  We can increase details of logwatch logs by editing '=/etc/logwatch/conf/logwatch.conf=' file and putting:

  ```
  Detail = High
  ```

in the file

- Custom logwatch service

  We can create custom logwatch service to get log reports of log messages received via rsyslog, especially in case we have configured some network device like firewall to send syslog messages to our host. To create a service we need to create two files

  - Service configuration file
  - Service script file

  - Creating service configuration file

    In order to get log messages of remote host though logwatch email we have to setup service for that hosts messages. First in folder /etc/logwatch/conf/services create a $\text{service}_{\text{name}}$.conf file. The only required line in this config file is Logfile = directive. Use logfile = messages. Here it is important to have space between LogFile and '=' and between '=' and messages. Also we have specified messages not because file is /var/log/messages but because /var/log/messages comes under messages Log Group.

    ```
    LogFile = messages
    ```

  - Creating service script file

    After this we have to create filter script that when given log file on standard input would print only relevant output on standard output. Filter scripts must be kept in /etc/logwatch/scripts/services directory and the name must be $\text{service}_{\text{name}}$. The file should also be executable, so if it is shell script do not forget to do chmod +x on it. It can also be a c program or python script, it is not necessary for filter to be a bash script. We mention loggroup in service configuration file. The loggroup configuration file contains names of logfiles inside that loggroup. For examples loggroup 'messages', contains logfile '/var/log/messages'. This script is given entire logfiles as standard input and its standard output is sent as log report.
    So if you want entire log file to be sent as log report the script file can contain just one 'cat' command without any arguments.

In case you want only lines containing word ASA to be sent as logreport then you can write only one line 'grep ASA' in the script file.

All the current logwatch scripts, configuration files and service files are located in '/usr/share/logwatch-&ltver>' directory. We can refer to these config files, service files etc. to create new files. One can also read `/usr/share/doc/logwatch-.*/HOWTO-Customize-LogWatch` for more details then given here.

1.6 Detecting unwanted changes

It is important to detect unwanted changes to file-system. Auditd is one of the tools which can help in monitoring changes. But sometimes we may want changes to be compared with previous system state to get idea of what got modified since last check. File based intrusion detection systems provide this functionality. With file based IDS we can keep information about system state at any given time and then compare it with system state at a later stage. Interesting differences (where we specify what is interesting) are shown. Such tools help in detecting unauthorized modifications of systems files. This also helps in detection of installation of back-doors and root-kits.

1.6.1 File based Intrusion Detection Systems (FIDS)

Two very famous file based intrusion detection system are discussed below. Basic difference between tripwire and AIDE is that tripwire supports cryptographic security by providing options for signing database generated while examining system state. Hence it is not possible for someone to update both system and tripwire database without knowing tripwire key passwords. But in case of AIDE an intelligent attacker who detects presence of such systems can update AIDE database to avoid detection. Off-course such attack vectors can be reduced by keeping AIDE database backups on other machines or by signing AIDE database and verifying the signature later. But managing systems where databases are stored remotely or where signatures have to generated and verified manually, is fairly complex.

- **TODO** Tripwire

  Tripwire installation and configuration can be learned from following links:

  - http://www.linuxjournal.com/article/8758?page=0,2

  - http://www.akadia.com/services/tripwire.html
  - http://centos.org/docs/2/rhl-rg-en-7.2/ch-tripwire.html

  Considerable information on how to use tripwire is present in following man pages:

  - `man twintro`
  - `man tripwire`
  - `man twconfig`
  - `man twpolicy`
  - `man twadmin`

- AIDE

  AIDE is very simple and easy to use (at least in comparison to tripwire). To learn AIDE one can start with below configuration file:

```
@@define AIDEDIR /home/saurabh/Desktop/aide
database=file://@@{AIDEDIR}/database/current_database.db.gz
database_out=file://@@{AIDEDIR}/database/new_database.db.gz
verbose=20
report_url=file://@@{AIDEDIR}/log/aide_report.txt
report_url=stdout
gzip_dbout=yes
warn_dead_symlinks=yes
config_version=1.0

/home/saurabh/Desktop/aide/test p+u+g+S
```

  Replace `/home/saurabh/Dekstop` in above configuration appropriately.

  Once configuration file is setup as mentioned above one can learn AIDE by performing operations in test folder and checking for reports using AIDE.

  Useful AIDE commands are:

  - Checking configuration

```
aide -c <configuration file> --config-check
```

- Initializing database

```
aide -c <configuration_file> -i
```

- Check system against current database

```
aide -c <configuration_file> -C
```

- Check system and also update database

```
aide -c <configuration_file> -u
```

We can run AIDE periodically using cron. Sample script which can be used for such checks is

```
#!/bin/bash
/usr/sbin/aide --update -V 20 | /bin/mail \
      -s "Weekly Aide Data" barjatiya.saurabh@gmail.com
cp <new_database> <current_database>
```

Note that AIDE does not provides cryptographic security when used as mentioned above. When using AIDE as mentioned above onus of protecting database from attacker is on administrator.

2 Security from network

2.1 Firewalls

2.1.1 `iptables`

`iptables` is very powerful and flexible host firewall available on all major Linux distributions by default. A very brief introduction to iptables is hosted at http://www.sbarjatiya.in/website/tutorials/iptables/iptables.pdf
A detailed project report on iptables is hosted at http://www.sbarjatiya.in/website/courses/2011/spring iptables.pdf

2.1.2 `ebtables`

Although iptables is very powerful and flexible, it cannot be used for firewalling purposes between VMs located on same base host when VMs communicate with each other using software bridges. In such cases `ebtables` comes to rescue. We can apply `ebtables` rules on bridge interfaces even when some of them do not have IP addresses assigned to them, making `iptables` unusable for such interfaces. Syntax, configuration files and man page of `ebtables` are very similar to `iptables` making it very easy to learn, once someone is comfortable with iptables.

2.1.3 `arptables`

If ARP protocol filtering is required then `arptables` firewall has more features and options then what are provided by `iptables` for ARP can be used.

2.2 TODO Intrusion prevention system (IPS) (`snort`)

2.3 Secure Shell

One among major advantages of using Linux Operating System is having SSH server installed and available on most distributions by default. If used properly SSH helps in securing remote access to machines (both command line and GUI) and also allows secure way of transferring or synchronizing files between two machines. SSH can also help in tunneling with support for Local, Remote and Dynamic port forwarding. Thus it is important to understand and use SSH properly so that it can aid as great security tool.

2.3.1 Scripting password based ssh login (`ssh-pass`, `expect`)

Many times users face problems in scripting when scripts require to run ssh / scp commands which would prompt for passwords. To supply passwords through scripts one can use `ssh-pass` hosted at http://linux.softpedia.com/get/Security/Sshpass-8693.shtml If more complex scripting requirements are present then one can learn `expect` scripting language.

Sample expect script that establishes SSH connection and runs `ls -l` command is:

```
#!/usr/bin/expect -f
spawn ssh saurabh.barjatiya@mirage.iiit.ac.in
expect "password:"
send "<password>\r"
```

```
expect "saurabh.barjatiya@mirage"
send "ls -a\r"
set timeout 1
expect "Not there"
send_user "$expect_out(buffer)"
close

#send "exit\r"
#expect "Not there"
#send_user "$expect_out(buffer)"
```

Another sample expect script that establishes a SSH connection and gives control to user after login:

```
#!/usr/bin/expect -f
spawn ssh saurabh.barjatiya@mirage.iiit.ac.in
expect "password:"
send "<password>\r"
expect "saurabh.barjatiya@mirage"
interact
```

Lastly sample script that allows copying of large VM images from one system to another using expect

```
spawn rsync -vazS --progress --partial \
    root@10.3.3.48:/mnt/data1/saurabh_do_not_delete/ \
    /mnt/data1/saurabh_do_not_delete/
set timeout -1
sleep 3
expect {
                yes/no {
                send "yes\r"
                exp_continue
        }
        password: {
                send "<password>\r"
                exp_continue
        }
        denied {
                send "<password>\r"
```

```
                exp_continue
        }
        eof {
                exit 0
        }
}
wait
exit 0
```

2.3.2 Key based ssh login (`ssh-keygen`, `~/.ssh/authorized_keys`)

If expect and ssh-pass based methods are not acceptable as they require storing of plain-text password in script source codes or prompting for them as command line argument then we can establish trust / key based authentication between two systems. For this one system generate keys using

```
ssh-keygen
```

Then copy file `~/.ssh/id_rsa.pub` to other machine at location `~/.ssh/authorized_keys`. It is important that permissions on folder `~/.ssh` are 700 and that `authorized_keys` file has 600 permissions. After setting the keys properly if we SSH from one system another then we don't get prompted for password and login is allowed based on keys.

2.3.3 Keys with pass-phrase (`ssh-agent`, `add-key`)

Key based login is also not acceptable sometimes as if system is left unattended or keys get compromised then attacker will get full access to system which trusts compromised keys. Hence ssh keys are often protected with passwords (called pass-phrases in this case). To assign pass-phrase to existing keys or to change pass-phrase of existing keys we can use following command:

```
ssh-keygen -p
```

Now every time the given key is used for authentication then user will get prompted for password. This can be annoying if we are going to use many consecutive ssh commands which will depend on keys as we will have to supply pass-phrase once for each command. To solve this problem we can run ssh-agent using command:

```
exec $(which ssh-agent) $SHELL
```

and on the obtained shell (which replaced parent shell transparently) we can use command:

```
add-key
```

and supply password (pass-phrase) only once. This causes key to get stored with ssh-agent and we do not need to supply pass-phrase for this particular running instance of shell. As soon as we exit from shell ssh-agent also exits automatically making system very secure.

2.3.4 Blocking brute-force attempts (`denyhosts`)

Anyone who has administered Linux server with public IP and SSH enabled for more than a day can check logs to see how many people attack a new server using brute-force to login as root user in just one day. The number is really surprising. Given that SSH as root will compromise system completely allowing attackers to continue trying different passwords on servers is not a good idea. Thus it is important to stop same person to try more than given number of password within an interval, especially for user accounts like root. Denyhosts script allows achieving this goal in very easy and efficient manner.

One should be able to install `denyhosts` on all major distributions using package managers like `yum`, `apt-get`, etc. Location of `denyhosts.cfg` file can be found using `locate` command (preceded by `updatedb`, if necessary). Lines that I personally modify in default `denyhosts.cfg` file are:

```
PURGE_DENY = 12w
PURGE_THRESHOLD = 2
BLOCK_SERVICE = ALL
DENY_THRESHOLD_ROOT = 10
ADMIN_EMAIL = barjatiya.saurabh@gmail.com
SYSLOG_REPORT = YES
AGE_RESET_VALID = 2d
AGE_RESET_ROOT = 2d
RESET_ON_SUCCESS = YES
DAEMON_SLEEP = 120S
(Uncomment) 'SYNC_SERVER = ...' line
SYNC_DOWNLOAD_THRESHOLD = 10
```

It is also recommended to install `denyhosts` as service and enable it on start-up so that it automatically runs on reboot. To prevent important trusted IPs from getting blocked they can be white-listed by creating file

named `allowed-hosts` in denyhosts `data` directory. We can add one IP per line in this file and all the mentioned IPs will not get black-listed irrespective of how many wrong password attempts are made from these IPs.

White-listing configuration must be done before denyhosts is used. If we try to white-list banned IP to remove ban, then it may not work without tinkering with other denyhosts data files.

2.3.5 Disabling command execution over SSH

- Restricted SSH (`rssh`, `scponly`)

  Sometimes on shared systems users do not need full ssh or command line access and only need to transfer files (`scp`, `rsync`, `sftp`, etc.) to / from shared host. In such cases giving full ssh permissions is not safe. To restrict users so that they can only perform `scp`, `rsync` etc. and nothing extra one can install `rssh` or `scponly`. Installation and configuration of these two tools is very easy and hence is not described here.

- Restricting sub-system to sftp with chroot

  If one has SSH version 4.9+ (can be installed using source on Cent-OS) then we can use chroot and sftp force options provided by latest SSH server without requiring to install or setup `rssh` or `scponly`. Configuration lines that would force users belonging to group `sftponly` to use sftp are

  ```
  Match Group sftponly
     ChrootDirectory %h
     ForceCommand internal-sftp
     AllowTcpForwarding no
     X11Forwarding no
  ```

  Apart from forcing users to use `sftp`, the above lines would also chroot them to their home directory. Hence users wont be able to access anything outside their home-directory. The last two lines disable TCP and X11 forwarding so that only thing users can do with SSH is file transfer.

2.3.6 Preventing misuse of SSH

SSH supports port forwards in form of Local, Remote and Dynamic port forwards. SSH man pages and documentation on Internet can be read to understand what each of these forwards provide and how to use them. Problem for administrators usually is that these features provide security loop holes as users can establish connections to/from host to which they SSH. Hence if SSH access to public server is provided then using Dynamic port forwarding one can turn server to a SOCKS proxy. To ensure that such security problems are not possible it is best to disable GUI login and TCP port forwarding for all non-root users.

Configuration lines that allow blocking of GUI login and port forwarding are:

```
AllowTcpForwarding no
X11Forwarding no
```

These can be preceded with filter `Match User !root` (Caution: experimental) so that these rules apply to all except root user.

2.4 Virtual Private Networks

2.4.1 OpenVPN

Many times SSH login to public servers are provided so that users can connect to a enterprise network from outside. But VPN is definitely a better and safer alternative in comparison to SSH for remote access. VPN is complex and hence cannot be discussed in this session. Interested administrators are encouraged to learn configuration of OpenVPN using How-To located at http://openvpn.net/index.php/open-source/documentation/howto.html

I have setup OpenVPN for IIIT Hyderabad and it has been in use by more than 1000 users (10-15 concurrent users) since many years and we do not face any problems in using OpenVPN. OpenVPN is ideal when users could connect from various operating systems like Windows, Mac OS X, Solaris, Linux etc. to OpenVPN server to use VPN services. OpenVPN works well for both 64-bit and 32-bit OS variants without any problem.

2.5 Port scanning

It is always good to verify after proper firewall (such as `iptables`) configuration has been done that only desired services are accessible from external users. An easy way to accomplish this is to port scan protected server from

prospective client machines / potential attackers to check which ports are open. One very versatile port scanning tool available for Linux (also works equally well with Windows) is `nmap`. `nmap` is well documented and we can use:

```
nmap
```

command without arguments to see various options supported by nmap and try each of them. `man nmap` can be used to get detailed information on each of the options available.

Some important nmap command line options are explained below:

Option	**Description**
-sP	Ping scan. Only determine whether host is on-line or not. Do not scan ports. This is very useful to find out which machines are on in given IP range.
-p	Scan only these ports. Useful to check for a particular service on given range of IPs
-n	Do not do DNS resolution. Very useful for passive information gathering.
-sS	TCP SYN Scan
-sA	TCP ACK Scan
-sT	TCP Connect Scan
-sU	UDP Scan
-sV	Probe open ports to determine service/version information
-O	Enable OS detection
-f [–mtu]	Fragment packets optionally with given MTU
-D	Use given set of decoys before scanning target with our address
-S <IP>	Use spoofed source address
-e	Use specified interface
–spoof-mac	Spoof MAC address
–ttl	Send packets with given TTL
–badsum	Send packets with bad check-sum
-v	Increase verbosity. Can be used twice for greater effect
–open	Only show open (or possibly open) ports
-6	Enable IPv6 scanning
-A	Enable OS detection, service detection and traceroute

3 General security tips

3.1 Securing Internet access within organization

Internet access is provided to employees in almost all organizations as access to email, on-line search engines, social networking websites, Wikipedia, etc. is required or at least improves productivity of employees. But if such access is provided without proper safe guards and without educating users about potential threats from Internet, then such access is very unsafe. Hence very basic Internet security related points are discussed below. Although these points do not come under 'open source security tools' which is main title or agenda for the lecture. But ignorance or mistakes with respect to mentioned points will render use of all sophisticated tools / techniques discussed so far useless, and expose users and in turn organization to variety of attacks.

3.1.1 Poor coding based attacks

Developers who code websites (especially active / dynamic websites) for company should be educated about various types of attacks that are tried on any dynamic website from malicious users. At least developers should ensure that the websites in use are safe from:

- SQL Injection attacks
- XSS attacks
- Brute force attacks

3.1.2 Firefox plugins for browsing security and privacy

Advertisements and social networking website APIs (facebook, twitter, etc.) allow tracking of user movement from one website to another. If privacy is a concern (which is usually the case) then it should not be possible for advertisement websites or social networking websites to know about all on-line activity of particular user. Hence use of following security plugins with Firefox browser is recommended for safe Internet use:

- Ghostery
- No Script
- Web of Trust (WOT)
- Ad-block plus

- Better privacy

3.1.3 Sensitive posting in social networking websites

It is important to ensure that users do not post anything sensitive (esp. photos and videos) on social networking and video sharing websites. Users should be explained that things once uploaded, **cannot be taken back / deleted**. Files which are uploaded will remain available to one or the other person in one form or another and it cannot be guaranteed that same file wont surface again somewhere else on Internet.

3.1.4 Saving passwords in browser

Users very often save passwords in browsers for convenience. It should be ensured that all users who save passwords also setup master password to ensure safety of saved passwords. It should be noted that master passwords can only be set in browsers like Opera, Firefox etc. Many famous browsers like Internet Explorer (at least as per my knowledge at time of writing) do not support configuration of master password. Hence use of Internet Explorer to save passwords is extremely bad idea. If password is stored by Internet explorer on system without encryption (in plain-text) then it can be read by other applications without any problem. Note that for login systems to work browsers cannot store hash of passwords, they are required to store complete password for auto password filling to work.

3.1.5 Saving passwords in chat client

Again as per my knowledge at time of writing there is no chat client including `pidgin`, Gtalk, Yahoo messenger to name a few which supports configuration of master password. This makes sense as usually users need to provide only one password in chat clients to login, so having another single password to protect this password does not makes much sense. But storing passwords in such chat clients is very unsafe, same as storing passwords in browsers without use of master password.

3.1.6 HTTP vs HTTPS

Often difference between HTTP and HTTPS and how these protocols are different in poorly understood. Very rarely users are aware of role of certificates, certificate authorities, etc. in making HTTPS secure. Hence these

concepts should be explained to all users so that their Internet browsing, especially authentication and banking are secure.

Topics that can be discussed under this heading are:

- ARP spoofing
- Port mirroring
- Untrusted ISP
- CA certificates
- squid HTTPS port and use of stunnel

3.1.7 IPv6

IPv6 support is now present **and enabled** by default on all operating systems. Thus failure to secure system on IPv6 addresses can lead to severe security problems. General awareness on IPv6 severely lacks understanding of even basics of IPv6 like addressing, routing, etc. Hence IPv6 provides a significant attack surface area in absence of tools and mechanisms that protect IPv6 based network attacks. Thus it is important to setup proper rules to prevent IPv6 based connections / attacks unless IPv6 concepts are properly understood and implemented by administrators and users.

For example IPv6 supports tunneling using 2002::/16 on IPv6 side and protocol 41 on IPv4 side. If firewall rules are not setup to prevent IPv4 to IPv6 tunnels and vice-versa then such tunnels can be used to establish connections to any remote machine without getting affected by devices / rules which do not take IPv6 into account. Even if ISP does not provides IPv6 services and addresses, free tunnel broker websites around the world allow users to connect IPv6 networks using tunnels without requiring any formal verified registration or paying any fee.

Most Linux services like SSH, HTTP etc. are configured such that they listen on both IPv4 and IPv6 addresses. Hence configuring firewalls like `iptables` without doing corresponding configuration in `ip6tables` would leave serious security blind-spots.

3.1.8 Anonymous browsing

Anonymous browsing is both security tool and potential security problem. With anonymous browsing one can open websites without being tracked. Note that anonymous browsing to avoid tracking would be useless without

plugins like Ghostery, BetterPrivacy etc. as advertisements, cookies, etc. would allow tracking of user without requiring him / her to make requests from same IP address.

Anonymous browsing is security problem as all the websites blocked due to company policy can be browsed via anonymous browsing websites. Blocking anonymous websites themselves is not easy as they keep changing IP addresses and keep coming up with ways to avoid blocking by all famous security software. Also failing to block even one such website can lead to access to all websites.

It should be noted that skilled developers / programmers can program such websites on their own in very little time and host them on cloud, personal servers outside organization boundary. Later they can use their own websites for visiting prohibited websites. This cannot be avoided as it is impossible to know purpose, owner etc. of each and every website visited by user.

Famous anonymous websites or tools that help in anonymous browsing are:

- http://www.anonymouse.org
- Freedom
- Tor
- JAP

3.1.9 Anti-virus

Availability of anti-viruses for Linux operating system have caused severe confusion that Linux operating systems are equally vulnerable to viruses as other operating systems like Windows, which is not necessarily true. Linux is very secure in its default configuration of firewalls, services, etc. at time of install. It is easy to have complete security breach if user accounts setup are not secure (weak passwords, etc.) as one can use any user account to SSH / sftp to server and run commands or transfer files unless such access is specifically blocked for particular user. Also once shell access is present, almost all operations including installation from source files can be performed by normal user without root privileges. Hence installing of back-doors, spying software etc. by attacker on systems that are not configured properly is very easy, as number of powerful tools like shell, compilers (gcc), languages(shell script, python, perl, php) etc. are available on most systems.

But all this requires severe mistakes from administrators side. If administrators keep strong passwords and user accounts (specifically SSH) in control then system can be considered to very safe. Specially installation of ad-ware, spyware due to clicking some bad link on browser / email client etc. is not possible. All downloaded files do not have execute permissions by default and users have to manually assign execute permissions to execute downloaded files. Hence possibility of having virus attacks accidentally is very rare.

One of the free anti-viruses available for Linux is Clamav. Anti-virus software are provided in Linux OS for scanning email attachments for viruses or ensuring virus files are not uploaded on common file shares, as these services can be used from other more vulnerable operating systems which require considerable care to protect them against viruses.

3.2 Do's and Dont's for Windows

- Avoid using Internet Explorer. Use Firefox with advised security plugins.
- Disable system restore if you do not want unknown backups of file to be created.
- Do not login as administrator user. Login as normal user for most of your work.
- Do not depend on anti-virus for complete security. Follow best practices to safe guard one self from virus.
  - Do not connect untrusted pen drives / CD to computer
  - Do not download / install untrusted software
  - Do not download / trust almost anything obtained via torrents / DC++ etc.
- Set password for administrator user (and remember it)
- Check users and their rights regularly
  - Check for new users in Computer Management regularly.
  - Check members of various groups regularly.
- Check for folders shared from your Computer regularly

  - Check for unwanted file shares in Computer Management.
  - Avoid using simple file sharing.
  - Avoid having writable shares unless absolutely necessary. It is better to have read-only share on source then having writable share on destination.

- Go through System Policy and modify it as per your needs.
- Consider moving pagefile to D drive, disabling GUI enhancements for performance.
- Always login only when you arrive at login screen after pressing 'Ctrl + Alt + Del' in domain environment.
- Configure screen-saver with password if you leave your system unattended for long durations
- Do not connect to unknown open wireless connections. Remove unrecognized SSID profiles regularly.
- Do not use open wireless at home / office.
- Do not depend too much on WEP security. Prefer WPA or better yet WPA2 for wireless security.
- Keep anti-virus and operating system updated
- Do not allow real player, vlc player, java, Windows media player etc. to contact Internet. Keep windows firewall enabled.
- Check for start-up programs (`msconfig`) and remove / disable unwanted entries
- Remove temporary files when deleting Internet history and recent documents

3.3 Do's and Dont's for email

- Do not trust any email claiming to come from police, bank, NSA, etc. Call proper organization by finding out their number from websites and confirm.
- Never send your ATM PIN number or password to **anyone**. (Not even Bank)

- Confidential attachments should not be sent with email without encryption. Encryption keys for encrypted documents must be conveyed using other channels (preferably non-online methods like phone).
- Do not install programs sent to you by someone over email.
- Do not trust emails that say you have won free gift (no free lunch)
- Do not trust emails that claim to be from soldiers in Afghanistan / Iraq who have discovered gold or fortune and they need your help in transferring it to other countries.
- Do not trust emails claiming to help poor, uneducated, deprived people in some random part of world.
- Do not forward emails so that something good happens to you. Nothing good happens by forwarding emails.
- Avoid sending joke, fun, entertainment emails to someone, unless they have specifically asked for it.
- While forwarding jokes, fun, entertainment emails always keep all recipients in BCC and remove all email addresses present in email.
- Do not trust sender address, recipient address (!!), Date, Time, Many initial headers present in emails.
- Do not click on links in emails unless necessary.
- Do not open / display external pictures embedded in email unless email is sent from trusted source. Such emails can be used to trace your locations.
- Do not try to un-subscribe from SPAM emails.

4 Installation of tools

Installation of various tools / command used in this lecture are mentioned below to help with installation of desired tools. For yum it is assumed that repositories `epel` (for Cent-OS / RHEL), rpmfusion, rpmforge (repoforge) have been properly configured.

S. No	Tool	Linux Installation	Windows Installation

1	`chmod`
	`chown`
	`chgrp`
	`usermod`
		Comes with full installation of OS (package coreutils)	

2	`sudo` `visudo`	Comes with full installation of OS (package sudo)	
3	`setenforce` `getenforce`	Comes with full installation of OS (package libselinux-utils)	
4	`svn` `svnadmin`	Comes with full installation of OS (package subversion)	
5	`git`	Comes with full installation of OS (package git)	
6	`rsync`	Comes with full installation of OS (package rsync)	
7	`dd`	Comes with full installation of OS (package coreutils)	
8	`mdadm`	Comes with full installation of OS (package mdadm)	
9	`gpg`	Comes with full installation of OS (package gnupg)	gpg4win can be download from http
10	`encfs` `encfsctl`	Can be installed via yum (package encfs)	Setup available at http://members.f First download and install Dokan Li encfs executable zip file.
11	`top`	Comes with full installation of OS (package procps)	
12	`htop`	Can be installed via yum (package htop)	
13	`free`	Comes with full installation of OS (package procps)	
14	`lsof`	Comes with full installation of OS (package lsof)	
15	`netstat`	Comes with full installation of OS (package net-tools)	
16	`tcptrack`	Can be installed via yum (package tcptrack)	
17	`iptraf`	Can be installed via yum (package iptraf)	
18	`w`	Comes with full installation of OS (package procps)	
19	`ps`	Comes with full installation of OS (package procps)	
20	`atop`	Can be installed via yum (package atop)	
21	`last` `lastb`	Comes with full installation of OS (package SysVinit)	
22	`lastlog`	Comes with full installation of OS	