A Cybersecurity Exercise In Linux: Student Learning Analysis

Jon D. Clark
jon.clark@colostate.edu

Stephen Hayne
stephen.hayne@colostate.edu

Computer Information Systems Department
College of Business
Colorado State University
Fort Collins, CO 80523

Abstract

The purpose of this experiment is to determine the level of learning involved in a Log Analysis exercise by novice graduate students who have very limited exposure to Linux. A pre-test was used to determine the level of understanding in cybersecurity and experience each student had relative to Linux. A post-test with many of the same assessments was administered to determine the amount of learning that occurred from the log analysis exercise. The log analysis required a set of six scripts to be written in Linux, applied incrementally as each was developed, producing the final results. The Linux commands required are those that novice students have been introduced to and have had a small degree of experience during the course. The log used in the exercise has approximately 10,800 rows, each documenting a login attempt and the outcome of that attempt. The log is sufficiently large that the students will find it impractical to determine the correct results using any brute force techniques.

Keywords: Cybersecurity, Log Analysis, Linux, Experiential Learning

1. INTRODUCTION

Colorado State University (CSU), a public land-grant research university, consists of eight academic colleges, and 55 departments. With leadership from the departments of Computer Science and Computer Information Systems, along with participation from Mathematics, Systems Engineering and Management, the Cybersecurity Center carries out funded research, and provides education and professional development. The College of Business currently offers four of thirteen courses consisting of Advanced Network and Security, Cyber Data Analytics, Information Assurance, and Supply Chain Security. Additionally, revisions to CIS606 and a title change to Software, Infrastructure and Security may be added to the curriculum.

Currently, CIS606 consists of three components and is intended as a foundation class for those who have minimal training and experience with computing. The first component addresses computer architecture at a very high level, how a computer works, the micro instruction cycle, data and storage structures as well as number systems including binary, octal, decimal and hexadecimal. The second component exposes students to Java followed by Python. Finally, component three covers Linux, and in particular, its use in access and security applications.

The purpose of this experiment is to determine the learning effectiveness of using a final assignment within the Linux component of the course material. In the exercise a series of

questions are posed about a large dataset of login attempts requiring simple Linux scripts.

2. ASSIGNMENT CONTEXT AND CONTENT

The Linux component of the class introduces UNIX, it's history, and in particular, the Linux dialect via Bourne-Again shell called bash. All commands are entered via the command line and the editor of choice is vi. Approximately 35 commands are covered, however, many others are provided in the instructional materials.

The first assignment covers a small set of commands allowing students to write, edit and save scripts as well as a set of permissions such that they may be executed (chmod). The second assignment requires that a .csv file be produced with records containing last name, first name, and date of birth. A loop allows multiple entries and each record is written to the screen (echo) as well as to a file, also defined at run-time. The file is then sorted on last name, output to the screen and evaluated in terms of lines, words and characters (wc).

The third assignment (Appendix A, Log Analysis Assignment), and the one that is part of this experiment, is common practice in a cybersecurity context. A login file (Appendix B, Sample Log File) with approximately 10,800 records of login attempts is provided and must be copied to a Linux virtual machine allocated to the class using vi. The file is sufficiently long that students will need to rely on a series of scripts to derive an answer to a set of six questions. Students are encouraged to develop the scripts incrementally to ensure that the various commands, separated by pipes (|) work as intended. The questions to be answered involve username identification of an upload, number of bytes downloaded, number of unique IP addresses, names of users with attempts at login, and finally, using a specific account reference, how many failed login attempts.

3. RESEARCH DESIGN

The research design used in this experiment is intended to measure the degree of confidence that students develop in Linux, including the command line interface. Students are also asked to rate their experience, confidence and familiarity with computer security concepts. All of these factors are measured on a 5 point Lickert scale, pre-experiment and post-experiment.

As a general profile of each student, age, sex and the MBTI (Myers Briggs Type Index) have been collected. In addition, each of the following four questions will be answered:

1. Years of experience with computers prior to programming? (type in a number to answer)
2. Years of experience with programming? (type in a number to answer)
3. Number of languages including computer OS that you are familiar and have experience with? (type in a number to answer)
4. Years of experience with Linux? (type in a number to answer)

Following the student profile, both the pre and post assessment will be based on the following nine questions and will use a 5 point Lickert scale where 1 is not at all, 2 slightly, 3 moderately, 4 very, and 5 extremely familiar or experienced. Note that awk, cat, grep, sort, uniq, wc and | are Linux commands needed to carry out the exercise.

1. How familiar are you with Computer Security? (type in a number from 1 to 5 to answer)
2. How experienced are you with Linux Command Line? (type in a number from 1 to 5 to answer)
3. How experienced are you with Linux command awk? (type in a number from 1 to 5 to answer)
4. How experienced are you with Linux command cat? (type in a number from 1 to 5 to answer)
   How experienced are you with Linux command grep? (type in a number from 1 to 5 to answer)
   How experienced are you with Linux command sort? (type in a number from 1 to 5 to answer)
5. How experienced are you with Linux command uniq? (type in a number from 1 to 5 to answer)
6. How experienced are you with Linux command wc? (type in a number from 1 to 5 to answer)
7. How experienced are you with Linux | (piping)? (type in a number from 1 to 5 to answer)

4. CYBERSECURITY ASSIGNMENT

The cybersecurity assignment consists of six questions as contained in the appendix, for which

a Linux script must be written to assess the contents of the login file. The login file is large enough that it is not feasible to scan its contents to derive a precise result without the application of scripts. It's recommended that students carry out each of their responses in an iterative manner such that the required result is found incrementally. The following tables demonstrate the development of a script through successive addition of Linux commands:

(1) What is the user name of the person who uploaded documents to the server?

```
head example.txt
cat example.txt | grep UPLOAD
cat example.txt | grep UPLOAD | awk '{print $8}'
cat example.txt | grep UPLOAD | awk '{print $8}' | uniq
```

(2) How many bytes were downloaded off the server?

```
head example.txt
cat example.txt | grep DOWNLOAD
cat example.txt | grep DOWNLOAD | awk '{print $14}'
cat example.txt | grep DOWNLOAD | awk '{SUM += $14} END { print SUM }'
```

(3) How many unique IP addresses tried to connect to the server?

```
head example.txt
cat example.txt | grep CONNECT
cat example.txt | grep CONNECT | wc -l
cat example.txt | grep -i CONNECT | wc -l
cat example.txt | grep -i CONNECT | awk '{print $10}'
cat example.txt | grep -i CONNECT | awk '{print $10}' | uniq
cat example.txt | grep -i CONNECT | awk '{print $10}' | uniq | wc -l
cat example.txt | grep -i CONNECT | awk '{print $10}' | uniq | sort
cat example.txt | grep -i CONNECT | awk '{print $10}' | uniq | sort | uniq
cat example.txt | grep -i CONNECT | awk '{print $10}' | uniq | sort | uniq | wc -l
cat example.txt | grep -i CONNECT | awk '{print $10}' | sort | uniq | wc -l
```

(4) What is the name of the user who failed to connect to the server the 3rd most times?

```
head example.txt
cat example.txt | grep 'FAIL' | head
cat example.txt | grep 'FAIL LOGIN' | wc -l
cat example.txt | grep -v 'FAIL' | wc -l
cat example.txt | grep 'FAIL LOGIN' | awk '{print $8}'
cat example.txt | grep 'FAIL LOGIN' | awk '{print $8}' | uniq -c
cat example.txt | grep 'FAIL LOGIN' | awk '{print $8}' | uniq -c | sort
cat example.txt | grep 'FAIL LOGIN' | awk '{print $8}' | sort | uniq -c
cat example.txt | grep 'FAIL LOGIN' | awk '{print $8}' | sort | uniq -c | sort
cat example.txt | grep 'FAIL LOGIN' | awk '{print $8}' | sort | uniq -c | sort -n
```

(5) Part A: How many times did someone attempt to log in (but failed!) with the "security" account?

cat example.txt | grep 'FAIL LOGIN' | grep 'prikumye' | wc -l (this returns 23 which is the answer "How many times")

Part B: What IP did they use?

cat example.txt | grep 'FAIL LOGIN' | grep 'prikumye' | awk '{print $8, $12}' | sort | uniq (this returns 4 unique lps which is to answer "What IP did they use")

5. DATA ANALYSIS

A total of 22 attribute values were collected from each subject participating in the experiment and all students registered in the class were represented in the final tabulation. The first four attributes are intended to establish a profile of the participant's background that might have an impact on the performance during the experiment and are: years of computing experience, years of programming, number of languages known, and years working with Linux.

As can be seen in Appendix C, Experimental Data, The mean number of years of computer experience is 10.91 with a standard deviation of 8.790. Experience is likely to be a bit skewed due to the nature of the class which tends to attract both novice and experienced but non degree holding students. Obviously, there is a great deal of variation, and this is typical in CIS606, which was intended to be a leveling experience. The second attribute is years of programming experience and the mean was 2.86 with a standard deviation of 2.897. Again, there is a great deal of variation in the attribute. The

number of programming languages known has a mean of 3.41 with a standard deviation of 2.623. Finally, Years of experience with Linux has a mean of 1.16 with a standard deviation of 2.84. On average, little Linux experience was represented, but with a fair amount of variation.

The remaining 18 attributes collected are evenly divided across a pre-test and post-test to capture the degree of possible change in confidence produce by the experience of log analysis. For each of the pre-test and post-test components, familiarity with security in general and eight Linux commands and command line use was self-assessed. The mean of the pre-test attributes was 1.65 (between not at all familiar and slightly familiar) with a standard deviation of 0.351. The post-test had a mean of 3.08 (a little beyond moderately familiar) with a standard deviation of 0.215.

Based on a simple analysis using a paired t-test with 21 degrees of freedom t=-9.8194 and p-value = 2.6633-09. Clearly students believe they have moderately mastered the material in this class. Considering that this was an experiment that took no longer than a week of class time, the result represents a significant improvement in perceived competence!

6. CONCLUSIONS

The most apparent conclusion of this experiment is that there was a significant increase in perceived mastery of Linux commands in a cyber security domain. This occurred during a period of a little over a week during which the experiment was being carried out. One might question whether the student's perception is accurate, but certainly the confidence in the material is in evidence.

A detailed assessment of the pre and post components indicates that the change in perceived mastery of cyber security topics was quite modest, while those related to specific Linux commands (awk, cat, grep, sort, unique, wc and |) were much more pronounced and statistically significant. Removing the security factor from the set of nine increases the difference between pre and post to 1.55 and 3.12 respectively. The purpose and content of the Linux module in the class was to develop programming skills and allowed the cyber security topic to be the application domain only. It is suspected that with a change in emphasis that includes cyber security topics, the increase in perceived mastery would improve as well.

Topic/Command	Pre-Familiarity Mean	Pre-Familiarity Stdev	Post-Familiarity Mean	Post-Familiarity Stdev
Security	2.45	0.988	2.68	0.924
CLI (cmd line)	1.77	0.950	2.91	0.900
AWK	1.36	0.710	2.95	0.878
CAT	1.82	1.466	3.36	1.068
GREP	1.73	1.286	3.18	0.936
SORT	1.41	0.887	2.95	0.824
UNIQ	1.36	0.771	3.09	0.949
WC	1.41	0.937	3.23	0.901
| (Pipe)	1.55	1.117	3.32	1.017
Mean	1.65		3.08	
Stdev	1.55		3.12	

Table 1 – Pre/Post Assessment Summary

7. POTENTIAL EXTENSIONS

If data were collected over two or more semesters, there might be sufficient data to consider any impact from age, sex and/or the MBTI of the participants. Future experiments might also be applied to other domains relevant to this course, including Java and Python programming. Third, it would be interesting to use an interval metric to determine the amount of mastery achieved.

8. ACKNOWLEDGEMENTS

I wish to acknowledge the contributions provided by two colleagues at CSU Dr. Stephen Hayne, Professor of Computer Information Systems and Deputy Director of the Cyber Security Center jointly supported by the departments of Computer Science and Computer Information Systems. It was Dr. Hayne's suggestion that a log analysis exercise be included in CIS606, IT Infrastructure as part of the Linux component. Spring Semester 2021 was the first semester that this content was provided. In addition, he provided guidance on how the experiment might be structured and analyzed.

Secondly, I wish to credit Gang Yue, Systems Architect Coordinator in the College of Business who served as the Instructional Coordinator for the class. His responsibilities included most day-to-day communication with students and loaded the various components of the exercise into Canvas, our learning management systems, and produced the various tables of data containing the results.

Finally, thanks are due to the students in this class for cooperating with an experiment that

wasn't envisioned or scheduled until midway through the semester. Not only were no complaints made, based on the results it seems that all profited from the exercise.

9. REFERENCES

CryptoKait (2019, August 27). Beginners Guide To Log Analysis For The National Cyber League Games. Retrieved from https://cryptokait.com/2019/08/27/leaping-into-log-analysis/

WebWitch's (2021, March 21) NCL Log Analysis Challenge. Retrieved from https://forms.gle/3jPhgot6rz9F7N9u6

CryptoKait (2021, February 10). Command-line log analysis. Retrieved from https://cryptokait.com/2021/02/10/command-line-log-analysis-for-the-win-1-3-how-to-approach-a-wild-log/

CryptoKait Log Analysis: Leaping. Retrieved from https://cryptokait.com/workshops/national-cyber-league-coaching-guide-v-2-1/ncl-coaching-guide-resources-by-category/log-analysis/log-analysis-leaping/

ColoradoStateUniversity (2021, April 15). About the Cybersecurity Center. Retrieved from https://cybersecurity.colostate.edu/

ColoradoStateUniversity (2021, April 15). Cybersecurity Course Areas and Faculty. Retrieved from https://cybersecurity.colostate.edu/education/

Appendix A
Log Analysis Assignment

For those of you who are not already familiar with them, a "log file", in a computing context, is the automatically produced and time-stamped documentation of events relevant to a particular system. Virtually all software applications and systems produce log files. Out in the wild, log analysis is often under-appreciated, but it becomes very important when you're trying to identify the source of a breach.

Use the commands below to analyze the log file for security breach attempts.

Command	Use	Examples	Common Flags
cat	Outputting text	cat output2.txt (takes text of file 'output2.txt' and prints it to the terminal window)	
|	Piping	cat output2.txt | [2nd command] (takes text of file output2.txt and uses it as an input for the next command)	N/A
grep	Pattern matching	grep -i 'example' Prints all lines with 'example'	-i, -v
awk	Everything	awk '{print $3}' (prints 3rd column of text)	
wc	Word count	cat file.txt | wc -l (Counts lines in file.txt)	-l, -c, -w
sort	Sorts output	cat file.txt | awk '{print $7}' | sort -d (takes the output from awk and sorts it in dictionary order)	-d, -n
uniq	Remove duplicates	cat file.txt | sort | uniq	-c

Answer the following questions:

1. What is the username of the person who uploaded documents to the server?
2. How many bytes were downloaded off the server?
3. How many unique IP addresses tried to connect to the server?
4. What is the name of the user who failed connect to the server the 3rd most times?
5. Part A: How many times did someone attempt to login (but failed!) with the "security" account?
   Part B: What IP did they use?

Appendix B
Sample Log File

Sun Mar 19 03:38:38 2017 [pid 24540] CONNECT: Client "59.188.221.110"
Sun Mar 19 03:38:42 2017 [pid 24539] [anonymous] FAIL LOGIN: Client "59.188.221.110"
Sun Mar 19 03:43:30 2017 [pid 26902] CONNECT: Client "121.206.121.31"
Sun Mar 19 03:43:42 2017 [pid 26901] [anonymous] FAIL LOGIN: Client "121.206.121.31"
Sun Mar 19 03:55:19 2017 [pid 29983] CONNECT: Client "222.223.143.107"
Sun Mar 19 03:55:21 2017 [pid 29982] [anonymous] FAIL LOGIN: Client "222.223.143.107"
Sun Mar 19 03:55:23 2017 [pid 30001] CONNECT: Client "222.223.143.107"
Sun Mar 19 03:55:26 2017 [pid 30000] [budclub] FAIL LOGIN: Client "222.223.143.107"
Sun Mar 19 03:55:27 2017 [pid 30011] CONNECT: Client "222.223.143.107"
Sun Mar 19 03:55:29 2017 [pid 30010] [budclub] FAIL LOGIN: Client "222.223.143.107"
Sun Mar 19 03:55:31 2017 [pid 30027] CONNECT: Client "222.223.143.107"
Sun Mar 19 03:55:33 2017 [pid 30026] [budclub] FAIL LOGIN: Client "222.223.143.107"
Sun Mar 19 03:55:35 2017 [pid 30035] CONNECT: Client "222.223.143.107"
Sun Mar 19 03:55:37 2017 [pid 30034] [budclub] FAIL LOGIN: Client "222.223.143.107"
Sun Mar 19 03:55:39 2017 [pid 30053] CONNECT: Client "222.223.143.107"
Sun Mar 19 03:55:42 2017 [pid 30052] [budclub] FAIL LOGIN: Client "222.223.143.107"
Sun Mar 19 03:55:44 2017 [pid 30061] CONNECT: Client "222.223.143.107"
Sun Mar 19 03:55:47 2017 [pid 30060] [budclub] FAIL LOGIN: Client "222.223.143.107"
Sun Mar 19 03:55:48 2017 [pid 30070] CONNECT: Client "222.223.143.107"
Sun Mar 19 03:55:50 2017 [pid 30069] [budclub] FAIL LOGIN: Client "222.223.143.107"
Sun Mar 19 03:55:52 2017 [pid 30081] CONNECT: Client "222.223.143.107"
Sun Mar 19 03:55:55 2017 [pid 30080] [budclub] FAIL LOGIN: Client "222.223.143.107"
Sun Mar 19 03:55:57 2017 [pid 30090] CONNECT: Client "222.223.143.107"
Sun Mar 19 03:55:59 2017 [pid 30089] [budclub] FAIL LOGIN: Client "222.223.143.107"
Sun Mar 19 03:56:01 2017 [pid 30100] CONNECT: Client "222.223.143.107"
Sun Mar 19 03:56:03 2017 [pid 30099] [budclub] FAIL LOGIN: Client "222.223.143.107"
Sun Mar 19 03:56:05 2017 [pid 30110] CONNECT: Client "222.223.143.107"
Sun Mar 19 03:56:07 2017 [pid 30109] [budclub] FAIL LOGIN: Client "222.223.143.107"

ISSN 2473-4901
v7 n5548

Appendix C
Experimental Data

	PreYearsComp	PreYearsProg	PreNumLang	PreYearsLinux	PreFamSec	PreFamCLI	PreFamAWK	PreFamCAT	PreFamGREP	PreFamSORT	PreFamUNIQ	PreFamWC	PreFamPIPE	PostFamSec	PostFamCLI	PostFamAWK	PostFamCAT	PostFamGREP	PostFamSORT	PostFamUNIQ	PostFamWC	PostFamPIPE
	5	2	2	0	2	1	1	1	1	1	1	1	1	3	3	4	4	4	3	3	3	3
	10	6	3	3	4	3	2	5	4	2	2	3	3	4	4	3	5	4	4	5	4	4
	20	3	2	0	2	2	1	1	1	1	1	1	1	3	3	3	4	3	3	3	4	4
	15	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	7	3	3	0	3	1	1	1	1	1	1	1	1	3	3	4	4	4	4	4	4	4
	15	7	10	4	5	2	2	2	2	2	2	2	2	3	3	4	4	4	4	4	4	4
	7	4	3	0	2	1	1	1	1	1	1	1	1	2	3	3	4	4	4	4	5	4
	10	2	8	1	4	3	3	4	4	4	4	4	4	4	4	4	5	4	4	4	5	4
	0	0	0	0	1	1	1	1	1	1	1	1	1	3	3	3	4	3	3	3	5	4
	16	1	4	0	3	1	1	1	1	1	1	1	1	3	3	3	4	3	2	3	3	3
	2	1	2	0	2	2	1	1	1	1	1	1	1	3	3	3	4	3	3	3	3	3
	30	1	3	1	3	2	1	1	1	1	1	1	1	2	2	2	3	2	2	2	3	3
	1	1	2	0	2	1	1	1	1	1	1	1	1	3	3	3	3	3	3	3	3	3
	1	7	6	0.5	2	2	1	1	1	1	1	1	1	3	3	3	3	3	3	3	3	3
	8	0	1	0	2	1	1	1	1	1	1	1	1	2	3	3	3	3	3	2	3	3
	30	10	7	11	3	4	3	5	5	2	2	1	5	3	4	3	5	5	5	3	3	5
	13	8	7	4	3	4	3	5	4	4	3	4	3	4	5	4	5	5	4	5	5	5
	0	0	0	0	2	1	1	1	1	1	1	1	1	1	1	2	2	2	2	2	2	2
	10	2	3	1	2	2	1	1	1	1	1	1	1	1	2	2	2	2	2	2	2	2
	5	0	1	0	2	1	1	1	1	1	1	1	1	4	2	2	2	2	2	3	3	2
	25	1	2	0	1	1	1	1	1	1	1	1	1	2	3	3	3	3	3	3	3	3
	10	4	5	0	3	2	1	3	3	1	1	1	1	3	3	3	4	3	3	3	3	4
Average	**10.91**	**2.86**	**3.41**	**1.16**	**2.45**	**1.77**	**1.36**	**1.82**	**1.73**	**1.41**	**1.36**	**1.41**	**1.55**	**2.68**	**2.91**	**2.95**	**3.36**	**3.18**	**2.95**	**3.09**	**3.23**	**3.32**
StdDev	8.790	2.897	2.623	2.484	0.988	0.950	0.710	1.466	1.286	0.887	0.771	0.937	1.117	0.924	0.900	0.878	1.068	0.936	0.824	0.949	0.901	1.017
Pre Mean	1.65																					
Pre StdDev	0.351																					
Post Mean	3.08																					
Post StdDev	0.215																					