2009/10 IT Training Programme for SMEs in General Industries

T10-Linux System Administration

What is Linux?

- Created by Linus Torvalds

What is Linux?

- Operating system
- Evolved from a kernel (The core or nucleus of an operating system).
- UNIX family
- Free software produced by GNU (Creating a totally "free" operating system in which the source code was available) in 1991.

What is Linux?

- Also called "GNU/Linux“.
- Tremendously stable and versatile operating system, particularly as a network server
- Can be installed on a home PC

Red Hat Enterprise Linux (RHEL)

- Produced by Red Hat
- First release: 2002
- Designed to be stable and with long-term support for enterprise users

CentOS

- Based on Red Hat Enterprise Linux
- Developed by North American Enterprise Linux vendor.
- First release: 2004
- Stands for Community ENTerprise Operating System

CentOS

- Each CentOS version is supported for 7 years (by means of security updates)
- New CentOS version is released every 2 years
- Each CentOS version is periodically updated (roughly every 6 months) to support newer hardware.
- Latest version: CentOS 5.4 Released on Oct 2009

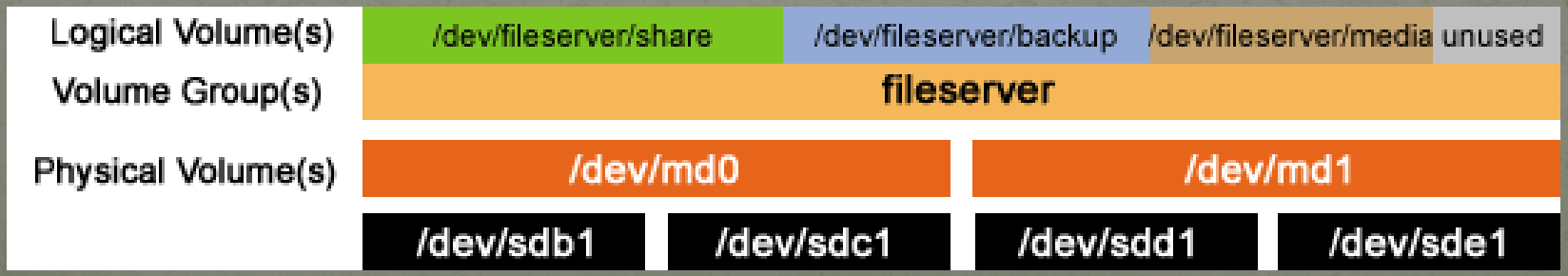
Excercise

- Install CentOS 5.4
- Remove all partitions on selected drives and create defalt layout
- Review and modify partitioning layout
- Set sitXX.t10 for the host name

- Modify the layout to
  /boot 100M
  /home remaining
  / 100G
  /swap 4G

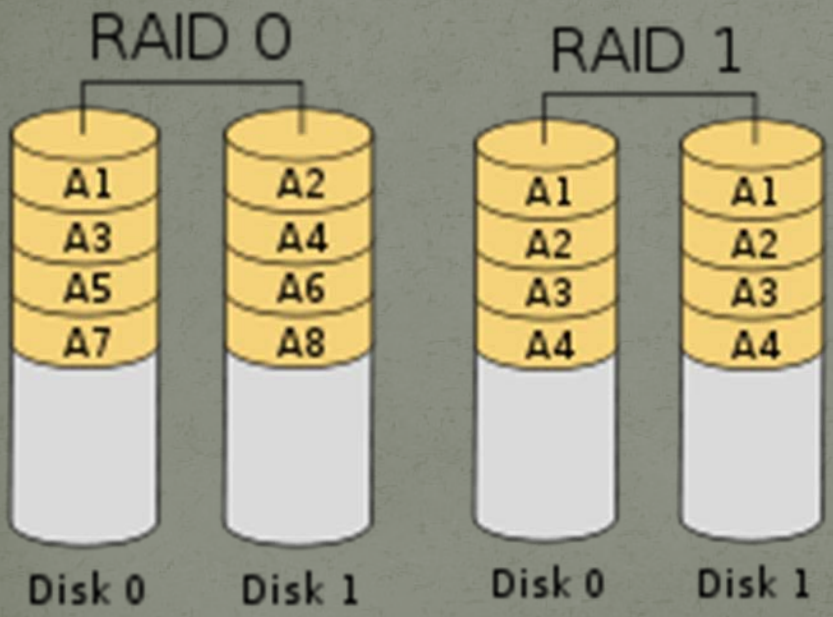
Terms

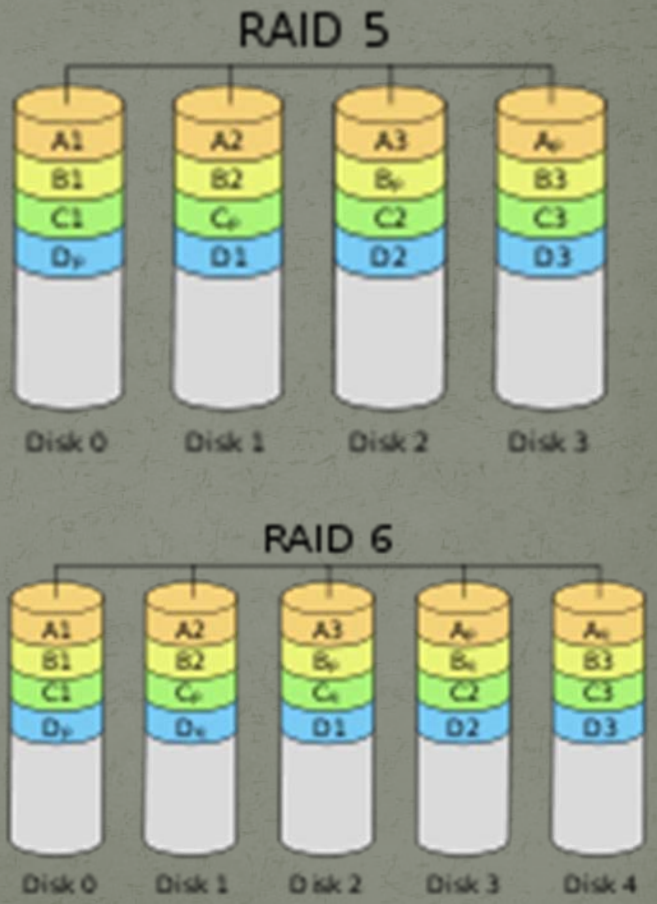
- Hard disk
  sda, sdb, etc.
- Hard disk partition
  sda1, sda2, sdb1, sdb2, etc.
- LVM
  logical volume management
- Volumn Group
  Group of hard disk partitions



Terms

- RAID
  RAID 0, 1, 5, 6, 10





Terms

- Journaling file system
  keeps track of the changes it intends to make in a journal (usually a
  circular log in a dedicated area of the file system) before committing
  them to the main file system. In the event of a system crash or power
  failure, such file systems are quicker to bring back online and less
  likely to become corrupted.

- EXT4
  a journaling file system developed as the successor to ext3.

- Swap partition
  extra memory boost. double of your ram memory.

Terms

- Boot loader
  Loaded into RAM by ROM's program
  Multiple-stage , several programs of increasing complexity sequentially load one after the other in a process of chain loading.

- GNU GRUB
  GNU GRand Unified Bootloader
  Enables a user to have multiple operating systems

- root
  The user name or account that by default has access to all commands and files on a Linux
  Not for routine work

- Administor user
  routine work with nearly root‘s power

Install Software

- Development: Java
- Server: FTP, Mail, MySQL, Configuration tools(select all), Web Server,
  Tomcat, Windows File server (Samba)
- Base: Administration tools (select all), Java, System tools, X windows system
- Languages: Language support

After reboot

- Enable firewall: FTP, Mail, SSh, Samba, https, http
- SELinux: Disabled
- Update system: Application->System tools->Software update

Basic directory structure

- /bin directory: All of the most used commands
  /etc directory: Houses most of the configuration files
  /etc/grub.d directory: OS configuration files.
  /usr directory: Contains files and programs meant to be used by all of the users
  /boot directory: Linux kernel
  /root directory: Root's home directory, a restricted area
  /sbin directory: like /bin, but only to be used by root. 'Shutdown' is in there.
  /tmp directory: Temporary files
  /var directory: Store files that has variable size, e.g. /var/lib/mysql and /var/mail
  /lib directory: Shared library files
  /home/<username>: personal files

In Linux, everything is a file

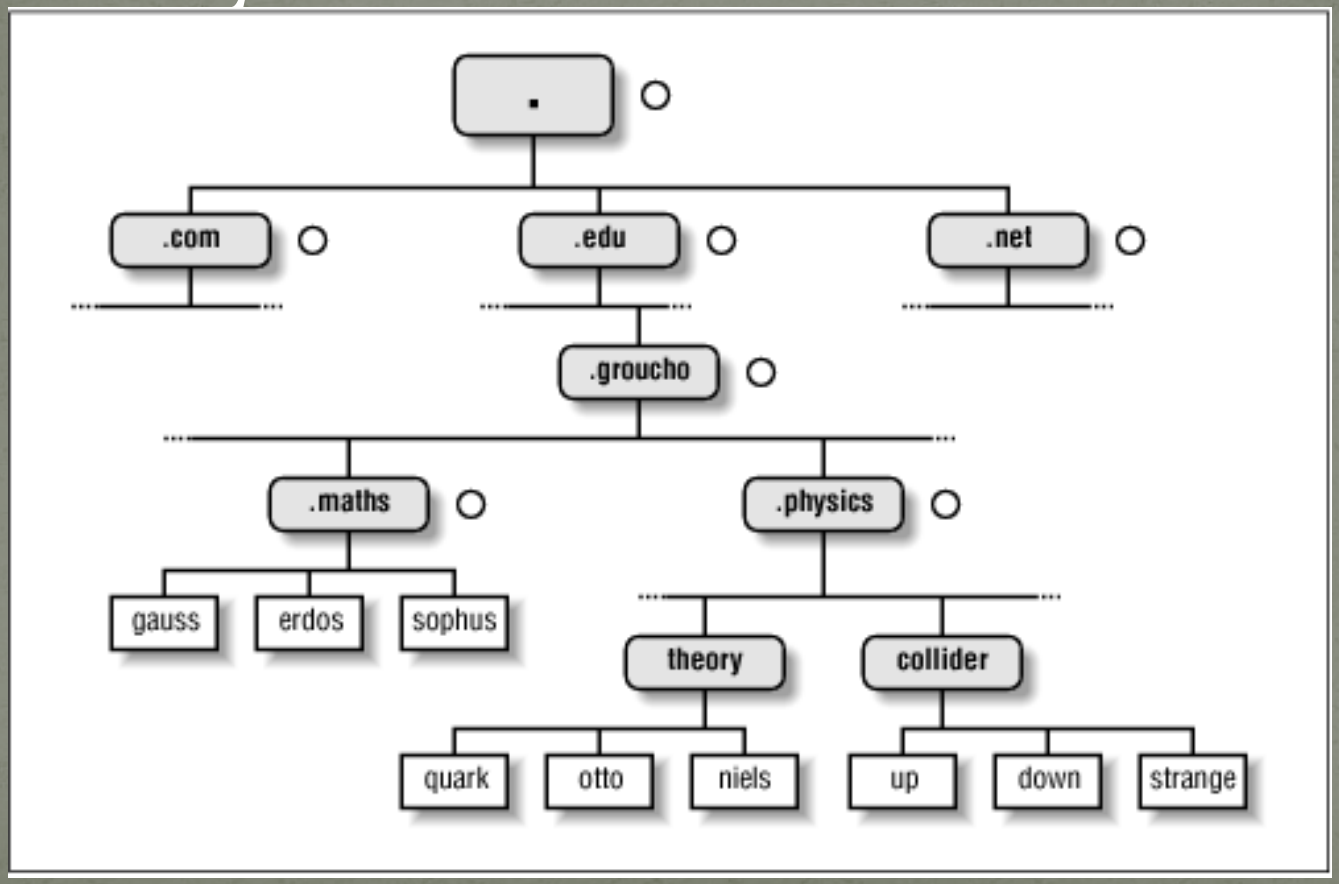
- /dev directory: Devices that your system uses or can use
- /dev/sd* : Hard disk devices

Linux Shell

- Get commands from the keyboard to the computer.
- Most popular shell is the bash shell. bash means "Bourne Again Shell“.
- Use up or down arrow to view and re-excute commands

DNS setup

- DNS: Organizes hostnames in a domain hierarchy.



/etc/profile

export http_proxy="http://proxy:8082/"
export ftp_proxy="http://proxy:8082/"
export gopher_proxy="http://proxy:8082/"
export wais_proxy="http://proxy:8082/"

/etc/yum.conf

http_proxy="http://proxy:8082/"

Uncommet /var/named/chroot/etc/named.conf

//query-source address * port 53;

DNS setup

- ifconfig: interface configurator, configure, control, and query TCP/IP
- network interface parameters
  /etc/host.conf: DN resolving order.
  /etc/hosts: map hostnames to IP addresses
- bind: BIND name service for host lookups
  /etc/resolv.conf: DNS list

Exercise

- Login root
- Right click desktop and open Terminal
  Type /sbin/ifconfig to get your IP address
- Open /etc/resolve to add
  nameserver <your IP>
- Open DNS sittings: Select System, Administration, Server Settings, Domain Name System,
  select DNS server, click new, click Zone
  Click OK for internet class, click OK for forward type.
  Type yourname.t10. to domain name
  Add "A" type entry for "www" and your IP.

  Select System, Administration, Server Settings, Services
  Select named, start

Exercise

- Go to terminal
  Type the following commands to test result.
  nslookup www.yourname.t10

- Add “A” and “NS” type for other classmates IP.
  nslookup www.others.t10 to test.

Remember to save before testing.

Exercise

- Allow port 53 UTP to pass through your firewall
- System, Adminstration, Security Level and firewall
- Stop named service
- Enter change your IP to other IP in /etc/reolv/conf
- Test other DNS.

Exercise: user and groups

- Use System, Administrations, users and groups to manipulate users and groups.
  Add a group t10
  Add a user t10student without private group.
  Add group t10 to the user t10student and change the primary group to t10,
  Right click computer and browse folders to go to home folder to check
  if the home directory of t10student to created.
  Show all hidden files and view as list.
- How to change your password? Use about me.
- How to change another user's password? Use users and groups.

Text editor "vi"

- vi <filename>
  Press <Esc> to exit insert mode
  i (insert mode)
  x (delete a character)
  dd (delete a line)
  :x (Save and exit)
  :w (Save)
  :q! (Abort and exit)
  /<characters> (seach <characters>)

Exercise: quota

- Make sure quota package is installed:
  Application, Add/remove package, search quota

- Edit /etc/fstab
  Add ,usrquota,grpquota to the first line:
  /dev/VolGroupoo/LogVoloo / ext3 defaults,usrquota,grpquota 1 1

- Reboot
  quotacheck -cugm /
  quotacheck –avugm

- df: check block size

- Edit quota
  edquota username (for each user)
  edquota -g groupname (for each group)
  edquota -t (for grace period)
  repquota -a (quota report)

System monitoring

- System monitor: System, Administration, System Monitor
- find and manage busy process
- kill process
- Monitor Resources and file system

Exercise:

- vi testing
- Use nice to change priority
- Kill process vim of this user.

File permission

- File browser:
  view, show hidden files
  Set file permission

- chown: Change ownership

Create zip files

- Use file browser to go to home directory
  Select multiple files, right click, create archive

Exercise: shell script

- Create a file "first" in home directory
  echo "My first script"
  echo $HOME
  echo $USERNAME

- Add execution permission in properties of "first"

- Open terminal
  ./first

Wild cards

- * Matches any string or group of characters.
  a* = any string starting with 'a'.
  *b = any string ending with 'b'.
  a*b = any strong starting with 'a' and ending with 'b'

- ? Matches any single character.
  a?b = any three characters starting with 'a' and ending with 'b'

- [..] Matches any one of the enclosed characters
  [ab]* = any string starting with 'a' or 'b'.
  [cde]? = any two characters starting with 'c', 'd' or 'e'.

- ! Not
  [!ab]* = any string not starting with 'a' or 'b'

Command Line arguments

- $0: command
- $1: first argument
- $2: second argument

Sample shell script:

- Use ssh to connect to a server to copy the script.

```
if [ "$1" == "" ]
then
  echo "Usage: chkhkid <HKID>"
  exit
fi
if echo $1|grep -q -e '^[A-Z][0-9]\{6\}([0-9A])$'
then
  hkid1=${1:0:7}
  hkid2=${1:8:1}
  hkid=$hkid1$hkid2
  asciival=`echo ${hkid:0:1}|tr -d "\n"|od -An -t dC`
  firstval=`expr $asciival - 64`
  chksum=`expr $firstval \* 8`
  for (( i = 1 ; i < 8 ; i++ ))
    do
    digit=`echo ${hkid:$i:1}`
    if [ $digit == "A" ]
    then
      digit=10
    fi
    multi=`expr 8 - $i`
    chkdigit=`expr $digit \* $multi`
    chksum=`expr $chksum + $chkdigit`
  done
  remainder=`expr $chksum % 11`
  if [ $remainder -eq 0 ]
  then
    echo "HKID is valid."
  else
    echo "HKID check sum error."
  fi
else
  echo 'Not valid patern'
fi
```

Exercise

- userdel <username>
  delete a user.

- rm -f -r <file/directory>
  remove files or directories
  -f: ignore nonexistent files, never prompt.
  -r: remove the contents of directories recursively.

- /home/<username>: home directory of <username>
  /var/mail/<username>: unread mail of <username>

- cat /etc/passwd: display user list

Exercise

- cat /etc/passwd|grep -q -e <regular expression>: Check if a username
  exist in user list.

- Write a program to delete a user and his home directory and unread mail.
  Name of program: deluser.sh
  Usage: deluser.sh <username>
  Note: If <username> is not given, display an error message. If
  <username> does not exist, display an error message. If <username>
  exists, delete this user and his directory and unread mail.

#!/bin/bash

```
if [ "$1" = "" ]
then
  echo "Error: username not found"
elif cat /etc/passwd|grep -q -e "^$1:"
then
  userdel $1
  rm -f -r /home/$1
  rm -f /var/mail/$1
  echo "User account and data deleted"
else
  echo "User not found."
fi
```

Scheduled task

- at time (to add a job run at time)
  atq (to list jobs)
  Enter and Ctrl-D to complete
  use mail to read result
  atrm <jobno> to remove job.
  /etc/at.deny: users who are not allowed to use 'at'.
  /etc/at.allow: users who are allowed to use 'at'. (higher priority)

  crontab -e: create scheduled job list

```
*    *    *   *    *       command to be executed
|    |    |   |    |
|    |    |   |    +----- day of week (0 - 6) (Sunday=0)
|    |    |   +------- month (1 - 12)
|    |    +--------- day of       month (1 - 31)
|    +------------ hour (0 - 23)
+-------------- min (0 - 59)
```

  crontab -l: list scheduled jobs
  crontab -r: remove job list
  /etc/cron.deny: users who are not allowed to use 'cron'.
  /etc/cron.allow: users who are allowed to use 'cron'.

  Example:
  30 18 * * * rm /home/someuser/tmp/*

Exercise:

- Create a cron job to execute "/usr/sbin/warnquota" at 3:00am everyday
  and "/usr/sbin/repquota -a" at 4:00am everyday.

/etc/sudoers
(list of which users may execute what)

Example:
dgb boulder = (operator : operator) /bin/ls, (root) /bin/kill, /usr/bin/lprm
dgb runs /bin/ls with either the user or group set to operator, but
/bin/kill and /usr/bin/lprm as root on machine boulder.

ray rushmore = NOPASSWD: /bin/kill, /bin/ls, /usr/bin/lprm
ray runs /bin/kill, /bin/ls, and /usr/bin/lprm as root on the machine
rushmore without authenticating himself.

Exercise:

- sudo: allows users to run programs with the security privileges of root
  Login <username>
  /sbin/shutdown -r now
  sudo /sbin/shutdown -r now

  Login root (run su -)
  visudo
  Add the following line
  <username> ALL=NOPASSWD: /sbin/shutdown -r now, /usr/bin/passwd root

  logout root (exit)
  /sbin/shutdown -r now
  sudo /sbin/shutdown -r now

Web service

- What is the web?
  A whole bunch of interconnected computers talking to one another

  What is the function of server software / programs?
  Server software is created to 'serve' web pages and web sites. Basically, the server computer has a bunch of web sites loaded on it and it just waits for people (via web browsers) to request or ask for a particular page. When the browser requests a page the server sends it out.

  How does the web surfer find a web site?
  By typing in the URL, Uniform Resource Locator, the web site address

  What are HTML tags?
  Specifically formatted text that creates 'markers' for web browser to read and interpret. These 'markers' tell the web browser what and how to display things on the web page. Tags are placed in and around text and images (text and images are some of the 'things') that your want to have appear in your web pages.

  e.g.
  <html>
  <body>
  This is Columny's page.
  </body>
  </html>

Apache webserver

- /etc/httpd/conf/httpd.conf: Main configuration file for the Apache webserver
  ServerRoot "/etc/httpd"
  Listen 80
  DocumentRoot "/var/www/html"
  ScriptAlias /cgi-bin/ "/var/www/cgi-bin"
  DirectoryIndex index.html index.html.var
  Include conf.d/*conf

- /etc/httpd/conf.d/ssl.conf
  Listen 443

- index.html.var: Deliver content-negotiated documents
  Example:
  URI: /en/
  Content-language: en
  Content-type: text/html

- URI: /zh/
  Content-language: zh
  Content-type: text/html

Exercise

- Install web browser.

  Create
  /var/www/html/index.html
  <html>
  <body>
  This is Columny's page.
  </body>
  </html>

- Create directory:
  /var/www/html/en
  /var/www/html/zh

- Copy index.html to en and zh
  Delete file index.html

•Change /var/www/html/en/index.html
<html>
<body>
English page
</body>
</html>

•Change /var/www/html/zh/index.html
<html>
<body>
Chinese page
</body>
</html>

•Set browser content language to english and then chinese to test.

•Try to access other students website.

Log file directories

- /var/log/httpd/

Continuously watching log

- cd /var/log/httpd
- tail –f access_log

Exercise: Allow specific machine to access a web site

- Search <Directory "/var/www/html"> of /etc/httpd/conf/httpd.conf
  Scroll down to "Order allow,deny"
  Remark "Order allow,deny"
  Remark "Allow from all"
  Add "Order deny,allow"
  Add "Deny from all"
  Add "Allow from XXX.XXX.XXX.XXX"

- Restart httpd service.
  Test if only XXX.XXX.XXX.XXX can access your page.

- Change "Allow from XXX.XXX.XXX.XXX"
  to "Allow from 127.0.0.1"

- Restart httpd service.
  Test if only you can access your page.

- Change "Allow from 127.0.0.1"
  to "Allow from XXX.XXX.XXX.0/24"

- Restart httpd service.
  Test if all except you can access your page.

Remove testing page

- What is /etc/httpd/conf.d/welcome.conf saying?
- Options –Indexes: If a URL which maps to a directory is requested, and there is no DirectoryIndex (e.g., index.html) in that directory, then a formatted listing of the directory is listed.

Exercise

- Remark all lines of welcome.conf
- Restart httpd service
- Test the error page
- Create some pages in /var/www/error
- Edit welcome.conf
- Restart httpd service and test different error pages.

.htaccess file

- Provide a way to make configuration changes on a per-directory basis.

- Two main reasons to avoid the use of .htaccess files:
  1. Performance: Apache looks in every directory for .htaccess files.
  The .htaccess file is loaded every time a document is requested.
  2. Security. You are permitting users to modify server configuration,
  which may result in changes over which you have no control.

Exercise

- Rollback the changes of /etc/httpd/conf/httpd.conf
  Change "AllowOveride None" to "AllowOveride All"

- Add a .htaccess to /var/www/html/en directory
  Order deny,allow
  deny from all
  allow from 127.0.0.1

- Restart httpd service
  Test if only you can access the english page. The chinese page is not affected.

Job Control

- Ctrl-Z: Suspend a job and return to shell.
- &: Run a job in background.
- bg: Resume a job in background.
- fg: Resume a job in foreground.
- jobs: List all jobs.

Exercise: Job Control

- Create a sample job: samplejob.sh
  echo "Line 1"
  sleep 10
  echo "Line 2"
  sleep 10
  echo "Line 3"

  Add executable permission to samlejob.sh

  Open System Monitor to watch your job.
  Open terminal to run:
  ./samplejob.sh (execute samplejob)
  Ctrl-C (stop job)
  ./samplejob.sh
  ./samplejob.sh > job.log (send output to job.log)

Exercise: Job Control

- Examine job.log

  ./samplejob.sh >> job.log (append out to job.log)

  Examine job.log

  ./samplejob.sh & (Run the job at backgroup)
  ./samplejob.sh > job.log &
  exit (exit terminal)

  Open a new terminal
  ./samplejob.sh
  Ctrl-Z (suspend job)
  jobs (list jobs)
  fg (continue job at foreground)
  Ctrl-Z
  bg (continue job at backgroud)
  jobs

Symbolic Links

- A pseudo-file which behaves as an alternative name for some other file
  or directory

  Open a terminal and type
  ls -l /usr/tmp

  Create a symbolic link to typing
  ln -s a.sh alink.sh

  A symbolic can also be created from GUI.

  ls -l (list directory)
  ./alink.sh (same as execute a.sh)

Hard links

- Refers to another file by inode number

ln a.sh ahard.sh (create hard link)
rm a.sh (remove a.sh)
./alink.sh (execute alink.sh)
./ahard.sh (execute ahard.sh)
ln ahard.sh a.sh
Modify a.sh
View ahard.sh

Special Directory Permissions: 'Sticky'(1)

- Only a file's owner may delete it from a sticky directory
  chmod +t <directory>

  Special Directory Permissions: Setgid (set group-id)

  Files created within it acquire the group ownership of the directory
  and directories created within it acquire both the group ownership and
  setgid permission
  Useful for a shared directory where all users working on its files are
  in a given group
  chmod g+s <directory>

Special Directory Permissions: 'Sticky'(2)

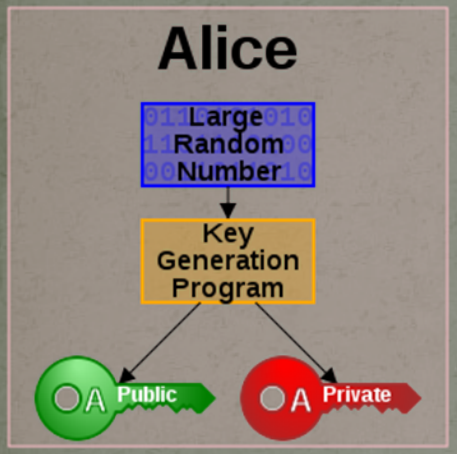
- Special File Permissions: Setgid

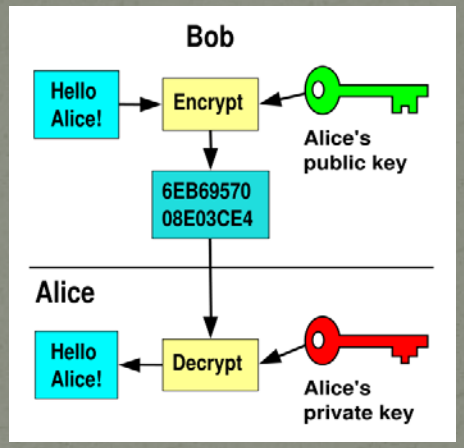
  A process run from a setgid file acquires the group id of the file
  Note: Linux doesn't directly allow scripts to be setgid — only compiled programs

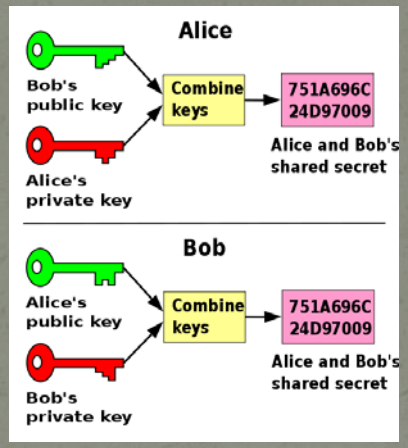
  Special File Permissions: Setuid (set user-id)
  chmod g+s <program>

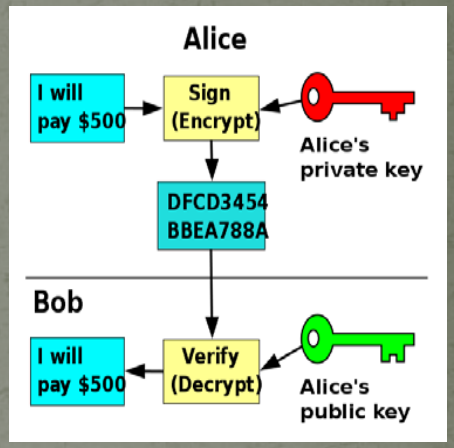
  Equivalent to setgid: a process run from a setuid file acquires the user id of the file
  As with setgid, Linux doesn't allow scripts to be setuid
  chmod u+s <program>

SSL









-

Test your web pages:
https://localhost

Certification Authority (CA)

- What is e-cert?

  Free: http://www.cacert.org/
  Paid: http://www.hongkongpost.gov.hk

Exercise: Install e-cert to your server(1):

- openssl genrsa -des3 -out columny.t10.key 1024
  (Generate private and public keys)

  openssl req -new -key columny.t10.key -out columny.t10.csr
  (Generate Certificate Signing Request, common name=www.columny.t10)

  cat columny.t10.csr
  Copy and paste to the your CA to request a signed certificate.
  Your CA will generate a cerficate file to you.
  Download the file and save as columny.t10.crt

  If you just want to test:
  openssl req -new -x509 -nodes -sha1 -days 365 -key server.key -out server.crt
  Create a self-signed Certificate

Exercise: Install e-cert to your server(2):

- Copy columny.t10.key to /etc/pki/tls/private
  Copy columny.t10.crt to /etc/pki/tls/certs

  Edit /etc/httpd/conf.d/ssl.conf
  SSLCertificateFile /etc/pki/tls/certs/columny.t10.crt
  SSLCertificateKeyFile /etc/pki/tls/private/columny.t10.key

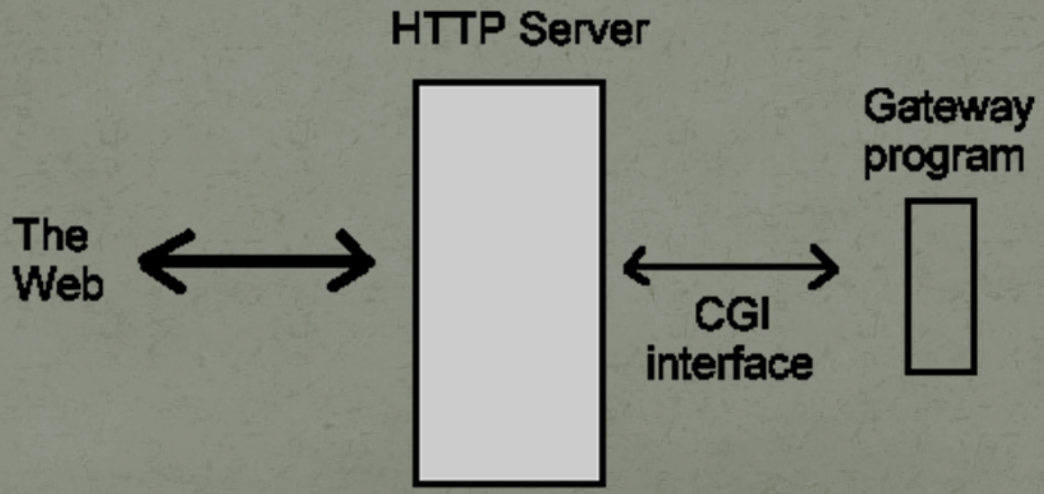
  Restart httpd service.

  /etc/init.d/httpd restart

  Remove password:
  mv columny.t10.key columny.t10.key.old
  openssl rsa -in columny.t10.key.old -out columny.t10.key

CGI

- Perl
  Perl is an ideal web programming language due to its text manipulation
  capabilities and rapid development cycle.



Exercise

- Create testpage.pl in /var/www/cgi-bin

```
print "<html>\r\n";
print "<body>\r\n";
print "This page is generated by PERL\r\n";
print "</body>\r\n";
print "</html>";
```

Add executable permission
Open terminal
perl testpage.pl

Use browser: http://localhost/cgi-bin/testpage.pl

Type
tail -f /var/log/httpd/error_log
to read error.

Add the following two lines.

```
#!/usr/bin/perl
print "Content-type: text/html\r\n\r\n";
```

Virtual host

- Allowed multiple websites to be hosted on one server . SSL is a virutal host: /etc/httpd/conf.d/ssl.conf

Exercise(1):

- Edit the end of httpd.conf:

NameVirtualHost:80

```
<VirtualHost *:80>
    ServerAdmin columnyl@columny.t10
    DocumentRoot /home/columnyl/www
    ServerName www.columnyl.hk
    ErrorLog logs/www.columnyl.hk-error_log
    CustomLog logs/www.columnyl.hk-access_log common
    ScriptAlias /cgi-bin/ "/home/columnyl/cgi-bin/"
</VirtualHost>
```

Add a VirtualHost session for the existing service.

Add www.columnyl.hk to DNS service.

Browse www.columnyl.hk

Create directory /home/columnyl/www and add index.html.

```
<html>
<body>
This is a virtual host's page.
</body>
</html>
```

Exercise(2):

- Browse www.columnyl.hk again and fix any error. (Check www.columnyl.hk-error_log)

  Create directory /home/columnyl/cgi-bin and add testpage.pl

```
#!/usr/bin/perl
print "Content-type: text/html\r\n\r\n";
print "<html>\r\n";
print "<body>\r\n";
print "testing virtual host cgi-bin\r\n";
print "</body>\r\n";
print "</html>\r\n";
```

  Browse www.columnyl.hk/cgi-bin/testpage.pl

Squirrelmail

- Webmail
  Written in PHP
  Configuration tool: written in PERL

Install Squirrelmail

- Use application, Add/Remove Software to search and install squirrelmail
  Try to find the URL of the main page. (Hint: search and read a file of conf.d)

  Start dovecot service.

  Optional: Add index.php to DirectoryIndex of httpd.conf

  Go to main page and login.
  Try to use root to login

  Edit to /etc/aliases to forward all mails sent to root.
  root: columnyl

  /etc/aliases can be used to create email group.
  e.g. finance: peterchan@gmail.com,john,vincentlee@yahoo.com.hk

  newaliases (update aliases.db)

  Try to send email to root.

Squirrelmail plugins

- Download from www.squirrelmail.org, plugins

  Optional: Install Change_passwd plugin

  Requirement: Compatibility plugin
  Check matrix of plugins
  Download compatibility-1.3.tar.gz and change_passwd-4.0-1.2.8.tar.gz to /usr/share/squirrelmail/plugins

  Install the plugin:
  Check /usr/share/squirrelmail/plugins/compatibility/INSTALL and /usr/share/squirrelmail/plugins/change_passwd/INSTALL

  cd /usr/share/squirrelmail/plugins/change_passwd
  cp config.php.sample config.php

  Turn on plugins:
  /usr/share/squirrelmail/config/conf.pl
  Select plugins and then Change_passwd and Compatibility plugin

  Login and select options to change password.

Exercise: Change options.php to force a strong password.

- Use mixed case letters (upper and lower case)
  Use more than one number
  Use special characters (!,@,#,$,%,^,&,*,?,_,~)
  Make your password 8 characters or more

  Find the source code of change password page:
  View the page properties from the browser.

  Change the original instruction to the above instruction.

```
if ($gochange) {
    $strength=0;
    if (preg_match("/[a-z]/i", $new_pw1)) {
      $strength += 5;
      if (preg_match("/([a-z].*[A-Z])|([A-Z].*[a-z])/", $new_pw1)) {
        $strength += 2;
      }
    }
    if (preg_match("/[0-9]/", $new_pw1)) {
      $strength += 5;
      if (preg_match("/[0-9].*[0-9]/", $new_pw1)) {
        $strength += 2;
      }
    }
    if (preg_match("/[\!\@\#\$\%\^\*\?\_\~\-]/", $new_pw1)) {
      $strength += 5;
      if (preg_match("/[\!\@\#\$\%\^\*\?\_\~\-].*[\!\@\#\$\%\^\*\?\_\~\-]/",
$new_pw1)) {
        $strength += 2;
      }
    }
    if ($strength < 14) {
      $msg = _("The passowrd is too weak. Please refer to the instruction.");
      $gochange = 0;
    }
  }
```

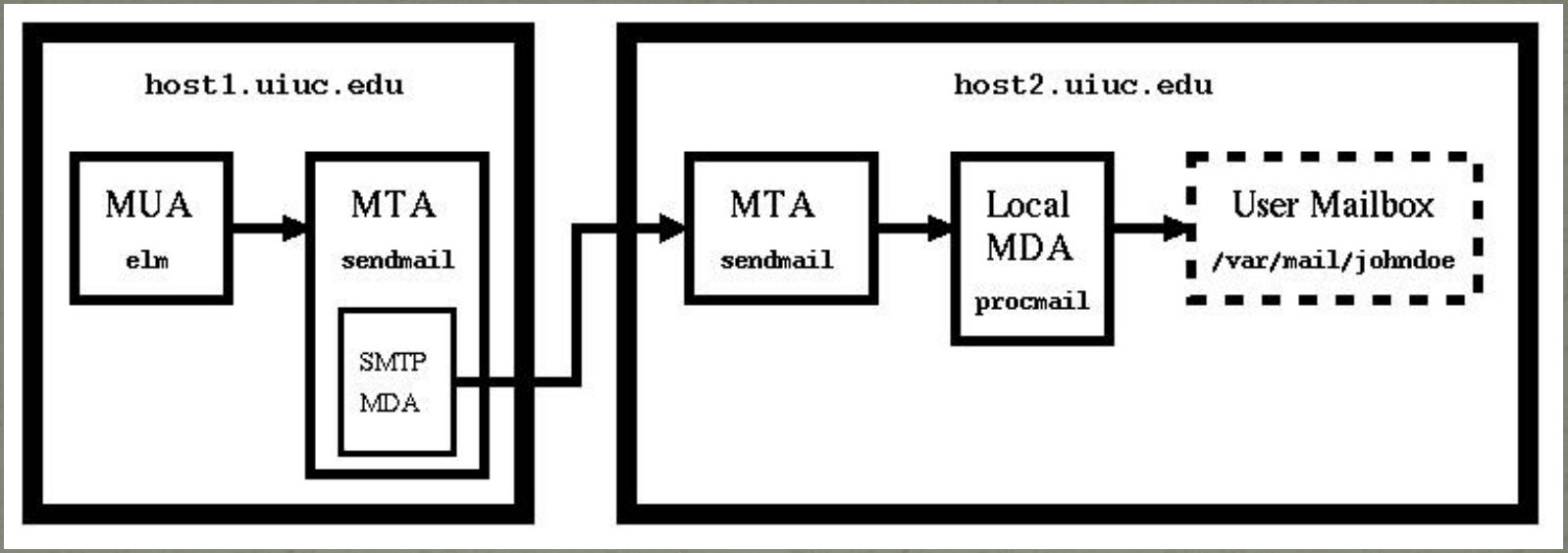
Force SSL connection

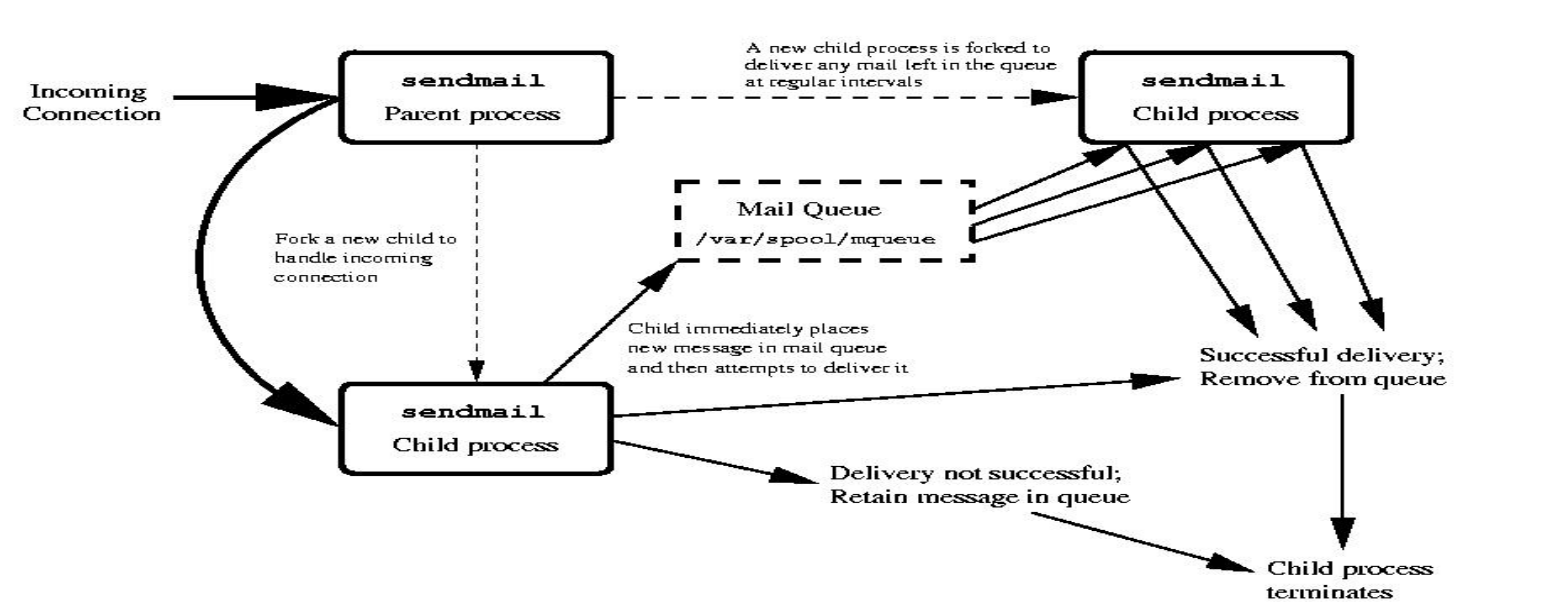
- Add the following code to /etc/httpd/conf.d/squirrelmail.conf

```
<Directory /usr/share/squirrelmail>
  SSLRequireSSL
</Directory>
```

Try http://localhost/webmail
Try https://localhost/webmail

Sendmail


host1.uiuc.edu
MUA
elm
MTA
sendmail
SMTP
MDA
host2.uiuc.edu
MTA
sendmail
Local
MDA
procmail
User Mailbox
/var/mail/johndoe


Incoming
Connection
sendmail
Parent process
A new child process is forked to
deliver any mail left in the queue
at regular intervals
sendmail
Child process
Mail Queue
/var/spool/mqueue
Fork a new child to
handle incoming
connection
Child immediately places
new message in mail queue
and then attempts to deliver it
sendmail
Child process
Successful delivery;
Remove from queue
Delivery not successful;
Retain message in queue
Child process
terminates

Sendmail

- Inbox: /var/mail/<username>
  Other mail folders: /home/<username>/mail/
  Log file: /var/log/maillog
  Configuration file: /etc/mail/sendmail.mc
  Execute the following command when sendmail.mc is changed.
  make -C /etc/mail

  .forward: forward email for each user
  e.g. in /home/columnyl/.forward
  \columnyl
  columnyl@gmail.com
  johnchan
  peter@hotmail.com

Sendmail

- Add Mail Exchange entry to DNS and then test by nslookup -querytype="MX" columny.t10

- Change /etc/mail/sendmail.mc
  From
  LOCAL_DOMAIN(`localhost.localdomain')dnl
  To
  LOCAL_DOMAIN(`columnyori.t10')dnl

Exercise

- Send mail to each other.

Spam

- start spamd service
  cp /etc/mail/spamassassin/spamassassin-spamc.rc /etc/procmailrc

  sa-update:
  Update rules
  Better to run once a day
  Restart spamd after each run.
  sa-update && service spamassassin restart

  sa-learn --spam ~/mail/saved-spam-folder
  sa-learn --ham ~/mail/inbox
  sa-learn --ham ~/other-nonspam-folder

Configuration file /etc/mail/spamassassin/local.cf

- whitelist_from joe@example.com fred@example.com
  whitelist_from *@example.com
  blacklist_from

  whitelist_from_rcvd
  whitelist_from_rcvd joe@example.com example.com
  whitelist_from_rcvd *@axkit.org sergeant.org

  whitelist_to
  more_spam_to
  all_spam_to

  blacklist_to

Exercise:

- Create three users: spamuser1, spamuser2, gooduser
  Add spamuser1@columny.t10 to blacklist of local.cf

  Watch the log: tail -f var/log/maillog
  Use spamuser1 to send email to gooduser.
  Use spamuser2 to send email to gooduser.

ftp

- Start vsftpd service.
  Config files: /etc/vsftpd/vsftpd.conf, ftpusers, user_list

freenx

- Use Applications, Add/Remove application to install freenx application/thin-client server
  Start freenx-server

  Go to www.nomachine.com to download and install NX clients for Linux.

  Run NX client
  Select configure, key, import
  Select file /var/lib/nxserver/home/.ssh/client.id_dsa.key

  Connect to other machines.

Generate own private and public keys

- ssh-keygen -t dsa
  no passphrase
  ftp id_dsa to client
  concatinate id_dsa.pub to /var/lib/nxserver/home/.ssh/authorized_keys2
  Optional: Remove the first part of authorized_keys2
  Use client to import id_dsa key.

  Connect to the server.

SSH

- ssh <other pc>

  Configuration file:
  /etc/ssh/sshd_config

Boot sequence

- BIOS
  First boot device (e.g. hard disk)
  GRUB stage 1 of Master Boot Record (MBR: The primary boot loader (446
  bytes) and the partition table (64 bytes))
  GRUB stage 2 (/boot/grub/grub.conf or /boot/grub/menu.lst)
  Kernel
    Initializes hardware
    loads drivers
    Initializes RAID and LVM
    Creates and mounts a root partition read-only
  /sbin/init
    /etc/rc.d/rc.sysinit
    /etc/inittab
    /etc/rc.d/init.d/functions
    /etc/rc.d/rc.local

Tomcat

- Start tomcat5 service
  Go to test page
  http://localhost:8080

- Allow 8080 TCP and 8443 TCP to pass through firewall.

- Log file: /var/log/tomcat5/catalina.out

Testing program

- Create a program
  HelloWorld.java

```
// Filename : HelloWorld.java
// Description : This servlet merely says hello!


import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class HelloWorld extends HttpServlet {

      public void doGet ( HttpServletRequest request,
HttpServletResponse response )
      throws ServletException, IOException    {

            response.setContentType("text/html");
            PrintWriter out = response.getWriter();

            out.println("<html>");
            out.println("<body>");
            out.println("This is servlet.");
            out.println("</body>");
            out.println("</html>");
      }// end doGet

}///:~
```

  Compile:
  javac -classpath /usr/share/java/tomcat5-servlet-2.4-api.jar HelloWorld.java

Deploy(1)

- Under this directory:
  /var/lib/tomcat5/webapps

  Create
  HelloWorld (Web application root directory)
  HelloWorld/WEB-INF (Configuration file web.xml)
  HelloWorld/WEB-INF/classes (Servlets)
  HelloWorld/WEB-INF/lib (Additional support libraries)

  Create
  HelloWorld.xml in /etc/tomcat5/Catalina/localhost
  <!-- HelloWorld Context -->
  <Context path="/HelloWorld" docBase="HelloWorld" debug="0"
  reloadable="true"/>

  Create
  web.xml in HelloWorld/WEB-INF
  <?xml version="1.0" encoding="ISO-8859-1"?>

  <!DOCTYPE web-app
  PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
  "http://java.sun.com/dtd/web-app_2_3.dtd">

  <web-app>

Deploy(2)

- <servlet>
  <servlet-name>HelloWorld</servlet-name>
  <servlet-class>HelloWorld</servlet-class>
  </servlet>

  <servlet-mapping>
  <servlet-name>HelloWorld</servlet-name>
  <url-pattern>/HelloWorld</url-pattern>
  </servlet-mapping>

  </web-app>

  Copy HelloWorld.class to HelloWorld/WEB-INF/classes

  Testing
  http://localhost:8080/HelloWorld/HelloWorld

  Configuration files:
  /etc/tomcat5

  User list
  /etc/tomcat5/tomcat-users.xml

jsp

- Create the helloworld.jsp in /var/lib/tomcat5/webapps/HelloWorld

```
<html>
<body>
This is jsp.
</body>
</html>
```

Testing http://localhost:8080/HelloWorld/helloworld.jsp

Admin page

- Admin page

  Add the following lines to /etc/tomcat5/
  <role rolename="manager"/>
  <role rolename="admin"/>
  <user username="admin" password="admin"
  roles="admin,manager"/>

  Use http://localhost:8080 to enter administraion page.

  /etc/tomcat5/web.xml: Global settings.
  e.g. index.jsp

Security(Basic)

- Add the following code to web.xml of HelloWorld

```
<security-constraint>
  <web-resource-collection>
    <web-resource-name>HelloWorld</web-resource-name>
    <url-pattern>/HelloWorld/*</url-pattern>
  </web-resource-collection>
  <auth-constraint>
     <role-name>manager</role-name>
  </auth-constraint>
</security-constraint>

<login-config>
  <auth-method>BASIC</auth-method>
  <realm-name>HelloWorld</realm-name>
</login-config>

<security-role>
  <description>
    The role that is required to log in to HelloWorld
  </description>
  <role-name>manager</role-name>
</security-role>
```

Test your page.

Security(Form)

- Change the above code to

```
<security-constraint>
  <display-name>HelloWorld</display-name>
  <web-resource-collection>
    <web-resource-name>Protected Area</web-resource-name>
    <url-pattern>/HelloWorld/*</url-pattern>
  </web-resource-collection>
  <auth-constraint>
    <role-name>manager</role-name>
  </auth-constraint>
</security-constraint>

<login-config>
  <auth-method>FORM</auth-method>
  <realm-name>HelloWorld</realm-name>
  <form-login-config>
    <form-login-page>/login.jsp</form-login-page>
    <form-error-page>/error.jsp</form-error-page>
  </form-login-config>
</login-config>

<security-role>
  <description>
    The role that is required to log in HelloWorld
  </description>
  <role-name>admin</role-name>
</security-role>
```

Security(Form 2)

- Add login.jsp to /var/lib/tomcat5/webapps/HelloWorld

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Login Page</title>
</head>
<body>
<h1>Login HelloWorld</h1>
<p>
Enter username and password
</p>
<form method="POST" action="j_security_check">
Username : <input type="text" size="15" maxlength="25"
name="j_username"><br><br>
Password : <input type="password" size="15" maxlength="25"
name="j_password"><br><br>
<input value="Login" type="submit">    <input
value="Clear" type="reset">
</form>
</body>
</html>
```

Security(Form 3)

- Add error.jsp

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Login Error</title>
</head>
<body>
<h1>HelloWorld login Error!</h1>
<p>
Wrong username or password.
</p>
</body>
</html>
```

Test your pages.

Security(IP)

- Change /etc/tomcat5/Catalina/localhost/HelloWorld.xml to

```
<!-- HelloWorld Context -->
 <Context path="/HelloWorld" docBase="HelloWorld"
    debug="0" reloadable="true">
   <Valve
    className="org.apache.catalina.valves.RemoteAddrValv
    e" allow="127.0.0.1,192.168.1.*"/>
 </Context>
```

Test your pages.

Tomcat SSL

- Generate private key and public key and store in keystore

  keytool -genkey -alias tomcat -keyalg RSA -keystore columny.t10.keystore

- Generate ecert request

  keytool -certreq -keyalg RSA -alias tomcat -file columny.t10.csr -keystore columny.t10.keystore

- Send the request to CA

Tomcat SSL (2)

- The following commands are for testing only.

  javac Base64Coder.java ExportPriv.java

  openssl genrsa -des3 -out columny.t10.cakey 1024

  openssl req -new -x509 -days 365 -key columny.t10.cakey -out columny.t10.cacrt

  java ExportPriv columny.t10.keystore tomcat changeit>columny.t10.pkcs8.key

  Generate ecert request

  Break each line of columny.t10.pkcs8.key to 64 characters.

  openssl pkcs8 -inform PEM -nocrypt –in columny.t10.pkcs8.key -out columny.t10.key

  openssl x509 -req -days 365 -in columny.t10.csr -signkey columny.t10.key -out columny.t10.crt

Tomcat SSL (3)

- Import CA ecert to keystore

  keytool -import -alias root -keystore columny.t10.keystore -trustcacerts -file columny.t10.cacrt

- Import columny.t10 ecert to keystore

  keytool -import -alias tomcat -keystore columny.t10.keystore -trustcacerts -file columny.t10.crt

Tomcat SSL (4)

- Move keystore to conf directory
  mv columny.t10.keystore /etc/tomcat5/.keystore
- Change /etc/tomcat5/server.xml:
  Remove the remark of
  <Connector port="8443" maxHttpHeaderSize="8192"
  maxThreads="150" minSpareThreads="25"
  maxSpareThreads="75"
  enableLookups="false" disableUploadTimeout="true"
  acceptCount="100" scheme="https" secure="true"
  clientAuth="false" sslProtocol="TLS"/>
  Add
  keystoreFile="/etc/tomcat5/.keystore"
  keystorePass="changeit“
- Test the page
  https://localhost:8443

Tomcat force SSL

- Change /var/lib/tomcat5/webapps/HelloWorld/WEB-INF/web.xml

```
<security-constraint>
  <display-name>HelloWorld</display-name>
  <web-resource-collection>
    <web-resource-name>Protected Area</web-resource-name>
    <url-pattern>/HelloWorld/*</url-pattern>
  </web-resource-collection>
  <auth-constraint>
    <role-name>manager</role-name>
  </auth-constraint>
  <user-data-constraint>
    <transport-guarantee>CONFIDENTIAL</transport-guarantee>
  </user-data-constraint>
</security-constraint>
```

- Test the page
  http://localhost:8080

MYSQL(1)

Start service mysqld
Log file: /var/log/mysqld.log
Database directories: /var/lib/mysqld/

Type mysql to enter mysql command prompt.
Type quit to exit.

select user();

create database t10; (drop database t10;)

show databases;

create user 'columnyl'@'localhost' identified by '123456'; (drop user columnyl;)
grant all on t10.* to 'columnyl'@'localhost' with grant option;
(revoke all on t10.* from 'columnyl'@'localhost';)

show grants for 'columnyl'@'localhost‘;

MYSQL(2)

- Logout and login columnyl.
  mysql -h localhost -u columnyl -p t10

  use t10
  select database();

  create table pc (name varchar(20), ip char(15)); (drop table pc;)
  show tables;
  describe pc;

  insert into pc values ('columny.t10', '172.20.54.28'); (delete from pc
  where name='columny.t10';)
  insert into pc values ('henry.t10', '172.20.54.125');
  select * from pc;

Connect MYSQL through jdbc(1)

- download connector/j from http://dev.mysql.com/

  copy mysql-connector-java-5.1.11-bin.jar
  to /var/lib/tomcat5/webapps/HelloWorld/WEB-INF/lib

  Create MysqlServlet.java

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;

public class MysqlServlet extends HttpServlet {

    public void doGet ( HttpServletRequest request,
HttpServletResponse response )
    throws ServletException, IOException {
        String ip="";
try {
        Class.forName("com.mysql.jdbc.Driver").newInstance();
        Connection dbCon =
```

Connect MYSQL through jdbc(2)

- DriverManager.getConnection"jdbc:mysql:///t10","columnyl","123456");
  ResultSet r = dbCon.createStatement().executeQuery("select ip from pc");
  r.next();
  ip = r.getString("ip");
  r.close();
  dbCon.close();
  } catch (Exception e)
  { e.printStackTrace(); }
  response.setContentType("text/html");
  PrintWriter out = response.getWriter();

  out.println("<html>");
  out.println("<body>");
  out.println("IP:");
  out.println(ip);
  out.println("</body>");
  out.println("</html>");
  }// end doGet

  }///:~

Connect MYSQL through jdbc(3)

- javac -classpath /usr/share/java/tomcat5-servlet-2.4-api.jar MysqlServlet.java
  cp MysqlServlet.class /var/lib/tomcat5/webapps/HelloWorld/WEB-INF/classes

  Add tags to /var/lib/tomcat5/webapps/HelloWorld/WEB-INF/web.xml

```
<servlet>
  <servlet-name>MysqlServlet</servlet-name>
  <servlet-class>MysqlServlet</servlet-class>
</servlet>

<servlet-mapping>
  <servlet-name>MysqlServlet</servlet-name>
  <url-pattern>/MysqlServlet</url-pattern>
</servlet-mapping>

</web-app>
```

  Testing
  https://localhost:8443/HelloWorld/MysqlServlet

JSP and javabean(1)

- Create a javabean MysqlJavabean.java

```
package chkip;

import java.sql.*;

public class MysqlJavabean implements java.io.Serializable {

    private String name;

    private String ip;

    private Connection dbCon;

    /** No-arg constructor (takes no arguments). */
    public MysqlJavabean() {
      try {
        Class.forName("com.mysql.jdbc.Driver").newInstance();
        dbCon =
DriverManager.getConnection("jdbc:mysql:///t10","columnyl","123456");
      } catch (Exception e)
      { e.printStackTrace(); }
    }

    /**
     * Property <code>name</code> (note capitalization) readable/writable.
     */
    public String getName() {
        return this.name;
    }
```

JSP and javabean(2)

```
    /**
     * Setter for property <code>name</code>.
     * @param name
     */
    public void setName(final String name) {
        this.name = name;
    }

    public String getIp() {
      try {
        PreparedStatement s = dbCon.prepareStatement("select ip from
pc where name=?");
        s.setString(1, this.name);
        ResultSet r = s.executeQuery();
        r.next();
        this.ip = r.getString("ip");
        r.close();
      } catch (Exception e)
      { e.printStackTrace(); }
      return this.ip;
    }

    public void setIp(final String ip) {
        this.ip = ip;
    }
}
```

JSP and javabean(3)

- javac MysqlJavabean.java
  mkdir /var/lib/tomcat5/webapps/HelloWorld/WEB-INF/classes/chkip
  mv MysqlJavabean.class /var/lib/tomcat5/webapps/HelloWorld/WEB-INF/classes/chkip

  Create a jsp file in /var/lib/tomcat5/webapps/HelloWorld/checkip.jsp

```
<jsp:useBean id="mysqlip" class="chkip.MysqlJavabean" scope="page"/>
<jsp:setProperty name="mysqlip" property="name"/>

<html>
<body>
Name: <jsp:getProperty name="mysqlip" property="name"/><br/>
IP: <jsp:getProperty name="mysqlip" property="ip"/><br/>
<br/>
<form name="checkip" method="POST" action="checkip.jsp">
Enter a name: <input type="text" name="name" size="50"><br/>
<input type="submit" value="getip">
</form>
</body>
</html>
```

  Testing:
  https://localhost:8443/HelloWorld/checkip.jsp