COURSE SYLLABUS

UNIX System Administration

50 Cragwood Rd, Suite 350
South Plainfield, NJ 07080

Victoria Commons, 613 Hope Rd Building #5,
Eatontown, NJ 07724

130 Clinton Rd,
Fairfield, NJ 07004

Avtech Institute of Technology Course

Instructor:
Course Duration: 60
Date/Time:
Training Location:

Course: UNIX System Administration

Text / Lab Books

Lecture Material and Workshop provided

Course Description

The UNIX certification program is concerned with the development and evolution of certification programs associated with the UNIX system trademark. This includes end-to-end certification development, from policy, product standards through to test suite development and acceptance.

Linux is an operating system for computers, comparable to Windows or Mac OSX, it resembles UNIX, is an open source software, you can download all the source code of the operating system and run it on your computer. Linux runs on a wide variety of hardware platforms, from huge mainframes to desktop PCs to cell phones. It is licensed under the Free Software Foundation's GNU Project's GNU General Public License, version 2, which lets users modify and redistribute the software.

This course provides complete information of the Single UNIX Specification in a Linux OS and comprehensive reference material as well the GUI and application programs, the vi, Borne/Korn Shell and Scripting also will teach in this class.

You can think of Linux as having two parts -- a kernel, which is the basic interface between the hardware and other system software, and the functions that run on top of it, such as a graphical user interface (GUI) and application programs.

Learning Objectives

Part I. History and Culture

1.0 The Open Group Single UNIX Specification

1.1. The Technical Standards
- 1.1.1. Base Definitions (XBD)
- 1.1.2. Shell and Utilities (XCU)
- 1.1.3. System Interfaces (XSH)

1.1.4. Rationale (Informative)

1.2. Getting to know an Operating System

1.2.1. UNIX System Chronology: Linux®, BSDI, IBM® 's OS/390, Windows® NT, Digital® UNIX, Hewlett Packard HP-UX®, IBM AIX®, SCO UnixWare®, SGI IRIX®, Sun Solaris®

1.2.2. Certification Programs

1.2.3. How to Use Minis and Mainframes

1.3. File System: Filenames, Dealing with Files: The Most Important Commands, The Text Editor, Command Files, Sending and Receiving Mail

1.4. General Advice: Filenames, Wildcards and File Deletion, Mail, The Text Editor and Line Editors, Full-Screen Editors, The Editing Buffer, Looking at Text Files, "Printing" on the Screen, Reading and Writing, Logging Off (or Out), Terminal Emulation and File Transfer, Emulated Terminals, Syntax Expressions, Comments and Suggestions

1.5. USENET

2.0 The UNIX vi Text Editor

2.1. Entering vi

2.2. Inserting Text, Deleting Text, Typing Over Existing Text, Searching for Text

2.3. Saving Changes

2.4. Quitting vi

2.5. Other vi commands

3.0 Working Knowledge of UNIX, VMS, OS/400, VM/CMS, and MVS

3.1. History of the operating system: OpenVMS, OS/400 IBM's AS/400, VM/CMS, IBM mainframe operating system, MVS. Another IBM mainframe operating system

3.2. Starting and ending a session

3.3. Filenames, how files are organized and how to navigate, Available on-line help, Creating, copying, renaming, and deleting files

3.4. Using the text editor and Printing text

3.5. Creating and running command files

3.6. Sending and receiving mail

4.0 Base Definitions (XBD)

4.1. Scope of the Base Specifications and the changes made in this revision, Normative references, terminology, and portability codes used throughout the Base Specifications

4.2. Implementation and Application Conformance requirements: the general terms and definitions that apply throughout the Base Specifications

4.3. General concepts that apply throughout the Base Specifications

4.4. Notation used to specify file input and output formats in XBD and XCU

4.5. Portable character set and the process of character set definition

4.6. Syntax for defining internationalization locales as well as the POSIX locale provided on all systems

4.7. Use of environment variables for internationalization and other purposes

4.8. Syntax of pattern matching using regular expressions employed by many utilities and matched by the regcomp() and regexec() functions

4.9. Files and devices found on all systems and their semantics. For example, the device /dev/null is an infinite data source and data sink
4.10. Asynchronous terminal interface for many of the functions in XSH and the stty utility in XCU
4.11. Policies for command line argument construction and parsing. It contains the utility argument syntax used throughout XCU, and also utility syntax guidelines for naming of utilities and the specification of their arguments and option-arguments and operands
4.12. Contents of headers which declare constants, macros, and data structures that are needed by programs using the services provided by the system interfaces defined in XSH

5.0 System Interfaces (XSH)

5.1. Status of this document and its relationship to other formal standards
- 5.1.1. Scope, conformance, and definitions (sections are pointers to the XBD document)
- 5.1.2. The meet of ISO/IEC rules
- 5.1.3. Terminology and portability (codes are identical to the section in XBD)
- 5.1.4. Important concepts, terms and caveats relating to the following: Compilation environment, the name space, definitions of error numbers, signal concepts, standard I/O streams, STREAMS, XSI IPC, realtime, threads, sockets, tracing, and data types
- 5.1.5. Functional interfaces to systems conformant of Single UNIX

5.2. The System Interfaces and Headers (XSH)
- 5.2.1. POSIX.2 C-language binding, Single UNIX Specification, Shared Memory
- 5.2.2. Enhanced Internationalization adding functions
- 5.2.3. XPG4 Base for conforming implementations only the base interfaces were mandatory

6.0 Shell and Utilities (XCU)

6.1. Formal standards, including the ISO C standard and also the XSH document
- 6.1.1. Utility limits, grammar conventions, defaults used by the utility descriptions, considerations for utilities in support of large files, and the list of required built-in utilities.
- 6.1.2. Scope, conformance, and definitions in XBD

6.2. Command language-the shell command language interpreter-used in systems conformant to the Single UNIX Specification
6.3. Implementation a set of services and utilities on systems supporting the Batch Environment option
6.4. Utilities available on systems conforming to the Single UNIX Specification

7.0 Using UNIX System

7.1. Printing Text Files. Checking the Print Queue, Canceling Your Print Job
7.2. Command Files
5.3. Starting Up and Finishing Your UNIX Session
5.4. Using Files in UNIX
- 5.4.1. The Eight Most Important Commands
- 5.4.2. Command Options: Switches, Common Error Messages
- 5.4.3. Listing Filenames, Listing More than One File Names

5.4.4. Displaying a Text File's Contents, Looking at Text Files One Screen at a Time
5.4.5. Copying Files, Renaming Files, Deleting Files
5.5. Controlling Access to a File, Creating Directories, Removing Directories

8.0 Rationale (XRAT)

Part II. Basic UNIX Environment

1.0 Introduction
1.1. The Core of Unix and Communication with Unix
1.2. Programs Are Designed to Work Together
1.3. Program Shell and run Shell
1.4. Internal and External Commands
1.5. The Kernel and Daemons
1.6. Filenames, Filename Extensions, and Wildcards
1.7. The Tree Structure of the Filesystem, Your Home Directory, Making Pathnames
1.8. File Access Permissions and The *superuser* (Root)

2.0 Getting Help
2.1. The man Command
2.2. *whatis*: One-Line Command Summaries, *whereis*: Finding Command is Located
2.3. Searching Online Manual Pages
2.4. How UNIX Systems Remember Their Names, Which Version Am I Using?
2.5. What *tty* Am I On? , Who's On?
2.6. The info Command

Part III. Customizing Environment

1.0 Setting Up Unix Shell
1.1. Log in to Mac OS X Terminal Application
1.2. Shell Setup Files, Login Shells, Interactive Shells
1.3. Automatic Setups for Different Terminals, Testing TERM, Testing Remote Hostname and X Display
1.4. Testing Port, Testing Environment Variables

2.0 Interacting with Environment
2.1. Basics of Setting the Prompt (Static, Dynamic, Simulating Dynamic)
2.2. C-Shell Prompt, Faster Prompt Setting with Built-ins, Multiline Shell Prompts
2.3. Session Info in Window Title or Status Line
2.4. A "Menu Prompt" for Naive Users, Highlighting and Color in Shell Prompts
2.5. Right-Side Prompts , Show Subshell Level with $SHLVL
2.6. Running Commands When You Log Out/at Bourne/Korn Shell Logout
2.7. Stop Accidental Bourne-Shell Logouts

3.0 Getting the Most out of Terminals, xterm, or X Windows
3.1. Terminals and Terminal Database, Setting the Terminal Type at Log In
3.2. Querying Terminal Type
3.3. Setting Erase, Kill, and Interrupt Characters

3.4. Working with xterm and Friends, and login xterms and rxvts
3.5. Problems with Large Selections
3.6. Tips for Copy and Paste Between Windows
3.7. Running a Single Command with xterm –e, Don't Quote Arguments to xterm -e

4.0 Your X Environment
4.1. Defining Keys and Button Presses with xmodmap, using xev to Learn Keysym Mappings
4.2. X Resource Syntax
4.3. X Event Translations
4.4. Setting X Resources: Setting Resources with the -xrm Option
4.5. How -name Affects Resources
4.6. Setting Resources with xrdb, Listing the Current Resources for a Client: appres Starting Remote X Clients

Part III. Working with Files and Directories

1.0 Directory Organization
1.1. *what*, *me*, Organized? Many Homes, Access to Directories
1.2. A bin Directory for Your Programs and Scripts, Private (Personal) Directories
1.3. Naming Files
1.4. Make More Directories and Making Directories Made Easier

2.0 Directories and Files
2.1. the *find* Command
2.2. The Three Unix File Times
2.3. Finding Oldest/Newest Files with *ls -t* and *ls –u*, List All Subdirectories with *ls –R*
2.4. The *ls -d* Option, Color *ls*, Some GNU *ls* Features, a *csh* Alias to List Recently Changed Files, Showing Hidden Files with *ls -A* and *–a*, Useful *ls* Aliases
2.5. Counting Files by Types, Listing Files by Age and Size
2.6. Picking a Unique Filename Automatically
2.7. Finding Files with *find*
2.8. Linking, Renaming, and Copying Files

3.0 Comparing Files
3.1. Checking Differences with *diff*, Comparing Three Different Versions with *diff3*
3.2. Comparing Two Files with *comm*, More Friendly *comm* Output
3.3. Showing What's in a File: *cat*
3.4. Searching Through Files: *grep*, Removing Files: *rm*
3.5. Optimizing Disk Space
3.6. Compressing a Directory Tree: Fine-Tuning
3.7. Save Space in Executable Files with strip
3.8. Disk Quotas

Part IV. Basic Editing

1.0 Spell Checking, Word Counting, and Textual Analysis
1.1. The Unix spell Command

1.2. Check Spelling Interactively with *ispell*

2.0 vi Tips and Tricks

2.1. Editing Multiple Files with vi, Edits Between Files, Local Settings for vi
2.2. Using Buffers to Move or Copy Text, Appending to an Existing File, Moving Blocks of Text by Patterns
2.3. Useful Global Commands (with Pattern Matches)
2.4. Per-File Setups in Separate Files, Filtering Text Through a Unix Command
2.5. vi File Recovery Versus Networked Filesystems, *vi -r* Recovered Buffers
2.6. Shell Escapes: Running One Unix Command While Using Another, Using vi Abbreviations as Commands (Cut and Paste Between vi's) and Fixing Typos
2.7. vi Line Commands versus Character Commands
2.8. Finding Your Place with *find, Undo*
2.9. Setting Up vi with the *.exrc* File

3.0 Creating Custom Commands in vi

3.1. Save Time and Typing with the *vi map* Commands
3.2. vi @-Functions
3.3. Keymaps for Pasting into a Window Running
3.4. Protecting Keys from Interpretation by *ex*
3.5. Maps for Repeated Edits, Repeating a vi Keymap
3.6. vi Macro for Splitting Long Lines
3.7. File-Backup Macros

4.0 GNU Emacs

4.1. Backup and Auto-Save Files
4.2. Putting Emacs in Overwrite Mode
4.3. Command Completion
4.4. Rational Searches
4.5. Inserting Binary Characters into Files
4.6. Using Word-Abbreviation Mode
4.7. Directories for Emacs Hacks
4.8. An Absurd Amusement

5.0 Batch Editing

5.1. Writing Editing Scripts
5.2. Line Addressing, Useful ex Commands
5.3. Running Editing Scripts Within vi
5.4. ed/ex Batch Edits, *patch*: Generalized Updating of Files of Differ, Quick Reference: *awk*
5.5. Neatening Text with *fmt*, Alternatives to *fmt*
5.6. Clean Up Program Comment Blocks

6.0 Sorting

6.1. Putting Things in Order
6.2. Sort Fields: How sort Sorts, Changing the sort Field Delimiter
6.3. Alphabetic and Numeric Sorting
6.4. Miscellaneous sort Hints
6.5. *lensort*: Sort Lines by Length

6.6. Sorting a List of People by Last Name

Part V. Processes and the Kernel

1.0 Job Control
- 1.1. Job Control in a Nutshell
- 1.2. Job Control Basics
- 1.3. Some Gotchas with Job Control
- 1.4. Job Control and *autowrite*: Real Timesavers
- 1.5. Stop Background Output with *stty tostop*
- 1.6. *nohup*
- 1.7. Disowning Processes
- 1.8. Linux Virtual Consoles
- 1.9. Stopping Remote Login Sessions

2.0 Starting, Stopping, and Killing Processes
- 2.1. Managing Processes: Overall Concepts
- 2.2. *fork* and *exec*, Subshells, The *ps* Command ,The Controlling Terminal
- 2.3. Tracking Down Processes, The */proc* Filesystem
- 2.4. Killing Foreground Jobs, Destroying Processes with *kill*
- 2.5. Printer Queue Watcher: A Restartable Daemon Shell Script
- 2.6. Killing All Your Processes, Killing Processes by Name, Kill Processes Interactively
- 2.7. Cleaning Up an Unkillable Process
- 2.8. The Process Chain to Your Window, Terminal Windows Without Shells
- 2.9. Close a Window by Killing Its Process(es)

3.0 Delayed Execution
- 3.1. Building Software Robots the Easy Way
- 3.2. Periodic Program Execution: The *cron* Facility, Adding *crontab* Entries, Including Standard Input Within a *cron* Entry
- 3.3. The *at* Command
- 3.4. Making Your at Jobs Quiet, Checking and Removing Jobs
- 3.5. Avoiding Other at and *cron Jobs*
- 3.6. Waiting a Little While: sleep

4.0 System Performance and Profiling
- 4.1. Timing Programs
- 4.2. Checking System Load: uptime
- 4.3. Know When to Be "nice" to Other Users-and When Not To
- 4.4. A nice Gotcha
- 4.5. Changing a Running Job's Niceness

Part VI. Scripting

1.0 Shell Interpretation

2.0 Saving Time on the Command Line

3.0 Custom Commands

4.0 The Use of History

5.0 Moving Around in a Hurry

6.0 Regular Expressions (Pattern Matching)

7.0 Wildcards (*)

8.0 The *sed* Stream Editor

9.0 Shell Programming for the Uninitiated

10.0 Shell Script Debugging and Gotchas

Part VII. Extending and Managing Your Environment

1.0 Backing Up Files

2.0 Creating and Reading Archives

3.0 Software Installation

4.0 Perl--High-Octane Shell Scripting

5.0 Python Installation and Distutils, Python Basics, Python and the Web

Part VIII. Communication and Connectivity

1.0 Redirecting Input and Output

2.0 Devices

3.0 Printing

4.0 Connectivity

5.0 Connecting to MS Windows

Part IX. Security

1.0 Security Basics

2.0 Root, Group, and User Management

3.0 File Security, Ownership, and Sharing

4.0 SSH

Part IIX. Linux HowTos

1.0 The Linux OS

1.1. Switching from Other Operating Systems
1.2. Distributions and Installation
1.3. Kernel, Boot Loaders and Booting the OS
1.4. Parallel Processing, Partitions and Filesystems
1.5. RAID, Printing, Shell and Using Linux

2.0 System Administration and Configuration
2.1. Configuration / Installation
2.2. Benchmarking
2.3. Clustering
2.4. Backup and Recovery
2.5. Security

3.0 Hardware
3.1. General and Platforms
3.2. Video Cards, CPUs / Architectures, CD-ROM / DVD-ROM Drives, Optical Disks, Keyboard and Console, Digital Cameras, Graphic Tablets, Diskettes, Hard Disks
3.3. Jaz and ZIP Drives, Mice, Modems, Printers / Scanners, Routers, SCSI, Serial Ports, Sound Cards, Tape Drives, Touchscreens, UPS
3.4. Wireless
3.5. Miscellaneous

4.0 Networking
4.1. General
4.2. Protocols, Dial-up, DNS, Virtual Private Networks (VPN)
4.3. Bridging, Routing, Security
4.4. Telephony / Satellite
4.5. Miscellaneous

5.0 Applications / GUI / Multimedia
5.1. Installing Applications
5.2. User Applications / Server Applications
5.3. DBMS / Databases
5.4. Mail
5.5. Usenet Network News
5.6. HTTP / FTP
5.7. Miscellaneous
5.8. GUI / Window Managers
5.9. X Window System
5.10. Window Managers
5.11. Fonts
5.12. Audio and Video

6.0 Programming
6.1. General
6.2. Compilers
6.3. Languages
6.4. Libraries

6.5. Security
6.6. Version Control
6.7. DBMS / Databases

7.0 Other (human) Languages
7.1. Language Support
7.2. Using Specific Languages

8.0 Miscellaneous
8.1. Authoring / Documentation
8.2. Linux Advocacy / Getting (and Staying) Involved
8.3. Hobbies and Special Interests

Prerequisite

Familiarity with PC & Windows OS
General understanding of the principles of computer programming

Contact Hours

_____ Contact Hours (Lecture ___ Hours / Lab ____ Hours)

Semester Credit Hours

_____ semester credit hours

Teaching Strategies

A variety of teaching strategies may be utilized in this course, including but not limited to, lecture, discussion, written classroom exercises, written lab exercises, performance based lab exercises, demonstrations, quizzes and examinations. Some quizzes may be entirely or contain lab based components. A mid-course and end course examination will be given.

Method of Evaluating Students

Grade Distribution

Class Attendance	10
Mid Term	30
Finals	50
Special Projects Makeup projects	10
Total	**100%**

Grading Policy

At the end of each course, each student is assigned a final grade as follows:

Point Range	Interpretation	Grade	Quality Points
90 – 100	Excellent	A	4.0
80 – 89	Very Good	B	3.0 – 3.9
70 – 79	Average	C	2.0 – 2.9
60 – 69	Poor	D	1.0 – 1.9
Below 60	Failure	F	0
N/A	Withdrawal	W	0
N/A	Pass	P	0
N/A	Incomplete	I	0

A student earning a grade of D or above is considered to have passed the course and is eligible to pursue further studies. A student receiving a grade of F has failed the course. A failed course must be repeated and passed to meet Avtech Institute's graduation requirements, in addition to an overall program GPA of 2.0.

Requirements for Successful Completion of the Course

At a minimum, students must achieve the following:

- A passing grade of **D** or above
- Completion of all required examinations
- Submission of all required lab exercises and projects and;
- Adherence to the school attendance policy.

Equipment Needed

Industry standard desktop computer for lab exercises.

Equipment Breakdown Lab room

Videos and Projector

Library Assignments

To be determined by the instructor.

Portfolio Assignment

Student program outcome portfolios are required to demonstrate student competencies. In conjunction with your course structure, please select a project/paper that best demonstrates what you have learned in this course and add it to your program portfolio.

Course Policies

Disruptive Behavior

Disruptive behavior is an activity that interferes with learning and teaching. Inappropriate talking during class, surfing inappropriate website, tardiness, cheating, alcohol or drug use, use of cell phone, playing lout music during class, etc. all disrupt the learning process.

Copyright Infringement

Specific exemptions to copyright infringement are made for student use in the context of learning activities. Graphic design students often download images from the Internet, or scan images from publications. As long as this work is for educational purpose, and subject to faculty permission, this is not a problem.

Plagiarism

Faculty cannot tolerate the ***misrepresentation of work as the student's own***. This often involves the use by one student or another student's design, whether voluntarily or involuntarily. In the event that plagiarism is evident and documented, all students involved in the conscious decision to misrepresent work must receive an F as the grade for the project. A second occurrence may result in suspension for the rest of the quarter, and return to the school only after a review by the Academic Standards Committee.

Attendance

Attendance and Lateness

In education and the workplace, regular attendance is necessary if individuals are to excel. There is a direct correlation between attendance and academic success. Attendance is mandatory. All students must arrive on time and prepared to learn at each class session. At the faculty member's discretion, students may be marked absent if they arrive more than 15 minutes late to any class. More that five absences in a class that meets twice per week or more that two absences in a class that meets once per week may result in a failure.

Make-Up Work

Late Projects and Homework

All projects and homework must be handed in on time. Homework should be emailed to your instructor if you are going to miss a class. Work that is submitted one week late will result in the loss of one full grade; and work that is submitted two weeks late will result in the loss of two full grades; more than two weeks late you will receive a failing grade on the project.