TNE30019 – Unix for Telecommunications

Learning Summary Report

JOSHUA REDOLFI (101601886)

Introduction

This report summarises what I have learnt in Unix for Telecommunications. It includes a summary of the completed tasks within the Unit Portfolio and a reflection on my learning.

Assessment Summary

Please complete the tables below to summarise completion of individual tasks in this Portfolio

	Pass	Credit	Distinction	High Distinction		
				HD1	HD2	HD3
Lab Report 1	✔					
Lab Report 2		✔				
Lab Report 3			✔			
Jails Report			✔			
Research Report					✔	
Project						
Presentation			✔			

Graded Portfolio Tasks

	Included (please tick)
Pass Tasks (14 tasks)	14
Credit Tasks (3 tasks)	3
Distinction Tasks (5 tasks)	5

Non-Graded Portfolio Tasks

Declaration

I declare that this portfolio is my individual work. I have not copied from any other student's work or from any other source except where due acknowledgment is made explicitly in the text, nor has any part of this submission been written for me by another person.

Signature: 

Reflection

The most important things I learnt:

From this unit I learnt a variety of things, about Unix, my studying habits and also what to expect in the future from what attending the industry speakers' lecturers. First and foremost, I gained a more in-depth understanding the Unix operating system. While I do use Unix (in particular, Fedora and Red Hat) at work, I learnt a lot more services such as BIND and LDAP which my work actually uses. I also gained a greater appreciation for Unix (not necessarily for FreeBSD) by seeing what tasks it is most appropriate for. While not surprising thinking about it now, I learnt why Unix is used everywhere – in part from the discussion about its modularity in the tutorials and also because of the plethora of services that we configured on it.

In terms of learning about studying, I learnt the benefit of keeping one week ahead of submissions in portfolio units, especially when it started to get hard and I fell behind. I hope I will be able to use similar tactics in the workforce to keep ahead of deadlines.

The industry speakers also gave insights into the industry which reinforced what I knew about the growth of the industry. I think the important takeaway from these lectures was how Netflix integrates itself into ISPs to deliver content around the world. The implications of this for other services could prove interesting in the future and I doubt Netflix will end up being the only one to do it.

The things that helped me most were:

In terms of what helped me most this semester, I found the wealth of information on the internet to be the best resource to help me. As I had some experience with Unix beforehand, it was often a matter of finding a guide to work through a lab or service. If that failed, the tutors helped with any query I had.

I found the following topics particularly challenging:

- **.cgi scripting:** it was challenging figuring out which options were required to execute a .cgi script on a web page. I believe this is an area I still need to work on.
- **Samba:** while not assessed, I found the format of the share at first confusing and it seemed no guide was able to help me what I was doing wrong. After being shown the correct implementation by my tutor I finally figured out what I was doing wrong however.
- **PCAP programming:** while the programming task itself was not hard, I found it challenging trying to make the program stop downloading the page once it has ended instead of rolling over and trying to download the page again.

I found the following topics particularly interesting:

- **Apache in general:** Learning how to set up a web server was interesting because it was something I'd never done before and also because I got to see my content actually displayed on the page.

Most of the other topics were also interesting and not too challenging (mostly due to guides on the internet).

I feel I learnt these topics, concepts, and/or tools really well:

- **Apache:** Despite the challenges in learning how execute .cgi scripts, I feel I have a journeyman understanding of how Apache servers can be set up. I completed all the tasks for labs relating to it a week ahead, and also set up an Apache server for my project in a short time.
- **Bash scripting:** while I already knew bash scripting from work, I learnt new features in it. For example, I learnt how to prematurely exit scripts (as seen in my user editing script on in my project) and more advanced uses of control statements.

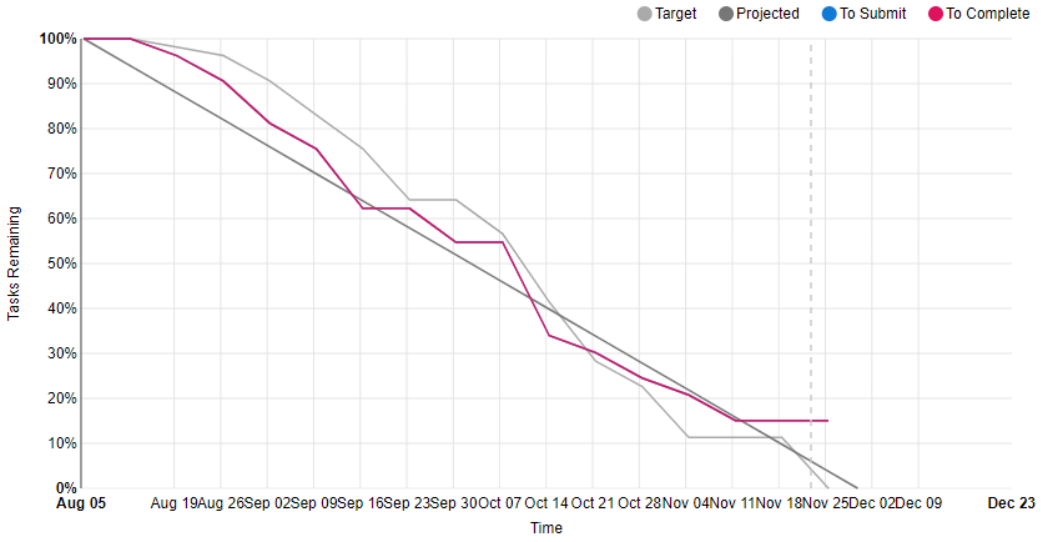
Most other areas I felt a gained a good understanding of the principles underlying them (for example: BIND)

I still need to work on the following areas:

.cgi scripting is the number one area I need to work on that comes to mind. Despite implementing it in my project and through the advanced Apache lab, I need to go over what commands work with .cgi scripting or how to set up the permissions/options in http.conf effectively.

My progress in this unit was …:

- Stayed a week ahead for the first eight weeks or so of the semester, set this as my goal to allow for more time for the project at the end.
- However, towards lab 8 I flew a week behind, which put me just on schedule.
- Due to issues with the last two labs (lab 9 and 10) that I couldn't resolve, I believe I fell behind further, but then the last two labs were removed.
- Thereafter completed all remaining tasks on time, but the final lab 9 section was left incomplete.



The graph shows that I stayed on top of required tasks (especially early on in the semester), with my progress only falling behind as exams started. This was because of the remaining lab 9 I hadn't completed and also because I had yet to submit/receive feedback on the project.

This unit will help me in the future:

As I mentioned earlier, by doing this unit I gained a greater understanding of how Unix operates. While I may not work in software engineering in the future like I do now, I believe that understanding Unix will benefit me wherever I go in the telecommunications field. For example, microcontrollers such as the Raspberry Pi use a Unix-based operating system, which I could be using even if I choose an electrical engineering path.

As mentioned earlier I feel I gained an understanding of the underlying principles of most of the services and tools used in the unit. This will help me in the future to understand if a service is working correctly and what could potentially be the issue. For example, with LDAP, I now have an understanding why it experiences a slow start.

In the research report, the topic of machine learning gave me a good overview of the topic and allowed to understand how machine learning works. As this field is growing, I believe having knowledge in this field will be put me at an advantage in the workplace.

If I did this unit again I would do the following things differently:

- Firstly, I would keep with my plan to stay at least a week ahead of the unit schedule (in essence: submit each lab in the lab a week before it was due). This worked well for most of the semester to allow me to focus on other units or tasks that needed attention.

- I would also spend more time on the project, both during the semester and after exams. I did not have time to research some of the ideas I had in mind for the extension HD tasks. I was planning adding LDAPS and also a webpage that tracked queries to the LDAP server and did not get time to figure out how to best do this.

Other…:

While I do not believe I achieved my full potential (due to other units and work) I feel I deserve a HD because I made sure to keep ahead in this unit throughout the majority of the semester. My Doubtfire progress graph shows this to be the case. Moreover, I was proactive and had an early start on the project, and when it was reset, I got back to where I was in three days while also writing the documentation in LaTeX. I believe I also demonstrated enthusiasm for the content we learnt and tried my best to engage in lectures and tutorials.

Swinburne University of Technology

Unix for Telecommunications

Joshua Redolfi

Portfolio Submission

Submitted By:
Joshua Redolfi
101601886

Tutor:
Farinaz Jowkarishasaltaneh

November 24, 2019

Contents

2 Overall Task Status

Task	Status	Times Assessed
Communications - Research Report	Complete	1
Practical - Lab 1: Introduction to Unix	Complete	1
Theory - Plagiarism Test (P)	Complete	1
Practical - Lab 2: LaTeX (D)	Complete	1
Practical - Lab 2: LaTeX (P)	Complete	1
Theory - Operating Systems (P)	Complete	1
Communications - Lab Report 1	Complete	1
Communications - Speaker Reflection 1	Complete	1
Practical - Lab 3: Bind (P)	Complete	1
Practical - Lab 4: Basic Apache (C)	Complete	1
Practical - Lab 4: Basic Apache (P)	Complete	1
Theory - Test 1 (P)	Complete	1
Practical - Lab 5: Advanced Apache (D)	Complete	1
Practical - Lab 5: Advanced Apache (P)	Complete	1
Theory - Jails	Complete	1
Practical - Project	Ready to Mark	0
Communications - Lab Report 2	Complete	1
Practical - Lab 6: PCAP Programming (D)	Complete	1
Practical - Lab 6: PCAP Programming (P)	Complete	1
Practical - Lab 7: NMap (C)	Complete	1
Practical - Lab 7: NMap (P)	Complete	1
Communications - Lab Report 3	Complete	1
Practical - Lab 8: TCPDump (C)	Complete	1
Practical - Lab 8: TCPDump (D)	Complete	1
Practical - Lab 8: TCPDump (P)	Complete	1
Practical - Programming Tutorial	Complete	1
Communications - Project Presentation	Complete	1
Practical - Lab 9: Samba (D)	Not Started	
Practical - Lab 9: Samba (P)	Complete	1
Communications - Speaker Reflection 2	Complete	1
Theory - Test 2 (P)	Complete	1

3 Learning Outcomes

3.1 ULO1

Use navigation processes in an installed Unix System

Task	Rating	Status	Times Assessed
Practical - Lab 4: Basic Apache (C)	♦◊◊◊	Complete	1
Practical - Lab 4: Basic Apache (P)	♦◊◊◊	Complete	1
Practical - Lab 5: Advanced Apache (D)	♦♦♦◊◊	Complete	1
Practical - Lab 5: Advanced Apache (P)	♦◊◊◊	Complete	1
Practical - Project	♦♦♦♦◊	Ready to Mark	0
Practical - Lab 6: PCAP Programming (D)	♦♦◊◊◊	Complete	1
Practical - Lab 6: PCAP Programming (P)	♦♦◊◊◊	Complete	1
Practical - Programming Tutorial	♦◊◊◊	Complete	1

3.2 ULO2

Appreciate the operation of a Unix-based Operating System

Task	Rating	Status	Times Assessed
Communications - Research Report	♦♦◊◊◊	Complete	1
Practical - Lab 1: Introduction to Unix	♦♦♦◊◊	Complete	1
Theory - Operating Systems (P)	♦♦♦◊◊	Complete	1
Communications - Speaker Reflection 1	♦♦◊◊◊	Complete	1
Practical - Lab 3: Bind (P)	♦◊◊◊◊	Complete	1
Theory - Test 1 (P)	♦♦♦◊◊	Complete	1
Practical - Project	♦♦♦◊◊	Ready to Mark	0
Practical - Lab 6: PCAP Programming (D)	♦♦♦◊◊	Complete	1
Practical - Lab 6: PCAP Programming (P)	♦♦♦◊◊	Complete	1
Practical - Lab 8: TCPDump (C)	♦◊◊◊◊	Complete	1
Practical - Lab 8: TCPDump (D)	♦◊◊◊◊	Complete	1
Practical - Lab 8: TCPDump (P)	♦◊◊◊◊	Complete	1
Practical - Programming Tutorial	♦♦♦◊◊	Complete	1
Communications - Speaker Reflection 2	♦♦◊◊◊	Complete	1
Theory - Test 2 (P)	♦♦♦◊◊	Complete	1

3.3 ULO3

Conduct the administration of a Unix server or workstation

Task	Rating	Status	Times Assessed
Communications - Research Report	♦◊◊◊◊	Complete	1
Practical - Lab 4: Basic Apache (C)	♦♦◊◊◊	Complete	1
Practical - Lab 4: Basic Apache (P)	♦♦◊◊◊	Complete	1
Practical - Project	♦♦♦♦♦	Ready to Mark	0
Communications - Project Presentation	♦♦◊◊◊	Complete	1

3.4 ULO4

Configure common network services, devices and security

Task	Rating	Status	Times Assessed
Practical - Lab 3: Bind (P)	♦♦♦◊◊	Complete	1
Practical - Lab 4: Basic Apache (C)	♦♦♦♦◊	Complete	1
Practical - Lab 4: Basic Apache (P)	♦♦♦♦◊	Complete	1
Practical - Lab 5: Advanced Apache (D)	♦♦♦♦◊	Complete	1
Practical - Lab 5: Advanced Apache (P)	♦♦♦♦◊	Complete	1
Practical - Project	♦♦♦♦♦	Ready to Mark	0
Practical - Lab 7: NMap (C)	♦♦◊◊◊	Complete	1
Practical - Lab 7: NMap (P)	♦♦◊◊◊	Complete	1
Practical - Programming Tutorial	♦♦◊◊◊	Complete	1

3.5 ULO5

Demonstrate the use of administration tools on Unix systems

Task	Rating	Status	Times Assessed
Practical - Lab 2: LaTeX (D)	♦♦♦◊◊	Complete	1
Practical - Lab 2: LaTeX (P)	♦♦♦◊◊	Complete	1
Practical - Lab 3: Bind (P)	♦♦♦◊◊	Complete	1
Practical - Project	♦♦♦♦♦	Ready to Mark	0
Practical - Lab 6: PCAP Programming (D)	♦♦♦◊◊	Complete	1
Practical - Lab 6: PCAP Programming (P)	♦♦♦◊◊	Complete	1
Practical - Lab 7: NMap (C)	♦♦♦◊◊	Complete	1
Practical - Lab 7: NMap (P)	♦♦♦◊◊	Complete	1
Practical - Lab 8: TCPDump (C)	♦♦◊◊◊	Complete	1
Practical - Lab 8: TCPDump (D)	♦♦◊◊◊	Complete	1
Practical - Lab 8: TCPDump (P)	♦♦◊◊◊	Complete	1
Practical - Lab 9: Samba (P)	♦♦♦◊◊	Complete	1

3.6 ULO6

Design and construct unfamiliar services

Task	Rating	Status	Times Assessed
Practical - Lab 3: Bind (P)	♦◊◊◊◊	Complete	1
Practical - Lab 5: Advanced Apache (D)	♦♦◊◊◊	Complete	1
Practical - Lab 5: Advanced Apache (P)	♦◊◊◊◊	Complete	1
Practical - Project	♦♦♦♦♦	Ready to Mark	0
Communications - Project Presentation	♦◊◊◊◊	Complete	1

3.7 ULO7

Generate documentation for laboratory work and a research assignment

Task	Rating	Status	Times Assessed
Communications - Research Report	♦♦♦♦♦	Complete	1
Theory - Plagiarism Test (P)	♦♦◊◊◊	Complete	1
Practical - Lab 2: LaTeX (D)	♦♦◊◊◊	Complete	1
Practical - Lab 2: LaTeX (P)	♦♦◊◊◊	Complete	1
Communications - Lab Report 1	♦♦♦♦◊	Complete	1
Communications - Speaker Reflection 1	♦♦♦◊◊	Complete	1
Theory - Test 1 (P)	♦♦♦◊◊	Complete	1
Communications - Lab Report 2	♦♦♦♦◊	Complete	1
Theory - Jails	♦♦♦◊◊	Complete	1
Practical - Project	♦♦♦♦♦	Ready to Mark	0
Communications - Lab Report 3	♦♦♦♦◊	Complete	1
Communications - Project Presentation	♦♦♦◊◊	Complete	1
Communications - Speaker Reflection 2	♦♦♦◊◊	Complete	1
Theory - Test 2 (P)	♦♦◊◊◊	Complete	1

4 Practical - Lab 2: LaTeX (D)

Lab task - LaTeX (Distinction)

Outcome	Weight
ULO7	♦♦◊◊◊

Outcome	Weight
ULO5	♦♦♦◊◊

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Practical - Lab 2: LaTeX (D)

Submitted By:
Joshua Redolfi
101601886
2019/08/20 15:21

Tutor:
Farinaz Jowkarishasaltaneh

August 20, 2019

Unix and LaTeX: Showcase

Joshua Redolfi
101601886@student.swin.edu.au
Swinburne University of Technology
Melbourne, Victoria

ABSTRACT

This document showcases several abilities of the IEEE format and LaTEX.

KEYWORDS

latex, unix, ieee

1 FORMATTING SHOWCASE

This section how text in the ACM LaTEX style. The following features will be used to showcase this:

- Subsections
- Lists
  - Bullet lists
  - Itemized lists
- Tables
- Images

The first five numbers in Roman numerals are:

(1) I
(2) II
(3) III
(4) IV
(5) V

1.1 Tables and Images

This subsection showcases the power in LaTeX for providing tables. By using this over Microsoft Word, the creator can tell LaTeX exactly how to create the document, which will automatically have a professional looking appearance [1]. For example, look at Table 1.

Table 1: An example of a table, showcasing the first letters of selected alphabets

English	Greek	Icelandic
ABC	ΑΒΓ	AÁB

Here is an image from my recent Europe trip, this shows one of the two lakes in Interlaken (Figure 1).

Figure 1: Lake Brienz in Interlaken, Switzerland.

REFERENCES

[1] Victoria Baramidze. 2014. Latex for technical writing. *Journal of Technical Science and Technologies* 2, 2 (2014), 45–48.

5 Practical - Lab 2: LaTeX (P)

Lab task - LaTeX

Outcome	Weight
ULO7	♦♦◊◊◊

Outcome	Weight
ULO5	♦♦♦◊◊

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Practical - Lab 2: LaTeX (P)

Submitted By:
Joshua Redolfi
101601886
2019/08/20 14:47

Tutor:
Farinaz Jowkarishasaltaneh

August 20, 2019

Unix and LaTeX: Showcase

1st Joshua Redolfi
Centre for Advanced Internet Architectures
Swinburne University of Technology
Melbourne, Australia
101601886@student.swin.edu.au

***Abstract*—This document showcases several abilities of the IEEE format and LaTEX.**
***Index Terms*—latex, unix, ieee**

I. Formatting Showcase

This section shows how text appears in the IEEE LaTEX style. The following features will be used to showcase this:

- Subsections
- Lists
  - Bullet lists
  - Itemized lists
- Tables
- Images

A. Tables and Images

This subsection showcases the power in LaTeX for providing tables. By using this over Microsoft Word, the creator can tell LaTeX exactly how to create the document, which will automatically have a professional looking appearance [1].

TABLE I
An example of a table, showcasing the first letters of selected alphabets

English	Greek	Icelandic
ABC	ΑΒΓ	A$\acute{A}$B

Here is an image from my recent Europe trip, this shows one of the two lakes in Interlaken.

Fig. 1. Lake Brienz in Interlaken, Switzerland.

References

[1] V. Baramidze, "Latex for technical writing," *Journal of Technical Science and Technologies*, vol. 2, no. 2, pp. 45–48, 2014.

6 Communications - Lab Report 1

Required Lab Report 1

Outcome	Weight
ULO7	♦♦♦♦◊

Date	Author	Comment
2019/09/08 18:22	Farinaz Jowkarishasa...	Great quality report. Your report presentation is consistent and easy to follow, great work. The followings are some suggestions to improve your future reports: - The section about adding zones needs more details. For instance, the zone names, their type (master / slave). Perhaps show the text you modified / added in the config file. - The DNS database files section is a bit short. You can expand this section by adding some figures and explaining / refering to the important lines.

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Communications - Lab Report 1

Submitted By:
Joshua Redolfi
101601886
2019/09/01 12:08

Tutor:
Farinaz Jowkarishasaltaneh

September 1, 2019

Using a BSD RULE Host as a DNS Server

Joshua Redolfi
Faculty of Science,
Engineering and, Technology
Swinburne University of Technology
Email: 101601886@student.swin.edu.au

***Abstract*—This report discusses configuring a RULE host to provide DNS information to a university provided marking script. The configuration provided in the lab manual provided correct information to the marking script and thus the experiment was successful. Skills were obtained in setting up a Bekerley Internet Name Domain Server.**

I. Aim

The aim of this experiment is to gain skills in setting up a DNS server by configuring the university provided rule host to provide DNS information to a marking script provided by the unit convener. Successful completion is providing a configuration that will allow for the script to obtain DNS data from the RULE host.

II. Equipment

This lab was completed on RULE host number 58, to connect to this host PuTTY (release 0.7 for 64 bit Windows) was used. This configuration was done on a computer located at the author's home which ran 64 bit Windows 10. Thus, the computer was not within the Swinburne private network Cisco AnyConnect™was used to connect to the RULE host. The testing of the configuration was achieved through a marking script located at the following address: ruleprimary1.caia.swin.edu.au (requires access to the Swinburne private network to view). On the RULE host itself, FreeBSD was used as the operating system and BIND (Bekerley Internet Name Domain) was used to provide a DNS server.

III. Method

As per the lab handout.

IV. Results

It is important to note that the results shown in here also show configuration for a later lab regarding combining Apache and BIND.

By configuring the lab as requested by the lab handout, the following results were produced. Testing the configuration against the marking script provided by the university shows a successful output. The list below displays the results of the configuration.

- The DNS server of `labbind.unix` is `136.186.230.58`
- The DNS server of `labnmap.unix` is `136.186.230.58`
- The DNS server of `caia.swin.edu.au` is `136.186.230.58`
- The DNS server of `136.186.230.58` is `ns1.unix`

Figure 1 shows the final DNS lookup performed above. The entry of `ns1.unix` shows a successful configuration.

```
student@rule58:~ % nslookup 136.186.230.58
Server:         136.186.230.58
Address:        136.186.230.58#53

58.230.186.136.in-addr.arpa     name = aristocrats.unix.
58.230.186.136.in-addr.arpa     name = ns1.unix.
58.230.186.136.in-addr.arpa     name = armitagechemicals.unix.
```

Fig. 1. The nslookup for the DNS IP.

Moreover, the full results are provided in the pdf generated by the marking script for RULE host 58. An example of it is shown below (the test was modified to fit the page):

```
Querying for (labbind.unix)
using DNS server (136.186.230.58)
- Correct answer (136.186.230.58)

Using domain server:
Name: 136.186.230.58
Address: 136.186.230.58#53
Aliases:

labbind.unix has address 136.186.230.58
Test Result: PASSED
```

By using `sockstat -4`, the PID of the BIND server was found to be 23184. Moreover, the Swinburne University domain name server was found to be `136.186.20.9`. The results of `sockstat -4` after the configuration is shown in Figure 2 and shows many BIND processes running. This shows that the BIND server is now operational.

```
student@rule58:~ % sockstat -4
USER     COMMAND    PID   FD PROTO  LOCAL ADDRESS         FOREIGN ADDRESS
student  sshd       46035 3  tcp4   136.186.230.58:22     136.186.247.128:55214
root     sshd       46032 3  tcp4   136.186.230.58:22     136.186.247.128:55214
www      httpd      86701 3  tcp4   136.186.230.58:80     *:*
www      httpd      86694 3  tcp4   136.186.230.58:80     *:*
www      httpd      86693 3  tcp4   136.186.230.58:80     *:*
www      httpd      86692 3  tcp4   136.186.230.58:80     *:*
www      httpd      86691 3  tcp4   136.186.230.58:80     *:*
www      httpd      86690 3  tcp4   136.186.230.58:80     *:*
root     httpd      86689 3  tcp4   136.186.230.58:80     *:*
bind     named      48811 20 tcp4   136.186.230.58:53     *:*
bind     named      48811 21 tcp4   136.186.230.58:953    *:*
bind     named      48811 512 udp4  136.186.230.58:53     *:*
bind     named      48811 513 udp4  136.186.230.58:53     *:*
bind     named      48811 514 udp4  136.186.230.58:53     *:*
bind     named      48811 515 udp4  136.186.230.58:53     *:*
bind     named      48811 516 udp4  136.186.230.58:53     *:*
bind     named      48811 517 udp4  136.186.230.58:53     *:*
bind     named      48811 518 udp4  136.186.230.58:53     *:*
bind     named      48811 519 udp4  136.186.230.58:53     *:*
bind     named      48811 520 udp4  136.186.230.58:53     *:*
bind     named      48811 521 udp4  136.186.230.58:53     *:*
bind     named      48811 522 udp4  136.186.230.58:53     *:*
bind     named      48811 523 udp4  136.186.230.58:53     *:*
bind     named      48811 524 udp4  136.186.230.58:53     *:*
bind     named      48811 525 udp4  136.186.230.58:53     *:*
bind     named      48811 526 udp4  136.186.230.58:53     *:*
bind     named      48811 527 udp4  136.186.230.58:53     *:*
bind     named      48811 528 udp4  136.186.230.58:53     *:*
bind     named      48811 529 udp4  136.186.230.58:53     *:*
bind     named      48811 530 udp4  136.186.230.58:53     *:*
bind     named      48811 531 udp4  136.186.230.58:53     *:*
bind     named      48811 532 udp4  136.186.230.58:53     *:*
bind     named      48811 533 udp4  136.186.230.58:53     *:*
bind     named      48811 534 udp4  136.186.230.58:53     *:*
root     sshd       85567 3  tcp4   136.186.230.58:22     *:*
```

Fig. 2. sockstat after configuration

V. Discussion

A. Preliminary Investigation

Three commands were used to initially verify the current of the RULE host, they are shown below and provide information about the DNS settings on the host.

`nslookup` is used to determine the name server of a given web address. For example: `nslookup www.swin.edu.au` gives the Swinburne name server at `136.186.20.9`. Before configuration, using `nslookup labbind.unix` returned no name server.

`host` shows the IP address and DNS information for the specified web address. This shows alternative web address and domain names as well.

`dig` retrieves detailed name server information, such as displaying alternative web addresses, along with the authoritative domain name servers.

Additionally, by using the command `ps -aux` the PID of the BIND server was discovered as mentioned in Results. By using `grep` the BIND server PID can be explicitly searched for.

B. Configuring the DNS Client

To configure the RULE host to use the right servers the file `resolv.conf` must be edited (located in `etc`). This files shows the domain name servers that the host can use. According to the manual, it uses them in order of listing in the file itself. Thus to make it use the BIND server created on the RULE host, it had to be edited as seen in Figure 3. The figure shows that the Swinburne DNS (`136.186.20.9`) has been replaced with the IP address of the RULE host (`136.186.230.58`). If this is not done, any command to verify the domain names hosted on the BIND server will return an NXDOMAIN error.

```
search caia.swin.edu.au
#nameserver 136.186.20.9
#nameserver 136.186.1.111
nameserver 136.186.230.58
```

Fig. 3. The configured resolv.conf file

C. Configuring the DNS Server

To make the RULE host function appropriately as a DNS server, the first step undertaken was to edit the file `named.conf`. It was first backed up in case the author accidentally created an unfixable error and also it is good practice.

The first edit undertaken was the removal of the listener line, which was previously used to make BIND only listen to internal requests. Thereafter the forwarder was added, this allows for the BIND server to forward a request to another server (for the purposes of this lab it was set to the default Swinburne DNS server) should it not have the entry in its database.

Next, zones were added to the file, these are the domains assigned to the BIND host. Some zones are configured to use the `empty.db` file for router benchmarking and testing.

D. Creating the DNS Database Files

These database files include mappings of IP address to domain name (and vice versa) as listed under Machine Names. It also includes time settings, such as the amount of seconds before refreshing an entry. The file `unix` is used to map the domain name to an IP address, whereas `230.186.136.in-addr.arpa` is used for the opposite. This is why both files were needed to allow for an nslookup of both the IP address and domain name.

E. Manually and Automatically Starting BIND

The output of `sockstat -4` was shown back in 2 back in the Results section. Prior to this, it had shown no bind entries meaning that the BIND was yet to operate. By using `nslookup`, all the entries configured now correctly pointed to the DNS server `136.186.230.58` which is the RULE host (the marking script also produced the same result). By editing `rc.conf` and adding the lines as required at the end of the file, BIND was successfully able to be started, stopped and restarted whenever needed. Not mentioned in the handout is that the command `/usr/local/etc/rc.d/named restart` will perform both a stopping and starting operation in one command.

VI. Conclusion

In conclusion, the RULE hosts provided by the university have been verified as able to be configured as a DNS server using BIND software.

Overall, from this lab the author learnt skills in both configuring a server to operate on Unix and an insight into how BIND (and other name servers) can work. By using a swath of new commands, further understanding of the terminal commands of Unix/FreeBSD was obtained. Overall, this will

allow for a sense of familiarity should future tasks similar to this be encountered in industry (for example, in network engineering).

ACKNOWLEDGMENT

The author would like to thank Farinaz Teneh for tutoring in the laboratory and Jonathan Kua for providing insight into the operation of FreeBSD.

7 Communications - Speaker Reflection 1

Reflection of presentation of first Industry Invited Speaker

Outcome	Weight
ULO2	♦♦◊◊◊

Outcome	Weight
ULO7	♦♦♦◊◊

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Communications - Speaker Reflection 1

Submitted By:
Joshua Redolfi
101601886
2019/09/04 12:14

Tutor:
Farinaz Jowkarishasaltaneh

September 4, 2019

Unix for Telecommunications

Portfolio Task – C-Speaker-Reflection-Form

Name: Joshua Rodolfi Date: 3/9/19

101601886

1

1) Reflect on one thing you learnt today that you were completely surprised by

The structure of the internet. Hearing that the modern internet and its structure (with large CDNs and large T2 ISPs), is not as layered as Cisco has taught it to be. This confirmed what I suspected, and I learnt just how big CDNs are which suprised me.

2) How do you think todays presentation changed the way you look at your career following graduation?

I liked hearing about how his research and hobbies gave him his job at Netflix. It is motivation to continue doing extra ciricular activities. As for a direction, I don't think it has changed, but it is a potential option to work with CDNs.

8 Practical - Lab 3: Bind (P)

Lab task - Bind

Outcome	Weight
ULO5	♦♦♦◊◊

Outcome	Weight
ULO6	♦◊◊◊◊

Outcome	Weight
ULO2	♦◊◊◊◊

Outcome	Weight
ULO4	♦♦♦◊◊

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Practical - Lab 3: Bind (P)

Submitted By:
Joshua Redolfi
101601886
2019/08/20 15:08

Tutor:
Farinaz Jowkarishasaltaneh

August 20, 2019

```
Marking P-Lab-03-Bind-P on RULE host rule58
TNE30019/TNE80014 - Portfolio Task - P-Lab03 - BIND - Pass
====================================
********************************************************************************
 Portfolio Task: P-Lab-03-Bind, Pass Task

 Configure your rule host (rule58) with a forwarding DNS server

 Tasks:
  Query for: labbind.unix        ; Correct answer: 136.186.230.58
  Query for: labnmap.unix        ; Correct answer: 136.186.230.21
  Query for: caia.swin.edu.au    ; Correct answer: 136.186.229.16
  Query for: 136.186.230.58      ; Correct answer: ns1.unix
********************************************************************************

Date: 20190820_1454
Last 5 Logons:
student    pts/6    10.1.16.119            Tue Aug 20 14:41   still logged in
student    pts/1    10.1.16.119            Mon Aug 19 16:43 - 17:08  (00:25)
student    pts/0    vpn247-159.cc.swin.edu Sat Aug 17 14:36 - 16:25  (01:48)
student    pts/0    10.1.13.187            Fri Aug 16 08:54 - 08:56  (00:01)
student    pts/0    10.1.13.187            Fri Aug 16 08:47 - 08:54  (00:06)

RULE host Socket information
tempuser sshd       55326 3  tcp4   136.186.230.58:22     10.1.16.119:58773
tempuser sshd       55326 7  tcp4   136.186.230.58:6010   *:*
root     sshd       55321 3  tcp4   136.186.230.58:22     10.1.16.119:58773
bind     named      23184 20 tcp4   136.186.230.58:53     *:*
bind     named      23184 21 tcp4   136.186.230.58:953    *:*
bind     named      23184 512 udp4  136.186.230.58:53     *:*
bind     named      23184 513 udp4  136.186.230.58:53     *:*
bind     named      23184 514 udp4  136.186.230.58:53     *:*
bind     named      23184 515 udp4  136.186.230.58:53     *:*
bind     named      23184 516 udp4  136.186.230.58:53     *:*
bind     named      23184 517 udp4  136.186.230.58:53     *:*
bind     named      23184 518 udp4  136.186.230.58:53     *:*
bind     named      23184 519 udp4  136.186.230.58:53     *:*
bind     named      23184 520 udp4  136.186.230.58:53     *:*
bind     named      23184 521 udp4  136.186.230.58:53     *:*
bind     named      23184 522 udp4  136.186.230.58:53     *:*
bind     named      23184 523 udp4  136.186.230.58:53     *:*
bind     named      23184 524 udp4  136.186.230.58:53     *:*
bind     named      23184 525 udp4  136.186.230.58:53     *:*
bind     named      23184 526 udp4  136.186.230.58:53     *:*
bind     named      23184 527 udp4  136.186.230.58:53     *:*
bind     named      23184 528 udp4  136.186.230.58:53     *:*
bind     named      23184 529 udp4  136.186.230.58:53     *:*
bind     named      23184 530 udp4  136.186.230.58:53     *:*
bind     named      23184 531 udp4  136.186.230.58:53     *:*
bind     named      23184 532 udp4  136.186.230.58:53     *:*
bind     named      23184 533 udp4  136.186.230.58:53     *:*
bind     named      23184 534 udp4  136.186.230.58:53     *:*
root     sshd       85567 3  tcp4   136.186.230.58:22     *:*
root     ntpd       1485  55 udp4   136.186.230.58:123    *:*

Performing DNS lookups on 136.186.230.58
========================================

Querying for (labbind.unix) using DNS server (136.186.230.58) - Correct answer (136.186.230.58)
Using domain server:
```

```
Name: 136.186.230.58
Address: 136.186.230.58#53
Aliases:

labbind.unix has address 136.186.230.58
Test Result: PASSED


Querying for (labnmap.unix) using DNS server (136.186.230.58) - Correct answer (136.186.230.21)
Using domain server:
Name: 136.186.230.58
Address: 136.186.230.58#53
Aliases:

labnmap.unix has address 136.186.230.21
Test Result: PASSED


Querying for (caia.swin.edu.au) using DNS server (136.186.230.58) - Correct answer (136.186.229.16)
Using domain server:
Name: 136.186.230.58
Address: 136.186.230.58#53
Aliases:

caia.swin.edu.au has address 136.186.229.16
Test Result: PASSED


Querying for (136.186.230.58) using DNS server (136.186.230.58) - Correct answer (ns1.unix.)
Using domain server:
Name: 136.186.230.58
Address: 136.186.230.58#53
Aliases:

58.230.186.136.in-addr.arpa domain name pointer ns1.unix.
Test Result: PASSED


********************************************************************************
PORTFOLIO TASK RESULT: PASSED
```

9 Practical - Lab 4: Basic Apache (C)

Lab task - Basic Apache (Credit)

Outcome	Weight
ULO4	♦♦♦♦◊

Outcome	Weight
ULO3	♦♦◊◊◊

Outcome	Weight
ULO1	♦◊◊◊◊

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Practical - Lab 4: Basic Apache (C)

Submitted By:
Joshua Redolfi
101601886
2019/08/27 14:41

Tutor:
Farinaz Jowkarishasaltaneh

August 27, 2019

```
Marking P-Lab-04-Basic-Apache-C on RULE host rule58
TNE30019/TNE80014 - Laboratory - Basic Apache
==============================================
*******************************************************************************
 Portfolio Task: P-Lab-04-Basic-Apache, Credit Task

 Configure your rule host (rule58) with Apache to serve a single web site

 Tasks:
  Apache running on rule58
  Browsing to http://rule58/private will fail with error code 401
  Browsing to http://rule58/private with username(ruleperson)
    password(ruleperson) will return a valid web page
*******************************************************************************

Date: 20190827_1439
Last 5 Logons:
student    pts/0    vpn247-179.cc.swin.edu Sun Aug 25 12:47 - 15:19  (02:32)
student    pts/1    10.1.16.119            Wed Aug 21 13:00 - 14:29  (01:29)
student    pts/6    10.1.16.119            Tue Aug 20 14:41 - 16:28  (01:47)
student    pts/1    10.1.16.119            Mon Aug 19 16:43 - 17:08  (00:25)
student    pts/0    vpn247-159.cc.swin.edu Sat Aug 17 14:36 - 16:25  (01:48)

RULE host Socket information
www      httpd      77938 3  tcp4   136.186.230.58:80     *:*
www      httpd      77937 3  tcp4   136.186.230.58:80     *:*
www      httpd      77936 3  tcp4   136.186.230.58:80     *:*
www      httpd      77935 3  tcp4   136.186.230.58:80     *:*
www      httpd      77934 3  tcp4   136.186.230.58:80     *:*
www      httpd      77933 3  tcp4   136.186.230.58:80     *:*
www      httpd      77932 3  tcp4   136.186.230.58:80     *:*
www      httpd      77931 3  tcp4   136.186.230.58:80     *:*
www      httpd      77930 3  tcp4   136.186.230.58:80     *:*
root     httpd      77929 3  tcp4   136.186.230.58:80     *:*
bind     named      23184 20 tcp4   136.186.230.58:53     *:*
bind     named      23184 21 tcp4   136.186.230.58:953    *:*
bind     named      23184 512 udp4  136.186.230.58:53     *:*
bind     named      23184 513 udp4  136.186.230.58:53     *:*
bind     named      23184 514 udp4  136.186.230.58:53     *:*
bind     named      23184 515 udp4  136.186.230.58:53     *:*
bind     named      23184 516 udp4  136.186.230.58:53     *:*
bind     named      23184 517 udp4  136.186.230.58:53     *:*
bind     named      23184 518 udp4  136.186.230.58:53     *:*
bind     named      23184 519 udp4  136.186.230.58:53     *:*
bind     named      23184 520 udp4  136.186.230.58:53     *:*
bind     named      23184 521 udp4  136.186.230.58:53     *:*
bind     named      23184 522 udp4  136.186.230.58:53     *:*
bind     named      23184 523 udp4  136.186.230.58:53     *:*
bind     named      23184 524 udp4  136.186.230.58:53     *:*
bind     named      23184 525 udp4  136.186.230.58:53     *:*
bind     named      23184 526 udp4  136.186.230.58:53     *:*
bind     named      23184 527 udp4  136.186.230.58:53     *:*
bind     named      23184 528 udp4  136.186.230.58:53     *:*
bind     named      23184 529 udp4  136.186.230.58:53     *:*
bind     named      23184 530 udp4  136.186.230.58:53     *:*
bind     named      23184 531 udp4  136.186.230.58:53     *:*
bind     named      23184 532 udp4  136.186.230.58:53     *:*
bind     named      23184 533 udp4  136.186.230.58:53     *:*
bind     named      23184 534 udp4  136.186.230.58:53     *:*
root     sshd       85567 3  tcp4   136.186.230.58:22     *:*
```

```
root     ntpd       1485  55 udp4   136.186.230.58:123    *:*
?        ?          ?     ?  tcp4   136.186.230.58:57881  136.186.230.58:80
?        ?          ?     ?  tcp4   136.186.230.58:57882  136.186.230.58:80
?        ?          ?     ?  tcp4   136.186.230.58:57883  136.186.230.58:80

HTTP QUERY http://rule58/private
====================================
--2019-08-27 14:39:41--  http://rule58/private
Resolving rule58 (rule58)... 136.186.230.58
Connecting to rule58 (rule58)|136.186.230.58|:80... connected.
HTTP request sent, awaiting response... 401 Unauthorized

Username/Password Authentication Failed.


Test Result: PASSED


HTTP QUERY http://rule58/private
 - Authentication parameters (--user=ruleperson --password=ruleperson)
====================================
--2019-08-27 14:39:41--  http://rule58/private
Resolving rule58 (rule58)... 136.186.230.58
Connecting to rule58 (rule58)|136.186.230.58|:80... connected.
HTTP request sent, awaiting response... 401 Unauthorized
Authentication selected: Basic realm="ANGRY"
Reusing existing connection to rule58:80.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: http://rule58/private/ [following]
--2019-08-27 14:39:41--  http://rule58/private/
Reusing existing connection to rule58:80.
HTTP request sent, awaiting response... 200 OK
Length: 626 [text/html]
Saving to: ’/tmp/wget_save.LLDpIjOZ’

     0K                                                       100% 74.6M=0s

2019-08-27 14:39:41 (74.6 MB/s) - ’/tmp/wget_save.LLDpIjOZ’ saved [626/626]


====================================
Downloaded Page contents
====================================
<!DOCTYPE html>
<html>
<body bgcolor="orange">
<title>Joshua’s Gallery of Angry Reacts</title>
<body>
<h1 style="color:blue; border:2px solid Tomato;">Joshua’s Gallery of Angry Reacts</h1>
<hr>
<h2>BEHOLD</h2>
<img src="1.jpg" alt="lol" width="256" height="256">
<img src="3.png" alt="lol" width="256" height="256">
<img src="4.jpg" alt="lol" width="256" height="256">

<h2>Even more!</h2>
<img src="2.jpg" alt="lol" width="480" height="678">
</body>

<footer>
```

```
<center>
<p>This website made for Unix for Telecommuncations (TNE30019).</p>
<p>Copyright Joshua Redolfi (101601886) 2019.</p>
</footer>
</html>==================================

Test Result: PASSED


********************************************************************************
PORTFOLIO TASK RESULT: PASSED
```

10 Practical - Lab 4: Basic Apache (P)

Lab task - Basic Apache

Outcome	Weight
ULO4	♦♦♦♦◊

Outcome	Weight
ULO3	♦♦◊◊◊

Outcome	Weight
ULO1	♦◊◊◊◊

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Practical - Lab 4: Basic Apache (P)

Submitted By:
Joshua Redolfi
101601886
2019/08/27 14:40

Tutor:
Farinaz Jowkarishasaltaneh

August 27, 2019

```
Marking P-Lab-04-Basic-Apache-P on RULE host rule58
TNE30019/TNE80014 - Laboratory - Basic Apache
==========================================
********************************************************************************
 Portfolio Task: P-Lab-04-Basic-Apache, Pass Task

 Configure your rule host (rule58) with Apache to serve a single web site

 Tasks:
  Apache running on rule58
  Browsing to http://rule58 will return a valid web page
********************************************************************************

Date: 20190827_1439
Last 5 Logons:
student    pts/0    vpn247-179.cc.swin.edu Sun Aug 25 12:47 - 15:19  (02:32)
student    pts/1    10.1.16.119            Wed Aug 21 13:00 - 14:29  (01:29)
student    pts/6    10.1.16.119            Tue Aug 20 14:41 - 16:28  (01:47)
student    pts/1    10.1.16.119            Mon Aug 19 16:43 - 17:08  (00:25)
student    pts/0    vpn247-159.cc.swin.edu Sat Aug 17 14:36 - 16:25  (01:48)

RULE host Socket information
www      httpd      77938 3  tcp4   136.186.230.58:80     *:*
www      httpd      77937 3  tcp4   136.186.230.58:80     *:*
www      httpd      77936 3  tcp4   136.186.230.58:80     *:*
www      httpd      77935 3  tcp4   136.186.230.58:80     *:*
www      httpd      77934 3  tcp4   136.186.230.58:80     *:*
www      httpd      77933 3  tcp4   136.186.230.58:80     *:*
www      httpd      77932 3  tcp4   136.186.230.58:80     *:*
www      httpd      77931 3  tcp4   136.186.230.58:80     *:*
www      httpd      77930 3  tcp4   136.186.230.58:80     *:*
root     httpd      77929 3  tcp4   136.186.230.58:80     *:*
bind     named      23184 20 tcp4   136.186.230.58:53     *:*
bind     named      23184 21 tcp4   136.186.230.58:953    *:*
bind     named      23184 512 udp4  136.186.230.58:53     *:*
bind     named      23184 513 udp4  136.186.230.58:53     *:*
bind     named      23184 514 udp4  136.186.230.58:53     *:*
bind     named      23184 515 udp4  136.186.230.58:53     *:*
bind     named      23184 516 udp4  136.186.230.58:53     *:*
bind     named      23184 517 udp4  136.186.230.58:53     *:*
bind     named      23184 518 udp4  136.186.230.58:53     *:*
bind     named      23184 519 udp4  136.186.230.58:53     *:*
bind     named      23184 520 udp4  136.186.230.58:53     *:*
bind     named      23184 521 udp4  136.186.230.58:53     *:*
bind     named      23184 522 udp4  136.186.230.58:53     *:*
bind     named      23184 523 udp4  136.186.230.58:53     *:*
bind     named      23184 524 udp4  136.186.230.58:53     *:*
bind     named      23184 525 udp4  136.186.230.58:53     *:*
bind     named      23184 526 udp4  136.186.230.58:53     *:*
bind     named      23184 527 udp4  136.186.230.58:53     *:*
bind     named      23184 528 udp4  136.186.230.58:53     *:*
bind     named      23184 529 udp4  136.186.230.58:53     *:*
bind     named      23184 530 udp4  136.186.230.58:53     *:*
bind     named      23184 531 udp4  136.186.230.58:53     *:*
bind     named      23184 532 udp4  136.186.230.58:53     *:*
bind     named      23184 533 udp4  136.186.230.58:53     *:*
bind     named      23184 534 udp4  136.186.230.58:53     *:*
root     sshd       85567 3  tcp4   136.186.230.58:22     *:*
root     ntpd       1485  55 udp4   136.186.230.58:123    *:*
?        ?          ?     ?  tcp4   136.186.230.58:57881  136.186.230.58:80
```

```
?       ?          ?     ?  tcp4   136.186.230.58:57882  136.186.230.58:80
?       ?          ?     ?  tcp4   136.186.230.58:57883  136.186.230.58:80
?       ?          ?     ?  tcp4   136.186.230.58:57884  136.186.230.58:80
?       ?          ?     ?  tcp4   136.186.230.58:57885  136.186.230.58:80

HTTP QUERY http://rule58
====================================
--2019-08-27 14:39:43--  http://rule58/
Resolving rule58 (rule58)... 136.186.230.58
Connecting to rule58 (rule58)|136.186.230.58|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 633 [text/html]
Saving to: '/tmp/wget_save.YBAmz7rD'

     0K                                                       100% 74.9M=0s

2019-08-27 14:39:43 (74.9 MB/s) - '/tmp/wget_save.YBAmz7rD' saved [633/633]


====================================
Downloaded Page contents
====================================
<!DOCTYPE html>
<html>
<body bgcolor="yellow">
<title>Joshua's Epic Website</title>
<body>
<h1 style="color:blue; border:2px solid Tomato;">Joshua's Epic Website</h1>
<hr>
<h2>What is his student number?</h2>
<p>101601886</p>
<h2>You when you see this website:</h2>
<img src="confuse.jpg" alt="lol" width="256" height="256">
<p>or</p>
<img src="wth.jpg" alt="lol" width="256" height="256">
<p></p>
<a href="private/index.html">Click if you dare</a>
</body>

<footer>
<center>
<p>This website made for Unix for Telecommuncations (TNE30019).</p>
<p>Copyright Joshua Redolfi (101601886) 2019.</p>
</footer>
</html>====================================

Test Result: PASSED


********************************************************************************
PORTFOLIO TASK RESULT: PASSED
```

11 Theory - Test 1 (P)

In class test run during tutorial in week 5 of semester

Outcome	Weight
ULO2	♦♦♦◊◊

Outcome	Weight
ULO7	♦♦♦◊◊

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Theory - Test 1 (P)

Submitted By:
Joshua Redolfi
101601886
2019/09/10 14:39

Tutor:
Farinaz Jowkarishasaltaneh

September 10, 2019

Unix for Telecommunications

Portfolio Task – T-Test-1
Pass Level Task

Student ID	101601886
Student Name	Joshua Redolfi
Test Date	Week 5
Time Allowed	30 minutes

Assessment – Staff Use Only

In order to pass the test you **MUST** score a minimum of **80%**

Question:	1	2	3	4	5	Total
Points:	4	6	7	6	9	32
Score:	4	4	4½	6	6	24½

PASS

12 Communications - Research Report

Research report required on Assignment topic

Outcome	Weight
ULO7	♦♦♦♦♦

Outcome	Weight
ULO3	♦◊◊◊◊

Outcome	Weight
ULO2	♦♦◊◊◊

Date	Author	Comment
2019/09/26 09:52	Jonathan Kua	Excellent work, well done!

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Communications - Research Report

Submitted By:
Joshua Redolfi
101601886
2019/09/20 08:46

Tutor:
Farinaz Jowkarishasaltaneh

September 20, 2019

Machine Learning: Applications for Internet of Things and Network Engineering

Joshua Redolfi
Faculty of Science,
Engineering and, Technology
Swinburne University of Technology
Email: 101601886@student.swin.edu.au

***Abstract*—Machine learning has many applications in combination with computer networks and Internet of Things. Firstly the various types of machine learning models and algorithms are discussed, such as deep learning networks and Bayes algorithms. The applications of machine learning in networking is then considered. It is found that most research primarily focuses on the benefits of using machine learning to detect network attacks, such as intrusions and denial of service. Adding onto this, its ability to detect botnets is discussed and then attacks that allow for the evasion of machine learning techniques are considered. The use of it to prevent malicious internet use is also looked into. Finally, the combination of machine learning and Internet of Things is discussed, along with instances of it being used in the mining industry and proposals for smart cities. Then its challenges are discussed, such as data quality. The report is concluded with a future outlook for the technology and the challenges it could face.**

I. Introduction

Machine learning has transformed data analysis and technology throughout the world and networking has not been an exception. Together with internet of things (henceforth IoT), machine learning has increasingly become a focus because of the rapid expansion of data collection and processing capabilities in the world [1]. While using it with IoT will likely have a tremendous effect on how data collection is performed, it also has applications in the field of networking [2].

In this report, a review shall firstly be done on the aspects of machine learning that have been used in networking and IoT applications. Thereafter applications of machine learning in networking and internet applications shall be covered. This will cover aspects ranging from monitoring internet usage to detecting attacks on networks. Afterwards, the implications and current research into the combination of IoT and machine learning shall be discussed. Areas covered include current innovations in the mining, industry then smart cities and finally the use of machine learning and IoT in Industry 4.0 (with a focus on predictive maintenance). Throughout this article, examples of current industrial use and future outlooks shall be made.

II. Machine Learning

Machine learning, according to IBM, is a form of artificial intelligence (henceforth AI) that can predict outcomes via the use of data rather than programming [3]. It can learn by integrating conclusions from the data automatically and also self-improve its algorithm [4]. The concept itself is not entirely new, with machine learning actually defined at the end of the 1950s [5].

Several machine learning technologies/algorithms will be discussed throughout this article, the following are brief explanations of how each technology works.

Bayes: Bayes (otherwise known as Naive Bayes) algorithms assume that all inputs to the system are independent of each other [6]. They are based on the Bayes Theorem and use the method of maximum likelihood to generate results [6].

Decision Tree: A decision tree is modelled as a binary tree with each leaf being a decision based on a variable [6]. By combining many decision trees into one algorithm, a random forest algorithm is created [7]. Decision trees are very fast to learn and make predictions and do not require any special preparation of the data inputted [6]. They also have a high ability to accurately generalise large amounts of data, allowing them to detect anomalies [8] [9].

Neural networks: An algorithm based on three components: an input layer, a hidden layer and an output layer [6]. It attempts to mimic human learning by having layers of interconnected units [3]. In the case of multiple hidden layers it is known as deep learning [3].

Reinforcement Learning: Reinforcement learning is where an algorithm rewards itself for producing an optimal decision and penalises itself for not doing so [10]. Thus, after running the algorithm for some time, it will produce optimal paths that it knows are the most rewarding [10].

Support Vector Machines: Support Vector Machines are heuristic algorithm that attempt to separate two attributes in data [6]. They solve a ”kernel function” to solve this classification function (which will separate the data) which can be in forms such as linear or polynomial [6].

III. Machine Learning in Networks

As mentioned earlier, machine learning has found use in network engineering and networks themselves. Most of the

literature available revolves around the concept of intrusion detection. Specifically, machine learning is used to detect DDoS attacks or locate their botnets so that they can be eliminated or mitigated. To achieve this, the algorithms are trained through the use of provided datasets, which are not consistently used [11].

A. Mitigating Network Attacks

Before 2000, machine learning was attempted to be applied to detecting network attacks [12]. One of the algorithms presented in [12] uses a decision tree based on the relation between services and ports. This can be applied to genetic algorithms that generate alerts when the intrusion is detected [12].

However, later research has found that machine learning has not been applied effectively, with the assumption often made that a network attack will often constitute a deviation from expected behaviour [2]. This however generates large numbers of false positives because machine learning is better at finding common patterns than detecting anomalies [2]. Moreover, the diversity of internet traffic is not often not researched and not easily replicated in a simulation [2].

Despite this, approaches using Bayes' algorithm have found up to a 99 percent success rate using test datasets [13]. However, this system was expected to work best in conjunction with "an expert system which is able to provide recommendations based on attack types" [13]. Unlike what was suggested in [2], there was no catering for a diverse range of traffic.

Distributed denial of service (DDoS) are another area of network attacks that machine learning has been applied to. Software Defined Networking (SDN), which is a new network model that divides the devices into being those only with data planes (switches) and control planes (a centralised controller) has resulted in new applications using machine learning to detect these attacks [8]. Machine learning to detect DDoS attacks relies on using a form of anomaly detection [8] [9]. To detect it in SDN networks, many options are presented, ranging from neural networks to decision trees [8]. Highlights of the overall analysis found that neural networks were better at generalizing from incomplete data and Bayesian algorithms were able to include previous knowledge [8]. Fuzzy logic, on the other hand, only required that approximate reasoning rather than precise, which would allow it to effectively counter port scans but consumed more resources than any method [8].

Outside of SDN networking, anomaly detection has been used in conjunction with a random tree algorithm (a set of decision trees) [9]. Figure 1 shows the system set up to use this random tree algorithm. All traffic is handled by a bait server, which authenticates the traffic based on what the algorithm decides (and what user settings are applied). Unauthenticated traffic in this case is then sent to the decoy server (which can later be authenticated) and authenticated traffic to the web server [9]. The algorithm chooses which traffic to send by sorting against previous traffic containing information such as source port, unique bytes and number of packets, which is the core of the decision tree [9].

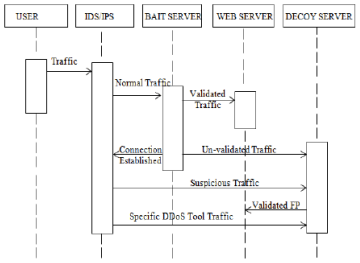


Fig. 1. The set up used for anomaly detection in [9].

Other approaches using machine learning to counter DDoS attacks include searching for botnets, which provide the backbone of DDoS attacks and shutting them down [11]. Botnets can operate in both client-server and peer-to-peer operation, however they all the same software [11]. Thus, machine learning can be used to detect which hosts are running botnets by sifting through large volumes of network data such as source ports [11]. For example, peer-to-peer networks (which are by usual methods, difficult to detect) have been found by using an artificial neural network that processes network traffic that is converted in traffic flows [14]. These flows are then compressed by taking an average of packet count and size and passed through the neural network [14]. The network then calculates if the flow is likely to be a bot or not (based on previously found botnet datasets) and then determines if it is confident in its prediction [14]. If the confidence is below the threshold, then it is passed through a decision tree algorithm to determine if it is a botnet based on preconfigured attributes [14].

However as detection technology becomes more sophisticated, so does the evasion methods. Attackers, especially those who are familiar with the aspects of the model used to detect botnets, are able to adapt the traffic produced by them to pass through the filters undetected [15]. To evade any machine learning algorithm, three main attacks have been identified. The first is a gradient-based attack, where if the model is known, the attacker can send information designed to confuse the algorithm [16]. Secondly there is score-based attacks that consist of probing the model and predicting the attributes of the model based on what results are returned [16]. Finally, by sending data through a network through trial and error, the basic outputs of a system can be determined, which is called a decision-based attack [16]. It is notable that most of these attacks can be countered by adding stochastic output in the model to confuse the attacker about what the gradient is [16].

The attack that is based off the above principles and described in [15] is a machine learning algorithm designed to counter other machine learning algorithms that detect botnets. It does this through the use of deep reinforcement learning that has the goal to learn good policies for sequential decision problems, by optimizing a cumulative future reward signal [15]. This future reward signal allows for the model to have reinforcement whenever it makes a beneficial decision [15].

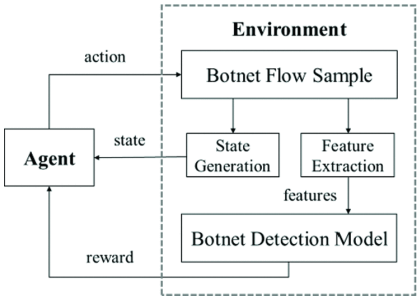


Fig. 2. The model used for detecting botnets in [15].

Figure 2 shows the flowchart that makes up the machine, which works by passing a flow through the policy (the algorithm designed to detect botnets). The flow's timestamps are modified or new packets (that are not related to the botnet) are appended to determine the settings of the policy [15]. The paper concedes that the more information that is known about an algorithm will result in better predictions [15].

Thus, machine learning has plenty of potential in network security to avoid DDoS attacks and other intrusion attacks such as the model seen in [8]. Conversely it can be used by attackers to evade the models created for network security to allow for attacks to breach policy undetected [16]. However, the reach of machine learning extends further than network security, it can also be used to detect malicious internet usage.

B. Detection of Spambots

Spambots are commonplace in major social media sites such as Facebook and Twitter, with sophisticated bots able to send unsolicited replies containing malicious links on the latter [17]. After analysing the behavior of the bots on Twitter (duplicate tweets and measuring the ratio of followers to followed), it was found that the most effective way to detect bots was using a Bayesian Classifier [17]. Similar technology using a support vector machine has been proposed for the detection of web spambots that rely on similar ways to spread spam (replying to forum boards and blogs) [18]. However, the Twitter spam bot detection used a probability based on the attributes of bots mentioned earlier to feed into the algorithm [17]. On the other hand, the web spam bot detector searched for user requests to the website, thus operating at a lower level [18]. In both cases, using the respective classifiers achieved a high level of accuracy in detecting spam [17] [18].

C. Network Resource Allocations

While the majority of literature for machine learning applications in relation to networks and the internet is used for internet security, there are exceptions. Machine learning can be applied to analyse computer networks to optimise their resource usage [19]. Research from the Massachusetts Institute of Technology and Microsoft have created an example product for resource management called DeepRM [10]. By using a reinforcement solution (which was found to be superior to other methods because it allows for an agent to learn directly from experience), it allocates resources to jobs [10]. To improve itself, DeepRM rewards itself for reducing the completion time of scheduled tasks [10]. Thus, the aglorthm will attempt to keep using the most rewarding resource allocation to tasks [10]. Compared to other job agents such as a Shortest Job First agent, DeepRM performed the best with a completion 50 per cent shorter on average than other software (Figure 2) [10]. Despite this solution being presented in 2016, there is no evidence of any organisation actually using it, suggesting that adoption is slow or that it has been superseded.

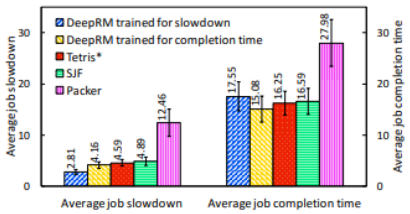


Fig. 3. The performance of DeepRM as presented by [10] versus competitors that do not use machine learning.

Two years after the previous paper and returning to the concept of Software Defined Networking, machine learning can be used in the controller of the network to provision resources in advance [20]. Like the previous paper, it uses reinforcement learning based on configuring the network correctly for optimal traffic usage [20]. However, on top of this is another model based on deep learning to predict network traffic based on previous data [20]. There appears to be no test data explaining the effectiveness of this model as of yet.

In conclusion, machine learning can be applied for many areas of network administration. It does have potential to be used in detecting many types of attacks ranging from network intrusion to DDoS attacks to spam bots. However, it must be noted that machine learning can be exploited by attackers to learn about the model used to secure the network, thus allowing for attackers to circumvent some models as seen in [16]. Despite the volume of data available in networking, for research purposes it is hard for researchers to obtain and therefore develop new models [19].

D. Applications in Network Engineering: Traffic Analysis at Cisco

An example where machine learning is already being in the network engineering industry is Ciscos network monitoring product called StealthWatch. This software provides threat detection and response mechanism that allows it to keep track of all network users and what traffic they are using [21]. Cisco also claims it is the only software on the market that can detect malware in encrypted traffic, however similar approaches were proposed by other authors were proposed several years ago [22] [23].

Stealthwatch collects a swath of data from already existing sources, such as Cisco routers and switches and analyses this to pinpoint threats within the network [21]. This can be as simple as something as device accessing a resource it should not and as advanced as malware that is hidden behind

encrypted traffic [21]. This detection is done in real time as the network devices constantly send data to the controller [24]. This data is then passed several layers of machine learning models to determine if an attack or malicious activity is occurring [24].

Machine learning is utilized in Stealthwatch is a form of statistical analysis to pick up unusual patterns within the network [21]. However, not all unusual behavior is malicious, and Cisco claims that the program is able to distinguish between these two types [21] [22]. It does in a few ways, the first being it connection with Ciscos cognitive intelligence machine learning engine that operates globally [22]. This allows for the software to determine if the threat not matches those previously seen on the network, but those seen on many networks around the globe [22]. This is one of the methods it uses to detect malicious traffic even though it is encrypted and is similar to what was found in [23] which used HTTP headers from previous attacks to determine if an application or network activity was malicious [22].

Previous methods detecting malicious files involved analysed the hashes of incoming and determining if they marched previous hashes of known viruses or malware [23] [22]. Malenfant in [22] notes that is sufficient to detect one malicious file, and the authors of [23] note that while it is able to determine known attacks, it cannot detect any new attacks. Stealthwatch, on the other hand takes into account 400 different attributes that could flag network activity as suspicious [22]. These are used as discrete identifiers in an unspecified machine learning model [22]. Other classifiers include SHA-256 hash values and web proxy logs [22]. Using a variety of tactics could mitigate the concerns mentioned in [2] which stated that machine learning is better at picking up patterns than anomalies.

Overall, Ciscos Stealthwatch is a potent application of machine learning already deployed within network engineering. It has the express goal of using machine learning as one of its tools to prevent network attacks and intrusions. Furthermore, it shows an application of machine learning detecting network attacks that is already in use globally.

IV. Machine Learning in Internet of Things

Up to 50 billion devices will be connected to the internet by 2020 and this is in part due to every device being connected to the internet - IoT [25]. This presents an opportunity for machine learning to learn from and provide solutions for large data sets [25]. Applications in the literature have ranged from smart cities to increasing the efficiency of mining [25]. However, the combination is not without its challenges, especially in the area of data privacy, this will be discussed for each area of machine learning and IoT covered.

A. Smart Cities

IoT and machine learning can be combined into smart cities to take advantage of both the masses of sensors present in the city and data analysis to benefit and assist citizens. These applications can range from areas such as health to transportation.

Firstly, is it noted that even a small city can produce a large amount of the data should there be a sufficient amount of sensors, therefore making it impossible for any smart city to be monitored without the use of machine learning [26]. Therefore, the authors of [26] propose a dashboard system capable of handling the data produced by the smart city. The dashboard is built on by relying off a deep neural network to provide analysis on the large volume of data coming into it [26]. This neural network features several important structures, the first being the use of drop out learning, which is a counter to a flaw in deep learning algorithms where they tend to overfit data and make unreasonable generalisations [27]. Drop out learning is where each unit in the model has a chance to be left out during processing, which removes most of these generalizations [27]. Another feature about this algorithm is the use of rectified linear units over nonlinear activation functions, which allow for much more efficient gradient propagation [27].

This model was tested using a sound dataset, with three goals: sound pollution monitoring, traffic monitoring using sound and search and rescue operations [26]. For the first goal of sound pollution, it was found that is not enough to use a camera system, and instead the use of both that and the sound system provided by machine learning allowed for offending vehicles that exceed the nose limit [26]. This can also be applied at night to identify cars that otherwise cannot be identified on camera and thus allowing for emergency vehicle detection [26]. Finally, this same technology can be applied to lost children and dogs by analysing their sound signature in addition to cameras [26].

However, this data must be collated into a dashboard and it is considered best practise in this article to provide a simple user interface to facilitate human decision making [26]. The neural network interpretation is provided to show how the model reached that conclusion [26]. But other approaches to building a dashboard for smart cities do not provide an option to analysis machine learning interpretation of the data [28].

Machine learning and IoT can also combined to potentially improve the health of the inhabitants of smart cities. This can be done by using IoT sensors to monitor users and the environment, and then using machine learning to provide actionable output [29]. A direct application of this can be monitoring the user through their smartphone or an indirect application such as monitoring air quality [29]. In this model a context engine is used; a context engine is a functional unit that has many inputs, but only one output [29]. By having a hierarchy of these within a machine learning model, compactness is traded off for more versatile responses, in turn allowing for better system stability and reducing the computational complexity that a flat hierarchy of units would require [29]. The ability to aggregate hierarchical information is also recommended to allow for the integration of already existing technology [30]. To achieve all of this, matrix-based stochastic learning models are used to transform the data [29]. This is then passed through an algorithm called the Taylor Expanded Analog Forecasting Algorithm which simply is based on a Taylor series and is

used to translate the data [29]. Each element in the series is multiplied coefficient learned with observation (a higher order member of the series has a higher correlation) [29]. This is then combined with linear algebra and the least squares approximation to determine the coefficients [29].

While technology like this can be found outside of a smart city context, the combination with smart cities allows for the above model to be applied to a greater effect. One of the examples provided is of air quality, which in the case of a single user using smart health technology, would be much harder to detect [29]. By monitoring the air quality in "Citisense" air monitors and GPS location data, a map can be built up of the air quality in areas of the city [29]. This information is passed through the model and therefore the algorithm generates a path through the city that can avoid the areas with lower air quality to the best of its ability [29].

However, there is a downside with any service related to personal health and using data for machine learning both within and outside smart cities. For example, with the dataset above for air pollution, a biased dataset or one created with dirty data could create a suboptimal path and thus risk the users health [30]. Moreover, in other cases more related to health (in essence: a direct application to monitor a users health), bias could be present from humans that could make its way into a dataset [30].

B. Industry 4.0

Machine learning is commonly combined with IoT in the field of Industry 4.0 (the next stage of manufacturing). Industry IoT, otherwise known as IIoT, will allow for machinery on the factory floor to have the ability to self-learn and diagnose problems without the help of a human engineer [31] [32]. The use of Big Data in Industry 4.0 will significantly change the skills required to operate a manufacturing plant, especially with data analysis [31].

Predictive Maintenance is an important issue in manufacturing, as it can determine when factory machinery needs maintenance. However, it is hard for a single solution to exist for all manufacturing plants [32]. Current methods of predictive maintenance rely on using PLC controllers to determine if a measured value has deviated from what is expected [32]. While this is simple and easy to implement, it can only have parameters adjustment can be done only having failure describing information [32]. But by using neural networks and sensors such as those PLCs mentioned before, the model can predict when the deviation from a required value has occurred (in essence, if the performance of the machine has decreased) [32] [33].

By combining concepts such as machine-to-machine communication, IoT and machine learning, alongside PLCs, predictive maintenance can be achieved easily. Firstly, by putting PLCs on a slitting machine and monitoring its output, data can be sent to a machine learning algorithm [33]. The algorithm in this case is an autoregressive integrated moving average can be used to predict future data points [33]. By the purposes of benchmarking, this algorithm was used in conjunction with four different models, including a Bayes algorithm and a deep neural network [33]. The whole set up looked like that as shown in Figure 4 [33].

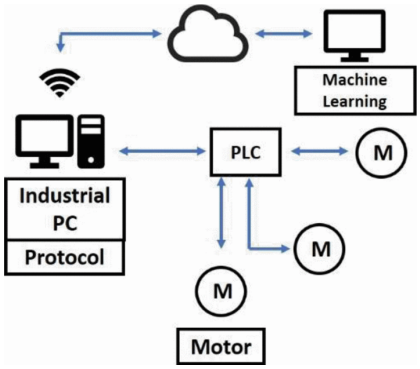


Fig. 4. A model of predictive maintenance that relies on machine learning [33].

It was found that the deep neural network was the best at performing predictive maintenance, however it was not without its flaws [33]. However, due to the low number of bad quality cycles, the model will have to continuously change the weights every time as each bad cycle is significant deviation [33]. The use of proactive anomaly detection can prevent this having a significant impact on the model [33].

In conclusion, IoT and machine learning are compliments of each other in many fields ranging from Smart cities to Industry 4.0. Smart cities, which cover most aspects of human life, can benefit from machine learning so as long as there is an adequate data dashboard to allow for easy human decision making [26]. There is also the aspect of health, which for example can benefit those with breathing issues by having real time data processed about how to avoid air pollution [29]. However, data from systems such as these must be treated with caution, as they could potentially be biased by even a small amount of corrupt data [30].

Industry 4.0 technologies are already possible, with the example of predictive maintenance showing how machine learning and IoT can be applied to directly benefit the manufacturing process [33]. Below there are examples where it is already implemented and show that there is much potential for future expansion.

C. Applications in Internet of Things: The Mining Sector

The mining sector has also taken advantage of the combination of IoT and machine learning. According to Micromine, a company specializing in solutions for mining, large volumes of data generated are by any mining operation [34]. Therefore, machine learning allows for better identification of issues and much faster decision making [34]. Another company by the name of Komatsu, that provides solutions for industries such as forestry and mining, has developed a holistic analytics system for large mining machines [35]. These machines can have approximately 125 sensors on it, which via use of IoT can feed data into MATLAB and Grafana [35]. Their system

has the capability of producing up to 200000 data points per second [35]. This requires machine learning to help sort through and process the data [35]. However, Komastsu did state that potential issues with their data analytics (including machine learning) includes the reliability of the data, getting the data to the right person at the right time and also timezones [35].

D. Applications in Internet of Things: IBM's Watson Platform for Industry

IBM puts significant resources in machine learning and develops a product called Watson Content [36]. Watson is an AI capable of performing machine learning providing data analysis and easy implementation of machine learning applications [36]. One of the places where it is used is in Industry 4.0 (and also cognitive manufacturing) [37].

As mentioned in literature, current manufacturing processes rely on PLCs providing data to a centralised location, where it is then stored and action taken if a value has changed from what is expected [32] [37]. Traditional manufacturing processes also assume that asset management maintenance management are different areas, despite their interconnectedness [37]. Real time data visualisation currently is also not feasible for most manufacturers [37].

IBM proposes to change the way manufacturing is done by introducing the aforementioned Industry Internet of Things (IIoT), of which Watson is the central product for data analysis, This include facets such as textual and image analysis, along with machine learning for processing this analysis [37]. For example, using acoustic analysis allows for the model to hear the sounds that a machine on the production line is making [37]. This allows for the anomalies in machine sounds to be detected without human help [37].

A real example of where Watson has been implemented in China for Shenzhen China Star Optoelectronics Technology Co., Ltd [38]. They use Watsons IoT technology, which includes machine learning as a form of data analysis to enhance their quality control process [37] [38]. They started by building a library of pictures of both faulty and acceptable products, reaching into the thousands [38]. This was then used to train an AI model which was then deployed on the factory floor in edge computers (edge devices being those closest to the factory floor) [37] [38]. This matches the data control as was recommended for machine learning implementations in [30].

Overall, implementing machine learning and other technologies through the use of Watson, the client claimed to reduce defect detection time significantly through the use of AI imaging versus a human operator [38].

V. Conclusion and Future Outlook

Overall, the effect that machine learning has made so far on the fields of network engineering and IoT has been substantial. With examples from IBM showing how often it is already used (especially in the context of IoT and Industry 4.0), it can be seen that future revelations in the field will be significant as well. In fact, IBM predicts that machine learning will eventually be embedded in most applications and also that most datasets will be superceded by models that have the ability to change in use [39].

In terms of network engineering, it was found that most applications revolves around network security, especially in fields relating to denial of service attacks. While machine learning features the capabilities to defend against such attacks and also sniff out botnets (even peer to peer botnets), attackers can send bogus data through a model [16]. Then based off the result they can potentially predict what data can evade the network (especially if the model is available open source) through the use of their own machine learning model [16]. Spam bots can also be detected through the use of machine learning, such as in the example on Twitter provided [17]. However, it is likely that this would suffer the aforementioned effects that botnet detectors can suffer from. It was also found that another area of use for machine learning in network engineering was developing resource scheduler tools which can perform better than currently existing tools [10].

As for machine learning and IoT, there are many applications of this already in existence that are using both as a part of Industry 4.0. Industry 4.0 has been found to be a major area of interest in machine learning given the many already existing applications for it. Smart cities are another area of interest, with areas such as health being explored [29]. In the context of smart health and outside of it, programmers must make sure that machine learning algorithms are not trained against dirty or corrupt data least they become biased and make indecipherable decisions [30].

In summary, machine learning, when combined with IoT or networking engineering, provides a powerful tool that can be used to comb through the large amount of data now being collected in the world [3]. With applications for machine learning in almost every engineering field, it is important to learn about and understand.

Acknowledgment

The author would like to thank Jonathan Kua for answering questions in a timely manner and providing assistance whenever the author needed help.

References

[1] J. McDonald. *The Internet of Things requires machine learning and AI*. Mar. 2019. URL: https://www.ibm.com/blogs/internet-of-things/iot-the-internet-of-things-requires-machine-learning/.

[2] R. Sommer and P. Vern. "Outside the closed world: On using machine learning for network intrusion detection". In: *2010 IEEE symposium on security and privacy*. IEEE. 2010, pp. 305–316.

[3] E. Guriankina. "Introduction to Machine Learning". In: May 2019. URL: https://www.ibm.com/ru-ru/events/think-summit/assets/pdf/Ekaterina_Guryankina.pdf.

[4] D.P. Vinchurkar and A. Reshamwala. *A Review of Intrusion Detection System Using Neural Network and Machine Learning*. 2012. URL: https://www.researchgate.net/profile/Alpa_Reshamwala/publication/233943805_A_Review_of_Intrusion_Detection_System_Using_Neural_Network_and_Machine_Learning_Technique/links/02bfe50d2f409cdafa000000.pdf.

[5] M. Tanskanen. *Applying machine learning to IoT data*. URL: https://www.sas.com/en_au/insights/articles/big-data/machine-learning-brings-concrete-aspect-to-iot.html.

[6] P. A. Harlianto, T. B. Adji, and N. A. Setiawan. "Comparison of machine learning algorithms for soil type classification". In: *2017 3rd International Conference on Science and Technology - Computer (ICST)*. July 2017, pp. 7–10. DOI: 10.1109/ICSTC.2017.8011843.

[7] Y. Liu et al. "An Improved Random Forest Algorithm Based on Attribute Compatibility". In: *2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*. Mar. 2019, pp. 2558–2561. DOI: 10.1109/ITNEC.2019.8729146.

[8] J. Ashraf and S. Latif. "Handling intrusion and DDoS attacks in Software Defined Networks using machine learning techniques". In: *2014 National Software Engineering Conference*. Nov. 2014, pp. 55–60. DOI: 10.1109/NSEC.2014.6998241.

[9] J. D. Ndibwile et al. "Web Server Protection against Application Layer DDoS Attacks Using Machine Learning and Traffic Authentication". In: *2015 IEEE 39th Annual Computer Software and Applications Conference*. Vol. 3. July 2015, pp. 261–267. DOI: 10.1109/COMPSAC.2015.240.

[10] H. Mao et al. "Resource Management with Deep Reinforcement Learning". In: *Proceedings of the 15th ACM Workshop on Hot Topics in Networks*. HotNets '16. Atlanta, GA, USA: ACM, 2016, pp. 50–56. ISBN: 978-1-4503-4661-0. DOI: 10.1145/3005745.3005750. URL: http://doi.acm.org.ezproxy.lib.swin.edu.au/10.1145/3005745.3005750.

[11] X. Dong, J. Hu, and Y. Cui. "Overview of Botnet Detection Based on Machine Learning". In: *2018 3rd International Conference on Mechanical, Control and Computer Engineering (ICMCCE)*. Sept. 2018, pp. 476–479. DOI: 10.1109/ICMCCE.2018.00106.

[12] C. Sinclair, L. Pierce, and S. Matzner. "An application of machine learning to network intrusion detection". In: *Proceedings 15th Annual Computer Security Applications Conference (ACSAC'99)*. Dec. 1999, pp. 371–377. DOI: 10.1109/CSAC.1999.816048.

[13] F. Jemili, M. Zaghdoud, and M. B. Ahmed. "A Framework for an Adaptive Intrusion Detection System using Bayesian Network". In: *2007 IEEE Intelligence and Security Informatics*. May 2007, pp. 66–70. DOI: 10.1109/ISI.2007.379535.

[14] S. Chen, Y. Chen, and W. Tzeng. "Effective Botnet Detection Through Neural Networks on Convolutional Features". In: *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*. Aug. 2018, pp. 372–378. DOI: 10.1109/TrustCom/BigDataSE.2018.00062.

[15] D. Wu et al. "Evading Machine Learning Botnet Detection Models via Deep Reinforcement Learning". In: *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*. May 2019, pp. 1–6. DOI: 10.1109/ICC.2019.8761337.

[16] W. Brendel, J. Rauber, and M. Bethge. "Decision-based adversarial attacks: Reliable attacks against black-box machine learning models". In: *arXiv preprint arXiv:1712.04248* (2017).

[17] A.H. Wang. "Detecting spam bots in online social networking sites: a machine learning approach". In: *IFIP Annual Conference on Data and Applications Security and Privacy*. Springer. 2010, pp. 335–342.

[18] P. Hayati et al. "Web Spambot Detection Based on Web Navigation Behaviour". In: *2010 24th IEEE International Conference on Advanced Information Networking and Applications*. Apr. 2010, pp. 797–803. DOI: 10.1109/AINA.2010.92.

[19] M. Wang et al. "Machine Learning for Networking: Workflow, Advances and Opportunities". In: *IEEE Network* 32.2 (Mar. 2018), pp. 92–99. DOI: 10.1109/MNET.2017.1700200.

[20] R. Alvizu et al. "Machine-Learning-Based Prediction and Optimization of Mobile Metro-Core Networks". In: *2018 IEEE Photonics Society Summer Topical Meeting Series (SUM)*. July 2018, pp. 155–156. DOI: 10.1109/PHOSST.2018.8456750.

[21] "Cisco Security Analytics". In: (2018). URL: https://www.cisco.com/c/dam/en/us/products/collateral/security/stealthwatch/white-paper-c11-740605.pdf.

[22] J. Malenfant. *How We Apply Machine Learning in Cisco Advanced Threat Solutions*. Aug. 2018. URL: https://blogs.cisco.com/security/how-we-apply-machine-learning-in-cisco-advanced-threat-solutions.

[23] S. Wang et al. "Detecting Android Malware Leveraging Text Semantics of Network Flows". In: *IEEE Transactions on Information Forensics and Security* 13.5 (May 2018), pp. 1096–1109. DOI: 10.1109/TIFS.2017.2771228.

[24] *Cisco Stealthwatch Enterprise Data Sheet*. Aug. 2019. URL: https://www.cisco.com/c/en/us/products/collateral/security/stealthwatch/datasheet-c78-739398.html.

[25] M. Mahdavinejad et al. "Machine learning for internet of things data analysis: a survey". In: *Digital Communications and Networks* 4.3 (2018), pp. 161–175. ISSN: 2352-8648. DOI: https://doi.org/10.1016/j.dcan.2017.10.002. URL: http://www.sciencedirect.com/science/article/pii/S235286481730247X.

[26] S. Sanaei, B. Majidi, and E. Akhtarkavan. “Deep Multisensor Dashboard for Composition Layer of Web of Things in the Smart City”. In: *2018 9th International Symposium on Telecommunications (IST)*. Dec. 2018, pp. 211–215. DOI: 10.1109/ISTEL.2018.8661092.

[27] K. J. Piczak. “Environmental sound classification with convolutional neural networks”. In: *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*. Sept. 2015, pp. 1–6. DOI: 10.1109/MLSP.2015.7324337.

[28] M. Patel and N. Chauhan. “Smart Dashboard: A Novel Approach for Sustainable Development of Smart Cities using Fog Computing”. In: *2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA)*. June 2019, pp. 632–636. DOI: 10.1109/ICECA.2019.8821813.

[29] J. Venkatesh et al. “Modular and Personalized Smart Health Application Design in a Smart City Environment”. In: *IEEE Internet of Things Journal* 5.2 (Apr. 2018), pp. 614–623. DOI: 10.1109/JIOT.2017.2712558.

[30] F. Ahamed and F. Farid. “Applying Internet of Things and Machine-Learning for Personalized Healthcare: Issues and Challenges”. In: *2018 International Conference on Machine Learning and Data Engineering (iCMLDE)*. Dec. 2018, pp. 19–21. DOI: 10.1109/iCMLDE.2018.00014.

[31] M. O. Gokalp et al. “Big Data for Industry 4.0: A Conceptual Framework”. In: *2016 International Conference on Computational Science and Computational Intelligence (CSCI)*. Dec. 2016, pp. 431–434. DOI: 10.1109/CSCI.2016.0088.

[32] K. Liulys. “Machine Learning Application in Predictive Maintenance”. In: *2019 Open Conference of Electrical, Electronic and Information Sciences (eStream)*. Apr. 2019, pp. 1–4. DOI: 10.1109/eStream.2019.8732146.

[33] A. Kanawaday and A. Sane. “Machine learning for predictive maintenance of industrial machines using IoT sensor data”. In: *2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS)*. Nov. 2017, pp. 87–90. DOI: 10.1109/ICSESS.2017.8342870.

[34] *Machine Learning and AI in Mining*. Jan. 2019. URL: https://www.micromine.com/machine-learning-and-ai-in-mining/.

[35] S. Terry. “How Komatsu is Improving Mining Efficiencies using IoT and Machine Learning”. In: Sept. 2018. URL: https://cdn.oreillystatic.com/en/assets/1/event/278/How%20Komatsu%20is%20improving%20mining%20efficiencies%20using%20the%20IoT%20and%20machine%20learning%20Presentation.pdf.

[36] *Watson Machine Learning - Overview*. URL: https://www.ibm.com/cloud/machine-learning.

[37] S. Bonnaud and C. Didier. “Industrie 4.0 and Cognitive Manufacturing”. In: (Sept. 2018). URL: https://www.ibm.com/downloads/cas/YKEDY8RD.

[38] *Shenzhen China Star Optoelectronics Technology Co., Ltd.* URL: https://www.ibm.com/case-studies/csot-watson-iot-visual-inspection.

[39] J. Hurwitz and D. Kirsch. *Machine Learning for Dummies*. 2018.

13 Practical - Lab 5: Advanced Apache (D)

Lab task - Advanced Apache (Distinction)

Outcome	Weight
ULO4	♦♦♦♦◊

Outcome	Weight
ULO6	♦♦◊◊◊

Outcome	Weight
ULO1	♦♦♦◊◊

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Practical - Lab 5: Advanced Apache (D)

Submitted By:
Joshua Redolfi
101601886
2019/08/28 09:11

Tutor:
Farinaz Jowkarishasaltaneh

August 28, 2019

```
Marking P-Lab-05-Advanced-Apache-D on RULE host rule58
TNE30019/TNE80014 - Laboratory - Advanced Apache
================================================
********************************************************************************
 Portfolio Task: P-Lab-04-Advanced-Apache, Pass Task

 Configure your rule host (rule58) with Apache to serve a virtual hosted web site

 Tasks:
  Set rule58 as DNS Server
  Query for: aristocrats.unix    ; Correct answer: 136.186.230.58
  Apache running on rule58
  Browsing to http://aristocrats.unix  will return a correct web page
********************************************************************************

Date: 20190828_0910
Last 5 Logons:
student    pts/2    10.1.16.119           Wed Aug 28 08:33 - 09:09  (00:35)
student    pts/1    10.1.16.119           Wed Aug 28 08:29 - 09:09  (00:39)
student    pts/0    10.1.16.119           Wed Aug 28 08:23 - 09:09  (00:45)
student    pts/26   10.1.16.119           Tue Aug 27 15:02 - 18:24  (03:21)
student    pts/26   10.1.16.119           Tue Aug 27 15:00 - 15:02  (00:02)

RULE host Socket information
root     sshd       86821 3  tcp4   136.186.230.58:22     10.1.16.119:51436
sshd     sshd       86820 3  tcp4   136.186.230.58:22     10.1.16.119:51436
root     sshd       86819 3  tcp4   136.186.230.58:22     10.1.16.119:51436
www      httpd      86701 3  tcp4   136.186.230.58:80     *:*
www      httpd      86694 3  tcp4   136.186.230.58:80     *:*
www      httpd      86693 3  tcp4   136.186.230.58:80     *:*
www      httpd      86692 3  tcp4   136.186.230.58:80     *:*
www      httpd      86691 3  tcp4   136.186.230.58:80     *:*
www      httpd      86690 3  tcp4   136.186.230.58:80     *:*
root     httpd      86689 3  tcp4   136.186.230.58:80     *:*
bind     named      48811 20 tcp4   136.186.230.58:53     *:*
bind     named      48811 21 tcp4   136.186.230.58:953    *:*
bind     named      48811 512 udp4  136.186.230.58:53     *:*
bind     named      48811 513 udp4  136.186.230.58:53     *:*
bind     named      48811 514 udp4  136.186.230.58:53     *:*
bind     named      48811 515 udp4  136.186.230.58:53     *:*
bind     named      48811 516 udp4  136.186.230.58:53     *:*
bind     named      48811 517 udp4  136.186.230.58:53     *:*
bind     named      48811 518 udp4  136.186.230.58:53     *:*
bind     named      48811 519 udp4  136.186.230.58:53     *:*
bind     named      48811 520 udp4  136.186.230.58:53     *:*
bind     named      48811 521 udp4  136.186.230.58:53     *:*
bind     named      48811 522 udp4  136.186.230.58:53     *:*
bind     named      48811 523 udp4  136.186.230.58:53     *:*
bind     named      48811 524 udp4  136.186.230.58:53     *:*
bind     named      48811 525 udp4  136.186.230.58:53     *:*
bind     named      48811 526 udp4  136.186.230.58:53     *:*
bind     named      48811 527 udp4  136.186.230.58:53     *:*
bind     named      48811 528 udp4  136.186.230.58:53     *:*
bind     named      48811 529 udp4  136.186.230.58:53     *:*
bind     named      48811 530 udp4  136.186.230.58:53     *:*
bind     named      48811 531 udp4  136.186.230.58:53     *:*
bind     named      48811 532 udp4  136.186.230.58:53     *:*
bind     named      48811 533 udp4  136.186.230.58:53     *:*
bind     named      48811 534 udp4  136.186.230.58:53     *:*
root     sshd       85567 3  tcp4   136.186.230.58:22     *:*
```

```
root     ntpd       1485  55 udp4   136.186.230.58:123    *:*
?        ?          ?     ?  tcp4   136.186.230.58:80     10.1.16.119:50036
?        ?          ?     ?  tcp4   136.186.230.58:56653  136.186.230.58:80
?        ?          ?     ?  tcp4   136.186.230.58:30424  136.186.230.58:80
?        ?          ?     ?  tcp4   136.186.230.58:24298  136.186.230.58:80

CHECKING DNS CONFIGURATION
==========================
Querying for (aristocrats.unix) using DNS server (136.186.230.58) - Correct answer (136.186.230.58)
Using domain server:
Name: 136.186.230.58
Address: 136.186.230.58#53
Aliases:

aristocrats.unix has address 136.186.230.58
Test Result: PASSED


ADDING DNS SERVER 136.186.230.58 TO SYSTEM RESOLV.CONF
==============================================

HTTP QUERY http://aristocrats.unix
====================================
--2019-08-28 09:10:14--  http://aristocrats.unix/
Resolving aristocrats.unix (aristocrats.unix)... 136.186.230.58
Connecting to aristocrats.unix (aristocrats.unix)|136.186.230.58|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [text/html]
Saving to: '/tmp/wget_save.O9zXZMYh'

     0K                                                       34.0K=0.01s

2019-08-28 09:10:14 (34.0 KB/s) - '/tmp/wget_save.O9zXZMYh' saved [495]


====================================
Downloaded Page contents
====================================
<html><head><title>Aristocrats Family Theater Resturant</title></head><body>
<center><h1>Hello!</h1></center><p>
<center><h2>The Aristocrats Family Theater Restaurant site is under construction...</h2></center></p
>
<center><h3>Check back soon for updates.</h3></center></p>
<center><h3>Remember: It is only <i>-406325414</i> seconds until our first location opens!</h3/><cen
ter>
<center><h3>Right now it is Wed Aug 28 09:10:14 EST 2019 and we open on Thu Oct 12 13:00:00 EST 2006

</body></html>
====================================

Checking Downloaded page contents

Test Result: PASS(download) + PASS(contents)


REMOVING DNS SERVER 136.186.230.58 FROM SYSTEM RESOLV.CONF
================================================

********************************************************************************
PORTFOLIO TASK RESULT: PASSED
```

14 Practical - Lab 5: Advanced Apache (P)

Lab task - Advanced Apache

Outcome	Weight
ULO4	♦♦♦♦◊

Outcome	Weight
ULO1	♦◊◊◊◊

Outcome	Weight
ULO6	♦◊◊◊◊

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Practical - Lab 5: Advanced Apache (P)

Submitted By:
Joshua Redolfi
101601886
2019/08/28 09:10

Tutor:
Farinaz Jowkarishasaltaneh

August 28, 2019

```
Marking P-Lab-05-Advanced-Apache-P on RULE host rule58
TNE30019/TNE80014 - Laboratory - Advanced Apache
================================================
********************************************************************************
 Portfolio Task: P-Lab-04-Advanced-Apache, Pass Task

 Configure your rule host (rule58) with Apache to serve a virtual hosted web site

 Tasks:
  Set rule58 as DNS Server
  Query for: armitagechemicals.unix    ; Correct answer: 136.186.230.58
  Apache running on rule58
  Browsing to http://armitagechemicals.unix  will return a correct web page
********************************************************************************

Date: 20190828_0910
Last 5 Logons:
student    pts/2    10.1.16.119           Wed Aug 28 08:33 - 09:09  (00:35)
student    pts/1    10.1.16.119           Wed Aug 28 08:29 - 09:09  (00:39)
student    pts/0    10.1.16.119           Wed Aug 28 08:23 - 09:09  (00:45)
student    pts/26   10.1.16.119           Tue Aug 27 15:02 - 18:24  (03:21)
student    pts/26   10.1.16.119           Tue Aug 27 15:00 - 15:02  (00:02)

RULE host Socket information
root     sshd       86821 3  tcp4   136.186.230.58:22     10.1.16.119:51436
sshd     sshd       86820 3  tcp4   136.186.230.58:22     10.1.16.119:51436
root     sshd       86819 3  tcp4   136.186.230.58:22     10.1.16.119:51436
www      httpd      86701 3  tcp4   136.186.230.58:80     *:*
www      httpd      86694 3  tcp4   136.186.230.58:80     *:*
www      httpd      86693 3  tcp4   136.186.230.58:80     *:*
www      httpd      86692 3  tcp4   136.186.230.58:80     *:*
www      httpd      86691 3  tcp4   136.186.230.58:80     *:*
www      httpd      86690 3  tcp4   136.186.230.58:80     *:*
root     httpd      86689 3  tcp4   136.186.230.58:80     *:*
bind     named      48811 20 tcp4   136.186.230.58:53     *:*
bind     named      48811 21 tcp4   136.186.230.58:953    *:*
bind     named      48811 512 udp4  136.186.230.58:53     *:*
bind     named      48811 513 udp4  136.186.230.58:53     *:*
bind     named      48811 514 udp4  136.186.230.58:53     *:*
bind     named      48811 515 udp4  136.186.230.58:53     *:*
bind     named      48811 516 udp4  136.186.230.58:53     *:*
bind     named      48811 517 udp4  136.186.230.58:53     *:*
bind     named      48811 518 udp4  136.186.230.58:53     *:*
bind     named      48811 519 udp4  136.186.230.58:53     *:*
bind     named      48811 520 udp4  136.186.230.58:53     *:*
bind     named      48811 521 udp4  136.186.230.58:53     *:*
bind     named      48811 522 udp4  136.186.230.58:53     *:*
bind     named      48811 523 udp4  136.186.230.58:53     *:*
bind     named      48811 524 udp4  136.186.230.58:53     *:*
bind     named      48811 525 udp4  136.186.230.58:53     *:*
bind     named      48811 526 udp4  136.186.230.58:53     *:*
bind     named      48811 527 udp4  136.186.230.58:53     *:*
bind     named      48811 528 udp4  136.186.230.58:53     *:*
bind     named      48811 529 udp4  136.186.230.58:53     *:*
bind     named      48811 530 udp4  136.186.230.58:53     *:*
bind     named      48811 531 udp4  136.186.230.58:53     *:*
bind     named      48811 532 udp4  136.186.230.58:53     *:*
bind     named      48811 533 udp4  136.186.230.58:53     *:*
bind     named      48811 534 udp4  136.186.230.58:53     *:*
root     sshd       85567 3  tcp4   136.186.230.58:22     *:*
```

```
root     ntpd       1485  55 udp4   136.186.230.58:123    *:*
?        ?          ?     ?  tcp4   136.186.230.58:80     10.1.16.119:50036
?        ?          ?     ?  tcp4   136.186.230.58:56653  136.186.230.58:80

CHECKING DNS CONFIGURATION
==========================
Querying for (armitagechemicals.unix) using DNS server (136.186.230.58) - Correct answer (136.186.23
0.58)
Using domain server:
Name: 136.186.230.58
Address: 136.186.230.58#53
Aliases:

armitagechemicals.unix has address 136.186.230.58
Test Result: PASSED


ADDING DNS SERVER 136.186.230.58 TO SYSTEM RESOLV.CONF
==============================================

HTTP QUERY http://armitagechemicals.unix
====================================
--2019-08-28 09:10:05--  http://armitagechemicals.unix/
Resolving armitagechemicals.unix (armitagechemicals.unix)... 136.186.230.58
Connecting to armitagechemicals.unix (armitagechemicals.unix)|136.186.230.58|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 303 [text/html]
Saving to: '/tmp/wget_save.emkQGD6c'

     0K                                                       100% 36.0M=0s

2019-08-28 09:10:05 (36.0 MB/s) - '/tmp/wget_save.emkQGD6c' saved [303/303]


====================================
Downloaded Page contents
====================================
<html>
<head><title>Welcome to Armitage Chemicals</title></head>
<body>
<center>
<h1>Welcome to Armitage Chemicals</h1>
<img src=armitagechemicals.jpg>
<p>Armitage Chemicals Headquarters<p>
To contact us, please deliver all letters/emails to the guard at the front gate.<br>

</center>

</body>
</html>
====================================

Checking Downloaded page contents

Test Result: PASS(download) + PASS(contents)


REMOVING DNS SERVER 136.186.230.58 FROM SYSTEM RESOLV.CONF
==============================================
```

```
********************************************************************************
PORTFOLIO TASK RESULT: PASSED
```

15 Communications - Lab Report 2

New Description

Outcome	Weight
ULO7	♦♦♦♦◊

Date	Author	Comment
2019/10/09 21:31	Farinaz Jowkarishasa...	Great report, good work.

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Communications - Lab Report 2

Submitted By:
Joshua Redolfi
101601886
2019/09/30 19:01

Tutor:
Farinaz Jowkarishasaltaneh

September 30, 2019

PCAP Programming: Individual Packet Analysis

Joshua Redolfi
Faculty of Science,
Engineering and, Technology
Swinburne University of Technology
Email: 101601886@student.swin.edu.au

***Abstract*—The goal of this lab was to process .pcap files and report their outputs on a packet-by-packet basis (along with an overall summary). By using the Python and the package dpkt, the contents of .pcap files were analysed. By using the marking script provided by the unit convener, it was found the results are acceptable. Python was used over other options (the C language) due to its data processing ability.**

I. Aim

The aim of this experiment is to create scripts that are capable of processing `.pcap` files and displaying their output without requiring the use of a program such as tcpdump. The output should be easy to understand and only display the needed information.

II. Equipment

This lab was completed on RULE host number 58, to connect to this host PuTTY (release 0.7 for 64 bit Windows) was used. This configuration was done on a computer located at the author's home which ran 64 bit Windows 10. Thus, the computer was not within the Swinburne private network Cisco AnyConnect™was used to connect to the RULE host. The testing of the configuration was achieved through a marking script located at the following address: ruleprimary1.caia.swin.edu.au (requires access to the Swinburne private network to view). To write the scripts, Notepad++ version 7.6.3 was used on the author's computer. On the RULE host itself, Python 2.7 and the package `dpkt` was used to execute the scripts. Finally, to test the script locally, two files called `test1.pcap` and `test2.pcap` were provided alongside the lab handout.

III. Method

As per the lab handout.

IV. Results

The results shown below are the output of the two scripts, and have been compared to the marking script. The marking script returned a postive result for the code created, thus vindicating the results as acceptable.

The first step that was done in the lab was installing Python 2.7. To make it usable on the RULE host it first had to be installed by locating the Python 2.7 directory and thereafter running `make install` to make it so it became the default Python in the system. The Python 2.7 directory Makefile was located in `/usr/ports/lang/python`. By using `make install` it allowed for Python to be executed by using `python` rather than `python2.7`. The script, however, had to be run with `#!/usr/local/bin/python2.7` in the first line rather than `python`.

The packet reader file `mtcpdump` was then created with the ability to read a `.pcap` file from the command line. Through the use of `dpkt` functions, the test `.pcap` files were successfully unpacked and the formatted results were displayed. Below in Table 1 is the summary of the obtained results.

TABLE I
Summary of packet captures.

Attribute	`test1.pcap`	`test2.pcap`
Total Packets	17	234
Total bytes (bytes)	10440	238056
Average packet size (bytes)	614	1017

The outputs as printed by the custom script are shown as Figure 1 for `test1.pcap` and Figure 2 for `test2.pcap`. The output of the latter had 234 packets in it and was reduced in Figure 2 for brevity.

```
student@rule58:~/pcap_lab % ./mtcpdump test1.pcap
[2018-08-16 02:48:10.238506] - 172.16.11.2:61016 -> 172.16.10.2:80 (tcp, len=60, ttl=63)
[2018-08-16 02:48:10.238523] - 172.16.10.2:80 -> 172.16.11.2:61016 (tcp, len=60, ttl=64)
[2018-08-16 02:48:10.288710] - 172.16.11.2:61016 -> 172.16.10.2:80 (tcp, len=52, ttl=63)
[2018-08-16 02:48:10.288738] - 172.16.11.2:61016 -> 172.16.10.2:80 (tcp, len=198, ttl=63)
[2018-08-16 02:48:10.288746] - 172.16.10.2:80 -> 172.16.11.2:61016 (tcp, len=52, ttl=64)
[2018-08-16 02:48:10.288860] - 172.16.10.2:80 -> 172.16.11.2:61016 (tcp, len=770, ttl=64)
[2018-08-16 02:48:10.288865] - 172.16.10.2:80 -> 172.16.11.2:61016 (tcp, len=52, ttl=64)
[2018-08-16 02:48:10.339317] - 172.16.11.2:61016 -> 172.16.10.2:80 (tcp, len=52, ttl=63)
[2018-08-16 02:48:10.339331] - 172.16.11.2:61016 -> 172.16.10.2:80 (tcp, len=52, ttl=63)
[2018-08-16 02:48:10.339563] - 172.16.11.2:61016 -> 172.16.10.2:80 (tcp, len=52, ttl=63)
[2018-08-16 02:48:10.339586] - 172.16.10.2:80 -> 172.16.11.2:61016 (tcp, len=52, ttl=64)
[2018-08-16 02:48:13.537505] - 172.16.11.3:37269 -> 172.16.10.2:12345 (udp, len=1498, ttl=63)
[2018-08-16 02:48:13.549001] - 172.16.11.3:37269 -> 172.16.10.2:12345 (udp, len=1498, ttl=63)
[2018-08-16 02:48:13.560213] - 172.16.11.3:37269 -> 172.16.10.2:12345 (udp, len=1498, ttl=63)
[2018-08-16 02:48:13.571413] - 172.16.11.3:37269 -> 172.16.10.2:12345 (udp, len=1498, ttl=63)
[2018-08-16 02:48:13.582614] - 172.16.11.3:37269 -> 172.16.10.2:12345 (udp, len=1498, ttl=63)
[2018-08-16 02:48:13.591438] - 172.16.10.2:12345 -> 172.16.11.3:37269 (udp, len=1498, ttl=64)

------------------ SUMMARY ------------------

Total packets: 17
Total bytes (bytes): 10440
Average packet size (bytes): 614

---------------------------------------------
```

Fig. 1. The output of test1.pcap.

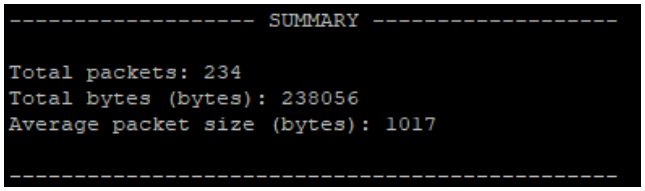


Fig. 2. The summary output of test2.pcap.

V. Discussion

The results of the lab show that `.pcap` files can be easily be processed by python scripting. However, the considerations behind using Python for this task shall be discussed here, along with the evaluation of other methods. Moreover, the script and its functions used to generate the results shall be also be considered.

A. Using Python

Python was one of the two options available for use in the lab, the other being in C (as functions were provided in both of these languages, they were the only two options to efficiently complete the task).

Python was selected due to its data-processing capabilities and the authors familiarity in using it for scripting in previous work experience. C would also require compilation via `./configure` and `make`, therefore requiring more rigorous testing than using Python. In contrast, a Python file can execute immediately after `#!/usr/local/bin/python2.7` is added to the head of the file or with only `python [filename]` as a terminal command.

Conversely, C does have the `libpcap` library which has similar abilities to `dpkt`, but this is negated by the previously mentioned rigid compilation process that a C program must have in Python before it can be used. C also features vectors as a result of its object-oriented nature, however Python features the `.append()` function which allows for an array to be extended as needed. This provides a much simpler data structure that can be expanded as more sets of data are found.

While it was challenging to install Python on the host, the data processing capabilities in Python are much easier to access and manipulate than in C. For the first task (the packet printing) there was no difficulty once the author went over the `dpkt` package documentation and issues with `sys.argv` were resolved. The use of the `.append()` function in Python also allowed for easier management of overall flow data in another scripting task that is outside the scope of this lab report.

B. The Packet Reader

The code for the packet printer is based off that found in [1]. `dpkt` was used to easily allow for the storage of the attributes of the packet and easily extraction of individual fields. The code consists of three functions, one of which is used to execute all others which is called `start`. Excluding the aforementioned function, the first two functions were provided by [1]. Thereafter, there is `inet_to_string`, which is used to convert an `inet` object (such as an IP address) into a string that can be printed.

The final function, `print_packets` is where the packets are actually displayed to the CLI. It first sets the values of the total packet and byte count to zero. Then, using a for loop which is controlled by the values of the buffer and timestamp within the `.pcap` file, it cycles through each packet. From each packet, it increases the value of packet count and adds the total bytes through the use of the `ip.len` flag. It also makes sure that the packet has Ethernet and IP data to check that it can be read. The timestamp is converted to a usable form for printing. Thereafter a control statement tests the value of `ip.p` which holds of the value of the layer four protocol in the packet. A value of 6 means the data is printed to the CLI with the protocol set as TCP, otherwise it is set to UDP if the value is found to be 17.

This control statement is shown below (edited to be displayed on the page). The code was not provided with an else statement as the script would simply ignore that packet. It was also assumed that the probability of a non-TCP or UDP packet that was also an IP packet existing would be close to zero.

```
if ip.p == 6: #tcp
    log_line = '[%s] - %s:%s -> %s:%s \
    (%s, len=%d, ttl=%d)' % \
    (utc_timestamp, inet_to_str(ip.src), \
    tcp.sport, inet_to_str(ip.dst), \
    tcp.dport, "tcp", ip.len, ip.ttl)

    print log_line

    total_bytes_count = \
    total_bytes_count + ip.len

elif ip.p == 17: #udp
    log_line = '[%s] - %s:%s -> %s:%s \
    (%s, len=%d, ttl=%d)' % \
    (utc_timestamp, inet_to_str(ip.src), \
    tcp.sport, inet_to_str(ip.dst), \
    tcp.dport, "udp", ip.len, ip.ttl)

    print log_line

    total_bytes_count = \
    total_bytes_count + ip.len
```

After the for loop finishes, an average packet size in bytes is determined by dividing between both values. Finally, the values of the total bytes and total packets are printed in the format requested by the lab handout (see Figure 2 in the Results section for an example).

C. Future Modifications

To expand on this laboratory, `mtcpdump` could be modified capture live packets entering on an interface. By replacing the file that is inputted with a while loop that reads and processes

each incoming packet a live update to the CLI would be achieved. The use of a pcap object or another library capable of reading a socket should provide this functionality.

VI. Conclusion

In conclusion, this lab was a success as the Python scripts could open the required `.pcap` files and emulate the results found in the marking script. It was also found that the package `dpkt` allows for this task to be completed to with ease.

Python scripting is commonly in the workforce, with the the author in fact having written several Python scripts in the past. Thus labs like this one allow for the development of skills using new packages and the adaption to new requirements. More experience in software development will allow for a greater understanding of scripting in the workforce and how Python can be an invaluable tool.

By installing Python on the RULE host, insight into the process of installing and maintaining programs on FreeBSD has been gained. Once again, as Unix is used in many workplaces, more experience with tasks like this will make work easier in the future. Moreover, by developing skills in Python, the author shall have a greater understanding of scripting in the future.

Acknowledgment

The author would like to thank Farinaz Teneh for tutoring in the laboratory and Jonathan Kua for providing insight into the operation of FreeBSD.

References

[1] D. Song. *Source code for examples.print$_packets$*. 2009. URL: https://dpkt.readthedocs.io/en/latest/_modules/examples/print_packets.html.

16 Practical - Lab 6: PCAP Programming (D)

Lab task - PCAP Programming (Distinction)

Outcome	Weight
ULO5	♦♦♦◊◊

Outcome	Weight
ULO1	♦♦◊◊◊

Outcome	Weight
ULO2	♦♦♦◊◊

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Practical - Lab 6: PCAP Programming (D)

Submitted By:
Joshua Redolfi
101601886
2019/09/09 08:52

Tutor:
Farinaz Jowkarishasaltaneh

September 9, 2019

```
Marking P-Lab-06-PCAP-Programming-D on RULE host rule58
TNE30019/TNE80014 - Laboratory - PCAP Programming
================================================
********************************************************************************
 Portfolio Task: P-Lab-06-PCAP-Programming, Distinction Task

 Develop a tcpsummary program to output a summary of all flows within a tcpdump
 PCAP file

 Tasks:
  Create an executable as /home/student/lab06/tcpsummary
  Executable takes one parameter - name of PCAP file
  Output standard summary block of text for each unique flow in PCAP file
********************************************************************************

Date: 20190909_0851
Last 5 Logons:
student    pts/0    10.1.16.119            Mon Sep  9 08:14   still logged in
student    pts/1    vpn247-80.cc.swin.edu. Sun Sep  8 15:21 - 16:01  (00:40)
student    pts/0    vpn247-80.cc.swin.edu. Sun Sep  8 12:31 - 15:41  (03:09)
student    pts/0    10.1.16.119            Wed Sep  4 19:47 - 20:22  (00:34)
student    pts/0    10.1.16.119            Wed Sep  4 15:43 - 15:45  (00:02)

CHECKING EXECUTABLE
===================
Checking executable (/usr/home/student/pcap_lab/tcpsummary), permissions(rwx??????)...

-rwxrwxrwx 1 1001 1001 4458 Sep 9 08:50 /usr/home/student/pcap_lab/tcpsummary

RUNNING TESTS
=============
============================================================
Running test: /usr/home/student/pcap_lab/tcpsummary test1.pcap

--- Program output --------------------------------------

--------------- PER-FLOW INFO -----------------

Flow: 172.16.11.2:61016-172.16.10.2:80
Protocol: tcp
Start time: 2018-08-16 02:48:10.238506
Total packets: 6
Total bytes: 466

Flow: 172.16.10.2:80-172.16.11.2:61016
Protocol: tcp
Start time: 2018-08-16 02:48:10.238523
Total packets: 5
Total bytes: 986

Flow: 172.16.11.3:37269-172.16.10.2:12345
Protocol: udp
Start time: 2018-08-16 02:48:13.537505
Total packets: 5
Total bytes: 7490

Flow: 172.16.10.2:12345-172.16.11.3:37269
Protocol: udp
Start time: 2018-08-16 02:48:13.591438
Total packets: 1
```

```
Total bytes: 1498

-----------------------------------------------
------------------------------------------------------------

Test Result: PASSED

============================================================
Running test: /usr/home/student/pcap_lab/tcpsummary test2.pcap

--- Program output ---------------------------------------

--------------- PER-FLOW INFO -----------------

Flow: 172.16.11.72:33711-172.16.10.62:5001
Protocol: tcp
Start time: 2018-08-01 03:08:34.575832
Total packets: 156
Total bytes: 234000

Flow: 172.16.10.62:5001-172.16.11.72:33711
Protocol: tcp
Start time: 2018-08-01 03:08:34.576840
Total packets: 78
Total bytes: 4056

-----------------------------------------------
------------------------------------------------------------

Test Result: PASSED

============================================================
Running test: /usr/home/student/pcap_lab/tcpsummary test3.pcap

Program output: Hidden

Test Result: PASSED

********************************************************************************
PORTFOLIO TASK RESULT: PASSED
```

17 Practical - Lab 6: PCAP Programming (P)

Lab task - PCAP Programming

Outcome	Weight
ULO5	♦♦♦◊◊

Outcome	Weight
ULO1	♦♦◊◊◊

Outcome	Weight
ULO2	♦♦♦◊◊

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Practical - Lab 6: PCAP Programming (P)

Submitted By:
Joshua Redolfi
101601886
2019/09/08 12:49

Tutor:
Farinaz Jowkarishasaltaneh

September 8, 2019

```
Marking P-Lab-06-PCAP-Programming-P on RULE host rule58
TNE30019/TNE80014 - Laboratory - PCAP Programming
================================================
********************************************************************************
 Portfolio Task: P-Lab-06-PCAP-Programming, Pass Task

 Develop a mini-tcpdump program to output a summary line for each packet in a
 tcpdump PCAP file

 Tasks:
  Create an executable as /home/student/lab06/mtcpdump
  Executable takes one parameter - name of PCAP file
  Output standard summary line for each packet in PCAP file
  Output summary of total bytes and packets captured, and average packet size
********************************************************************************

Date: 20190908_1248
Last 5 Logons:
student    pts/0    vpn247-80.cc.swin.edu. Sun Sep  8 12:31   still logged in
student    pts/0    10.1.16.119            Wed Sep  4 19:47 - 20:22  (00:34)
student    pts/0    10.1.16.119            Wed Sep  4 15:43 - 15:45  (00:02)
student    pts/15   10.1.16.119            Tue Sep  3 14:41 - 16:20  (01:39)
student    pts/2    vpn247-58.cc.swin.edu. Sun Sep  1 11:59 - 12:11  (00:12)

CHECKING EXECUTABLE
===================
Checking executable (/usr/home/student/pcap_lab/mtcpdump), permissions(rwx??????)...

-rwxrwxrwx 1 1001 1001 3201 Sep 8 12:43 /usr/home/student/pcap_lab/mtcpdump

RUNNING TESTS
=============
============================================================
Running test: /usr/home/student/pcap_lab/mtcpdump test1.pcap

--- Program output ---------------------------------------
[2018-08-16 02:48:10.238506] - 172.16.11.2:61016 -> 172.16.10.2:80 (tcp, len=60, ttl=63)
[2018-08-16 02:48:10.238523] - 172.16.10.2:80 -> 172.16.11.2:61016 (tcp, len=60, ttl=64)
[2018-08-16 02:48:10.288710] - 172.16.11.2:61016 -> 172.16.10.2:80 (tcp, len=52, ttl=63)
[2018-08-16 02:48:10.288738] - 172.16.11.2:61016 -> 172.16.10.2:80 (tcp, len=198, ttl=63)
[2018-08-16 02:48:10.288746] - 172.16.10.2:80 -> 172.16.11.2:61016 (tcp, len=52, ttl=64)
[2018-08-16 02:48:10.288860] - 172.16.10.2:80 -> 172.16.11.2:61016 (tcp, len=770, ttl=64)
[2018-08-16 02:48:10.288865] - 172.16.10.2:80 -> 172.16.11.2:61016 (tcp, len=52, ttl=64)
[2018-08-16 02:48:10.339317] - 172.16.11.2:61016 -> 172.16.10.2:80 (tcp, len=52, ttl=63)
[2018-08-16 02:48:10.339331] - 172.16.11.2:61016 -> 172.16.10.2:80 (tcp, len=52, ttl=63)
[2018-08-16 02:48:10.339563] - 172.16.11.2:61016 -> 172.16.10.2:80 (tcp, len=52, ttl=63)
[2018-08-16 02:48:10.339586] - 172.16.10.2:80 -> 172.16.11.2:61016 (tcp, len=52, ttl=64)
[2018-08-16 02:48:13.537505] - 172.16.11.3:37269 -> 172.16.10.2:12345 (udp, len=1498, ttl=63)
[2018-08-16 02:48:13.549001] - 172.16.11.3:37269 -> 172.16.10.2:12345 (udp, len=1498, ttl=63)
[2018-08-16 02:48:13.560213] - 172.16.11.3:37269 -> 172.16.10.2:12345 (udp, len=1498, ttl=63)
[2018-08-16 02:48:13.571413] - 172.16.11.3:37269 -> 172.16.10.2:12345 (udp, len=1498, ttl=63)
[2018-08-16 02:48:13.582614] - 172.16.11.3:37269 -> 172.16.10.2:12345 (udp, len=1498, ttl=63)
[2018-08-16 02:48:13.591438] - 172.16.10.2:12345 -> 172.16.11.3:37269 (udp, len=1498, ttl=64)

------------------- SUMMARY -------------------

Total packets: 17
Total bytes (bytes): 10440
Average packet size (bytes): 614
```

```
-------------------------------------------
------------------------------------------------------------

Test Result: PASSED

============================================================
Running test: /usr/home/student/pcap_lab/mtcpdump test2.pcap

--- Program output ---------------------------------------
[2018-08-01 03:08:34.575832] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.576818] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.576840] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.577830] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.578830] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.578863] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.579829] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.580817] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.580849] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.581830] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.582831] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.582863] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.583830] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.584818] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.584853] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.585855] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.586855] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.586927] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.587854] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.588842] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.588911] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.589854] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.590895] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.590953] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.591899] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.592885] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.592942] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.593898] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.594898] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.594956] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.595898] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.596893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.596948] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.597872] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.598855] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.598882] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.599855] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.600883] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.600928] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.601898] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.602897] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.602942] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.603899] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.604884] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.604930] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.605896] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.606898] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.606943] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.607900] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.608883] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.608940] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
```

```
[2018-08-01 03:08:34.609896] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.610897] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.610954] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.611872] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.612860] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.612904] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.613872] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.614873] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.614916] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.615872] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.616859] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.616902] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.617868] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.618871] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.618913] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.619867] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.620884] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.620941] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.621897] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.622897] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.622954] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.623897] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.624884] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.624940] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.625898] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.626876] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.626921] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.627855] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.628842] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.628913] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.629891] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.630893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.630934] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.631892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.632880] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.632921] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.633892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.634892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.634933] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.635892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.636880] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.636920] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.637892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.638867] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.638898] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.639893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.640854] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.640885] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.641893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.642869] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.642900] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.643894] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.644856] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.644886] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.645892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.646867] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.646897] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.647892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.648839] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.648888] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
```

```
[2018-08-01 03:08:34.649892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.650871] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.650907] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.651893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.652880] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.652914] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.653893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.654893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.654928] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.655892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.656879] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.656912] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.657892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.658893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.658926] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.659892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.660878] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.660911] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.661891] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.662893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.662926] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.663892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.664879] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.664913] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.665892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.666892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.666926] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.667892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.668879] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.668912] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.669892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.670892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.670925] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.671892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.672878] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.672912] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.673891] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.674892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.674925] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.675891] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.676878] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.676911] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.677894] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.678892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.678924] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.679893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.680878] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.680911] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.681893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.682893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.682926] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.683891] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.684880] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.684913] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.685891] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.686893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.686926] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.687893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.688879] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.688913] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
```

```
[2018-08-01 03:08:34.689893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.690892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.690926] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.691893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.692880] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.692913] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.693900] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.694891] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.694924] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.695893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.696877] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.696910] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.697893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.698891] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.698924] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.699890] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.700881] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.700914] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.701892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.702894] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.702927] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.703892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.704878] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.704911] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.705893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.706893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.706926] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.707893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.708880] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.708913] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.709890] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.710893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.710926] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.711891] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.712881] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.712914] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.713894] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.714893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.714926] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.715891] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.716878] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.716912] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.717893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.718892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.718925] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.719891] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.720879] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.720913] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.721893] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.722891] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.722926] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.723868] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.724878] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.724911] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.725892] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.726868] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.726887] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
[2018-08-01 03:08:34.727867] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.728857] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.728873] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)
```

```
[2018-08-01 03:08:34.729868] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.730868] - 172.16.11.72:33711 -> 172.16.10.62:5001 (tcp, len=1500, ttl=63)
[2018-08-01 03:08:34.730884] - 172.16.10.62:5001 -> 172.16.11.72:33711 (tcp, len=52, ttl=64)

------------------- SUMMARY -------------------

Total packets: 234
Total bytes (bytes): 238056
Average packet size (bytes): 1017

-----------------------------------------------
-------------------------------------------------------------

Test Result: PASSED

=============================================================
Running test: /usr/home/student/pcap_lab/mtcpdump test3.pcap

Program output: Hidden

Test Result: PASSED

*******************************************************************************
PORTFOLIO TASK RESULT: PASSED
```

18 Practical - Lab 7: NMap (C)

Lab task - Nmap (Credit)

Outcome	Weight
ULO5	♦♦♦◊◊

Outcome	Weight
ULO4	♦♦◊◊◊

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Practical - Lab 7: NMap (C)

Submitted By:
Joshua Redolfi
101601886
2019/09/09 11:04

Tutor:
Farinaz Jowkarishasaltaneh

September 9, 2019

No spitzensparken for the new RULE system

19 Practical - Lab 7: NMap (P)

Lab task - Nmap

Outcome	Weight
ULO5	♦♦♦◊◊

Outcome	Weight
ULO4	♦♦◊◊◊

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Practical - Lab 7: NMap (P)

Submitted By:
Joshua Redolfi
101601886
2019/09/10 15:32

Tutor:
Farinaz Jowkarishasaltaneh

September 10, 2019

```
student@rule58:~ % nmap -PS rule21.caia.swin.edu.au
Starting Nmap 7.70 ( https://nmap.org ) at 2019-09-09 10:22 EST
Nmap scan report for rule21.caia.swin.edu.au (136.186.230.21)
Host is up (0.000058s latency).
rDNS record for 136.186.230.21: labnmap.unix
Not shown: 65525 closed ports
PORT    STATE SERVICE
7/tcp   open  echo
13/tcp  open  daytime
21/tcp  open  ftp
22/tcp  open  ssh
23/tcp  open  telnet
79/tcp  open  finger
80/tcp  open  http
110/tcp open  pop3
143/tcp open  imap
55246/tcp open unknown

Nmap done: 1 IP address (1 host up) scanned in 6.30 seconds
```

20 Practical - Programming Tutorial

Successful completion of programming task in tutorial

Outcome	Weight
ULO4	♦♦◊◊◊

Outcome	Weight
ULO2	♦♦♦◊◊

Outcome	Weight
ULO1	♦◊◊◊◊

Date	Author	Comment
2019/10/11 17:37	Jonathan Kua	Well done!

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Practical - Programming Tutorial

Submitted By:
Joshua Redolfi
101601886
2019/10/10 17:18

Tutor:
Farinaz Jowkarishasaltaneh

October 10, 2019

```c
#include <stdio.h>
#include <stdlib.h>

#define __USE_GNU
#include <string.h>

#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>

#include <errno.h>

#define WWW_PORT    80
#define BUF_SIZE    10000

//-----------------------------------------------------------------------
// Print an error message to stderr and terminate the program immediately
void
printerror(char *pcErrStr)
{
    fprintf(stderr, "ERROR: %s\n\n", pcErrStr);
    exit(1);
}

//-----------------------------------------------------------------------
// Input parameters are the name of the server to query and a pointer to a
// sockaddr_in structure to fill.  The function performs a lookup of the
// specified Internet name and stores the address along with the web port
// number (WWW_PORT) in the provided sockaddr_in structure
void
FillAddress(char *pcURL, struct sockaddr_in *psAddress)
{
    struct hostent      *psHostDetails;

    if ((psHostDetails = gethostbyname(pcURL)) == NULL) printerror("Can't determine
    ↪  address of website");

    memcpy(&(psAddress->sin_addr), psHostDetails->h_addr_list[0], 4);
    psAddress->sin_family = AF_INET;
    psAddress->sin_port = htons(WWW_PORT);

    printf("Server details:\n");
    printf("  Server:\t%s\n", pcURL);
    printf("  IP Address:\t%s\n", inet_ntoa(psAddress->sin_addr));
}
//-----------------------------------------------------------------------
int
main(int iArgC, char **ppcArgV)
{
    struct sockaddr_in  sAddress;
    char                *pcWWWServer, *pcPage, *pcPos;
    int sockfd, bytes, sent, received, total;
```

```
53
54      if (iArgC != 2) printerror("Command Line requires one parameter = website");
55
56      // If the provided URL begins with "http://" then set pcWWWServer to point
57      // to the next character otherwise set it to point to the entire provided URL
58      if (pcWWWServer = strstr(ppcArgV[1], "http://")) pcWWWServer+= 7; else
        ↪  pcWWWServer = ppcArgV[1];
59
60      // If there is a "/" in the URL (after the http:// has been stripped) then
61      // replace it with a 0 so the string terminates as just the name.  Create
62      // a new string pointer (pcPage) to be a copy of the remaining string OR
63      // just "/" if no page details have been provided
64      if (pcPos = strstr(pcWWWServer, "/"))
65      {
66          pcPage = strdup(pcPos);
67          pcPos[0] = 0;
68      } else
69          pcPage = strdup("/");
70
71      // Call FillAddress() to fill sAddress with the IP address details (and port
72      // number 80) of the requested URL
73      FillAddress(pcWWWServer, &sAddress);
74
75      FILE * fp;
76      fp = fopen ("data.txt", "w");
77      fclose(fp);
78      fp = fopen ("data.txt", "a");
79      int iListenSocket;
80      printf("OPENING SOCKET\n");
81      if ((iListenSocket = socket(AF_INET, SOCK_STREAM, 0)) < 0)
82      {
83          fprintf(stderr, "ERROR: Cannot create socket\n");
84          exit(1);
85      }
86
87      printf("CONNECTING SOCKET\n");
88       sAddress.sin_family = AF_INET;
89      sAddress.sin_port = htons(WWW_PORT);
90      if (connect(iListenSocket, (struct sockaddr *) &sAddress, sizeof(sAddress)) < 0)
91      {
92          fprintf(stderr, "ERROR: Unable to connect to port\n");
93          exit(1);
94      }
95
96      printf("LISTENING...\n");
97
98      listen(iListenSocket, 5);
99
100
101         int                 iServerSocket, iBytesRead;
102         struct sockaddr_in  sServerAddress;
103         socklen_t           iAddressLength = sizeof(sServerAddress);
104         char                pcBuffer[BUF_SIZE];
```

```
105         char    *message = "GET /\r\n";
106 
107         printf("CONNECTION ACCEPTED\n");
108 
109         printf("SENDING GET REQUEST\n");
110 
111         if (send(iListenSocket, message, strlen(message), 0) < 0)
112         {
113             fprintf(stderr, "ERROR: Sending data to client\n");
114         }
115         printf("READING DATA\n");
116         while(iBytesRead = (recv(iListenSocket, pcBuffer, BUF_SIZE , 0)))
117         {
118             printf(pcBuffer,"\n");
119             fprintf (fp, "%s", pcBuffer);
120         }
121 
122         printf("CONNECTION CLOSED\n");
123 
124     return 0;
125 }
```

21 Theory - Jails

Brief report summarising jail operations

Outcome	Weight
ULO7	♦♦♦◊◊

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Theory - Jails

Submitted By:
Joshua Redolfi
101601886
2019/10/10 08:45

Tutor:
Farinaz Jowkarishasaltaneh

October 10, 2019

Virtualisation and Jails

Joshua Redolfi
Email: 101601886@student.swin.edu.au
AIMING FOR DISTINCTION

***Abstract*—Virtualisation is achieved in many ways in the FreBSD system, and in this report two types are discussed. The first is jails and the other is virtualisation through the use of bhyve. They are found to have different uses and abilities for development and system administration. Thereafter other methods of virtualisation in other operating systems are discussed.**

I. Introduction

Jails and virtualisation are major components of the FreeBSD operating system. While related, they exhibit different characteristics and uses.

II. Jails versus Virtualisation

On FreeBSD, there is two main options for dividing the operating system and achieving virtualisation. Firstly, there is jails, where processes are kept within an isolated, unique root directory but still within the FreeBSD operating system [1] [2]. On the other hand, the forms of virtualisation available on FreeBSD include bhyve, which will be focused on for the purposes of this report [3]. bhyve is a virtual machine manager designed for machines running FreeBSD with an Intel or AMD processor bhyve [3].

Jails allow for a set of superuser privileges to be given to a user for only a certain area of the operating system/file space and essentially modify the root directory for a certain set of processes [1] [2]. This keeps them in a container where they cannot access files outside of the designated root directory [2]. Jails are based the `chroot(2)` call, an earlier method of limited compartmentalisation that has several shortcomings (such as the inability to prevent interference to processes outside their compartment) [4].

For virtualisation, bhyve is software within FreeBSD that allows for a variety of operating systems to be run under it [3]. For example, through the use of UEFI, bhyve can be used to run Windows 10 [3]. bhyve also supports setting the number of virtual CPUs and the memory required and will operate until it is commanded to reset [5].

Both bhyve and jails are now part of the FreeBSD distribution (jails from version 4.0 and bhyve from 12.0) [2] [6]. Both of them require no installation as they are now part of the FreeBSD operating system [2] [5]. A jail can simply be created by editing a file called `jails.conf` (which specifies the parameters for jails in FreeBSD) and the command `service jail start [name]` [2]. On the other hand, executing bhyve with the image `/my/image` with 1 GB of RAM and 2 virtual CPUs would look like `bhyve -c 2 -s 2,virtio-blk,/my/image -m 1G` *(shortened for brevity, see manual for full command)* [6]. It is notable that bhyve has command line options to allocate memory and virtual CPUs, whereas jails have only had a limited ability to do this since FreeBSD 9.0 [1] [6]. Finally the security of jails has increased significantly since their first introduction and even more so than its predecessor chroot [1]. Examples include jail-aware device drivers and restriction of system calls [1]. According to [7] bhyve was actually excluded because the authors could not find any major security risks in bhyve.

Table I summarises the features of jails and bhyve.

TABLE I
Comparison of Jails and Virtualisation

Jails	Virtualisation
The jail is a contained area in the FreeBSD OS [1].	bhyve has memory access restricted by shadow tables [8].
Jails, as a part of the FreeBSD OS, requires no installation [1].	Like jails, bhyve is a part of FreeBSD as it is in the manual [5]
The jail is stored in a part of storage, as subset of the FreeBSD system [4]	A bhyve kernel image is stored in a virtual disk with the file extension `.img` [3].
A jail can restrict system calls and access to device drivers [1].	A 2016 study found that bhyve had no significant security issues [7].

In reflection, bhyve and jails provide different sorts of virtualisation for FreeBSD. One is not better than the other as they provide different services and are both part of the OS. In the workplace, I believe knowledge of both would be needed to be successful in operating a variety of services.

III. Using Jails and Virtualisation

Jails and bhyve have vastly different uses, the most obvious being that bhyve has the advantage of being able to run many different sorts of operating systems and choose how many virtual CPUs/memory the container can have [3] [6]. Based on this, it would be better to use jails for simple tasks that only require a FreeBSD process that needs to be contained due to it needing more lax security than the rest of the system. It also allows for an easy division of the operating system, as seen in [9] where it is used to allow many students access to a Unix system to learn how to use it. This allowed for many students to use a small number of FreeBSD hosts [9]. In contrast, bhyve has been used in applications such as hosting the monitoring server Graphite where it is used for its ability to provide a "summary of virtual machine (VM) operation with an emphasis on its kernel resource utilization but few

with insights into its relative performance to the host" [10]. Thus using bhyve in a virtual computing environment were resources are restricted and any operating system is required would be best. This would be very formidable when combined with the option to customise the number of CPUs and disk space it has [6].

Thus, it is best to use jails where the application can run on FreeBSD and there is no need to restrict the system resources that the application will consume. They are also known for their performance and reliability [2]. On the other hand, bhyve is more customisable based on the image loaded and the extensive parameters provided [3] [6].

Based on what was said in the reflection for the last section. I believe both are have their different attributes and will both be required to be a successful network engineer.

IV. Jail-like Tools in Other Operating Systems

Jail-like and virtualisation technologies also exist outside of FreeBSD, such as Docker. Unlike Jails which is inside the FreeBSD OS, they are separate software either maintained as proprietary and open source software [1]. A brief description of each selected tool is provided and their features are then compared.

A. Linux Containers

Linux Containers (LXC) is a container solution that consists of user space tools that it can take advantage of because it only uses features integrated into the Linux terminal [7] [1]. It allows for users to manage containers via the use of Linux tools such as `chroot`, like jails, of which it is a descendant [11] [12]. Unlike jails, LXC works by having separate kernel namespaces, whereas the former used the same kernel namespace for each container [1]. This increases the complexity of the system, but allows for more flexibility than using a structure-based container like jails [1].

B. Docker

Docker is an open-source container system designed for developers and system admins for most Unix/Linux platforms [13]. It has been described as a light weight virtual machine and creates a contained environment based off LXC [13] [14]. Essentially Docker isolates its processes entirely, but still uses the same kernel [14]. Despite this, Docker has been found to have several security flaws, mostly involving access control and data handling [7]. This is in comparison to bhyve, which the authors have noted does not have any known major security issues [7]. This is also in comparison to jails, which is considered outdated by [7].

TABLE II
Comparison of Jail-like Tools

Docker	Docker can run on any Unix operating system.	Uses the same kernel but otherwise completely contained [14].	Renowned for being lightweight [13] [14].
Linux Containers	LXC is designed for the Linux kernel [11]	Uses a separate kernel namespace [1].	LXC was found to be the least efficient container [15].

Finally, in reflection the different tools such as Docker show that jails is the foundation on which many tools are built, but in some cases it has been superseded. Thus, I believe it is important to learn Docker in particular, which is used in many workplaces such as my own.

V. Conclusion and Reflection

From this, it can be seen that there are many different solutions to containerisation and virtualisation for many operating systems, but in particular for Unix-like ones. In my opinion, the jail subsystem's power is in the fact that it comes pre-installed as a part of FreeBSD and is simple to set up. However, I do not believe it is useful for any secure application (such as data storage) where options like Docket and bhyve (which has little known security issues) [7]. For example, at my work where services need to be secure, Docker is used always. But in the case of personal use or for education like it is used at Swinburne University, I think it is incredibly useful. Thus, while jails is apparently outdated, from the standpoint of simplicity and mediocre security, it will do a fine job in most applications [7].

Acknowledgment

The author would like to thank Jonathan Kua for providing insight into the operation of FreeBSD.

References

[1] E. Reshetova et al. "Security of OS-level virtualization technologies". In: *Nordic Conference on Secure IT Systems*. Springer. 2014, pp. 77–93.

[2] "FreeBSD Developers' Handbook". In: 2000. Chap. 14. URL: https://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/jails.html.

[3] *bhyve, the BSD Hypervisor*. 2018. URL: https://wiki.freebsd.org/bhyve.

[4] P-H. Kamp and R. N. M. Watson. *Jails: Confining the omnipotent root*. 2000. URL: https://docs.freebsd.org/44doc/papers/jail/jail.html.

[5] N. Natu and P. Grehan. *bhyve*. 2018. URL: https://www.freebsd.org/cgi/man.cgi?bhyve.

[6] *bhyve(8) - FreeBSD Manual Pages*. 2019. URL: https://www.freebsd.org/cgi/man.cgi?query=bhyve&apropos=0&sektion=8&manpath=FreeBSD+12.0-RELEASE+and+Ports&arch=default&format=html.

[7] A. Gkortzis, S. Rizou, and D. Spinellis. "An empirical analysis of vulnerabilities in virtualization technologies". In: *2016 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*. IEEE. 2016, pp. 533–538.

[8] N. Natu and P. Grehan. "Nested Paging in bhyve". In: *The FreeBSD Project, http://people. freebsd. org/neel/bhyve/bhyve—nested—paging. pdf* (2014).

[9] G. J. Armitage. “Maximising Student Exposure to Unix Networking using FreeBSD Virtual Hosts”. In: *Center for Advanced Internet Architecture, Swinburne University of Technology* (2003).

[10] M. Dexter. “Visualizing Unix: Graphing bhyve, ZFS and PF with Graphite”. In: *AsiaBSDCon 2014* (2014), p. 7.

[11] *LXC - Introduction*. URL: https://linuxcontainers.org/lxc/introduction/.

[12] D. Bernstein. “Containers and cloud: From lxc to docker to kubernetes”. In: *IEEE Cloud Computing* 1.3 (2014), pp. 81–84.

[13] C. Anderson. “Docker [software engineering]”. In: *IEEE Software* 32.3 (2015), pp. 102–c3.

[14] D. Merkel. “Docker: lightweight linux containers for consistent development and deployment”. In: *Linux Journal* 2014.239 (2014), p. 2.

[15] M.G. Xavier et al. “Performance evaluation of container-based virtualization for high performance computing environments”. In: *2013 21st Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*. IEEE. 2013, pp. 233–240.

22 Communications - Lab Report 3

New Description

Outcome	Weight
ULO7	♦♦♦♦◊

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Communications - Lab Report 3

Submitted By:
Joshua Redolfi
101601886
2019/10/14 19:12

Tutor:
Farinaz Jowkarishasaltaneh

October 14, 2019

Flow Analysis via TCPDump

Joshua Redolfi
Faculty of Science,
Engineering and, Technology
Swinburne University of Technology
Email: 101601886@student.swin.edu.au

***Abstract*—In this report a .pcap provided by the unit convener is analysed. Firstly the various options within tcpdump are tested to find out their capabilities. Information is also provided on the output that tcpdump provides. Thereafter the .pcap file is analysed and conclusions are about the four different sorts of traffic in it (SSH, HTTP, POP3 and FTP).**

I. Aim

To determine the types of traffic in a provided `.pcapng` file and draw conclusions on the characteristics of traffic produced by protocols such as SSH and HTTP. This will be achieved through the use of tcpdump and other Unix commands such as grep to manipulate the output of the former.

II. Equipment

This lab was completed on RULE host number 58, to connect to this host PuTTY (release 0.7 for 64 bit Windows) was used. This configuration was done on the author's laptop (Acer Aspire E 15) which runs Windows 10. Provided as a part of the lab materials was `lab7.pcapng` which contained the data that needed to be analysed. The data was then copied into Google Sheets and Microsoft Excel 365 where it was then graphed. The program Lightshot (version 5.4.0.1) was then used to capture and save the graphs from Excel.

III. Method

As per the lab handout.

IV. Results

In this results section firstly the commands used in tcpdump are discussed and thereafter the analysis of `lab7.pcapng` will be shown.

To analyse the file a variety of commands were used. To see the ACSII data inside the packets the command `tcpdump -r lab7.pcapng -XX` was used extensively. The stream was also filtered by the use of `grep` to find port numbers and protocols, through `tcpdump -r lab7.pcapng 'tcp port [service]'` could have easily been used. Below is a sample of the output by running `tcpdump -r lab7.pcapng` which is the very last line of the .pcap file.

```
1159861586.356266 IP
nakter2.caia.swin.edu.au.58052 >
hhoang2.caia.swin.edu.au.ssh: Flags [.],
ack 4811, win 33303,
options [nop,nop,
TS val 600714053 ecr 8267478],
length 0
```

Each line means the following:

```
TIMESTAMP IP
SOURCE_HOST_NAME.SOURCE_PORT >
DESTINATION_HOST_NAME.DEST_PORT: Flags [.],
ack ACK_NUMBER, win WINDOW_SIZE,
options [PACKET_OPTIONS],
length PAYLOAD_SIZE
```

With this knowledge, the .pcap file was deciphered.

A. Results from the .pcap File

The file `lab7.pcapng` was then analysed via the use of a variety of the aforementioned tcpdump commands. Table 1 shows the two hosts that interacted over the course of approximately 69 seconds (the timestamps of the packets range from `17:45:17.195008` to `17:46:26.356266` UTC+10).

TABLE I
The hosts used

IP Address	136.186.229.139	136.186.229.138
Hostname	nakter2.caia.swin.edu.au	hhoang2.caia.swin.edu.au

Four services are shown below in Table 2, along with the number of packets and (unidirectional) flows detected. It is notable that the average size of a HTTP packet is much larger than that of any of the other three protocols found in the .pcap file. One email was found to be received over the course of the capture and POP3 shows how little packets it required to do this. The order that these protocols were used in is shown by the table.

TABLE II
A summary of the flows and services detected

Service	**Packets detected**	**Flows detected**	**Total data sent (bytes)**	**Average packet size (bytes)**
HTTP	117	10	68119	582
POP3	25	2	1402	56
FTP	62	4	1319	21
SSH	68	2	6298	93

B. Notable Events

Figure 1 shows a sample of the tcpdump output of the file. Through looking through output with the `-XX` the contents of the packets were analysed. Notable events that happened in the are listed below.

```
18:46:26.355164 IP nakter2.caia.swin.edu.au.58052 > hhoang2.caia.swin.edu.au.ssh
: Flags [P.], seq 2088:2120, ack 4810, win 33304, options [nop,nop,TS val 600714
052 ecr 8267478], length 32
18:46:26.355211 IP nakter2.caia.swin.edu.au.58052 > hhoang2.caia.swin.edu.au.ssh
: Flags [F.], seq 2120, ack 4810, win 33304, options [nop,nop,TS val 600714052 e
cr 8267478], length 0
18:46:26.355241 IP hhoang2.caia.swin.edu.au.ssh > nakter2.caia.swin.edu.au.58052
: Flags [.], ack 2121, win 33288, options [nop,nop,TS val 8267478 ecr 600714052]
, length 0
18:46:26.356053 IP hhoang2.caia.swin.edu.au.ssh > nakter2.caia.swin.edu.au.58052
: Flags [F.], seq 4810, ack 2121, win 33288, options [nop,nop,TS val 8267478 ecr
 600714052], length 0
18:46:26.356266 IP nakter2.caia.swin.edu.au.58052 > hhoang2.caia.swin.edu.au.ssh
: Flags [.], ack 4811, win 33303, options [nop,nop,TS val 600714053 ecr 8267478]
, length 0
```

Fig. 1. A sample of the output.

FTP Login Failure: The client attempts to log into FTP but fails on the first attempt because they use the valid account `anonymous` (Figure 2).

```
18:45:43.422714 IP nakter2.caia.swin.edu.au.51688 > hhoang2.caia.swin.edu.au.ftp
], length 16
        0x0000:  000e a649 f155 0002 b3e7 bc21 0800 4540  ...I.U.....!..E@
        0x0010:  0044 2965 4000 4006 3484 88ba e58b 88ba  .D)e@.@.4.......
        0x0020:  e58a c9e8 0015 ec43 45c4 f554 21af 8018  .......CE..T!...
        0x0030:  8218 bf32 0000 0101 080a 23cd 8392 007e  ...2......#....~
        0x0040:  160b 5553 4552 2061 6e6f 6e79 6d6f 7573  ..USER.anonymous
        0x0050:  0d0a                                     ..
18:45:43.423668 IP hhoang2.caia.swin.edu.au.ftp > nakter2.caia.swin.edu.au.51688
2], length 29
        0x0000:  0002 b3e7 bc21 000e a649 f155 0800 4510  .....!...I.U..E.
        0x0010:  0051 1563 4000 4006 48a9 88ba e58a 88ba  .Q.c@.@.H.......
        0x0020:  e58b 0015 c9e8 f554 21af ec43 45d4 8018  .......T!..CE...
        0x0030:  8218 99ba 0000 0101 080a 007e 1611 23cd  ...........~..#.
        0x0040:  8392 3533 3020 5573 6572 2061 6e6f 6e79  ..530.User.anony
        0x0050:  6d6f 7573 2075 6e6b 6e6f 776e 2e0d 0a    mous.unknown...
```

Fig. 2. FTP user 'anonymous' being rejected.

FTP File Download: The client eventually connects to the FTP server and navigates several directories as seen by the transmission of FTP commands over the interface. Eventually, he chooses to download the image `/usr/home/lslowman/Blkmarble-desktopImage.jpg` from the server.

POP3 Email Delivery: At 17:08:30 (UTC+10), The client receives an email via POP3, which is shown below.

```
Date: Tue, 3 Oct 2006 17:08:30 +1000 (EST)
From: Warren Harrop
    <wharrop@wharrop2.
    caia.swin.edu.au>
Message-Id:
    <200610030708.k9378Ums005778@
    wharrop2.caia.swin.edu.au>
To: lslowman@wharrop2.caia.swin.edu.au
Subject: Labs?

Lawrence Slowman, you have yet to hand
in any labs. Is there any reason for
this? Are you still enrolled?

Warren Harrop
```

HTTP 404 Error: Twice the client attempts to connect to a webpage that doesn't exist and is presented with a 404 error page. Figure 3 shows the packet that downloads the 404 error page.

```
18:45:45.204293 IP hhoang2.caia.swin.edu.au.http > nakter2.caia.swin.edu.au.6315
904], length 581
        0x0000:  4500 0279 156b 4000 4006 4689 88ba e58a  E..y.k@.@.F.....
        0x0010:  88ba e58b 0050 f6b5 5d3e 1a3f 1d8a 96a5  .....P..]>.?....
        0x0020:  8018 8218 bd7f 0000 0101 080a 007e 16c3  ............~..
        0x0030:  23cd 8a88 4854 5450 2f31 2e31 2034 3034  #...HTTP/1.1.404
        0x0040:  204e 6f74 2046 6f75 6e64 0d0a 5365 7276  .Not.Found..Serv
        0x0050:  6572 3a20 7468 7474 7064 2f32 2e32 3562  er:.thttpd/2.25b
```

Fig. 3. The HTTP server transmitting a 404 error.

HTTP Successful Connection: The client successfully connects to the website `wharrop2.caia.swin.edu.au` and downloads the web page 'Armitage Chemicals'.

Plain-text Password: FTP transmits its password in the clear, which allows for tcpdump to pick it up as `mypassword`.

V. Discussion

The results of the lab show that tcpdump can be used to analyse captures both in real time and later on via the `-r` option. It was also found that each protocol has a different characteristics, with HTTP standing out in particular due to its high data usage.

A. Using TCPDump

While tcpdump was installed previously on the RULE host, to install it could easily be done via finding the most recent package through the command `pkg search tcpdump`. Once the appropriate package has been located, it can be installed through `pkg install [package name]`.

Moreover, the name tcpdump is also misleading, as while it does dump TCP data, it also can track UDP and other layer 4 protocols easily.

To read the .pcap file provided, a variety of `tcpdump`commands were used to manipulate the output of a packet capture. The following options include some of these found useful for this task.

To search for only HTTP traffic (which is on port 80):
`tcpdump -r lab7.pcapng 'tcp port 80'`
This would be useful to filter only for HTTP packets or any other protocol based on port number.

To print the contents of the packet in hexadecimal format:
`tcpdump -r lab7.pcapng -xx`
This would be useful to the see the contents of the packet.

To print the contents of the packet in ASCII format:
`tcpdump -r lab7.pcapng -XX`
Like above, but even more so because plain text characters in the packet can be made out.

To print all the details of the packet:
`tcpdump -r lab7.pcapng -v`

This would be useful in seeing all the options within the packet and detecting events like a three way handshake.

To prevent tcpdump from resolving IP addresses into host names (and thus showing the IP addresses):

```
tcpdump -r lab7.pcapng -n
```

This would be useful to see the actual IP addresses.

It was found that tcpdump also has to be run as root to do a live capture (none of which are present in this lab however) to allow for access to the socket. This is because root access to the virtual network devices in `/dev` are needed [1].

B. Analysis of Protocols

Overall, major differences in the protocols could be seen. Primarily, the disproportionate size of a HTTP packet in comparison to the other three protocols. This shows the amount of data that a HTTP connection has to consume in comparison to other protocols that are built around issuing commands or small amounts of data (emails in POP3 and files in FTP).

Firstly, HTTP is shown to consume much more data than any of the other protocols, this is likely because HTTP is downloading both a HTTP document and some images. While FTP does download an image, the number and size of packets imply that it is much smaller.

The difference of the encryption of the protocols versus SSH is striking. The HTTP page and the POP3 email could be read as plain text just by analysing the packets. Moreover, the FTP password and commands could be determined by the same method. In contrast, in an attempt to learn what commands the client had used in an SSH session proved useless as they were encrypted. The only part of the SSH connection that could read was the Diffie-Hellman exchange, which despite transmitting numbers related to the keys used, it is almost impossible to figure out what key is being used. This shows the importance of secure versions of these protocols (such as SFTP and HTTPS). These secure protocols prevent third party users from using tools such as tcpdump and Wireshark from analysing network traffic and gaining illegitimate access to information.

Each protocol also had a difference in connection time, which was determined by how many flows there were. Protocols like HTTP and POP3 would open a socket via the three way handshake, transfer data and then immediately close with the `FIN` flag. On the other hand, SSH and FTP kept sessions open for a much longer time, ranging well into the seconds (note that webpages and emails can be downloaded almost instantaneously). This could be seen in Table 2 in the Results section by the small number of flows from both protocols.

C. Quality of Service

From this, it could be seen how Quality of Service could impact a network, as HTTP uses a large amount of data in a short amount of time, which could impact other services. For example, if there were a VoIP service, it could be negatively impacted by a large amount of HTTP traffic. The other protocols, however, would need much more flows to negatively affect the in comparison. Further investigation would thus best involve more HTTP services such as streaming to see the size and length of the flows that it generates.

VI. Conclusion

In conclusion, the laboratory was a success as many conclusions were drawn about the nature of the traffic in the .pcap file provided. Moreover, the many of the flags in tcpdump were learnt, even if if the data provided was not in real time as is usually done. The characteristics of the four protocols (HTTP, FTP, POP3, SSH) were analysed too. It was found that HTTP has much larger packets in comparison to the other protocols and with the exception of SSH, all these basic protocols transmit in plain-text. Thus, the importance of secure protocols is reinforced.

In reflection, network engineers should take note of tools like this and their outputs as hey show how different each protocol is terms of transmission rates. This shows that not all network traffic is the same and thus networks must be planned around what sort of protocols will be flowing through them in addition to the data rate. It reinforces the concept of QoS which will be an important issue once in the workforce, especially if working in a large data-centre or ISP.

By using these tools and seeing how powerful they are at detecting information that should be kept secret such as passwords and emails. It shows that network engineers must be cautious about using insecure protocols like HTTP and instead should use HTTPS. The laboratory also reinforced why in the industry protocols SSH and high security measures are needed, such as in the author's own workplace, as tcpdump/Wireshark on a wireless interface will detect all traffic that is being transmitted.

Acknowledgment

The author would like to thank Farinaz Teneh for tutoring in the laboratory and Jonathan Kua for providing insight into the operation of FreeBSD.

References

[1] *Manpage of PCAP*. 2019. URL: https://www.tcpdump.org/manpages/pcap.3pcap.html.

23 Practical - Lab 8: TCPDump (C)

Lab task - TCPDump (Credit)

Outcome	Weight
ULO2	♦◊◊◊◊

Outcome	Weight
ULO5	♦♦◊◊◊

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

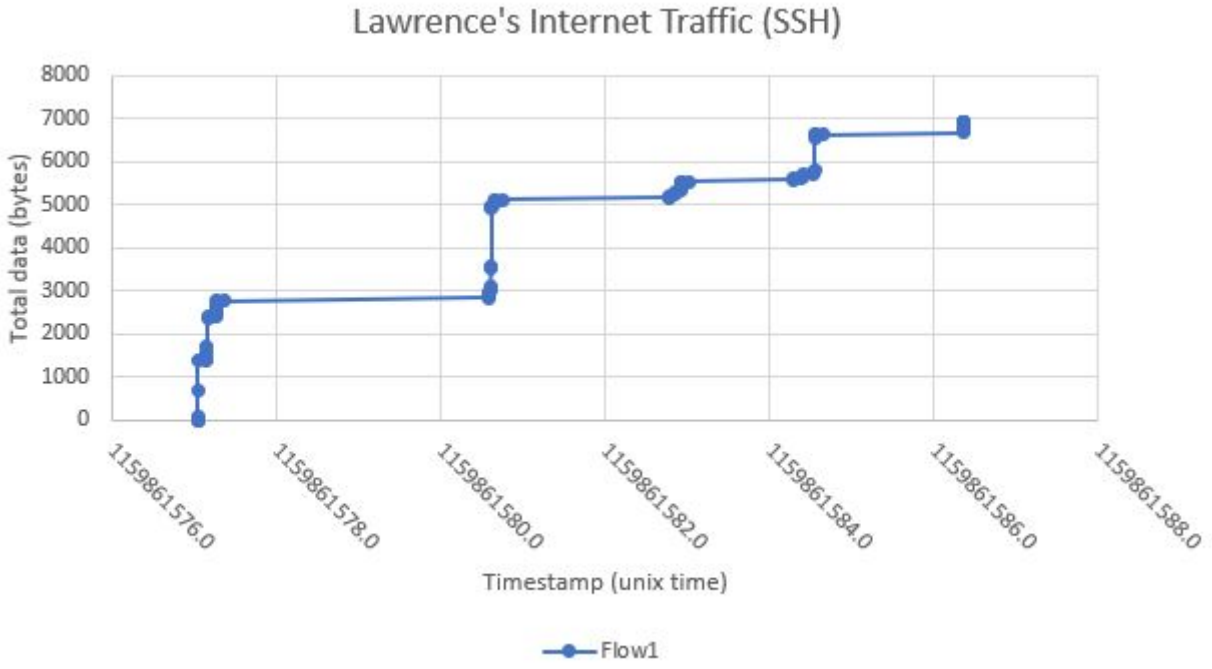
Practical - Lab 8: TCPDump (C)

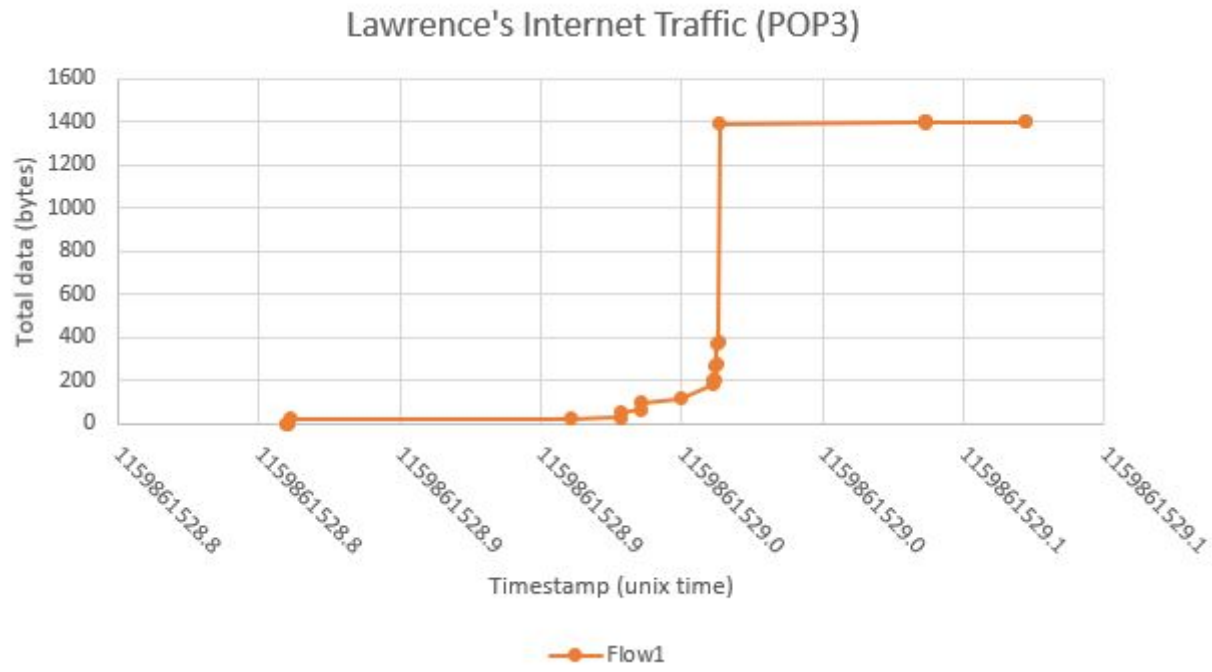
Submitted By:
Joshua Redolfi
101601886
2019/10/14 10:41

Tutor:
Farinaz Jowkarishasaltaneh

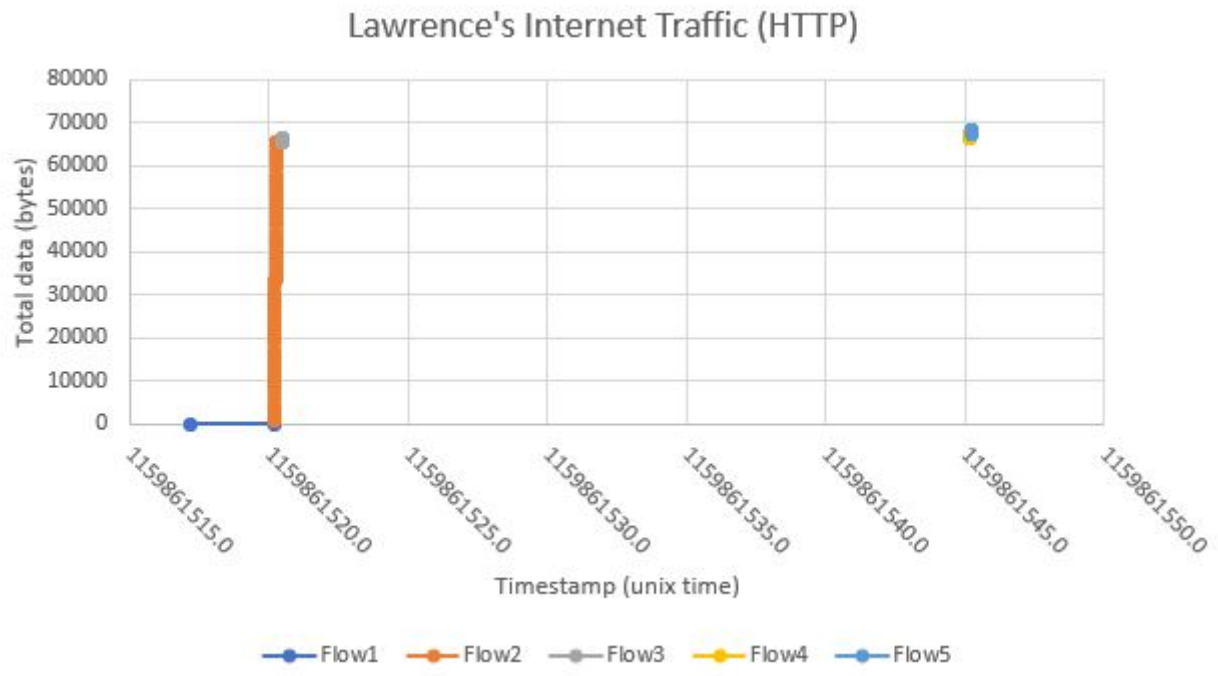
October 14, 2019

JOSHUA REDOLFI
101601886
NOTE THAT THIS HAS BI-DIRECTIONAL FLOWS

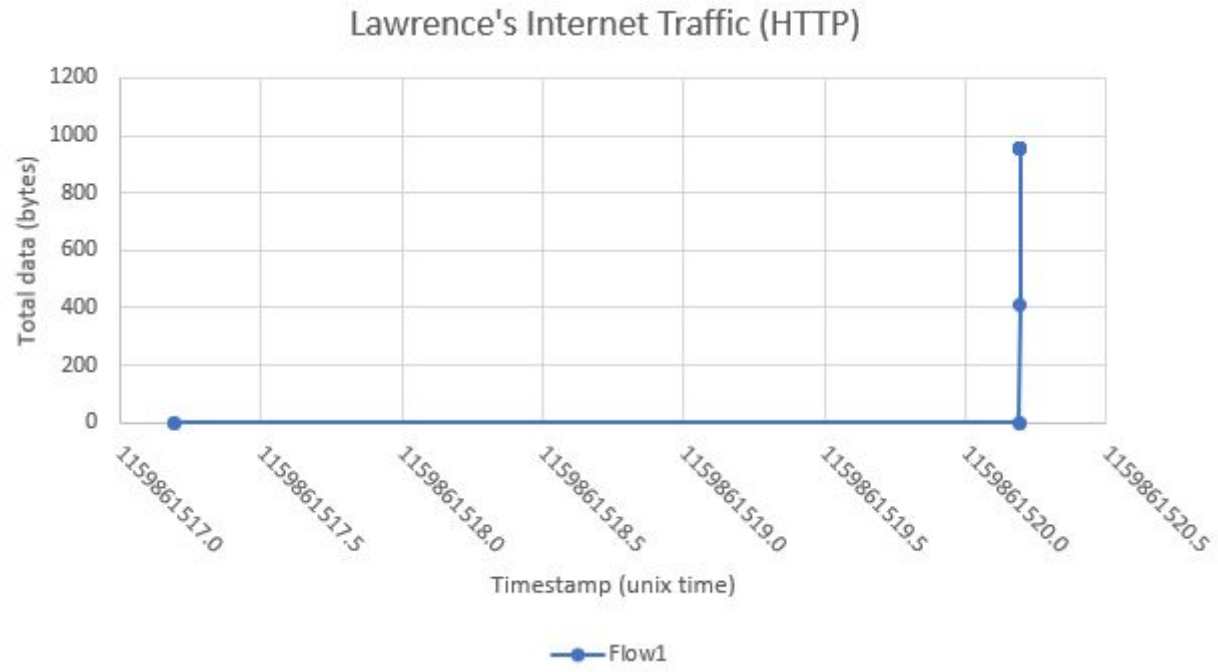




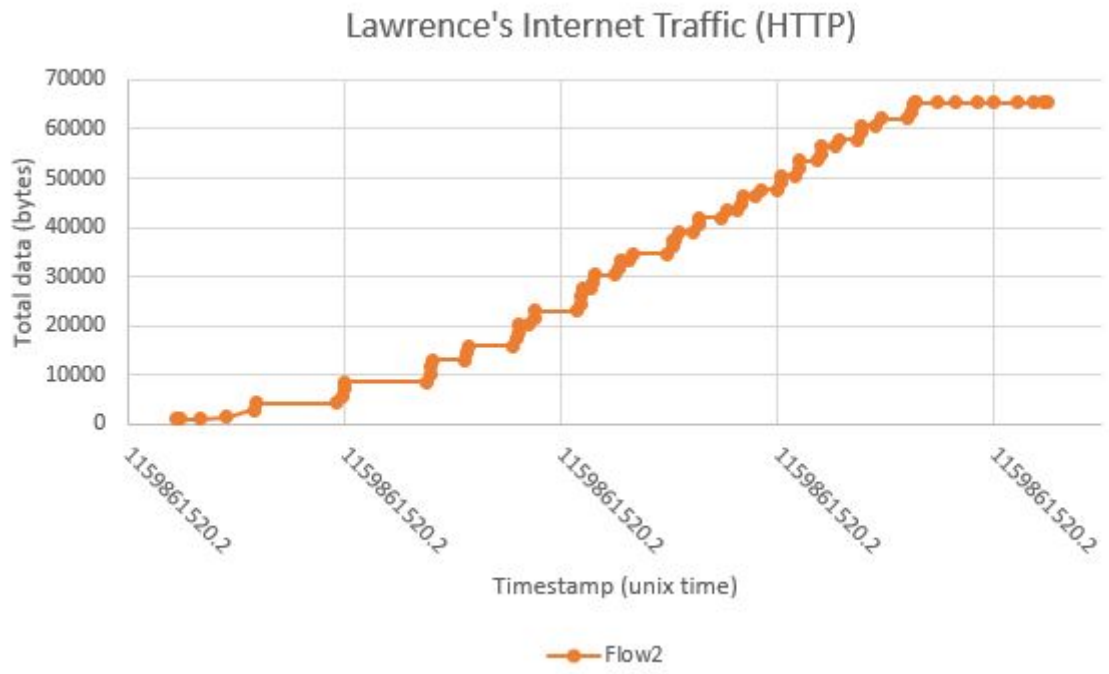
OVERALL HTTP FLOW:
As each HTTP flow varies in size greatly, I have arranged them with an overall flow, then each one individually.


Lawrence's Internet Traffic (HTTP)
Total data (bytes)
80000
70000
60000
50000
40000
30000
20000
10000
0
1159861515.0
1159861520.0
1159861525.0
1159861530.0
1159861535.0
1159861540.0
1159861545.0
1159861550.0
Timestamp (unix time)
Flow1
Flow2
Flow3
Flow4
Flow5

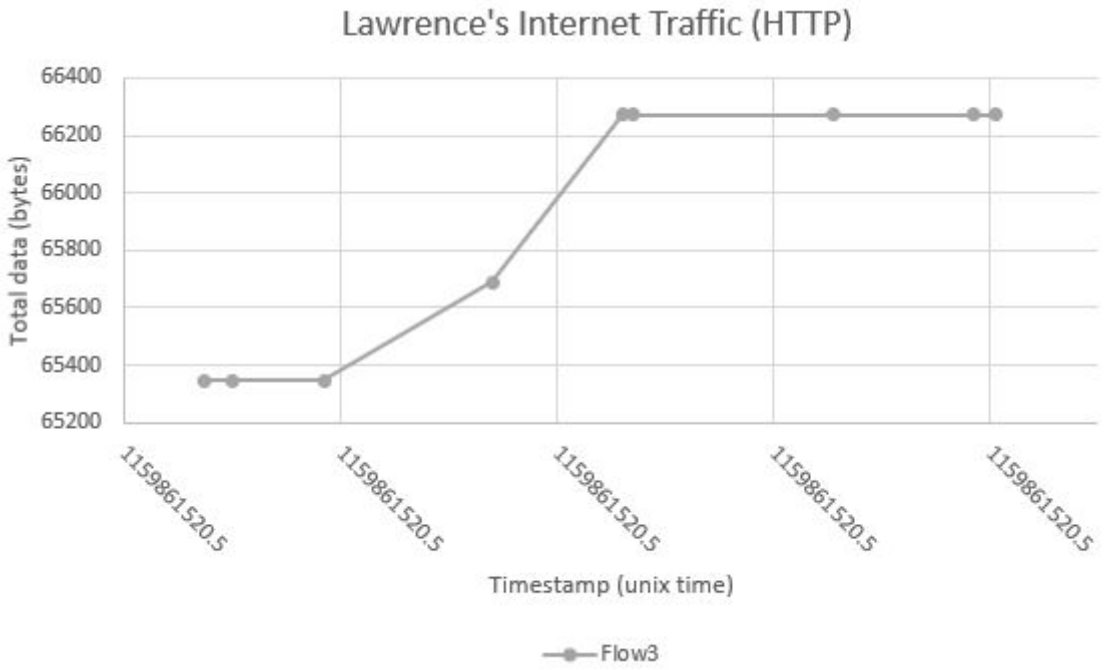
FLOW 1:


Lawrence's Internet Traffic (HTTP)
Total data (bytes)
1200
1000
800
600
400
200
0
1159861517.0
1159861517.5
1159861518.0
1159861518.5
1159861519.0
1159861519.5
1159861520.0
1159861520.5
Timestamp (unix time)
Flow1

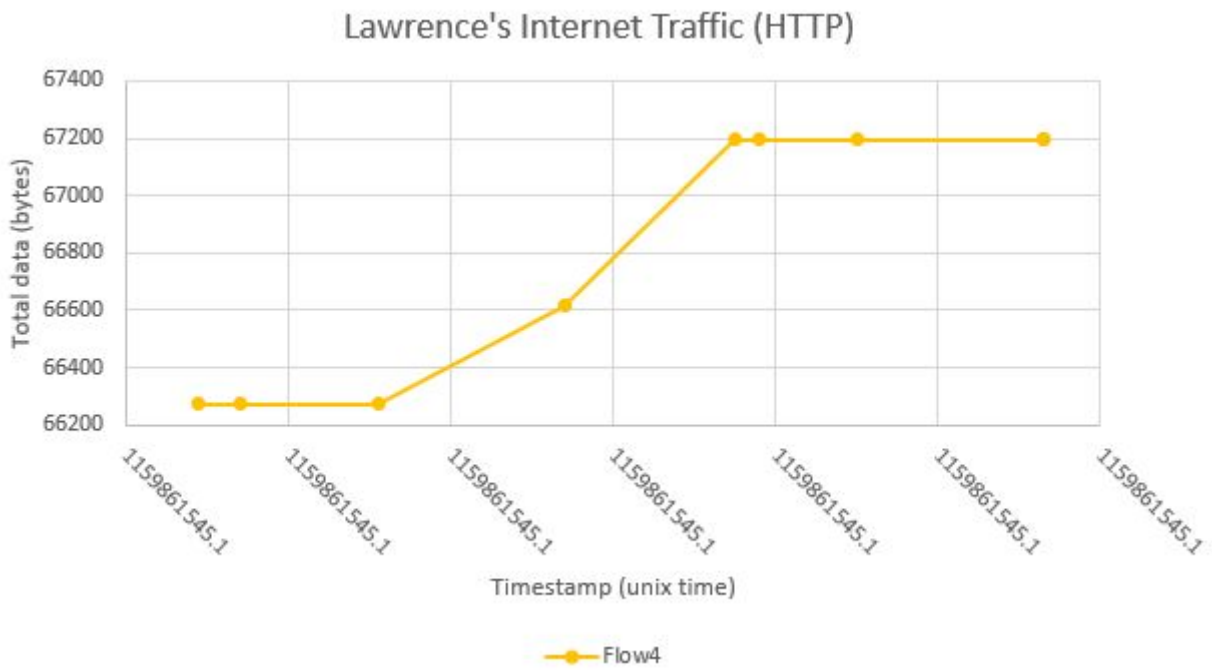
FLOW 2:


Lawrence's Internet Traffic (HTTP)
70000
60000
50000
40000
30000
20000
10000
0
Total data (bytes)
1159861520.2
1159861520.2
1159861520.2
1159861520.2
1159861520.2
Timestamp (unix time)
Flow2

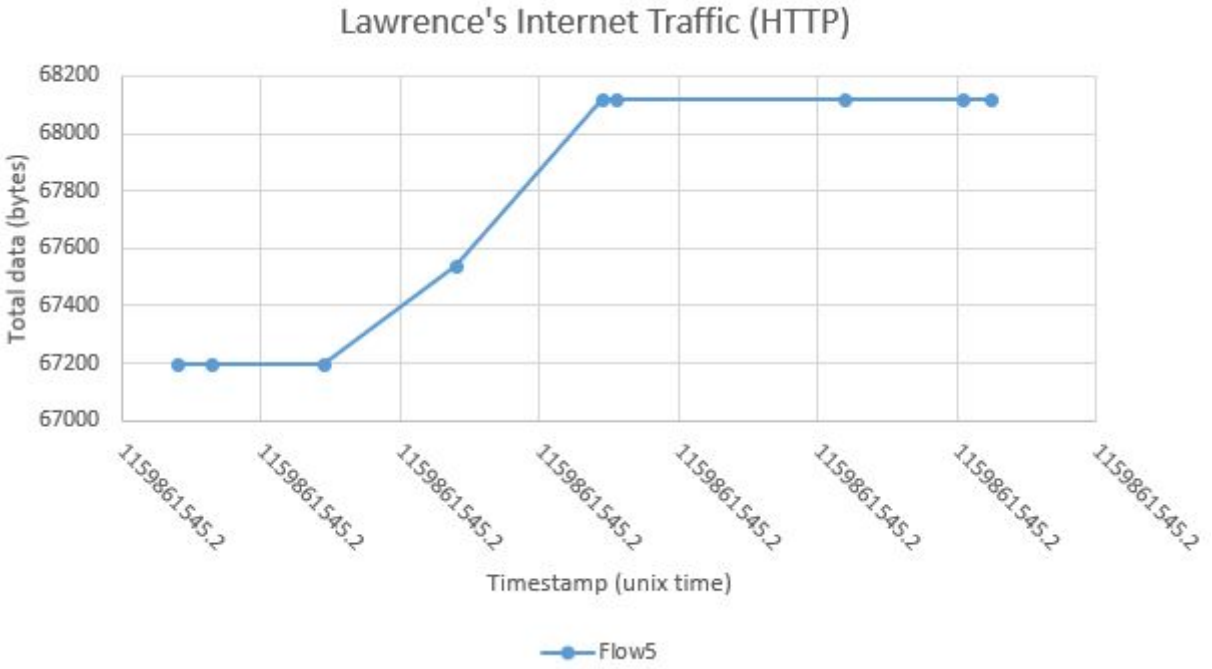
FLOW 3:

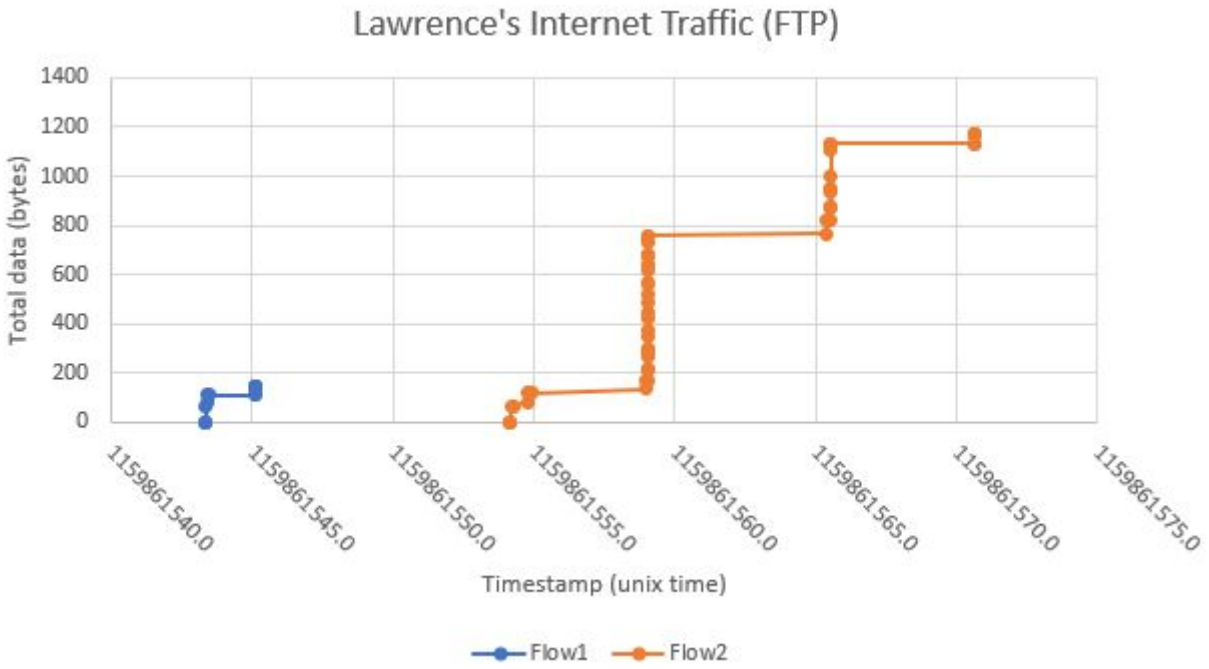

Lawrence's Internet Traffic (HTTP)
66400
66200
66000
65800
65600
65400
65200
Total data (bytes)
1159861520.5
1159861520.5
1159861520.5
1159861520.5
1159861520.5
Timestamp (unix time)
Flow3

FLOW 4:


Lawrence's Internet Traffic (HTTP)
Total data (bytes)
67400
67200
67000
66800
66600
66400
66200
1159861545.1
Timestamp (unix time)
Flow4

FLOW 5:


Lawrence's Internet Traffic (HTTP)
Total data (bytes)
68200
68000
67800
67600
67400
67200
67000
1159861545.2
Timestamp (unix time)
Flow5


Lawrence's Internet Traffic (FTP)
Total data (bytes)
1400
1200
1000
800
600
400
200
0
1159861540.0
1159861545.0
1159861550.0
1159861555.0
1159861560.0
1159861565.0
1159861570.0
1159861575.0
Timestamp (unix time)
Flow1
Flow2

24 Practical - Lab 8: TCPDump (D)

Lab task - TCPDump (Distinction)

Outcome	Weight
ULO2	♦◊◊◊◊

Outcome	Weight
ULO5	♦♦◊◊◊

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Practical - Lab 8: TCPDump (D)

Submitted By:
Joshua Redolfi
101601886
2019/10/04 19:59

Tutor:
Farinaz Jowkarishasaltaneh

October 4, 2019

JOSHUA REDOLFI
101601886

a) What do the graphs say about the nature of the traffic generated?
b) If there is more than one unique flow, provide insight into what is happening?
c) Is the traffic bursty/smooth? What does this mean in terms of impact on a typical home gateway?

A. SSH
   a. SSH produces a small amount of traffic (on average about 2000 bytes a second) over a longer period of time than the other protocols presented here. This traffic occurs in small bursts.
   b. There is one flow, which increases at a consistent rate every time there is a new command.
   c. The traffic is the smoothest of all the protocols provided, with almost increasing in a linear fashion. This should a present, small but constant stream of traffic through a home router than uses little resources.

B. POP3
   a. POP3 generates no traffic except when an email is sent. The size of the packets sent are small, and in the case of the graph, occur over a very limited time.
   b. There is one flow, which consists of many small packets and what appears to be one large one.
   c. Excluding the infrequentess of emails, the traffic is fairly bursty, with small packets occurs either side of a very large packet that presumably contains the email content. Due to the small size, it would likely be handled well by a home router but bursty traffic otherwise means the router will have poor efficiency.

C. HTTP
   a. They show that there are several instances of little bandwidth being used (small packets), but then there is occasional large bursts of traffic that happen periodically (this is likely HTTP downloading a big file from a webpage).
   b. There are four flows, all of which appear to be very small. Each one is likely a different webpage and thus the size of each flow is proportional to the size of the webpage that has to be downloaded. The first flow is probably the same page being refreshed.
   c. The traffic is very bursty, with a large spike in the second flow. This would mean that most of the time the resources of a home gateway aren’t being used, but when they are used, it will be resource intensive.

D. FTP
   a. They show multiple spikes on top of each other in each flow. They likely show many small packets to initiate the connection,then large file downloads.
   b. There are two flows, both showing how long a FTP connection can hold (the second lasts well over 20 seconds). They appear to be two separate FTP connections to different servers.
   c. The traffic is very bursty as it occurs over a long period of time and consists of large bursts. The effect on the home router could be, once again, that its resources are often unused except when a large burst comes through.

25 Practical - Lab 8: TCPDump (P)

Lab task - TCPDump

Outcome	Weight
ULO2	♦◊◊◊◊

Outcome	Weight
ULO5	♦♦◊◊◊

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Practical - Lab 8: TCPDump (P)

Submitted By:
Joshua Redolfi
101601886
2019/10/08 14:57

Tutor:
Farinaz Jowkarishasaltaneh

October 8, 2019

JOSHUA REDOLFI
101601886

Part C:

1. 17:45:17.195008 to 17:46:26.356266 = 69.161258 seconds
2. They are:
    a. On port 80, which is a HTTP service.
    b. On port 110, which is POP3 or email.
    c. On port 21, which is FTP.
    d. On port 22, which is SSH (can be used for SFTP too).
3. wharrop2.caia.swin.edu.au
4. From 17:45:43.360929 to 17:46:05.642999 Attempted to log into the account anonymous which was unknown. No password was entered.
5. /usr/home/lslowman/Blkmarble-desktopImage.jpg
6. You can't read SSH commands as they are encrypted!
7. The email was sent by Warren Harrop at 5:08:30

```
Date: Tue, 3 Oct 2006 17:08:30 +1000 (EST)
From: Warren Harrop <wharrop@wharrop2.caia.swin.edu.au>
Message-Id: <200610030708.k9378Ums005778@wharrop2.caia.swin.edu.au>
To: lslowman@wharrop2.caia.swin.edu.au
Subject: Labs?

Lawrence Slowman, you have yet to hand in any labs. Is there any reason for
this? Are
you still enrolled?

Warren Harrop

.
```

8. mypassword

26 Practical - Lab 9: Samba (P)

Lab task - Samba

Outcome	Weight
ULO5	♦♦♦◊◊

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Practical - Lab 9: Samba (P)

Submitted By:
Joshua Redolfi
101601886
2019/10/08 17:42

Tutor:
Farinaz Jowkarishasaltaneh

October 8, 2019

```
Marking P-Lab-09-Samba-P on RULE host rule58
TNE30019/TNE80014 - Laboratory - Samba
=====================================
********************************************************************************
 Portfolio Task: P-Lab-09-Samba, Pass Task

 Configure your rule host (rule58) with Samba running private home shares

 Tasks:
  Two Samba users: samba(password=samba) and autocollector(password=autocollector)
  Samba share available to both users: //rule58/home
  Each share maps to /home/&ltusername>
  Users access their own (respective) home directories only
********************************************************************************

Date: 20191008_1742
Last 5 Logons:
student    pts/1    10.1.16.119           Tue Oct  8 17:34   still logged in
root       pts/2                          Tue Oct  8 17:32 - 17:32  (00:00)
root       pts/2                          Tue Oct  8 16:55 - 17:32  (00:37)
student    pts/12   10.1.16.119           Tue Oct  8 14:39 - 16:09  (01:30)
student    pts/3    10.1.16.119           Tue Oct  8 10:19 - 10:30  (00:10)

RULE host Socket information
tempuser sshd       29875 3  tcp4   136.186.230.58:22     10.1.16.119:56971
tempuser sshd       29875 7  tcp4   136.186.230.58:6010   *:*
root     sshd       29870 3  tcp4   136.186.230.58:22     10.1.16.119:56971
root     smbd       28780 35 tcp4   136.186.230.58:445    *:*
root     smbd       28780 36 tcp4   136.186.230.58:139    *:*
root     nmbd       28776 16 udp4   136.186.230.58:137    *:*
root     nmbd       28776 17 udp4   136.186.230.58:138    *:*
root     nmbd       28776 18 udp4   136.186.230.58:137    *:*
root     nmbd       28776 19 udp4   136.186.230.58:137    *:*
root     nmbd       28776 20 udp4   136.186.230.58:138    *:*
root     nmbd       28776 21 udp4   136.186.230.58:138    *:*
www      httpd      51029 3  tcp4   136.186.230.58:80     *:*
www      httpd      51028 3  tcp4   136.186.230.58:80     *:*
www      httpd      51027 3  tcp4   136.186.230.58:80     *:*
www      httpd      51026 3  tcp4   136.186.230.58:80     *:*
www      httpd      51025 3  tcp4   136.186.230.58:80     *:*
www      httpd      51021 3  tcp4   136.186.230.58:80     *:*
www      httpd      51018 3  tcp4   136.186.230.58:80     *:*
www      httpd      51016 3  tcp4   136.186.230.58:80     *:*
www      httpd      86694 3  tcp4   136.186.230.58:80     *:*
www      httpd      86693 3  tcp4   136.186.230.58:80     *:*
root     httpd      86689 3  tcp4   136.186.230.58:80     *:*
bind     named      48811 20 tcp4   136.186.230.58:53     *:*
bind     named      48811 21 tcp4   136.186.230.58:953    *:*
bind     named      48811 512 udp4  136.186.230.58:53     *:*
bind     named      48811 513 udp4  136.186.230.58:53     *:*
bind     named      48811 514 udp4  136.186.230.58:53     *:*
bind     named      48811 515 udp4  136.186.230.58:53     *:*
bind     named      48811 516 udp4  136.186.230.58:53     *:*
bind     named      48811 517 udp4  136.186.230.58:53     *:*
bind     named      48811 518 udp4  136.186.230.58:53     *:*
bind     named      48811 519 udp4  136.186.230.58:53     *:*
bind     named      48811 520 udp4  136.186.230.58:53     *:*
bind     named      48811 521 udp4  136.186.230.58:53     *:*
bind     named      48811 522 udp4  136.186.230.58:53     *:*
bind     named      48811 523 udp4  136.186.230.58:53     *:*
```

```
bind     named      48811 524 udp4  136.186.230.58:53     *:*
bind     named      48811 525 udp4  136.186.230.58:53     *:*
bind     named      48811 526 udp4  136.186.230.58:53     *:*
bind     named      48811 527 udp4  136.186.230.58:53     *:*
bind     named      48811 528 udp4  136.186.230.58:53     *:*
bind     named      48811 529 udp4  136.186.230.58:53     *:*
bind     named      48811 530 udp4  136.186.230.58:53     *:*
bind     named      48811 531 udp4  136.186.230.58:53     *:*
bind     named      48811 532 udp4  136.186.230.58:53     *:*
bind     named      48811 533 udp4  136.186.230.58:53     *:*
bind     named      48811 534 udp4  136.186.230.58:53     *:*
root     sshd       85567 3  tcp4   136.186.230.58:22     *:*
root     ntpd       1485  55 udp4   136.186.230.58:123    *:*

TEMPORARILY EDITING nsmb.conf
=============================
Setting access to RULE58
  Username: SAMBA
  Password: samba

RUNNING SAMBA ACCESS TEST
=========================

Checking access to //samba@RULE58/home...

Mounting (//samba@RULE58/home) to (/var/tmp/rule/collect/mnt)...

Checking Mount Status...
//SAMBA@RULE58/HOME on /var/tmp/rule/collect/mnt (smbfs)

Creating file (auto_collect_test)...

Checking file access (read)...
=< contents >==============================================
[RULE58]
addr=rule58

[RULE58:SAMBA]
password=samba
============================================================

Checking file properties on actual RULE Host...
-rwxr--r--  1 1002  1002  52 Oct  8 17:42 /usr/jails/unix/lab/rule58/usr/home/samba/auto_collect_tes
t

Checking contents of written file...

Unmounting (/var/tmp/rule/collect/mnt)


Testing summary:
  Mounting share:           Pass
  File creation:            Pass
  Accessing mounted file:   Pass
  File contents correct:    Pass
  File location on host:    Pass
  File Owner permissions:   Pass
  File Group permissions:   Pass
  File Public permissions:  Pass
```

```
REINSTATING OLD nsmb.conf
=========================

TEMPORARILY EDITING nsmb.conf
=============================
Setting access to RULE58
  Username: AUTOCOLLECTOR
  Password: autocollector

RUNNING SAMBA ACCESS TEST
=========================

Checking access to //autocollector@RULE58/home...

Mounting (//autocollector@RULE58/home) to (/var/tmp/rule/collect/mnt)...

Checking Mount Status...
//AUTOCOLLECTOR@RULE58/HOME on /var/tmp/rule/collect/mnt (smbfs)

Creating file (auto_collect_test)...

Checking file access (read)...
=< contents >==============================================
[RULE58]
addr=rule58

[RULE58:AUTOCOLLECTOR]
password=autocollector
============================================================

Checking file properties on actual RULE Host...
-rwxr--r--  1 1003  1003  68 Oct  8 17:42 /usr/jails/unix/lab/rule58/usr/home/autocollector/auto_col
lect_test

Checking contents of written file...

Unmounting (/var/tmp/rule/collect/mnt)


Testing summary:
  Mounting share:           Pass
  File creation:            Pass
  Accessing mounted file:   Pass
  File contents correct:    Pass
  File location on host:    Pass
  File Owner permissions:   Pass
  File Group permissions:   Pass
  File Public permissions:  Pass

REINSTATING OLD nsmb.conf
=========================

*******************************************************************************
PORTFOLIO TASK RESULT: PASSED
```

27 Communications - Project Presentation

New Description

Outcome	Weight
ULO7	♦♦♦◊◊

Outcome	Weight
ULO3	♦♦◊◊◊

Outcome	Weight
ULO6	♦◊◊◊◊

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Communications - Project Presentation

Submitted By:
Joshua Redolfi
101601886
2019/10/29 16:43

Tutor:
Farinaz Jowkarishasaltaneh

October 29, 2019

Implementing OpenLDAP

Joshua Redolfi | 101601886

Aiming for distinction

Current Situation

- IT currently has to manually add accounts.
- Updating password will only affect one computer.
- Security risk if a password is compromised.

What is LDAP?

- Lightweight Directory Access Protocol
- Stores accounts and passwords on a single server.
- Client queries server for account details.

Benefits of LDAP

- LDAP has a centralised account server.
- This means that only one account has to be set up.
- Changing password will also apply to every computer.

Financial Benefits of LDAP

- Assuming $20/hr for IT Technician.
- Setting up accounts on 10 computers is (10 * 0.25 hours * $20 = $50)
- With LDAP, setting up an account takes 10 min (0.2 * $20 = $4)

Risk Analysis

	LOW	MEDIUM	HIGH
LOW	LOW	LOW	MEDIUM
MEDIUM	LOW	MEDIUM	HIGH
HIGH	MEDIUM	HIGH	HIGH

Risk Event	Probability	Impact Rating	Score
Old password is compromised by a hacker, who then uses it to obtain access to via a host which hasn't had its password changed.	MEDIUM	HIGH	HIGH
Account is not set up a computer, resulting in the inability for a user to log in.	LOW	LOW	LOW
Access to accounts is recorded locally, meaning that there is centralised place to track logins. This means that malicious use cannot be detected via a central source.	MEDIUM	MEDIUM	MEDIUM

Potential cost of current solution: thousands in lost revenue from a data breach

LDAP Risks:

- Single point of failure with one server.
- Transmitting passwords in the clear.

Demonstration

- Demonstration of the following functions (with and without LDAP):
  - Adding and removing user accounts via our current method and using LDAP scripts.

28 Communications - Speaker Reflection 2

Reflection of presentation of second Industry Invited Speaker

Outcome	Weight
ULO7	♦♦♦◊◊

Outcome	Weight
ULO2	♦♦◊◊◊

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Communications - Speaker Reflection 2

Submitted By:
Joshua Redolfi
101601886
2019/10/22 17:06

Tutor:
Farinaz Jowkarishasaltaneh

October 22, 2019

Unix for Telecommunications

Portfolio Task – C-Speaker-Reflection-Form

Name: Joshua Redolf; 101601886 Date: 22/10/19

1) Reflect on one thing you learnt today that you were completely surprised by

- that the world Economic Forum rates cyber attacks as the 4th biggest global risk.
- the trend of data science being used in security

2) How do you think todays presentation changed the way you look at your career following graduation?

I really want to avoid security, but it was emphaised that everyone should be involved in it - even network engineers!

PASS

29 Theory - Test 2 (P)

In class test run during tutorial in week 11 of semester

Outcome	Weight
ULO7	♦♦◊◊◊

Outcome	Weight
ULO2	♦♦♦◊◊

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Theory - Test 2 (P)

Submitted By:
Joshua Redolfi
101601886
2019/10/29 08:39

Tutor:
Farinaz Jowkarishasaltaneh

October 29, 2019

Theory Test 2

Due No due date **Points** 29 **Questions** 18
Available 24 Oct at 0:00 - 25 Oct at 23:59 2 days **Time limit** 40 Minutes

Instructions

Trial attempt at Canvas version of test

This quiz was locked 25 Oct at 23:59.

Attempt history

	Attempt	Time	Score
LATEST	Attempt 1	21 minutes	22 out of 29

Submission details:

Time:	21 minutes
Current score:	22 out of 29
Kept score:	22 out of 29

30 Practical - Project

project submission

Outcome	Weight
ULO7	♦♦♦♦♦

Outcome	Weight
ULO6	♦♦♦♦♦

Outcome	Weight
ULO3	♦♦♦♦♦

Outcome	Weight
ULO2	♦♦♦◊◊

Outcome	Weight
ULO4	♦♦♦♦♦

Outcome	Weight
ULO5	♦♦♦♦♦

Outcome	Weight
ULO1	♦♦♦♦◊

Swinburne University of Technology

Unix for Telecommunications

Doubtfire Submission

Practical - Project

Submitted By:
Joshua Redolfi
101601886
2019/11/20 17:56

Tutor:
Farinaz Jowkarishasaltaneh

November 20, 2019

LDAP Project Documentation

Joshua Redolfi
Faculty of Science,
Engineering and, Technology
Swinburne University of Technology
Email: 101601886@student.swin.edu.au

Semester 2 2019

Abstract

This document describes the operation of the LDAP (Light-weight Directory Access Protocol) on the RULE host rule241.caia.swin.edu.au. It contains information on how the server was configured and how to configure clients to connect to it. Thereafter, account and script details are provided to show how the system has been automated. Finally, the operation of an LDAP web monitor is provided. **This document contains all the documentation required for the project**.

Contents

1 Introduction

This documentation contains information of the LDAP server set up on the following hosts:

Server Address: 136.186.230.241 (rule241.caia.swin.edu.au)
Client Address: 136.186.230.240 (rule240.caia.swin.edu.au)

2 Location of Files and CLI Scripts

This contains the locations of the files that were edited and created for the use of LDAP. It also has the locations of all the scripts for adding, editing and removing user accounts, along with the ability for users to change their passwords.

2.1 Server Locations

SLAPD config: `/usr/local/etc/openldap/slapd.conf`
Initial .ldif file: `/usr/local/etc/openldap/twoacc.ldif`
LDAP config: `/usr/local/etc/openldap/ldap.conf`
PAM-LDAP config: `/usr/local/etc/ldap.conf`
NSS-LDAP config: `/usr/local/etc/nss_ldap.conf`
NSS config: `/etc/nsswitch.conf`
PAM config (system): `/etc/pam.d/system`
PAM config (ssh): `/etc/pam.d/sshd`
PAM config (su): `/etc/pam.d/su`
Apache config: `/usr/local/etc/apache24/http.conf`
Web Monitor: `/usr/local/www/apache24/data/index.cgi`

2.2 Client Locations

LDAP config: `/usr/local/etc/openldap/ldap.conf`
PAM-LDAP config: `/usr/local/etc/ldap.conf`
NSS-LDAP config: `/usr/local/etc/nss_ldap.conf`
NSS config: `/etc/nsswitch.conf`
PAM config (system): `/etc/pam.d/system`
PAM config (ssh): `/etc/pam.d/sshd`
PAM config (su): `/etc/pam.d/su`

2.3 Script Locations

Adding users (root): `/root/addusr.sh`
Removing users (root): `/root/rmusr.sh`
Editing users (root): `/root/editusr.sh`
Changing passwords (user): `/home/changepasswd.sh`

3 Script Commands

To assist in the operating LDAP, several script commands have been provided to save time on tasks. The code for these scripts can be found in the Appendix.

3.1 Adding Accounts

To add an account, use `addusr.sh`, it is solely for root and located in `/root/`. It can be executed as follows: `/root/addusr.sh [uid] [First Name] [Last Name] \\ [Group Number/User Number]` (remove the \\ when in the console). Thereafter it will ask for a password.

3.2 Removing Accounts

To remove an account, use `rmusr.sh`, which is located in `/root/` and can only be used by root. To execute it, enter the command like the following:

```
/root/rmusr.sh [uid] [First Name] [Last Name]
```

3.3 Editing Accounts

Root is also able to modify account details using the script `editusr.sh`. Like the other scripts intended for use by `root`, it can be found in `/root/`. It can be executed with `editusr.sh [option]` and the script will prompt root for the `uid` of the user and other needed information. The options and what they change are shown below:

- `name` - changes the user's `givenName` and `sn`, not their `uid`.
- `home` - changes the user home directory.
- `password` - chnages the user password.
- `shell` - changes the shell between `bash`, `csh` and `sh`.
- `help` - shows how to use the command.

3.4 Changing User Passwords

This script is for the use of a non-root user to change their own password. It can be executed by the use of `./changepasswd`; it will return an error if a local account attempts to use it.

4 Instructions

Below are the instructions to use root to modify accounts and how a user can change their account password.

4.1 Using root to add user accounts

To create a account via `root`, firstly a `.ldif` file must created with the details of the account. This file should contain both a group and user identity, but the former can be ignored if you are using a preexisting group. The following fields must configured:

- `gidNumber`, group number
- `uid`, user ID
- `uidNunmber`, user number
- `cn`, entry/full name
- `givenName`, first name
- `sn`, surname

Below an example configuration is shown. This will create an account with the user ID `jsmith` and a member of the group `jsmith`. It is notable that the group ID of an account can be set to 0, meaning it will be a member of the `wheel` group, which means it can use the `su` command.

To work with the scripts provided, the `cn` of an account **must be equal** to the uid appended with a '`u_`'. For example: `u_[uid]`. For groups the `cn` is equal to `g_[uid]`.

Listing 1: /usr/local/etc/openldap/test.ldif

```
# John Smith, Group, rule241.caia.swin.edu.au
dn: cn=jsmith,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
cn: g_jsmith
gidNumber: 6002
objectClass: posixGroup
objectClass: top

# Johm Smith, People, rule241.caia.swin.edu.au
dn: cn=John Smith,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
givenName: John
sn: Smith
cn: u_jsmith
uid: jsmith
objectClass: inetOrgPerson
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
uidNumber: 6002
gidNumber: 6002
```

```
homeDirectory: /home/jsmith
loginShell: /usr/local/bin/bash
userPassword: test
```

Thereafter, this account can be added by the use of `slapadd -l test.ldif`. Thereafter check that the account has been added by the use of `ldapsearch`.

4.2 Using root to remove user accounts

Root can delete an LDAP account if the `rootdn` username and password are known. The command is `ldapdelete` and the format is the like the following:

```
ldapdelete -D "[rootdn user]" -w [rootdn password] -h [hostname of server] \\
-p [ldap port] "[user1 to be deleted]" "[user2 to be deleted]" ..
```

Going back to the previous example with user `John Smith`, the following command can be used to delete an LDAP account.

```
ldapdelete -D "cn=Manager,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au" \\
-w secret -h rule241.caia.swin.edu.au \\
-p 389 "cn=jsmith,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au" \\
"cn=John Smith,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
```

Verify that the account has been deleted via `ldapsearch`.

4.3 Using root to change account passwords

Similar to the above section, the command to change a user's password (`ldappasswd`) requires the details of the `rootdn`. There are several different options that can be used that achieve the same result.

- `-a` allows for the old password to be received in the command itself.
- `-A` will make the console prompt root for the old password.
- `-s` allows for the new password to be received in the command itself.
- `-S` will make the console prompt root for the new password.

Overall, the command follows this format:

```
ldappasswd -D "[rootdn user]" -w [rootdn password] \\
-h [hostname of server] -p [ldap port] -A -S "[user]"
```

Once again, changing the password of user `jsmith` is as follows.

```
ldappasswd -D "cn=Manager,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au" \\
-w secret -h rule241.caia.swin.edu.au -p 389 -A -S \\
"cn=John Smith,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
```

4.4 How a user can change their own password

There are two methods provided to allow for a user to change their own password. The first is identical to the above method, but expects that the user has knowledge of the manager password (an ACL prevents the user from accessing any other accounts or details) and other server details. There is no method to change passwords without the use of the manager password.

4.4.1 Direct CLI Method

Below a method using `ldappasswd` to change the user passwords is shown. The user needs to know the domain of the server, the manager password and their `uid`. See the above section for the different password changing options in `ldappasswd`.

1. The user should find their `uid` using `whoami`. The account, if created by script, should have a `cn` that looks like `u_[uid]`.
2. The command below should be then executed, and the prompts it has should be filled in correctly (old and new passwords).

```
ldappasswd -D "[rootdn user]" -w [rootdn password] \\
-h [hostname of server] -p [ldap port] -A -S "[user]"
```

4.4.2 Script CLI Method

Running the script `./changepasswd` in the `/home/` directory will allow for a prompt to come up. The user should be enter their old and new passwords to change it.

5 Server Configuration Documentation

The server was configured with the following details. `/usr/local/etc/openldap/slapd.conf` was modified like below to allow for the operation of LDAP server. For brevity, only some configuration options are shown.

Listing 2: /usr/local/etc/openldap/slapd.conf

```
include /usr/local/etc/openldap/schema/core.schema
include /usr/local/etc/openldap/schema/cosine.schema
include /usr/local/etc/openldap/schema/inetorgperson.schema
include /usr/local/etc/openldap/schema/misc.schema
include /usr/local/etc/openldap/schema/nis.schema

moduleload      back_mdb

database        mdb
```

```
maxsize         1073741824
suffix          "dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
rootdn          "cn=Manager,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
```

To ensure `slapd` can be started via `service slapd start` the line `slapd_enable="YES"` was added to `/etc/rc.conf`. The two demonstration accounts were added via the following `.ldif` file and the `slapadd` command.

Listing 3: /usr/local/etc/openldap/twoacc.ldif

```
# rule241.caia.swin.edu.au
dn: dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
dc: rule241
objectClass: top
objectClass: domain
objectClass: domainRelatedObject
associatedDomain: rule241.caia.swin.edu.au

# People, rule241.caia.swin.edu.au
dn: ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
ou: People
objectClass: top
objectClass: organizationalUnit
objectClass: domainRelatedObject
associatedDomain: rule241.caia.swin.edu.au

# Group, rule241.caia.swin.edu.au
dn: ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
ou: Group
objectClass: top
objectClass: organizationalUnit
objectClass: domainRelatedObject
associatedDomain: rule241.caia.swin.edu.au

# Alan Smithee, Group, rule241.caia.swin.edu.au
dn: cn=g_asmithee,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
cn: g_asmithee
gidNumber: 6000
objectClass: posixGroup
objectClass: top

# gspelvin, Group, rule241.caia.swin.edu.au
dn: cn=g_gspelvin,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
cn: g_gspelvin
gidNumber: 6001
```

```
objectClass: posixGroup
objectClass: top

# Alan Smithee, People, rule241.caia.swin.edu.au
dn: cn=u_asmithee,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
givenName: Alan
sn: Smithee
cn: u_asmithee
uid: asmithee
objectClass: inetOrgPerson
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
uidNumber: 6000
gidNumber: 6000
homeDirectory: /home/asmithee
loginShell: /usr/local/bin/bash
userPassword: dune

# George Spelvin, People, rule241.caia.swin.edu.au
dn: cn=u_gspelvin,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
givenName: George
sn: Spelvin
cn: u_gspelvin
uid: gspelvin
objectClass: inetOrgPerson
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
uidNumber: 6001
gidNumber: 6001
homeDirectory: /home/gspelvin
loginShell: /usr/local/bin/bash
userPassword: broadway
```

To enable LDAP users to edit their own passwords, the following access control configuration should be placed in `slapd.conf`.

Listing 4: /usr/local/etc/openldap/slapd.conf

```
access to attrs=userPassword
        by self write
        by anonymous auth
        by users none

access to * by * read
```

5.1 Demonstration Account Details

As a demonstration, two accounts have been added to ensure the operation of the LDAP server, which are shown in Table **LOOK AT TABLE NUMBER**.

Table 1: Demonstration Accounts

Full Name	Alan Smithee	George Spelvin
Account Name	asmithee	gspelvin
Password	dune	broadway

6 Client Configuration Documentation

To install another work station with `root`, the following software is required to ensure LDAP can run. They should all be installed via `pkg install [package name]`

- `openldap-server-2.4.46_4`
- `pam_ldap-1.8.6_3`
- `nss_ldap-1.265_13`

After this, the first file that should be configured is `/usr/local/etc/openldap/ldap.conf`. Three lines should then be added to this file, which are listed below.

Listing 5: /usr/local/etc/openldap/ldap.conf

```
BASE    dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
URI     ldap://rule241.caia.swin.edu.au

pam_login_attribute uid
```

Save this file and then confirm that it was successful by running `ldapsearch`. If successful, it should show that the accounts `asmithee` and `gspelvin` exist and return an error code of 0.

Next, NSS LDAP must be configured, this can be done by opening the file `/usr/local/etc/nss_ldap.conf` and adding the lines as shown in Listing 2. Change the `host` and `base` fields to suit. If you are configuring the LDAP server to receive LDAP requests, also add the line that is commented out and remove the line `host 136.186.230.241`. Furthermore, to ensure the `slapd` server starts instantly (rather than taking about five minutes, add the `bind_policy` line (seen commented below).

Listing 6: /usr/local/etc/nss ldap.conf

```
#host    127.0.0.1
host     136.186.230.241
base     dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
```

```
uri       ldap://rule241.caia.swin.edu.au

#bind_policy soft
```

Moreover, the file `/etc/nsswitch.conf` must have the following lines edited. Ensure you can log in after changing the file and before logging out.

Listing 7: /etc/nsswitch.conf

```
group: files ldap
passwd: files ldap
```

Create the home directory for both users and use `ls -l` to ensure that the group and user names are appearing correctly (instead of their user IDs).

Then we must set up PAM, to do this several configuration files must be changed. It is advised that after each change is made you attempt to log into the client using another PuTTy session to ensure you have not lost access to the system. Firstly, copy `/usr/local/etc/openldap/ldap.conf` to `/usr/local/etc/`. This can easily be done in the `openldap` directory by entering `cp ldap.conf ..`, this act as the configuration file for PAM LDAP.

Thereafter move to the directory `/etc/pam.d/` and first open `system`, then `sshd` and `su`. Note that \\ is used where the line splits on the page.

The configuration for the `auth` and `account` settings must be as follows:

Listing 8: /etc/pam.d/system

```
auth sufficient pam_opie.so no_warn no_fake_prompts
auth requisite pam_opieaccess.so no_warn allow_local
auth sufficient /usr/local/lib/pam_ldap.so no_warn
auth required pam_unix.so  no_warn try_first_pass nullok

account required pam_login_access.so
account required pam_unix.so
account required /usr/local/lib/pam_ldap.so \\
    no_warn ignore_authinfo_unavail ignore_unknown_user
```

Listing 9: /etc/pam.d/sshd

```
auth sufficient pam_opie.so no_warn no_fake_prompts
auth requisite  pam_opieaccess.so no_warn allow_local
auth  sufficient /usr/local/lib/pam_ldap.so  no_warn
auth required pam_unix.so no_warn try_first_pass

account required pam_nologin.so
account required pam_login_access.so
account required pam_unix.so
account  required /usr/local/lib/pam_ldap.so \\
    no_warn ignore_authinfo_unavail ignore_unknown_user
```

Listing 10: /etc/pam.d/su

```
auth sufficient pam_rootok.so no_warn
auth sufficient pam_self.so no_warn
auth requisite pam_group.so no_warn group=wheel root
_only fail_safe ruser
auth include system

account include system
account required /usr/local/lib/pam_ldap.so \\
    no_warn ignore_authinfo_unavail ignore_unknown_user
```

Thereafter, the LDAP client should be setup. Make sure to log into all accounts (both local and LDAP based) via `ssh` and `su`.

7 Additional Features and Configuration

While the above sections deal with the creation and operation of an LDAP server, below are modifications that were made to the server to benefit the troubleshooting process.

7.1 LDAP Web Monitor

The web monitor provides an alternative method of checking the status of the LDAP server. It provides information on if the server is online and what performance it has (in essence: CPU and memory). This means that any user can check if the LDAP is reachable and if connection issues are due to the LDAP server being offline or due to another network fault. An example is a firewall misconfiguration, which could could allow HTTP through but not LDAP packets.

This page can be accessed via `http://rule241.caia.swin.edu.au`

7.1.1 Configuring the Web Monitor

To configure the `.cgi` file, it was placed in `/usr/local/www/apache24/data`. The following steps were then undertaken to make sure it works:

1. The address of the web server was set to `rule241.caia.swin.edu.au:80`.
2. `Options +ExecCGI` was added to `/usr/local/etc/apache24/http.conf` directory.
3. `AddHandler cgi-script .cgi` was added to `/usr/local/etc/apache24/http.conf` for the main directory.
4. `DirectoryIndex` was edited to `index.cgi` from `index.html`.
5. `mpm-prefork modules` was uncommented.

Thereafter the web monitor was functional.

8 Appendix (Code)

Below is the code of the four scripts; \\ is used when a line is too long for the page.

8.1 Add Users

Listing 11: addusr.sh

```
#!/usr/local/bin/bash

# addusr
# Author: Joshua Redolfi
# Add a user to the LDAP server on rule241.caia.swin.edu.au
# Takes the following arguments
# [uid] [First name] [Last name] [Password] OPTIONAL: [uidNumber]

#Get our args
uid=$1
fname=$2
lname=$3
idnum=$4

domain="dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
host="rule241.caia.swin.edu.au"
#make sure the above two match
rootdn="Manager"
rootpw=secret
port=389

if [ $1 == "help" ];
then
        echo "Command usage: ./addusr.sh [uid] [First name] \\
        [Last name] [uid number]"
        retVal=1
        return ${retVal} 2>/dev/null # this will attempt to return
        exit "${retVal}" # this will get executed if the above failed.
fi

#enter password
read -p "Enter the password: " pass

#create file tmp.ldif (and remove old one if it exists)
rm /tmp/tmp.ldif
echo 'dn: cn='g_$uid',ou=Group,'$domain'' >> /tmp/tmp.ldif
echo 'cn: 'g_$uid'' >> /tmp/tmp.ldif
```

```
echo 'gidNumber: '$idnum'' >> /tmp/tmp.ldif
echo 'objectClass: posixGroup' >> /tmp/tmp.ldif
echo 'objectClass: top' >> /tmp/tmp.ldif
echo '' >> /tmp/tmp.ldif
#now let's make the user
echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
echo 'givenName: '$fname'' >> /tmp/tmp.ldif
echo 'sn: '$lname'' >> /tmp/tmp.ldif
echo 'cn: 'u_$uid'' >> /tmp/tmp.ldif
echo 'uid: '$uid'' >> /tmp/tmp.ldif
echo 'objectClass: inetOrgPerson' >> /tmp/tmp.ldif
echo 'objectClass: top' >> /tmp/tmp.ldif
echo 'objectClass: posixAccount' >> /tmp/tmp.ldif
echo 'objectClass: shadowAccount' >> /tmp/tmp.ldif
echo 'uidNumber: '$idnum'' >> /tmp/tmp.ldif
echo 'gidNumber: '$idnum'' >> /tmp/tmp.ldif
echo 'homeDirectory: /home/'$uid'' >> /tmp/tmp.ldif
echo 'loginShell: /usr/local/bin/bash' >> /tmp/tmp.ldif
echo 'userPassword: '$pass'' >> /tmp/tmp.ldif

#add this ldif file to slapd
#slapadd -l /tmp/tmp.ldif

#adding to slapd via ldapadd
ldapadd -D cn=$rootdn,$domain -w $rootpw -h $host:$port -f /tmp/tmp.ldif

#check error code to see if user was actually added
if [ $? -eq 1 ]
then
        echo 'There was an error in adding the .ldif file'
else
        echo 'User '$uid'/'$fname' '$lname' added successfully'
fi

rm /tmp/tmp.ldif
if [ $? -eq 1 ]
then
        echo 'tmp.ldif could not be removed!'
else
        echo 'tmp.ldif removed from /tmp/'
fi
```

8.2 Remove Users

Listing 12: rmusr.sh

```
#!/usr/local/bin/bash

# rmusr
# Author: Joshua Redolfi
# Remove a user off the LDAP server on rule241.caia.swin.edu.au
# Takes the following arguments
# [uid] [First name] [Last name]

#Get our args
uid=$1

if [ $1 == "help" ];
then
        echo "Command usage: ./rmusr.sh [uid]"
        retVal=1
        return ${retVal} 2>/dev/null # this will attempt to return
        exit "${retVal}" # this will get executed if the above failed.
fi

domain="dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
host="rule241.caia.swin.edu.au"
#make sure the above two match
rootdn="Manager"
rootpw=secret
port=389

ldapdelete -D cn=$rootdn,$domain -w $rootpw -h \\
$host -p $port "cn=g_$uid,ou=Group,$domain" \\
"cn=u_$uid,ou=People,$domain"

#check error code to see if user was actually removed
if [ $? -eq 0 ]
then
        echo 'User '$uid' removed successfully'
else
        echo 'There was an error in removing the user'
fi
```

8.3 Edit Users

Listing 13: editusr.sh

```
#!/usr/local/bin/bash
```

```
# editusr
# Author: Joshua Redolfi
# Edit a user's details on the LDAP server \\
on rule241.caia.swin.edu.au

cname="u_"$name
domain="dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
host="rule241.caia.swin.edu.au"
#make sure the above two match
rootdn="Manager"
rootpw=secret
port=389

#control statement for options \\
(help, password, name, shell, home)
if [ $1 == "help" ]
then
        echo "Command usage: ./editusr.sh [option]"
        echo "password - change a user's password"
        echo "name - change a user's name \\
        (first name and last name)"
        echo "shell - change a user's shell (bash, csh and sh)"
        echo "home - change a user's home directory"
        retVal=1
        return ${retVal} 2>/dev/null # this will attempt to return
        exit "${retVal}" # this will get \\
        executed if the above failed.
elif [ $1 == "password" ]
        then
        echo "Enter user id to change password for"
        read -p "User ID:" uid
        ldappasswd -D cn=$rootdn,$domain -w $rootpw -h $host \\
        -p $port -A -S cn="u_"$uid,ou=People,$domain
        #check error code to see if password was actually changed
        if [ $? -eq 0 ]
        then
                echo 'User password changed successfully'
        else
                echo 'There was an error in changing the password'
        fi
elif [ $1 == "name" ]
        then
        echo "Enter user id of the account to change the name of"
        read -p "User ID:" uid
        rm /tmp/tmp.ldif
        echo "Enter the new first name:"
```

```
        read -p "First name:" fname
        echo "Enter the new surname:"
        read -p "Surname:" lname
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        echo 'changetype: modify' >> /tmp/tmp.ldif
        echo 'replace: givenName' >> /tmp/tmp.ldif
        echo 'givenName: '$fname'' >> /tmp/tmp.ldif
        #modify first name first, done in two steps because \\
        ldapmodify doesnt like it otherwise
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        echo 'changetype: modify' >> /tmp/tmp.ldif
        echo 'replace: sn' >> /tmp/tmp.ldif
        echo 'sn: '$lname'' >> /tmp/tmp.ldif
        #then modify last name
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
elif [ $1 == "shell" ]
        then
        echo "Enter user id of the account to change the shell of"
        read -p "User ID:" uid
        echo "Type name of the shell you wish \\
        to change the user shell to:"
        echo "Options: bash, sh, csh"
        read -p "Shell:" shell
        rm /tmp/tmp.ldif
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        #control statement for shells, reject if invalid
        if [ $shell == "bash" ]
        then
                echo 'changetype: modify' >> /tmp/tmp.ldif
                echo 'replace: loginShell' >> /tmp/tmp.ldif
                echo 'loginShell: /usr/local/bin/bash' >> /tmp/tmp.ldif
        elif [ $shell = "sh" ]
        then
                echo 'changetype: modify' >> /tmp/tmp.ldif
                echo 'replace: loginShell' >> /tmp/tmp.ldif
                echo 'loginShell: /bin/sh' >> /tmp/tmp.ldif
        elif [ $shell = "csh" ]
        then
                echo 'changetype: modify' >> /tmp/tmp.ldif
                echo 'replace: loginShell' >> /tmp/tmp.ldif
                echo 'loginShell: /bin/csh' >> /tmp/tmp.ldif
```

```
        else
                echo "Invalid option, won't change shell type"
                echo "Exiting..."
                retVal=1
                return ${retVal} 2>/dev/null # this will \\
                attempt to return
                exit "${retVal}" # this will get executed \\
                if the above failed.
        fi
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
elif [ $1 == "home" ]
then
        echo "Enter user id of the account to \\
        change the home directory of"
        read -p "User ID:" uid
        echo "Enter the new directory (absolute reference):"
        echo "Example: /newlocation/"
        read -p "Home directory:" home
        rm /tmp/tmp.ldif
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        echo 'changetype: modify' >> /tmp/tmp.ldif
        echo 'replace: homeDirectory' >> /tmp/tmp.ldif
        echo 'homeDirectory: '$home'' >> /tmp/tmp.ldif
        #modify home directory here
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
else
        echo "Option invalid, type \\
        './editusr.sh help' to see all options."
fi
```

8.4 Change Passwords

Listing 14: changepasswd.sh

```
#!/usr/local/bin/bash

# changepasswd
# Author: Joshua Redolfi
# Edit a user's password on the LDAP \\
server on rule241.caia.swin.edu.au
# Takes no arguments
```

```
name=$(whoami)

domain='dc=rule241,dc=caia,dc=swin,dc=edu,dc=au'
host=rule241.caia.swin.edu.au
#make sure the above two match
rootdn=Manager
rootpw=secret
port=389

ldappasswd -D cn=$rootdn,$domain -w $rootpw -h \\
$host -p $port -A -S cn="u_"$name,ou=People,$domain

#check error code to see if password was actually changed
if [ $? -eq 0 ]
then
        echo 'User_'$name'_password_successfully'
else
        echo 'There_was_an_error_in_changing_the_password'
fi


#ldappasswd -D cn=Manager,dc=rule241,dc=caia,dc=swin, \\
dc=edu,dc=au -w secret -h rule241.caia.swin.edu.au \\
-p 389 -A -S cn=asmithee,ou=People,dc=rule241, \\
dc=caia,dc=swin,dc=edu,dc=au
```

LDAP Project Documentation

Joshua Redolfi
Faculty of Science,
Engineering and, Technology
Swinburne University of Technology
Email: 101601886@student.swin.edu.au

Semester 2 2019

Abstract

This document describes the operation of the LDAP (Light-weight Directory Access Protocol) on the RULE host rule241.caia.swin.edu.au. It contains information on how the server was configured and how to configure clients to connect to it. Thereafter, account and script details are provided to show how the system has been automated. Finally, the operation of an LDAP web monitor is provided. **This document contains all the documentation required for the project**.

Contents

1 Introduction

This documentation contains information of the LDAP server set up on the following hosts:

Server Address: 136.186.230.241 (rule241.caia.swin.edu.au)
Client Address: 136.186.230.240 (rule240.caia.swin.edu.au)

2 Location of Files and CLI Scripts

This contains the locations of the files that were edited and created for the use of LDAP. It also has the locations of all the scripts for adding, editing and removing user accounts, along with the ability for users to change their passwords.

2.1 Server Locations

SLAPD config: `/usr/local/etc/openldap/slapd.conf`
Initial .ldif file: `/usr/local/etc/openldap/twoacc.ldif`
LDAP config: `/usr/local/etc/openldap/ldap.conf`
PAM-LDAP config: `/usr/local/etc/ldap.conf`
NSS-LDAP config: `/usr/local/etc/nss_ldap.conf`
NSS config: `/etc/nsswitch.conf`
PAM config (system): `/etc/pam.d/system`
PAM config (ssh): `/etc/pam.d/sshd`
PAM config (su): `/etc/pam.d/su`
Apache config: `/usr/local/etc/apache24/http.conf`
Web Monitor: `/usr/local/www/apache24/data/index.cgi`

2.2 Client Locations

LDAP config: `/usr/local/etc/openldap/ldap.conf`
PAM-LDAP config: `/usr/local/etc/ldap.conf`
NSS-LDAP config: `/usr/local/etc/nss_ldap.conf`
NSS config: `/etc/nsswitch.conf`
PAM config (system): `/etc/pam.d/system`
PAM config (ssh): `/etc/pam.d/sshd`
PAM config (su): `/etc/pam.d/su`

2.3 Script Locations

Adding users (root): `/root/addusr.sh`
Removing users (root): `/root/rmusr.sh`
Editing users (root): `/root/editusr.sh`
Changing passwords (user): `/home/changepasswd.sh`

3 Script Commands

To assist in the operating LDAP, several script commands have been provided to save time on tasks. The code for these scripts can be found in the Appendix.

3.1 Adding Accounts

To add an account, use `addusr.sh`, it is solely for root and located in `/root/`. It can be executed as follows: `/root/addusr.sh [uid] [First Name] [Last Name] \\ [Group Number/User Number]` (remove the \\ when in the console). Thereafter it will ask for a password.

3.2 Removing Accounts

To remove an account, use `rmusr.sh`, which is located in `/root/` and can only be used by root. To execute it, enter the command like the following:

```
/root/rmusr.sh [uid] [First Name] [Last Name]
```

3.3 Editing Accounts

Root is also able to modify account details using the script `editusr.sh`. Like the other scripts intended for use by `root`, it can be found in `/root/`. It can be executed with `editusr.sh [option]` and the script will prompt root for the `uid` of the user and other needed information. The options and what they change are shown below:

- `name` - changes the user's `givenName` and `sn`, not their `uid`.
- `home` - changes the user home directory.
- `password` - chnages the user password.
- `shell` - changes the shell between `bash`, `csh` and `sh`.
- `help` - shows how to use the command.

3.4 Changing User Passwords

This script is for the use of a non-root user to change their own password. It can be executed by the use of `./changepasswd`; it will return an error if a local account attempts to use it.

4 Instructions

Below are the instructions to use root to modify accounts and how a user can change their account password.

4.1 Using root to add user accounts

To create a account via `root`, firstly a `.ldif` file must created with the details of the account. This file should contain both a group and user identity, but the former can be ignored if you are using a preexisting group. The following fields must configured:

- `gidNumber`, group number
- `uid`, user ID
- `uidNunmber`, user number
- `cn`, entry/full name
- `givenName`, first name
- `sn`, surname

Below an example configuration is shown. This will create an account with the user ID `jsmith` and a member of the group `jsmith`. It is notable that the group ID of an account can be set to 0, meaning it will be a member of the `wheel` group, which means it can use the `su` command.

To work with the scripts provided, the `cn` of an account **must be equal** to the uid appended with a '`u_`'. For example: `u_[uid]`. For groups the `cn` is equal to `g_[uid]`.

Listing 1: /usr/local/etc/openldap/test.ldif

```
# John Smith, Group, rule241.caia.swin.edu.au
dn: cn=jsmith,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
cn: g_jsmith
gidNumber: 6002
objectClass: posixGroup
objectClass: top

# Johm Smith, People, rule241.caia.swin.edu.au
dn: cn=John Smith,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
givenName: John
sn: Smith
cn: u_jsmith
uid: jsmith
objectClass: inetOrgPerson
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
uidNumber: 6002
gidNumber: 6002
```

```
homeDirectory: /home/jsmith
loginShell: /usr/local/bin/bash
userPassword: test
```

Thereafter, this account can be added by the use of `slapadd -l test.ldif`. Thereafter check that the account has been added by the use of `ldapsearch`.

4.2 Using root to remove user accounts

Root can delete an LDAP account if the `rootdn` username and password are known. The command is `ldapdelete` and the format is the like the following:

```
ldapdelete -D "[rootdn user]" -w [rootdn password] -h [hostname of server] \\
-p [ldap port] "[user1 to be deleted]" "[user2 to be deleted]" ..
```

Going back to the previous example with user `John Smith`, the following command can be used to delete an LDAP account.

```
ldapdelete -D "cn=Manager,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au" \\
-w secret -h rule241.caia.swin.edu.au \\
-p 389 "cn=jsmith,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au" \\
"cn=John Smith,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
```

Verify that the account has been deleted via `ldapsearch`.

4.3 Using root to change account passwords

Similar to the above section, the command to change a user's password (`ldappasswd`) requires the details of the `rootdn`. There are several different options that can be used that achieve the same result.

- `-a` allows for the old password to be received in the command itself.
- `-A` will make the console prompt root for the old password.
- `-s` allows for the new password to be received in the command itself.
- `-S` will make the console prompt root for the new password.

Overall, the command follows this format:

```
ldappasswd -D "[rootdn user]" -w [rootdn password] \\
-h [hostname of server] -p [ldap port] -A -S "[user]"
```

Once again, changing the password of user `jsmith` is as follows.

```
ldappasswd -D "cn=Manager,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au" \\
-w secret -h rule241.caia.swin.edu.au -p 389 -A -S \\
"cn=John Smith,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
```

4.4 How a user can change their own password

There are two methods provided to allow for a user to change their own password. The first is identical to the above method, but expects that the user has knowledge of the manager password (an ACL prevents the user from accessing any other accounts or details) and other server details. There is no method to change passwords without the use of the manager password.

4.4.1 Direct CLI Method

Below a method using `ldappasswd` to change the user passwords is shown. The user needs to know the domain of the server, the manager password and their `uid`. See the above section for the different password changing options in `ldappasswd`.

1. The user should find their `uid` using `whoami`. The account, if created by script, should have a `cn` that looks like `u_[uid]`.
2. The command below should be then executed, and the prompts it has should be filled in correctly (old and new passwords).

```
ldappasswd -D "[rootdn user]" -w [rootdn password] \\
-h [hostname of server] -p [ldap port] -A -S "[user]"
```

4.4.2 Script CLI Method

Running the script `./changepasswd` in the `/home/` directory will allow for a prompt to come up. The user should be enter their old and new passwords to change it.

5 Server Configuration Documentation

The server was configured with the following details. `/usr/local/etc/openldap/slapd.conf` was modified like below to allow for the operation of LDAP server. For brevity, only some configuration options are shown.

Listing 2: /usr/local/etc/openldap/slapd.conf

```
include /usr/local/etc/openldap/schema/core.schema
include /usr/local/etc/openldap/schema/cosine.schema
include /usr/local/etc/openldap/schema/inetorgperson.schema
include /usr/local/etc/openldap/schema/misc.schema
include /usr/local/etc/openldap/schema/nis.schema

moduleload      back_mdb

database        mdb
```

```
maxsize         1073741824
suffix          "dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
rootdn          "cn=Manager,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
```

To ensure `slapd` can be started via `service slapd start` the line `slapd_enable="YES"` was added to `/etc/rc.conf`. The two demonstration accounts were added via the following `.ldif` file and the `slapadd` command.

Listing 3: /usr/local/etc/openldap/twoacc.ldif

```
# rule241.caia.swin.edu.au
dn: dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
dc: rule241
objectClass: top
objectClass: domain
objectClass: domainRelatedObject
associatedDomain: rule241.caia.swin.edu.au

# People, rule241.caia.swin.edu.au
dn: ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
ou: People
objectClass: top
objectClass: organizationalUnit
objectClass: domainRelatedObject
associatedDomain: rule241.caia.swin.edu.au

# Group, rule241.caia.swin.edu.au
dn: ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
ou: Group
objectClass: top
objectClass: organizationalUnit
objectClass: domainRelatedObject
associatedDomain: rule241.caia.swin.edu.au

# Alan Smithee, Group, rule241.caia.swin.edu.au
dn: cn=g_asmithee,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
cn: g_asmithee
gidNumber: 6000
objectClass: posixGroup
objectClass: top

# gspelvin, Group, rule241.caia.swin.edu.au
dn: cn=g_gspelvin,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
cn: g_gspelvin
gidNumber: 6001
```

```
objectClass: posixGroup
objectClass: top

# Alan Smithee, People, rule241.caia.swin.edu.au
dn: cn=u_asmithee,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
givenName: Alan
sn: Smithee
cn: u_asmithee
uid: asmithee
objectClass: inetOrgPerson
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
uidNumber: 6000
gidNumber: 6000
homeDirectory: /home/asmithee
loginShell: /usr/local/bin/bash
userPassword: dune

# George Spelvin, People, rule241.caia.swin.edu.au
dn: cn=u_gspelvin,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
givenName: George
sn: Spelvin
cn: u_gspelvin
uid: gspelvin
objectClass: inetOrgPerson
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
uidNumber: 6001
gidNumber: 6001
homeDirectory: /home/gspelvin
loginShell: /usr/local/bin/bash
userPassword: broadway
```

To enable LDAP users to edit their own passwords, the following access control configuration should be placed in `slapd.conf`.

Listing 4: /usr/local/etc/openldap/slapd.conf

```
access to attrs=userPassword
        by self write
        by anonymous auth
        by users none

access to * by * read
```

5.1 Demonstration Account Details

As a demonstration, two accounts have been added to ensure the operation of the LDAP server, which are shown in Table **LOOK AT TABLE NUMBER**.

Table 1: Demonstration Accounts

Full Name	Alan Smithee	George Spelvin
Account Name	asmithee	gspelvin
Password	dune	broadway

6 Client Configuration Documentation

To install another work station with `root`, the following software is required to ensure LDAP can run. They should all be installed via `pkg install [package name]`

- `openldap-server-2.4.46_4`
- `pam_ldap-1.8.6_3`
- `nss_ldap-1.265_13`

After this, the first file that should be configured is `/usr/local/etc/openldap/ldap.conf`. Three lines should then be added to this file, which are listed below.

Listing 5: /usr/local/etc/openldap/ldap.conf

```
BASE    dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
URI     ldap://rule241.caia.swin.edu.au

pam_login_attribute uid
```

Save this file and then confirm that it was successful by running `ldapsearch`. If successful, it should show that the accounts `asmithee` and `gspelvin` exist and return an error code of 0.

Next, NSS LDAP must be configured, this can be done by opening the file `/usr/local/etc/nss_ldap.conf` and adding the lines as shown in Listing 2. Change the `host` and `base` fields to suit. If you are configuring the LDAP server to receive LDAP requests, also add the line that is commented out and remove the line `host 136.186.230.241`. Furthermore, to ensure the `slapd` server starts instantly (rather than taking about five minutes, add the `bind_policy` line (seen commented below).

Listing 6: /usr/local/etc/nss ldap.conf

```
#host    127.0.0.1
host     136.186.230.241
base     dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
```

```
uri       ldap://rule241.caia.swin.edu.au

#bind_policy soft
```

Moreover, the file `/etc/nsswitch.conf` must have the following lines edited. Ensure you can log in after changing the file and before logging out.

Listing 7: /etc/nsswitch.conf

```
group: files ldap
passwd: files ldap
```

Create the home directory for both users and use `ls -l` to ensure that the group and user names are appearing correctly (instead of their user IDs).

Then we must set up PAM, to do this several configuration files must be changed. It is advised that after each change is made you attempt to log into the client using another PuTTy session to ensure you have not lost access to the system. Firstly, copy `/usr/local/etc/openldap/ldap.conf` to `/usr/local/etc/`. This can easily be done in the `openldap` directory by entering `cp ldap.conf ..`, this act as the configuration file for PAM LDAP.

Thereafter move to the directory `/etc/pam.d/` and first open `system`, then `sshd` and `su`. Note that \\ is used where the line splits on the page.

The configuration for the `auth` and `account` settings must be as follows:

Listing 8: /etc/pam.d/system

```
auth sufficient pam_opie.so no_warn no_fake_prompts
auth requisite pam_opieaccess.so no_warn allow_local
auth sufficient /usr/local/lib/pam_ldap.so no_warn
auth required pam_unix.so  no_warn try_first_pass nullok

account required pam_login_access.so
account required pam_unix.so
account required /usr/local/lib/pam_ldap.so \\
    no_warn ignore_authinfo_unavail ignore_unknown_user
```

Listing 9: /etc/pam.d/sshd

```
auth sufficient pam_opie.so no_warn no_fake_prompts
auth requisite  pam_opieaccess.so no_warn allow_local
auth  sufficient /usr/local/lib/pam_ldap.so  no_warn
auth required pam_unix.so no_warn try_first_pass

account required pam_nologin.so
account required pam_login_access.so
account required pam_unix.so
account  required /usr/local/lib/pam_ldap.so \\
    no_warn ignore_authinfo_unavail ignore_unknown_user
```

Listing 10: /etc/pam.d/su

```
auth sufficient pam_rootok.so no_warn
auth sufficient pam_self.so no_warn
auth requisite pam_group.so no_warn group=wheel root
_only fail_safe ruser
auth include system

account include system
account required /usr/local/lib/pam_ldap.so \\
    no_warn ignore_authinfo_unavail ignore_unknown_user
```

Thereafter, the LDAP client should be setup. Make sure to log into all accounts (both local and LDAP based) via `ssh` and `su`.

7 Additional Features and Configuration

While the above sections deal with the creation and operation of an LDAP server, below are modifications that were made to the server to benefit the troubleshooting process.

7.1 LDAP Web Monitor

The web monitor provides an alternative method of checking the status of the LDAP server. It provides information on if the server is online and what performance it has (in essence: CPU and memory). This means that any user can check if the LDAP is reachable and if connection issues are due to the LDAP server being offline or due to another network fault. An example is a firewall misconfiguration, which could could allow HTTP through but not LDAP packets.

This page can be accessed via `http://rule241.caia.swin.edu.au`

7.1.1 Configuring the Web Monitor

To configure the `.cgi` file, it was placed in `/usr/local/www/apache24/data`. The following steps were then undertaken to make sure it works:

1. The address of the web server was set to `rule241.caia.swin.edu.au:80`.
2. `Options +ExecCGI` was added to `/usr/local/etc/apache24/http.conf` directory.
3. `AddHandler cgi-script .cgi` was added to `/usr/local/etc/apache24/http.conf` for the main directory.
4. `DirectoryIndex` was edited to `index.cgi` from `index.html`.
5. `mpm-prefork modules` was uncommented.

Thereafter the web monitor was functional.

8 Appendix (Code)

Below is the code of the four scripts; \\ is used when a line is too long for the page.

8.1 Add Users

Listing 11: addusr.sh

```
#!/usr/local/bin/bash

# addusr
# Author: Joshua Redolfi
# Add a user to the LDAP server on rule241.caia.swin.edu.au
# Takes the following arguments
# [uid] [First name] [Last name] [Password] OPTIONAL: [uidNumber]

#Get our args
uid=$1
fname=$2
lname=$3
idnum=$4

domain="dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
host="rule241.caia.swin.edu.au"
#make sure the above two match
rootdn="Manager"
rootpw=secret
port=389

if [ $1 == "help" ];
then
        echo "Command usage: ./addusr.sh [uid] [First name] \\
        [Last name] [uid number]"
        retVal=1
        return ${retVal} 2>/dev/null # this will attempt to return
        exit "${retVal}" # this will get executed if the above failed.
fi

#enter password
read -p "Enter the password: " pass

#create file tmp.ldif (and remove old one if it exists)
rm /tmp/tmp.ldif
echo 'dn: cn='g_$uid',ou=Group,'$domain'' >> /tmp/tmp.ldif
echo 'cn: 'g_$uid'' >> /tmp/tmp.ldif
```

```
echo 'gidNumber: '$idnum'' >> /tmp/tmp.ldif
echo 'objectClass: posixGroup' >> /tmp/tmp.ldif
echo 'objectClass: top' >> /tmp/tmp.ldif
echo '' >> /tmp/tmp.ldif
#now let's make the user
echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
echo 'givenName: '$fname'' >> /tmp/tmp.ldif
echo 'sn: '$lname'' >> /tmp/tmp.ldif
echo 'cn: 'u_$uid'' >> /tmp/tmp.ldif
echo 'uid: '$uid'' >> /tmp/tmp.ldif
echo 'objectClass: inetOrgPerson' >> /tmp/tmp.ldif
echo 'objectClass: top' >> /tmp/tmp.ldif
echo 'objectClass: posixAccount' >> /tmp/tmp.ldif
echo 'objectClass: shadowAccount' >> /tmp/tmp.ldif
echo 'uidNumber: '$idnum'' >> /tmp/tmp.ldif
echo 'gidNumber: '$idnum'' >> /tmp/tmp.ldif
echo 'homeDirectory: /home/'$uid'' >> /tmp/tmp.ldif
echo 'loginShell: /usr/local/bin/bash' >> /tmp/tmp.ldif
echo 'userPassword: '$pass'' >> /tmp/tmp.ldif

#add this ldif file to slapd
#slapadd -l /tmp/tmp.ldif

#adding to slapd via ldapadd
ldapadd -D cn=$rootdn,$domain -w $rootpw -h $host:$port -f /tmp/tmp.ldif

#check error code to see if user was actually added
if [ $? -eq 1 ]
then
        echo 'There was an error in adding the .ldif file'
else
        echo 'User '$uid'/'$fname' '$lname' added successfully'
fi

rm /tmp/tmp.ldif
if [ $? -eq 1 ]
then
        echo 'tmp.ldif could not be removed!'
else
        echo 'tmp.ldif removed from /tmp/'
fi
```

8.2 Remove Users

Listing 12: rmusr.sh

```
#!/usr/local/bin/bash

# rmusr
# Author: Joshua Redolfi
# Remove a user off the LDAP server on rule241.caia.swin.edu.au
# Takes the following arguments
# [uid] [First name] [Last name]

#Get our args
uid=$1

if [ $1 == "help" ];
then
        echo "Command usage: ./rmusr.sh [uid]"
        retVal=1
        return ${retVal} 2>/dev/null # this will attempt to return
        exit "${retVal}" # this will get executed if the above failed.
fi

domain="dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
host="rule241.caia.swin.edu.au"
#make sure the above two match
rootdn="Manager"
rootpw=secret
port=389

ldapdelete -D cn=$rootdn,$domain -w $rootpw -h \\
$host -p $port "cn=g_$uid,ou=Group,$domain" \\
"cn=u_$uid,ou=People,$domain"

#check error code to see if user was actually removed
if [ $? -eq 0 ]
then
        echo 'User '$uid' removed successfully'
else
        echo 'There was an error in removing the user'
fi
```

8.3 Edit Users

Listing 13: editusr.sh

```
#!/usr/local/bin/bash
```

```
# editusr
# Author: Joshua Redolfi
# Edit a user's details on the LDAP server \\
on rule241.caia.swin.edu.au

cname="u_"$name
domain="dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
host="rule241.caia.swin.edu.au"
#make sure the above two match
rootdn="Manager"
rootpw=secret
port=389

#control statement for options \\
(help, password, name, shell, home)
if [ $1 == "help" ]
then
        echo "Command usage: ./editusr.sh [option]"
        echo "password - change a user's password"
        echo "name - change a user's name \\
        (first name and last name)"
        echo "shell - change a user's shell (bash, csh and sh)"
        echo "home - change a user's home directory"
        retVal=1
        return ${retVal} 2>/dev/null # this will attempt to return
        exit "${retVal}" # this will get \\
        executed if the above failed.
elif [ $1 == "password" ]
        then
        echo "Enter user id to change password for"
        read -p "User ID:" uid
        ldappasswd -D cn=$rootdn,$domain -w $rootpw -h $host \\
        -p $port -A -S cn="u_"$uid,ou=People,$domain
        #check error code to see if password was actually changed
        if [ $? -eq 0 ]
        then
                echo 'User password changed successfully'
        else
                echo 'There was an error in changing the password'
        fi
elif [ $1 == "name" ]
        then
        echo "Enter user id of the account to change the name of"
        read -p "User ID:" uid
        rm /tmp/tmp.ldif
        echo "Enter the new first name:"
```

```
        read -p "First name:" fname
        echo "Enter the new surname:"
        read -p "Surname:" lname
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        echo 'changetype: modify' >> /tmp/tmp.ldif
        echo 'replace: givenName' >> /tmp/tmp.ldif
        echo 'givenName: '$fname'' >> /tmp/tmp.ldif
        #modify first name first, done in two steps because \\
        ldapmodify doesnt like it otherwise
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        echo 'changetype: modify' >> /tmp/tmp.ldif
        echo 'replace: sn' >> /tmp/tmp.ldif
        echo 'sn: '$lname'' >> /tmp/tmp.ldif
        #then modify last name
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
elif [ $1 == "shell" ]
        then
        echo "Enter user id of the account to change the shell of"
        read -p "User ID:" uid
        echo "Type name of the shell you wish \\
        to change the user shell to:"
        echo "Options: bash, sh, csh"
        read -p "Shell:" shell
        rm /tmp/tmp.ldif
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        #control statement for shells, reject if invalid
        if [ $shell == "bash" ]
        then
                echo 'changetype: modify' >> /tmp/tmp.ldif
                echo 'replace: loginShell' >> /tmp/tmp.ldif
                echo 'loginShell: /usr/local/bin/bash' >> /tmp/tmp.ldif
        elif [ $shell = "sh" ]
        then
                echo 'changetype: modify' >> /tmp/tmp.ldif
                echo 'replace: loginShell' >> /tmp/tmp.ldif
                echo 'loginShell: /bin/sh' >> /tmp/tmp.ldif
        elif [ $shell = "csh" ]
        then
                echo 'changetype: modify' >> /tmp/tmp.ldif
                echo 'replace: loginShell' >> /tmp/tmp.ldif
                echo 'loginShell: /bin/csh' >> /tmp/tmp.ldif
```

```
        else
                echo "Invalid option, won't change shell type"
                echo "Exiting..."
                retVal=1
                return ${retVal} 2>/dev/null # this will \\
                attempt to return
                exit "${retVal}" # this will get executed \\
                if the above failed.
        fi
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
elif [ $1 == "home" ]
then
        echo "Enter user id of the account to \\
        change the home directory of"
        read -p "User ID:" uid
        echo "Enter the new directory (absolute reference):"
        echo "Example: /newlocation/"
        read -p "Home directory:" home
        rm /tmp/tmp.ldif
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        echo 'changetype: modify' >> /tmp/tmp.ldif
        echo 'replace: homeDirectory' >> /tmp/tmp.ldif
        echo 'homeDirectory: '$home'' >> /tmp/tmp.ldif
        #modify home directory here
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
else
        echo "Option invalid, type \\
        './editusr.sh help' to see all options."
fi
```

8.4 Change Passwords

Listing 14: changepasswd.sh

```
#!/usr/local/bin/bash

# changepasswd
# Author: Joshua Redolfi
# Edit a user's password on the LDAP \\
server on rule241.caia.swin.edu.au
# Takes no arguments
```

```
name=$(whoami)

domain='dc=rule241,dc=caia,dc=swin,dc=edu,dc=au'
host=rule241.caia.swin.edu.au
#make sure the above two match
rootdn=Manager
rootpw=secret
port=389

ldappasswd -D cn=$rootdn,$domain -w $rootpw -h \\
$host -p $port -A -S cn="u_"$name,ou=People,$domain

#check error code to see if password was actually changed
if [ $? -eq 0 ]
then
        echo 'User_'$name'_password_successfully'
else
        echo 'There_was_an_error_in_changing_the_password'
fi


#ldappasswd -D cn=Manager,dc=rule241,dc=caia,dc=swin, \\
dc=edu,dc=au -w secret -h rule241.caia.swin.edu.au \\
-p 389 -A -S cn=asmithee,ou=People,dc=rule241, \\
dc=caia,dc=swin,dc=edu,dc=au
```

LDAP Project Documentation

Joshua Redolfi
Faculty of Science,
Engineering and, Technology
Swinburne University of Technology
Email: 101601886@student.swin.edu.au

Semester 2 2019

Abstract

This document describes the operation of the LDAP (Light-weight Directory Access Protocol) on the RULE host rule241.caia.swin.edu.au. It contains information on how the server was configured and how to configure clients to connect to it. Thereafter, account and script details are provided to show how the system has been automated. Finally, the operation of an LDAP web monitor is provided. **This document contains all the documentation required for the project**.

Contents

1 Introduction

This documentation contains information of the LDAP server set up on the following hosts:

Server Address: 136.186.230.241 (rule241.caia.swin.edu.au)
Client Address: 136.186.230.240 (rule240.caia.swin.edu.au)

2 Location of Files and CLI Scripts

This contains the locations of the files that were edited and created for the use of LDAP. It also has the locations of all the scripts for adding, editing and removing user accounts, along with the ability for users to change their passwords.

2.1 Server Locations

SLAPD config: `/usr/local/etc/openldap/slapd.conf`
Initial .ldif file: `/usr/local/etc/openldap/twoacc.ldif`
LDAP config: `/usr/local/etc/openldap/ldap.conf`
PAM-LDAP config: `/usr/local/etc/ldap.conf`
NSS-LDAP config: `/usr/local/etc/nss_ldap.conf`
NSS config: `/etc/nsswitch.conf`
PAM config (system): `/etc/pam.d/system`
PAM config (ssh): `/etc/pam.d/sshd`
PAM config (su): `/etc/pam.d/su`
Apache config: `/usr/local/etc/apache24/http.conf`
Web Monitor: `/usr/local/www/apache24/data/index.cgi`

2.2 Client Locations

LDAP config: `/usr/local/etc/openldap/ldap.conf`
PAM-LDAP config: `/usr/local/etc/ldap.conf`
NSS-LDAP config: `/usr/local/etc/nss_ldap.conf`
NSS config: `/etc/nsswitch.conf`
PAM config (system): `/etc/pam.d/system`
PAM config (ssh): `/etc/pam.d/sshd`
PAM config (su): `/etc/pam.d/su`

2.3 Script Locations

Adding users (root): `/root/addusr.sh`
Removing users (root): `/root/rmusr.sh`
Editing users (root): `/root/editusr.sh`
Changing passwords (user): `/home/changepasswd.sh`

3 Script Commands

To assist in the operating LDAP, several script commands have been provided to save time on tasks. The code for these scripts can be found in the Appendix.

3.1 Adding Accounts

To add an account, use `addusr.sh`, it is solely for root and located in `/root/`. It can be executed as follows: `/root/addusr.sh [uid] [First Name] [Last Name] \\ [Group Number/User Number]` (remove the \\ when in the console). Thereafter it will ask for a password.

3.2 Removing Accounts

To remove an account, use `rmusr.sh`, which is located in `/root/` and can only be used by root. To execute it, enter the command like the following:

```
/root/rmusr.sh [uid] [First Name] [Last Name]
```

3.3 Editing Accounts

Root is also able to modify account details using the script `editusr.sh`. Like the other scripts intended for use by `root`, it can be found in `/root/`. It can be executed with `editusr.sh [option]` and the script will prompt root for the `uid` of the user and other needed information. The options and what they change are shown below:

- `name` - changes the user's `givenName` and `sn`, not their `uid`.
- `home` - changes the user home directory.
- `password` - chnages the user password.
- `shell` - changes the shell between `bash`, `csh` and `sh`.
- `help` - shows how to use the command.

3.4 Changing User Passwords

This script is for the use of a non-root user to change their own password. It can be executed by the use of `./changepasswd`; it will return an error if a local account attempts to use it.

4 Instructions

Below are the instructions to use root to modify accounts and how a user can change their account password.

4.1 Using root to add user accounts

To create a account via `root`, firstly a `.ldif` file must created with the details of the account. This file should contain both a group and user identity, but the former can be ignored if you are using a preexisting group. The following fields must configured:

- `gidNumber`, group number
- `uid`, user ID
- `uidNunmber`, user number
- `cn`, entry/full name
- `givenName`, first name
- `sn`, surname

Below an example configuration is shown. This will create an account with the user ID `jsmith` and a member of the group `jsmith`. It is notable that the group ID of an account can be set to 0, meaning it will be a member of the `wheel` group, which means it can use the `su` command.

To work with the scripts provided, the `cn` of an account **must be equal** to the uid appended with a '`u_`'. For example: `u_[uid]`. For groups the `cn` is equal to `g_[uid]`.

Listing 1: /usr/local/etc/openldap/test.ldif

```
# John Smith, Group, rule241.caia.swin.edu.au
dn: cn=jsmith,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
cn: g_jsmith
gidNumber: 6002
objectClass: posixGroup
objectClass: top

# Johm Smith, People, rule241.caia.swin.edu.au
dn: cn=John Smith,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
givenName: John
sn: Smith
cn: u_jsmith
uid: jsmith
objectClass: inetOrgPerson
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
uidNumber: 6002
gidNumber: 6002
```

```
homeDirectory: /home/jsmith
loginShell: /usr/local/bin/bash
userPassword: test
```

Thereafter, this account can be added by the use of `slapadd -l test.ldif`. Thereafter check that the account has been added by the use of `ldapsearch`.

4.2 Using root to remove user accounts

Root can delete an LDAP account if the `rootdn` username and password are known. The command is `ldapdelete` and the format is the like the following:

```
ldapdelete -D "[rootdn user]" -w [rootdn password] -h [hostname of server] \\
-p [ldap port] "[user1 to be deleted]" "[user2 to be deleted]" ..
```

Going back to the previous example with user `John Smith`, the following command can be used to delete an LDAP account.

```
ldapdelete -D "cn=Manager,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au" \\
-w secret -h rule241.caia.swin.edu.au \\
-p 389 "cn=jsmith,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au" \\
"cn=John Smith,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
```

Verify that the account has been deleted via `ldapsearch`.

4.3 Using root to change account passwords

Similar to the above section, the command to change a user's password (`ldappasswd`) requires the details of the `rootdn`. There are several different options that can be used that achieve the same result.

- `-a` allows for the old password to be received in the command itself.
- `-A` will make the console prompt root for the old password.
- `-s` allows for the new password to be received in the command itself.
- `-S` will make the console prompt root for the new password.

Overall, the command follows this format:

```
ldappasswd -D "[rootdn user]" -w [rootdn password] \\
-h [hostname of server] -p [ldap port] -A -S "[user]"
```

Once again, changing the password of user `jsmith` is as follows.

```
ldappasswd -D "cn=Manager,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au" \\
-w secret -h rule241.caia.swin.edu.au -p 389 -A -S \\
"cn=John Smith,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
```

4.4 How a user can change their own password

There are two methods provided to allow for a user to change their own password. The first is identical to the above method, but expects that the user has knowledge of the manager password (an ACL prevents the user from accessing any other accounts or details) and other server details. There is no method to change passwords without the use of the manager password.

4.4.1 Direct CLI Method

Below a method using `ldappasswd` to change the user passwords is shown. The user needs to know the domain of the server, the manager password and their `uid`. See the above section for the different password changing options in `ldappasswd`.

1. The user should find their `uid` using `whoami`. The account, if created by script, should have a `cn` that looks like `u_[uid]`.
2. The command below should be then executed, and the prompts it has should be filled in correctly (old and new passwords).

```
ldappasswd -D "[rootdn user]" -w [rootdn password] \\
-h [hostname of server] -p [ldap port] -A -S "[user]"
```

4.4.2 Script CLI Method

Running the script `./changepasswd` in the `/home/` directory will allow for a prompt to come up. The user should be enter their old and new passwords to change it.

5 Server Configuration Documentation

The server was configured with the following details. `/usr/local/etc/openldap/slapd.conf` was modified like below to allow for the operation of LDAP server. For brevity, only some configuration options are shown.

Listing 2: /usr/local/etc/openldap/slapd.conf

```
include /usr/local/etc/openldap/schema/core.schema
include /usr/local/etc/openldap/schema/cosine.schema
include /usr/local/etc/openldap/schema/inetorgperson.schema
include /usr/local/etc/openldap/schema/misc.schema
include /usr/local/etc/openldap/schema/nis.schema

moduleload      back_mdb

database        mdb
```

```
maxsize         1073741824
suffix          "dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
rootdn          "cn=Manager,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
```

To ensure `slapd` can be started via `service slapd start` the line `slapd_enable="YES"` was added to `/etc/rc.conf`. The two demonstration accounts were added via the following `.ldif` file and the `slapadd` command.

Listing 3: /usr/local/etc/openldap/twoacc.ldif

```
# rule241.caia.swin.edu.au
dn: dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
dc: rule241
objectClass: top
objectClass: domain
objectClass: domainRelatedObject
associatedDomain: rule241.caia.swin.edu.au

# People, rule241.caia.swin.edu.au
dn: ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
ou: People
objectClass: top
objectClass: organizationalUnit
objectClass: domainRelatedObject
associatedDomain: rule241.caia.swin.edu.au

# Group, rule241.caia.swin.edu.au
dn: ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
ou: Group
objectClass: top
objectClass: organizationalUnit
objectClass: domainRelatedObject
associatedDomain: rule241.caia.swin.edu.au

# Alan Smithee, Group, rule241.caia.swin.edu.au
dn: cn=g_asmithee,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
cn: g_asmithee
gidNumber: 6000
objectClass: posixGroup
objectClass: top

# gspelvin, Group, rule241.caia.swin.edu.au
dn: cn=g_gspelvin,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
cn: g_gspelvin
gidNumber: 6001
```

```
objectClass: posixGroup
objectClass: top

# Alan Smithee, People, rule241.caia.swin.edu.au
dn: cn=u_asmithee,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
givenName: Alan
sn: Smithee
cn: u_asmithee
uid: asmithee
objectClass: inetOrgPerson
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
uidNumber: 6000
gidNumber: 6000
homeDirectory: /home/asmithee
loginShell: /usr/local/bin/bash
userPassword: dune

# George Spelvin, People, rule241.caia.swin.edu.au
dn: cn=u_gspelvin,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
givenName: George
sn: Spelvin
cn: u_gspelvin
uid: gspelvin
objectClass: inetOrgPerson
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
uidNumber: 6001
gidNumber: 6001
homeDirectory: /home/gspelvin
loginShell: /usr/local/bin/bash
userPassword: broadway
```

To enable LDAP users to edit their own passwords, the following access control configuration should be placed in `slapd.conf`.

Listing 4: /usr/local/etc/openldap/slapd.conf

```
access to attrs=userPassword
        by self write
        by anonymous auth
        by users none

access to * by * read
```

5.1 Demonstration Account Details

As a demonstration, two accounts have been added to ensure the operation of the LDAP server, which are shown in Table **LOOK AT TABLE NUMBER**.

Table 1: Demonstration Accounts

Full Name	Alan Smithee	George Spelvin
Account Name	asmithee	gspelvin
Password	dune	broadway

6 Client Configuration Documentation

To install another work station with `root`, the following software is required to ensure LDAP can run. They should all be installed via `pkg install [package name]`

- `openldap-server-2.4.46_4`
- `pam_ldap-1.8.6_3`
- `nss_ldap-1.265_13`

After this, the first file that should be configured is `/usr/local/etc/openldap/ldap.conf`. Three lines should then be added to this file, which are listed below.

Listing 5: /usr/local/etc/openldap/ldap.conf

```
BASE    dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
URI     ldap://rule241.caia.swin.edu.au

pam_login_attribute uid
```

Save this file and then confirm that it was successful by running `ldapsearch`. If successful, it should show that the accounts `asmithee` and `gspelvin` exist and return an error code of 0.

Next, NSS LDAP must be configured, this can be done by opening the file `/usr/local/etc/nss_ldap.conf` and adding the lines as shown in Listing 2. Change the `host` and `base` fields to suit. If you are configuring the LDAP server to receive LDAP requests, also add the line that is commented out and remove the line `host 136.186.230.241`. Furthermore, to ensure the `slapd` server starts instantly (rather than taking about five minutes, add the `bind_policy` line (seen commented below).

Listing 6: /usr/local/etc/nss ldap.conf

```
#host    127.0.0.1
host     136.186.230.241
base     dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
```

```
uri       ldap://rule241.caia.swin.edu.au

#bind_policy soft
```

Moreover, the file `/etc/nsswitch.conf` must have the following lines edited. Ensure you can log in after changing the file and before logging out.

Listing 7: /etc/nsswitch.conf

```
group: files ldap
passwd: files ldap
```

Create the home directory for both users and use `ls -l` to ensure that the group and user names are appearing correctly (instead of their user IDs).

Then we must set up PAM, to do this several configuration files must be changed. It is advised that after each change is made you attempt to log into the client using another PuTTy session to ensure you have not lost access to the system. Firstly, copy `/usr/local/etc/openldap/ldap.conf` to `/usr/local/etc/`. This can easily be done in the `openldap` directory by entering `cp ldap.conf ..`, this act as the configuration file for PAM LDAP.

Thereafter move to the directory `/etc/pam.d/` and first open `system`, then `sshd` and `su`. Note that \\ is used where the line splits on the page.

The configuration for the `auth` and `account` settings must be as follows:

Listing 8: /etc/pam.d/system

```
auth sufficient pam_opie.so no_warn no_fake_prompts
auth requisite pam_opieaccess.so no_warn allow_local
auth sufficient /usr/local/lib/pam_ldap.so no_warn
auth required pam_unix.so  no_warn try_first_pass nullok

account required pam_login_access.so
account required pam_unix.so
account required /usr/local/lib/pam_ldap.so \\
    no_warn ignore_authinfo_unavail ignore_unknown_user
```

Listing 9: /etc/pam.d/sshd

```
auth sufficient pam_opie.so no_warn no_fake_prompts
auth requisite  pam_opieaccess.so no_warn allow_local
auth  sufficient /usr/local/lib/pam_ldap.so  no_warn
auth required pam_unix.so no_warn try_first_pass

account required pam_nologin.so
account required pam_login_access.so
account required pam_unix.so
account  required /usr/local/lib/pam_ldap.so \\
    no_warn ignore_authinfo_unavail ignore_unknown_user
```

Listing 10: /etc/pam.d/su

```
auth sufficient pam_rootok.so no_warn
auth sufficient pam_self.so no_warn
auth requisite pam_group.so no_warn group=wheel root
_only fail_safe ruser
auth include system

account include system
account required /usr/local/lib/pam_ldap.so \\
    no_warn ignore_authinfo_unavail ignore_unknown_user
```

Thereafter, the LDAP client should be setup. Make sure to log into all accounts (both local and LDAP based) via `ssh` and `su`.

7 Additional Features and Configuration

While the above sections deal with the creation and operation of an LDAP server, below are modifications that were made to the server to benefit the troubleshooting process.

7.1 LDAP Web Monitor

The web monitor provides an alternative method of checking the status of the LDAP server. It provides information on if the server is online and what performance it has (in essence: CPU and memory). This means that any user can check if the LDAP is reachable and if connection issues are due to the LDAP server being offline or due to another network fault. An example is a firewall misconfiguration, which could could allow HTTP through but not LDAP packets.

This page can be accessed via `http://rule241.caia.swin.edu.au`

7.1.1 Configuring the Web Monitor

To configure the `.cgi` file, it was placed in `/usr/local/www/apache24/data`. The following steps were then undertaken to make sure it works:

1. The address of the web server was set to `rule241.caia.swin.edu.au:80`.
2. `Options +ExecCGI` was added to `/usr/local/etc/apache24/http.conf` directory.
3. `AddHandler cgi-script .cgi` was added to `/usr/local/etc/apache24/http.conf` for the main directory.
4. `DirectoryIndex` was edited to `index.cgi` from `index.html`.
5. `mpm-prefork modules` was uncommented.

Thereafter the web monitor was functional.

8 Appendix (Code)

Below is the code of the four scripts; \\ is used when a line is too long for the page.

8.1 Add Users

Listing 11: addusr.sh

```
#!/usr/local/bin/bash

# addusr
# Author: Joshua Redolfi
# Add a user to the LDAP server on rule241.caia.swin.edu.au
# Takes the following arguments
# [uid] [First name] [Last name] [Password] OPTIONAL: [uidNumber]

#Get our args
uid=$1
fname=$2
lname=$3
idnum=$4

domain="dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
host="rule241.caia.swin.edu.au"
#make sure the above two match
rootdn="Manager"
rootpw=secret
port=389

if [ $1 == "help" ];
then
        echo "Command usage: ./addusr.sh [uid] [First name] \\
        [Last name] [uid number]"
        retVal=1
        return ${retVal} 2>/dev/null # this will attempt to return
        exit "${retVal}" # this will get executed if the above failed.
fi

#enter password
read -p "Enter the password: " pass

#create file tmp.ldif (and remove old one if it exists)
rm /tmp/tmp.ldif
echo 'dn: cn='g_$uid',ou=Group,'$domain'' >> /tmp/tmp.ldif
echo 'cn: 'g_$uid'' >> /tmp/tmp.ldif
```

```
echo 'gidNumber: '$idnum'' >> /tmp/tmp.ldif
echo 'objectClass: posixGroup' >> /tmp/tmp.ldif
echo 'objectClass: top' >> /tmp/tmp.ldif
echo '' >> /tmp/tmp.ldif
#now let's make the user
echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
echo 'givenName: '$fname'' >> /tmp/tmp.ldif
echo 'sn: '$lname'' >> /tmp/tmp.ldif
echo 'cn: 'u_$uid'' >> /tmp/tmp.ldif
echo 'uid: '$uid'' >> /tmp/tmp.ldif
echo 'objectClass: inetOrgPerson' >> /tmp/tmp.ldif
echo 'objectClass: top' >> /tmp/tmp.ldif
echo 'objectClass: posixAccount' >> /tmp/tmp.ldif
echo 'objectClass: shadowAccount' >> /tmp/tmp.ldif
echo 'uidNumber: '$idnum'' >> /tmp/tmp.ldif
echo 'gidNumber: '$idnum'' >> /tmp/tmp.ldif
echo 'homeDirectory: /home/'$uid'' >> /tmp/tmp.ldif
echo 'loginShell: /usr/local/bin/bash' >> /tmp/tmp.ldif
echo 'userPassword: '$pass'' >> /tmp/tmp.ldif

#add this ldif file to slapd
#slapadd -l /tmp/tmp.ldif

#adding to slapd via ldapadd
ldapadd -D cn=$rootdn,$domain -w $rootpw -h $host:$port -f /tmp/tmp.ldif

#check error code to see if user was actually added
if [ $? -eq 1 ]
then
        echo 'There was an error in adding the .ldif file'
else
        echo 'User '$uid'/'$fname' '$lname' added successfully'
fi

rm /tmp/tmp.ldif
if [ $? -eq 1 ]
then
        echo 'tmp.ldif could not be removed!'
else
        echo 'tmp.ldif removed from /tmp/'
fi
```

8.2 Remove Users

Listing 12: rmusr.sh

```
#!/usr/local/bin/bash

# rmusr
# Author: Joshua Redolfi
# Remove a user off the LDAP server on rule241.caia.swin.edu.au
# Takes the following arguments
# [uid] [First name] [Last name]

#Get our args
uid=$1

if [ $1 == "help" ];
then
        echo "Command usage: ./rmusr.sh [uid]"
        retVal=1
        return ${retVal} 2>/dev/null # this will attempt to return
        exit "${retVal}" # this will get executed if the above failed.
fi

domain="dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
host="rule241.caia.swin.edu.au"
#make sure the above two match
rootdn="Manager"
rootpw=secret
port=389

ldapdelete -D cn=$rootdn,$domain -w $rootpw -h \\
$host -p $port "cn=g_$uid,ou=Group,$domain" \\
"cn=u_$uid,ou=People,$domain"

#check error code to see if user was actually removed
if [ $? -eq 0 ]
then
        echo 'User '$uid' removed successfully'
else
        echo 'There was an error in removing the user'
fi
```

8.3 Edit Users

Listing 13: editusr.sh

```
#!/usr/local/bin/bash
```

```
# editusr
# Author: Joshua Redolfi
# Edit a user's details on the LDAP server \\
on rule241.caia.swin.edu.au

cname="u_"$name
domain="dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
host="rule241.caia.swin.edu.au"
#make sure the above two match
rootdn="Manager"
rootpw=secret
port=389

#control statement for options \\
(help, password, name, shell, home)
if [ $1 == "help" ]
then
        echo "Command usage: ./editusr.sh [option]"
        echo "password - change a user's password"
        echo "name - change a user's name \\
        (first name and last name)"
        echo "shell - change a user's shell (bash, csh and sh)"
        echo "home - change a user's home directory"
        retVal=1
        return ${retVal} 2>/dev/null # this will attempt to return
        exit "${retVal}" # this will get \\
        executed if the above failed.
elif [ $1 == "password" ]
        then
        echo "Enter user id to change password for"
        read -p "User ID:" uid
        ldappasswd -D cn=$rootdn,$domain -w $rootpw -h $host \\
        -p $port -A -S cn="u_"$uid,ou=People,$domain
        #check error code to see if password was actually changed
        if [ $? -eq 0 ]
        then
                echo 'User password changed successfully'
        else
                echo 'There was an error in changing the password'
        fi
elif [ $1 == "name" ]
        then
        echo "Enter user id of the account to change the name of"
        read -p "User ID:" uid
        rm /tmp/tmp.ldif
        echo "Enter the new first name:"
```

```
        read -p "First name:" fname
        echo "Enter the new surname:"
        read -p "Surname:" lname
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        echo 'changetype: modify' >> /tmp/tmp.ldif
        echo 'replace: givenName' >> /tmp/tmp.ldif
        echo 'givenName: '$fname'' >> /tmp/tmp.ldif
        #modify first name first, done in two steps because \\
        ldapmodify doesnt like it otherwise
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        echo 'changetype: modify' >> /tmp/tmp.ldif
        echo 'replace: sn' >> /tmp/tmp.ldif
        echo 'sn: '$lname'' >> /tmp/tmp.ldif
        #then modify last name
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
elif [ $1 == "shell" ]
        then
        echo "Enter user id of the account to change the shell of"
        read -p "User ID:" uid
        echo "Type name of the shell you wish \\
        to change the user shell to:"
        echo "Options: bash, sh, csh"
        read -p "Shell:" shell
        rm /tmp/tmp.ldif
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        #control statement for shells, reject if invalid
        if [ $shell == "bash" ]
        then
                echo 'changetype: modify' >> /tmp/tmp.ldif
                echo 'replace: loginShell' >> /tmp/tmp.ldif
                echo 'loginShell: /usr/local/bin/bash' >> /tmp/tmp.ldif
        elif [ $shell = "sh" ]
        then
                echo 'changetype: modify' >> /tmp/tmp.ldif
                echo 'replace: loginShell' >> /tmp/tmp.ldif
                echo 'loginShell: /bin/sh' >> /tmp/tmp.ldif
        elif [ $shell = "csh" ]
        then
                echo 'changetype: modify' >> /tmp/tmp.ldif
                echo 'replace: loginShell' >> /tmp/tmp.ldif
                echo 'loginShell: /bin/csh' >> /tmp/tmp.ldif
```

```
        else
                echo "Invalid option, won't change shell type"
                echo "Exiting..."
                retVal=1
                return ${retVal} 2>/dev/null # this will \\
                attempt to return
                exit "${retVal}" # this will get executed \\
                if the above failed.
        fi
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
elif [ $1 == "home" ]
then
        echo "Enter user id of the account to \\
        change the home directory of"
        read -p "User ID:" uid
        echo "Enter the new directory (absolute reference):"
        echo "Example: /newlocation/"
        read -p "Home directory:" home
        rm /tmp/tmp.ldif
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        echo 'changetype: modify' >> /tmp/tmp.ldif
        echo 'replace: homeDirectory' >> /tmp/tmp.ldif
        echo 'homeDirectory: '$home'' >> /tmp/tmp.ldif
        #modify home directory here
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
else
        echo "Option invalid, type \\
        './editusr.sh help' to see all options."
fi
```

8.4 Change Passwords

Listing 14: changepasswd.sh

```
#!/usr/local/bin/bash

# changepasswd
# Author: Joshua Redolfi
# Edit a user's password on the LDAP \\
server on rule241.caia.swin.edu.au
# Takes no arguments
```

```
name=$(whoami)

domain='dc=rule241,dc=caia,dc=swin,dc=edu,dc=au'
host=rule241.caia.swin.edu.au
#make sure the above two match
rootdn=Manager
rootpw=secret
port=389

ldappasswd -D cn=$rootdn,$domain -w $rootpw -h \\
$host -p $port -A -S cn="u_"$name,ou=People,$domain

#check error code to see if password was actually changed
if [ $? -eq 0 ]
then
        echo 'User_'$name'_password_successfully'
else
        echo 'There_was_an_error_in_changing_the_password'
fi


#ldappasswd -D cn=Manager,dc=rule241,dc=caia,dc=swin, \\
dc=edu,dc=au -w secret -h rule241.caia.swin.edu.au \\
-p 389 -A -S cn=asmithee,ou=People,dc=rule241, \\
dc=caia,dc=swin,dc=edu,dc=au
```

LDAP Project Documentation

Joshua Redolfi
Faculty of Science,
Engineering and, Technology
Swinburne University of Technology
Email: 101601886@student.swin.edu.au

Semester 2 2019

Abstract

This document describes the operation of the LDAP (Light-weight Directory Access Protocol) on the RULE host rule241.caia.swin.edu.au. It contains information on how the server was configured and how to configure clients to connect to it. Thereafter, account and script details are provided to show how the system has been automated. Finally, the operation of an LDAP web monitor is provided. **This document contains all the documentation required for the project**.

Contents

1 Introduction

This documentation contains information of the LDAP server set up on the following hosts:

Server Address: 136.186.230.241 (rule241.caia.swin.edu.au)
Client Address: 136.186.230.240 (rule240.caia.swin.edu.au)

2 Location of Files and CLI Scripts

This contains the locations of the files that were edited and created for the use of LDAP. It also has the locations of all the scripts for adding, editing and removing user accounts, along with the ability for users to change their passwords.

2.1 Server Locations

SLAPD config: `/usr/local/etc/openldap/slapd.conf`
Initial .ldif file: `/usr/local/etc/openldap/twoacc.ldif`
LDAP config: `/usr/local/etc/openldap/ldap.conf`
PAM-LDAP config: `/usr/local/etc/ldap.conf`
NSS-LDAP config: `/usr/local/etc/nss_ldap.conf`
NSS config: `/etc/nsswitch.conf`
PAM config (system): `/etc/pam.d/system`
PAM config (ssh): `/etc/pam.d/sshd`
PAM config (su): `/etc/pam.d/su`
Apache config: `/usr/local/etc/apache24/http.conf`
Web Monitor: `/usr/local/www/apache24/data/index.cgi`

2.2 Client Locations

LDAP config: `/usr/local/etc/openldap/ldap.conf`
PAM-LDAP config: `/usr/local/etc/ldap.conf`
NSS-LDAP config: `/usr/local/etc/nss_ldap.conf`
NSS config: `/etc/nsswitch.conf`
PAM config (system): `/etc/pam.d/system`
PAM config (ssh): `/etc/pam.d/sshd`
PAM config (su): `/etc/pam.d/su`

2.3 Script Locations

Adding users (root): `/root/addusr.sh`
Removing users (root): `/root/rmusr.sh`
Editing users (root): `/root/editusr.sh`
Changing passwords (user): `/home/changepasswd.sh`

3 Script Commands

To assist in the operating LDAP, several script commands have been provided to save time on tasks. The code for these scripts can be found in the Appendix.

3.1 Adding Accounts

To add an account, use `addusr.sh`, it is solely for root and located in `/root/`. It can be executed as follows: `/root/addusr.sh [uid] [First Name] [Last Name] \\ [Group Number/User Number]` (remove the \\ when in the console). Thereafter it will ask for a password.

3.2 Removing Accounts

To remove an account, use `rmusr.sh`, which is located in `/root/` and can only be used by root. To execute it, enter the command like the following:

```
/root/rmusr.sh [uid] [First Name] [Last Name]
```

3.3 Editing Accounts

Root is also able to modify account details using the script `editusr.sh`. Like the other scripts intended for use by `root`, it can be found in `/root/`. It can be executed with `editusr.sh [option]` and the script will prompt root for the `uid` of the user and other needed information. The options and what they change are shown below:

- `name` - changes the user's `givenName` and `sn`, not their `uid`.
- `home` - changes the user home directory.
- `password` - chnages the user password.
- `shell` - changes the shell between `bash`, `csh` and `sh`.
- `help` - shows how to use the command.

3.4 Changing User Passwords

This script is for the use of a non-root user to change their own password. It can be executed by the use of `./changepasswd`; it will return an error if a local account attempts to use it.

4 Instructions

Below are the instructions to use root to modify accounts and how a user can change their account password.

4.1 Using root to add user accounts

To create a account via `root`, firstly a `.ldif` file must created with the details of the account. This file should contain both a group and user identity, but the former can be ignored if you are using a preexisting group. The following fields must configured:

- `gidNumber`, group number
- `uid`, user ID
- `uidNunmber`, user number
- `cn`, entry/full name
- `givenName`, first name
- `sn`, surname

Below an example configuration is shown. This will create an account with the user ID `jsmith` and a member of the group `jsmith`. It is notable that the group ID of an account can be set to 0, meaning it will be a member of the `wheel` group, which means it can use the `su` command.

To work with the scripts provided, the `cn` of an account **must be equal** to the uid appended with a '`u_`'. For example: `u_[uid]`. For groups the `cn` is equal to `g_[uid]`.

Listing 1: /usr/local/etc/openldap/test.ldif

```
# John Smith, Group, rule241.caia.swin.edu.au
dn: cn=jsmith,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
cn: g_jsmith
gidNumber: 6002
objectClass: posixGroup
objectClass: top

# Johm Smith, People, rule241.caia.swin.edu.au
dn: cn=John Smith,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
givenName: John
sn: Smith
cn: u_jsmith
uid: jsmith
objectClass: inetOrgPerson
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
uidNumber: 6002
gidNumber: 6002
```

```
homeDirectory: /home/jsmith
loginShell: /usr/local/bin/bash
userPassword: test
```

Thereafter, this account can be added by the use of `slapadd -l test.ldif`. Thereafter check that the account has been added by the use of `ldapsearch`.

4.2 Using root to remove user accounts

Root can delete an LDAP account if the `rootdn` username and password are known. The command is `ldapdelete` and the format is the like the following:

```
ldapdelete -D "[rootdn user]" -w [rootdn password] -h [hostname of server] \\
-p [ldap port] "[user1 to be deleted]" "[user2 to be deleted]" ..
```

Going back to the previous example with user `John Smith`, the following command can be used to delete an LDAP account.

```
ldapdelete -D "cn=Manager,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au" \\
-w secret -h rule241.caia.swin.edu.au \\
-p 389 "cn=jsmith,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au" \\
"cn=John Smith,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
```

Verify that the account has been deleted via `ldapsearch`.

4.3 Using root to change account passwords

Similar to the above section, the command to change a user's password (`ldappasswd`) requires the details of the `rootdn`. There are several different options that can be used that achieve the same result.

- `-a` allows for the old password to be received in the command itself.
- `-A` will make the console prompt root for the old password.
- `-s` allows for the new password to be received in the command itself.
- `-S` will make the console prompt root for the new password.

Overall, the command follows this format:

```
ldappasswd -D "[rootdn user]" -w [rootdn password] \\
-h [hostname of server] -p [ldap port] -A -S "[user]"
```

Once again, changing the password of user `jsmith` is as follows.

```
ldappasswd -D "cn=Manager,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au" \\
-w secret -h rule241.caia.swin.edu.au -p 389 -A -S \\
"cn=John Smith,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
```

4.4 How a user can change their own password

There are two methods provided to allow for a user to change their own password. The first is identical to the above method, but expects that the user has knowledge of the manager password (an ACL prevents the user from accessing any other accounts or details) and other server details. There is no method to change passwords without the use of the manager password.

4.4.1 Direct CLI Method

Below a method using `ldappasswd` to change the user passwords is shown. The user needs to know the domain of the server, the manager password and their `uid`. See the above section for the different password changing options in `ldappasswd`.

1. The user should find their `uid` using `whoami`. The account, if created by script, should have a `cn` that looks like `u_[uid]`.
2. The command below should be then executed, and the prompts it has should be filled in correctly (old and new passwords).

```
ldappasswd -D "[rootdn user]" -w [rootdn password] \\
-h [hostname of server] -p [ldap port] -A -S "[user]"
```

4.4.2 Script CLI Method

Running the script `./changepasswd` in the `/home/` directory will allow for a prompt to come up. The user should be enter their old and new passwords to change it.

5 Server Configuration Documentation

The server was configured with the following details. `/usr/local/etc/openldap/slapd.conf` was modified like below to allow for the operation of LDAP server. For brevity, only some configuration options are shown.

Listing 2: /usr/local/etc/openldap/slapd.conf

```
include /usr/local/etc/openldap/schema/core.schema
include /usr/local/etc/openldap/schema/cosine.schema
include /usr/local/etc/openldap/schema/inetorgperson.schema
include /usr/local/etc/openldap/schema/misc.schema
include /usr/local/etc/openldap/schema/nis.schema

moduleload      back_mdb

database        mdb
```

```
maxsize         1073741824
suffix          "dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
rootdn          "cn=Manager,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
```

To ensure `slapd` can be started via `service slapd start` the line `slapd_enable="YES"` was added to `/etc/rc.conf`. The two demonstration accounts were added via the following `.ldif` file and the `slapadd` command.

Listing 3: /usr/local/etc/openldap/twoacc.ldif

```
# rule241.caia.swin.edu.au
dn: dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
dc: rule241
objectClass: top
objectClass: domain
objectClass: domainRelatedObject
associatedDomain: rule241.caia.swin.edu.au

# People, rule241.caia.swin.edu.au
dn: ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
ou: People
objectClass: top
objectClass: organizationalUnit
objectClass: domainRelatedObject
associatedDomain: rule241.caia.swin.edu.au

# Group, rule241.caia.swin.edu.au
dn: ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
ou: Group
objectClass: top
objectClass: organizationalUnit
objectClass: domainRelatedObject
associatedDomain: rule241.caia.swin.edu.au

# Alan Smithee, Group, rule241.caia.swin.edu.au
dn: cn=g_asmithee,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
cn: g_asmithee
gidNumber: 6000
objectClass: posixGroup
objectClass: top

# gspelvin, Group, rule241.caia.swin.edu.au
dn: cn=g_gspelvin,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
cn: g_gspelvin
gidNumber: 6001
```

```
objectClass: posixGroup
objectClass: top

# Alan Smithee, People, rule241.caia.swin.edu.au
dn: cn=u_asmithee,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
givenName: Alan
sn: Smithee
cn: u_asmithee
uid: asmithee
objectClass: inetOrgPerson
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
uidNumber: 6000
gidNumber: 6000
homeDirectory: /home/asmithee
loginShell: /usr/local/bin/bash
userPassword: dune

# George Spelvin, People, rule241.caia.swin.edu.au
dn: cn=u_gspelvin,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
givenName: George
sn: Spelvin
cn: u_gspelvin
uid: gspelvin
objectClass: inetOrgPerson
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
uidNumber: 6001
gidNumber: 6001
homeDirectory: /home/gspelvin
loginShell: /usr/local/bin/bash
userPassword: broadway
```

To enable LDAP users to edit their own passwords, the following access control configuration should be placed in `slapd.conf`.

Listing 4: /usr/local/etc/openldap/slapd.conf

```
access to attrs=userPassword
        by self write
        by anonymous auth
        by users none

access to * by * read
```

5.1 Demonstration Account Details

As a demonstration, two accounts have been added to ensure the operation of the LDAP server, which are shown in Table **LOOK AT TABLE NUMBER**.

Table 1: Demonstration Accounts

Full Name	Alan Smithee	George Spelvin
Account Name	asmithee	gspelvin
Password	dune	broadway

6 Client Configuration Documentation

To install another work station with `root`, the following software is required to ensure LDAP can run. They should all be installed via `pkg install [package name]`

- `openldap-server-2.4.46_4`
- `pam_ldap-1.8.6_3`
- `nss_ldap-1.265_13`

After this, the first file that should be configured is `/usr/local/etc/openldap/ldap.conf`. Three lines should then be added to this file, which are listed below.

Listing 5: /usr/local/etc/openldap/ldap.conf

```
BASE    dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
URI     ldap://rule241.caia.swin.edu.au

pam_login_attribute uid
```

Save this file and then confirm that it was successful by running `ldapsearch`. If successful, it should show that the accounts `asmithee` and `gspelvin` exist and return an error code of 0.

Next, NSS LDAP must be configured, this can be done by opening the file `/usr/local/etc/nss_ldap.conf` and adding the lines as shown in Listing 2. Change the `host` and `base` fields to suit. If you are configuring the LDAP server to receive LDAP requests, also add the line that is commented out and remove the line `host 136.186.230.241`. Furthermore, to ensure the `slapd` server starts instantly (rather than taking about five minutes, add the `bind_policy` line (seen commented below).

Listing 6: /usr/local/etc/nss ldap.conf

```
#host    127.0.0.1
host     136.186.230.241
base     dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
```

```
uri       ldap://rule241.caia.swin.edu.au

#bind_policy soft
```

Moreover, the file `/etc/nsswitch.conf` must have the following lines edited. Ensure you can log in after changing the file and before logging out.

Listing 7: /etc/nsswitch.conf

```
group: files ldap
passwd: files ldap
```

Create the home directory for both users and use `ls -l` to ensure that the group and user names are appearing correctly (instead of their user IDs).

Then we must set up PAM, to do this several configuration files must be changed. It is advised that after each change is made you attempt to log into the client using another PuTTy session to ensure you have not lost access to the system. Firstly, copy `/usr/local/etc/openldap/ldap.conf` to `/usr/local/etc/`. This can easily be done in the `openldap` directory by entering `cp ldap.conf ..`, this act as the configuration file for PAM LDAP.

Thereafter move to the directory `/etc/pam.d/` and first open `system`, then `sshd` and `su`. Note that \\ is used where the line splits on the page.

The configuration for the `auth` and `account` settings must be as follows:

Listing 8: /etc/pam.d/system

```
auth sufficient pam_opie.so no_warn no_fake_prompts
auth requisite pam_opieaccess.so no_warn allow_local
auth sufficient /usr/local/lib/pam_ldap.so no_warn
auth required pam_unix.so  no_warn try_first_pass nullok

account required pam_login_access.so
account required pam_unix.so
account required /usr/local/lib/pam_ldap.so \\
    no_warn ignore_authinfo_unavail ignore_unknown_user
```

Listing 9: /etc/pam.d/sshd

```
auth sufficient pam_opie.so no_warn no_fake_prompts
auth requisite  pam_opieaccess.so no_warn allow_local
auth  sufficient /usr/local/lib/pam_ldap.so  no_warn
auth required pam_unix.so no_warn try_first_pass

account required pam_nologin.so
account required pam_login_access.so
account required pam_unix.so
account  required /usr/local/lib/pam_ldap.so \\
    no_warn ignore_authinfo_unavail ignore_unknown_user
```

Listing 10: /etc/pam.d/su

```
auth sufficient pam_rootok.so no_warn
auth sufficient pam_self.so no_warn
auth requisite pam_group.so no_warn group=wheel root
_only fail_safe ruser
auth include system

account include system
account required /usr/local/lib/pam_ldap.so \\
    no_warn ignore_authinfo_unavail ignore_unknown_user
```

Thereafter, the LDAP client should be setup. Make sure to log into all accounts (both local and LDAP based) via `ssh` and `su`.

7 Additional Features and Configuration

While the above sections deal with the creation and operation of an LDAP server, below are modifications that were made to the server to benefit the troubleshooting process.

7.1 LDAP Web Monitor

The web monitor provides an alternative method of checking the status of the LDAP server. It provides information on if the server is online and what performance it has (in essence: CPU and memory). This means that any user can check if the LDAP is reachable and if connection issues are due to the LDAP server being offline or due to another network fault. An example is a firewall misconfiguration, which could could allow HTTP through but not LDAP packets.

This page can be accessed via `http://rule241.caia.swin.edu.au`

7.1.1 Configuring the Web Monitor

To configure the `.cgi` file, it was placed in `/usr/local/www/apache24/data`. The following steps were then undertaken to make sure it works:

1. The address of the web server was set to `rule241.caia.swin.edu.au:80`.
2. `Options +ExecCGI` was added to `/usr/local/etc/apache24/http.conf` directory.
3. `AddHandler cgi-script .cgi` was added to `/usr/local/etc/apache24/http.conf` for the main directory.
4. `DirectoryIndex` was edited to `index.cgi` from `index.html`.
5. `mpm-prefork modules` was uncommented.

Thereafter the web monitor was functional.

8 Appendix (Code)

Below is the code of the four scripts; \\ is used when a line is too long for the page.

8.1 Add Users

Listing 11: addusr.sh

```
#!/usr/local/bin/bash

# addusr
# Author: Joshua Redolfi
# Add a user to the LDAP server on rule241.caia.swin.edu.au
# Takes the following arguments
# [uid] [First name] [Last name] [Password] OPTIONAL: [uidNumber]

#Get our args
uid=$1
fname=$2
lname=$3
idnum=$4

domain="dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
host="rule241.caia.swin.edu.au"
#make sure the above two match
rootdn="Manager"
rootpw=secret
port=389

if [ $1 == "help" ];
then
        echo "Command usage: ./addusr.sh [uid] [First name] \\
        [Last name] [uid number]"
        retVal=1
        return ${retVal} 2>/dev/null # this will attempt to return
        exit "${retVal}" # this will get executed if the above failed.
fi

#enter password
read -p "Enter the password: " pass

#create file tmp.ldif (and remove old one if it exists)
rm /tmp/tmp.ldif
echo 'dn: cn='g_$uid',ou=Group,'$domain'' >> /tmp/tmp.ldif
echo 'cn: 'g_$uid'' >> /tmp/tmp.ldif
```

```
echo 'gidNumber: '$idnum'' >> /tmp/tmp.ldif
echo 'objectClass: posixGroup' >> /tmp/tmp.ldif
echo 'objectClass: top' >> /tmp/tmp.ldif
echo '' >> /tmp/tmp.ldif
#now let's make the user
echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
echo 'givenName: '$fname'' >> /tmp/tmp.ldif
echo 'sn: '$lname'' >> /tmp/tmp.ldif
echo 'cn: 'u_$uid'' >> /tmp/tmp.ldif
echo 'uid: '$uid'' >> /tmp/tmp.ldif
echo 'objectClass: inetOrgPerson' >> /tmp/tmp.ldif
echo 'objectClass: top' >> /tmp/tmp.ldif
echo 'objectClass: posixAccount' >> /tmp/tmp.ldif
echo 'objectClass: shadowAccount' >> /tmp/tmp.ldif
echo 'uidNumber: '$idnum'' >> /tmp/tmp.ldif
echo 'gidNumber: '$idnum'' >> /tmp/tmp.ldif
echo 'homeDirectory: /home/'$uid'' >> /tmp/tmp.ldif
echo 'loginShell: /usr/local/bin/bash' >> /tmp/tmp.ldif
echo 'userPassword: '$pass'' >> /tmp/tmp.ldif

#add this ldif file to slapd
#slapadd -l /tmp/tmp.ldif

#adding to slapd via ldapadd
ldapadd -D cn=$rootdn,$domain -w $rootpw -h $host:$port -f /tmp/tmp.ldif

#check error code to see if user was actually added
if [ $? -eq 1 ]
then
        echo 'There was an error in adding the .ldif file'
else
        echo 'User '$uid'/'$fname' '$lname' added successfully'
fi

rm /tmp/tmp.ldif
if [ $? -eq 1 ]
then
        echo 'tmp.ldif could not be removed!'
else
        echo 'tmp.ldif removed from /tmp/'
fi
```

8.2 Remove Users

Listing 12: rmusr.sh

```
#!/usr/local/bin/bash

# rmusr
# Author: Joshua Redolfi
# Remove a user off the LDAP server on rule241.caia.swin.edu.au
# Takes the following arguments
# [uid] [First name] [Last name]

#Get our args
uid=$1

if [ $1 == "help" ];
then
        echo "Command usage: ./rmusr.sh [uid]"
        retVal=1
        return ${retVal} 2>/dev/null # this will attempt to return
        exit "${retVal}" # this will get executed if the above failed.
fi

domain="dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
host="rule241.caia.swin.edu.au"
#make sure the above two match
rootdn="Manager"
rootpw=secret
port=389

ldapdelete -D cn=$rootdn,$domain -w $rootpw -h \\
$host -p $port "cn=g_$uid,ou=Group,$domain" \\
"cn=u_$uid,ou=People,$domain"

#check error code to see if user was actually removed
if [ $? -eq 0 ]
then
        echo 'User '$uid' removed successfully'
else
        echo 'There was an error in removing the user'
fi
```

8.3 Edit Users

Listing 13: editusr.sh

```
#!/usr/local/bin/bash
```

```
# editusr
# Author: Joshua Redolfi
# Edit a user's details on the LDAP server \\
on rule241.caia.swin.edu.au

cname="u_"$name
domain="dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
host="rule241.caia.swin.edu.au"
#make sure the above two match
rootdn="Manager"
rootpw=secret
port=389

#control statement for options \\
(help, password, name, shell, home)
if [ $1 == "help" ]
then
        echo "Command usage: ./editusr.sh [option]"
        echo "password - change a user's password"
        echo "name - change a user's name \\
        (first name and last name)"
        echo "shell - change a user's shell (bash, csh and sh)"
        echo "home - change a user's home directory"
        retVal=1
        return ${retVal} 2>/dev/null # this will attempt to return
        exit "${retVal}" # this will get \\
        executed if the above failed.
elif [ $1 == "password" ]
        then
        echo "Enter user id to change password for"
        read -p "User ID:" uid
        ldappasswd -D cn=$rootdn,$domain -w $rootpw -h $host \\
        -p $port -A -S cn="u_"$uid,ou=People,$domain
        #check error code to see if password was actually changed
        if [ $? -eq 0 ]
        then
                echo 'User password changed successfully'
        else
                echo 'There was an error in changing the password'
        fi
elif [ $1 == "name" ]
        then
        echo "Enter user id of the account to change the name of"
        read -p "User ID:" uid
        rm /tmp/tmp.ldif
        echo "Enter the new first name:"
```

```
        read -p "First name:" fname
        echo "Enter the new surname:"
        read -p "Surname:" lname
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        echo 'changetype: modify' >> /tmp/tmp.ldif
        echo 'replace: givenName' >> /tmp/tmp.ldif
        echo 'givenName: '$fname'' >> /tmp/tmp.ldif
        #modify first name first, done in two steps because \\
        ldapmodify doesnt like it otherwise
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        echo 'changetype: modify' >> /tmp/tmp.ldif
        echo 'replace: sn' >> /tmp/tmp.ldif
        echo 'sn: '$lname'' >> /tmp/tmp.ldif
        #then modify last name
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
elif [ $1 == "shell" ]
        then
        echo "Enter user id of the account to change the shell of"
        read -p "User ID:" uid
        echo "Type name of the shell you wish \\
        to change the user shell to:"
        echo "Options: bash, sh, csh"
        read -p "Shell:" shell
        rm /tmp/tmp.ldif
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        #control statement for shells, reject if invalid
        if [ $shell == "bash" ]
        then
                echo 'changetype: modify' >> /tmp/tmp.ldif
                echo 'replace: loginShell' >> /tmp/tmp.ldif
                echo 'loginShell: /usr/local/bin/bash' >> /tmp/tmp.ldif
        elif [ $shell = "sh" ]
        then
                echo 'changetype: modify' >> /tmp/tmp.ldif
                echo 'replace: loginShell' >> /tmp/tmp.ldif
                echo 'loginShell: /bin/sh' >> /tmp/tmp.ldif
        elif [ $shell = "csh" ]
        then
                echo 'changetype: modify' >> /tmp/tmp.ldif
                echo 'replace: loginShell' >> /tmp/tmp.ldif
                echo 'loginShell: /bin/csh' >> /tmp/tmp.ldif
```

```
        else
                echo "Invalid option, won't change shell type"
                echo "Exiting..."
                retVal=1
                return ${retVal} 2>/dev/null # this will \\
                attempt to return
                exit "${retVal}" # this will get executed \\
                if the above failed.
        fi
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
elif [ $1 == "home" ]
then
        echo "Enter user id of the account to \\
        change the home directory of"
        read -p "User ID:" uid
        echo "Enter the new directory (absolute reference):"
        echo "Example: /newlocation/"
        read -p "Home directory:" home
        rm /tmp/tmp.ldif
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        echo 'changetype: modify' >> /tmp/tmp.ldif
        echo 'replace: homeDirectory' >> /tmp/tmp.ldif
        echo 'homeDirectory: '$home'' >> /tmp/tmp.ldif
        #modify home directory here
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
else
        echo "Option invalid, type \\
        './editusr.sh help' to see all options."
fi
```

8.4 Change Passwords

Listing 14: changepasswd.sh

```
#!/usr/local/bin/bash

# changepasswd
# Author: Joshua Redolfi
# Edit a user's password on the LDAP \\
server on rule241.caia.swin.edu.au
# Takes no arguments
```

```
name=$(whoami)

domain='dc=rule241,dc=caia,dc=swin,dc=edu,dc=au'
host=rule241.caia.swin.edu.au
#make sure the above two match
rootdn=Manager
rootpw=secret
port=389

ldappasswd -D cn=$rootdn,$domain -w $rootpw -h \\
$host -p $port -A -S cn="u_"$name,ou=People,$domain

#check error code to see if password was actually changed
if [ $? -eq 0 ]
then
        echo 'User_'$name'_password_successfully'
else
        echo 'There_was_an_error_in_changing_the_password'
fi


#ldappasswd -D cn=Manager,dc=rule241,dc=caia,dc=swin, \\
dc=edu,dc=au -w secret -h rule241.caia.swin.edu.au \\
-p 389 -A -S cn=asmithee,ou=People,dc=rule241, \\
dc=caia,dc=swin,dc=edu,dc=au
```

LDAP Project Documentation

Joshua Redolfi
Faculty of Science,
Engineering and, Technology
Swinburne University of Technology
Email: 101601886@student.swin.edu.au

Semester 2 2019

Abstract

This document describes the operation of the LDAP (Light-weight Directory Access Protocol) on the RULE host rule241.caia.swin.edu.au. It contains information on how the server was configured and how to configure clients to connect to it. Thereafter, account and script details are provided to show how the system has been automated. Finally, the operation of an LDAP web monitor is provided. **This document contains all the documentation required for the project**.

Contents

1 Introduction

This documentation contains information of the LDAP server set up on the following hosts:

Server Address: 136.186.230.241 (rule241.caia.swin.edu.au)
Client Address: 136.186.230.240 (rule240.caia.swin.edu.au)

2 Location of Files and CLI Scripts

This contains the locations of the files that were edited and created for the use of LDAP. It also has the locations of all the scripts for adding, editing and removing user accounts, along with the ability for users to change their passwords.

2.1 Server Locations

SLAPD config: `/usr/local/etc/openldap/slapd.conf`
Initial .ldif file: `/usr/local/etc/openldap/twoacc.ldif`
LDAP config: `/usr/local/etc/openldap/ldap.conf`
PAM-LDAP config: `/usr/local/etc/ldap.conf`
NSS-LDAP config: `/usr/local/etc/nss_ldap.conf`
NSS config: `/etc/nsswitch.conf`
PAM config (system): `/etc/pam.d/system`
PAM config (ssh): `/etc/pam.d/sshd`
PAM config (su): `/etc/pam.d/su`
Apache config: `/usr/local/etc/apache24/http.conf`
Web Monitor: `/usr/local/www/apache24/data/index.cgi`

2.2 Client Locations

LDAP config: `/usr/local/etc/openldap/ldap.conf`
PAM-LDAP config: `/usr/local/etc/ldap.conf`
NSS-LDAP config: `/usr/local/etc/nss_ldap.conf`
NSS config: `/etc/nsswitch.conf`
PAM config (system): `/etc/pam.d/system`
PAM config (ssh): `/etc/pam.d/sshd`
PAM config (su): `/etc/pam.d/su`

2.3 Script Locations

Adding users (root): `/root/addusr.sh`
Removing users (root): `/root/rmusr.sh`
Editing users (root): `/root/editusr.sh`
Changing passwords (user): `/home/changepasswd.sh`

3 Script Commands

To assist in the operating LDAP, several script commands have been provided to save time on tasks. The code for these scripts can be found in the Appendix.

3.1 Adding Accounts

To add an account, use `addusr.sh`, it is solely for root and located in `/root/`. It can be executed as follows: `/root/addusr.sh [uid] [First Name] [Last Name] \\ [Group Number/User Number]` (remove the \\ when in the console). Thereafter it will ask for a password.

3.2 Removing Accounts

To remove an account, use `rmusr.sh`, which is located in `/root/` and can only be used by root. To execute it, enter the command like the following:

```
/root/rmusr.sh [uid] [First Name] [Last Name]
```

3.3 Editing Accounts

Root is also able to modify account details using the script `editusr.sh`. Like the other scripts intended for use by `root`, it can be found in `/root/`. It can be executed with `editusr.sh [option]` and the script will prompt root for the `uid` of the user and other needed information. The options and what they change are shown below:

- `name` - changes the user's `givenName` and `sn`, not their `uid`.
- `home` - changes the user home directory.
- `password` - chnages the user password.
- `shell` - changes the shell between `bash`, `csh` and `sh`.
- `help` - shows how to use the command.

3.4 Changing User Passwords

This script is for the use of a non-root user to change their own password. It can be executed by the use of `./changepasswd`; it will return an error if a local account attempts to use it.

4 Instructions

Below are the instructions to use root to modify accounts and how a user can change their account password.

4.1 Using root to add user accounts

To create a account via `root`, firstly a `.ldif` file must created with the details of the account. This file should contain both a group and user identity, but the former can be ignored if you are using a preexisting group. The following fields must configured:

- `gidNumber`, group number
- `uid`, user ID
- `uidNunmber`, user number
- `cn`, entry/full name
- `givenName`, first name
- `sn`, surname

Below an example configuration is shown. This will create an account with the user ID `jsmith` and a member of the group `jsmith`. It is notable that the group ID of an account can be set to 0, meaning it will be a member of the `wheel` group, which means it can use the `su` command.

To work with the scripts provided, the `cn` of an account **must be equal** to the uid appended with a '`u_`'. For example: `u_[uid]`. For groups the `cn` is equal to `g_[uid]`.

Listing 1: /usr/local/etc/openldap/test.ldif

```
# John Smith, Group, rule241.caia.swin.edu.au
dn: cn=jsmith,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
cn: g_jsmith
gidNumber: 6002
objectClass: posixGroup
objectClass: top

# Johm Smith, People, rule241.caia.swin.edu.au
dn: cn=John Smith,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
givenName: John
sn: Smith
cn: u_jsmith
uid: jsmith
objectClass: inetOrgPerson
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
uidNumber: 6002
gidNumber: 6002
```

```
homeDirectory: /home/jsmith
loginShell: /usr/local/bin/bash
userPassword: test
```

Thereafter, this account can be added by the use of `slapadd -l test.ldif`. Thereafter check that the account has been added by the use of `ldapsearch`.

4.2 Using root to remove user accounts

Root can delete an LDAP account if the `rootdn` username and password are known. The command is `ldapdelete` and the format is the like the following:

```
ldapdelete -D "[rootdn user]" -w [rootdn password] -h [hostname of server] \\
-p [ldap port] "[user1 to be deleted]" "[user2 to be deleted]" ..
```

Going back to the previous example with user `John Smith`, the following command can be used to delete an LDAP account.

```
ldapdelete -D "cn=Manager,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au" \\
-w secret -h rule241.caia.swin.edu.au \\
-p 389 "cn=jsmith,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au" \\
"cn=John Smith,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
```

Verify that the account has been deleted via `ldapsearch`.

4.3 Using root to change account passwords

Similar to the above section, the command to change a user's password (`ldappasswd`) requires the details of the `rootdn`. There are several different options that can be used that achieve the same result.

- `-a` allows for the old password to be received in the command itself.
- `-A` will make the console prompt root for the old password.
- `-s` allows for the new password to be received in the command itself.
- `-S` will make the console prompt root for the new password.

Overall, the command follows this format:

```
ldappasswd -D "[rootdn user]" -w [rootdn password] \\
-h [hostname of server] -p [ldap port] -A -S "[user]"
```

Once again, changing the password of user `jsmith` is as follows.

```
ldappasswd -D "cn=Manager,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au" \\
-w secret -h rule241.caia.swin.edu.au -p 389 -A -S \\
"cn=John Smith,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
```

4.4 How a user can change their own password

There are two methods provided to allow for a user to change their own password. The first is identical to the above method, but expects that the user has knowledge of the manager password (an ACL prevents the user from accessing any other accounts or details) and other server details. There is no method to change passwords without the use of the manager password.

4.4.1 Direct CLI Method

Below a method using `ldappasswd` to change the user passwords is shown. The user needs to know the domain of the server, the manager password and their `uid`. See the above section for the different password changing options in `ldappasswd`.

1. The user should find their `uid` using `whoami`. The account, if created by script, should have a `cn` that looks like `u_[uid]`.
2. The command below should be then executed, and the prompts it has should be filled in correctly (old and new passwords).

```
ldappasswd -D "[rootdn user]" -w [rootdn password] \\
-h [hostname of server] -p [ldap port] -A -S "[user]"
```

4.4.2 Script CLI Method

Running the script `./changepasswd` in the `/home/` directory will allow for a prompt to come up. The user should be enter their old and new passwords to change it.

5 Server Configuration Documentation

The server was configured with the following details. `/usr/local/etc/openldap/slapd.conf` was modified like below to allow for the operation of LDAP server. For brevity, only some configuration options are shown.

Listing 2: /usr/local/etc/openldap/slapd.conf

```
include /usr/local/etc/openldap/schema/core.schema
include /usr/local/etc/openldap/schema/cosine.schema
include /usr/local/etc/openldap/schema/inetorgperson.schema
include /usr/local/etc/openldap/schema/misc.schema
include /usr/local/etc/openldap/schema/nis.schema

moduleload      back_mdb

database        mdb
```

```
maxsize         1073741824
suffix          "dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
rootdn          "cn=Manager,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
```

To ensure `slapd` can be started via `service slapd start` the line `slapd_enable="YES"` was added to `/etc/rc.conf`. The two demonstration accounts were added via the following `.ldif` file and the `slapadd` command.

Listing 3: /usr/local/etc/openldap/twoacc.ldif

```
# rule241.caia.swin.edu.au
dn: dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
dc: rule241
objectClass: top
objectClass: domain
objectClass: domainRelatedObject
associatedDomain: rule241.caia.swin.edu.au

# People, rule241.caia.swin.edu.au
dn: ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
ou: People
objectClass: top
objectClass: organizationalUnit
objectClass: domainRelatedObject
associatedDomain: rule241.caia.swin.edu.au

# Group, rule241.caia.swin.edu.au
dn: ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
ou: Group
objectClass: top
objectClass: organizationalUnit
objectClass: domainRelatedObject
associatedDomain: rule241.caia.swin.edu.au

# Alan Smithee, Group, rule241.caia.swin.edu.au
dn: cn=g_asmithee,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
cn: g_asmithee
gidNumber: 6000
objectClass: posixGroup
objectClass: top

# gspelvin, Group, rule241.caia.swin.edu.au
dn: cn=g_gspelvin,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
cn: g_gspelvin
gidNumber: 6001
```

```
objectClass: posixGroup
objectClass: top

# Alan Smithee, People, rule241.caia.swin.edu.au
dn: cn=u_asmithee,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
givenName: Alan
sn: Smithee
cn: u_asmithee
uid: asmithee
objectClass: inetOrgPerson
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
uidNumber: 6000
gidNumber: 6000
homeDirectory: /home/asmithee
loginShell: /usr/local/bin/bash
userPassword: dune

# George Spelvin, People, rule241.caia.swin.edu.au
dn: cn=u_gspelvin,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
givenName: George
sn: Spelvin
cn: u_gspelvin
uid: gspelvin
objectClass: inetOrgPerson
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
uidNumber: 6001
gidNumber: 6001
homeDirectory: /home/gspelvin
loginShell: /usr/local/bin/bash
userPassword: broadway
```

To enable LDAP users to edit their own passwords, the following access control configuration should be placed in `slapd.conf`.

Listing 4: /usr/local/etc/openldap/slapd.conf

```
access to attrs=userPassword
        by self write
        by anonymous auth
        by users none

access to * by * read
```

5.1 Demonstration Account Details

As a demonstration, two accounts have been added to ensure the operation of the LDAP server, which are shown in Table **LOOK AT TABLE NUMBER**.

Table 1: Demonstration Accounts

Full Name	Alan Smithee	George Spelvin
Account Name	asmithee	gspelvin
Password	dune	broadway

6 Client Configuration Documentation

To install another work station with `root`, the following software is required to ensure LDAP can run. They should all be installed via `pkg install [package name]`

- `openldap-server-2.4.46_4`
- `pam_ldap-1.8.6_3`
- `nss_ldap-1.265_13`

After this, the first file that should be configured is `/usr/local/etc/openldap/ldap.conf`. Three lines should then be added to this file, which are listed below.

Listing 5: /usr/local/etc/openldap/ldap.conf

```
BASE    dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
URI     ldap://rule241.caia.swin.edu.au

pam_login_attribute uid
```

Save this file and then confirm that it was successful by running `ldapsearch`. If successful, it should show that the accounts `asmithee` and `gspelvin` exist and return an error code of 0.

Next, NSS LDAP must be configured, this can be done by opening the file `/usr/local/etc/nss_ldap.conf` and adding the lines as shown in Listing 2. Change the `host` and `base` fields to suit. If you are configuring the LDAP server to receive LDAP requests, also add the line that is commented out and remove the line `host 136.186.230.241`. Furthermore, to ensure the `slapd` server starts instantly (rather than taking about five minutes, add the `bind_policy` line (seen commented below).

Listing 6: /usr/local/etc/nss ldap.conf

```
#host    127.0.0.1
host     136.186.230.241
base     dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
```

```
uri       ldap://rule241.caia.swin.edu.au

#bind_policy soft
```

Moreover, the file `/etc/nsswitch.conf` must have the following lines edited. Ensure you can log in after changing the file and before logging out.

Listing 7: /etc/nsswitch.conf

```
group: files ldap
passwd: files ldap
```

Create the home directory for both users and use `ls -l` to ensure that the group and user names are appearing correctly (instead of their user IDs).

Then we must set up PAM, to do this several configuration files must be changed. It is advised that after each change is made you attempt to log into the client using another PuTTy session to ensure you have not lost access to the system. Firstly, copy `/usr/local/etc/openldap/ldap.conf` to `/usr/local/etc/`. This can easily be done in the `openldap` directory by entering `cp ldap.conf ..`, this act as the configuration file for PAM LDAP.

Thereafter move to the directory `/etc/pam.d/` and first open `system`, then `sshd` and `su`. Note that \\ is used where the line splits on the page.

The configuration for the `auth` and `account` settings must be as follows:

Listing 8: /etc/pam.d/system

```
auth sufficient pam_opie.so no_warn no_fake_prompts
auth requisite pam_opieaccess.so no_warn allow_local
auth sufficient /usr/local/lib/pam_ldap.so no_warn
auth required pam_unix.so  no_warn try_first_pass nullok

account required pam_login_access.so
account required pam_unix.so
account required /usr/local/lib/pam_ldap.so \\
    no_warn ignore_authinfo_unavail ignore_unknown_user
```

Listing 9: /etc/pam.d/sshd

```
auth sufficient pam_opie.so no_warn no_fake_prompts
auth requisite  pam_opieaccess.so no_warn allow_local
auth  sufficient /usr/local/lib/pam_ldap.so  no_warn
auth required pam_unix.so no_warn try_first_pass

account required pam_nologin.so
account required pam_login_access.so
account required pam_unix.so
account  required /usr/local/lib/pam_ldap.so \\
    no_warn ignore_authinfo_unavail ignore_unknown_user
```

Listing 10: /etc/pam.d/su

```
auth sufficient pam_rootok.so no_warn
auth sufficient pam_self.so no_warn
auth requisite pam_group.so no_warn group=wheel root
_only fail_safe ruser
auth include system

account include system
account required /usr/local/lib/pam_ldap.so \\
    no_warn ignore_authinfo_unavail ignore_unknown_user
```

Thereafter, the LDAP client should be setup. Make sure to log into all accounts (both local and LDAP based) via `ssh` and `su`.

7 Additional Features and Configuration

While the above sections deal with the creation and operation of an LDAP server, below are modifications that were made to the server to benefit the troubleshooting process.

7.1 LDAP Web Monitor

The web monitor provides an alternative method of checking the status of the LDAP server. It provides information on if the server is online and what performance it has (in essence: CPU and memory). This means that any user can check if the LDAP is reachable and if connection issues are due to the LDAP server being offline or due to another network fault. An example is a firewall misconfiguration, which could could allow HTTP through but not LDAP packets.

This page can be accessed via `http://rule241.caia.swin.edu.au`

7.1.1 Configuring the Web Monitor

To configure the `.cgi` file, it was placed in `/usr/local/www/apache24/data`. The following steps were then undertaken to make sure it works:

1. The address of the web server was set to `rule241.caia.swin.edu.au:80`.
2. `Options +ExecCGI` was added to `/usr/local/etc/apache24/http.conf` directory.
3. `AddHandler cgi-script .cgi` was added to `/usr/local/etc/apache24/http.conf` for the main directory.
4. `DirectoryIndex` was edited to `index.cgi` from `index.html`.
5. `mpm-prefork modules` was uncommented.

Thereafter the web monitor was functional.

8 Appendix (Code)

Below is the code of the four scripts; \\ is used when a line is too long for the page.

8.1 Add Users

Listing 11: addusr.sh

```
#!/usr/local/bin/bash

# addusr
# Author: Joshua Redolfi
# Add a user to the LDAP server on rule241.caia.swin.edu.au
# Takes the following arguments
# [uid] [First name] [Last name] [Password] OPTIONAL: [uidNumber]

#Get our args
uid=$1
fname=$2
lname=$3
idnum=$4

domain="dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
host="rule241.caia.swin.edu.au"
#make sure the above two match
rootdn="Manager"
rootpw=secret
port=389

if [ $1 == "help" ];
then
        echo "Command usage: ./addusr.sh [uid] [First name] \\
        [Last name] [uid number]"
        retVal=1
        return ${retVal} 2>/dev/null # this will attempt to return
        exit "${retVal}" # this will get executed if the above failed.
fi

#enter password
read -p "Enter the password: " pass

#create file tmp.ldif (and remove old one if it exists)
rm /tmp/tmp.ldif
echo 'dn: cn='g_$uid',ou=Group,'$domain'' >> /tmp/tmp.ldif
echo 'cn: 'g_$uid'' >> /tmp/tmp.ldif
```

```
echo 'gidNumber: '$idnum'' >> /tmp/tmp.ldif
echo 'objectClass: posixGroup' >> /tmp/tmp.ldif
echo 'objectClass: top' >> /tmp/tmp.ldif
echo '' >> /tmp/tmp.ldif
#now let's make the user
echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
echo 'givenName: '$fname'' >> /tmp/tmp.ldif
echo 'sn: '$lname'' >> /tmp/tmp.ldif
echo 'cn: 'u_$uid'' >> /tmp/tmp.ldif
echo 'uid: '$uid'' >> /tmp/tmp.ldif
echo 'objectClass: inetOrgPerson' >> /tmp/tmp.ldif
echo 'objectClass: top' >> /tmp/tmp.ldif
echo 'objectClass: posixAccount' >> /tmp/tmp.ldif
echo 'objectClass: shadowAccount' >> /tmp/tmp.ldif
echo 'uidNumber: '$idnum'' >> /tmp/tmp.ldif
echo 'gidNumber: '$idnum'' >> /tmp/tmp.ldif
echo 'homeDirectory: /home/'$uid'' >> /tmp/tmp.ldif
echo 'loginShell: /usr/local/bin/bash' >> /tmp/tmp.ldif
echo 'userPassword: '$pass'' >> /tmp/tmp.ldif

#add this ldif file to slapd
#slapadd -l /tmp/tmp.ldif

#adding to slapd via ldapadd
ldapadd -D cn=$rootdn,$domain -w $rootpw -h $host:$port -f /tmp/tmp.ldif

#check error code to see if user was actually added
if [ $? -eq 1 ]
then
        echo 'There was an error in adding the .ldif file'
else
        echo 'User '$uid'/'$fname' '$lname' added successfully'
fi

rm /tmp/tmp.ldif
if [ $? -eq 1 ]
then
        echo 'tmp.ldif could not be removed!'
else
        echo 'tmp.ldif removed from /tmp/'
fi
```

8.2 Remove Users

Listing 12: rmusr.sh

```
#!/usr/local/bin/bash

# rmusr
# Author: Joshua Redolfi
# Remove a user off the LDAP server on rule241.caia.swin.edu.au
# Takes the following arguments
# [uid] [First name] [Last name]

#Get our args
uid=$1

if [ $1 == "help" ];
then
        echo "Command usage: ./rmusr.sh [uid]"
        retVal=1
        return ${retVal} 2>/dev/null # this will attempt to return
        exit "${retVal}" # this will get executed if the above failed.
fi

domain="dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
host="rule241.caia.swin.edu.au"
#make sure the above two match
rootdn="Manager"
rootpw=secret
port=389

ldapdelete -D cn=$rootdn,$domain -w $rootpw -h \\
$host -p $port "cn=g_$uid,ou=Group,$domain" \\
"cn=u_$uid,ou=People,$domain"

#check error code to see if user was actually removed
if [ $? -eq 0 ]
then
        echo 'User '$uid' removed successfully'
else
        echo 'There was an error in removing the user'
fi
```

8.3 Edit Users

Listing 13: editusr.sh

```
#!/usr/local/bin/bash
```

```
# editusr
# Author: Joshua Redolfi
# Edit a user's details on the LDAP server \\
on rule241.caia.swin.edu.au

cname="u_"$name
domain="dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
host="rule241.caia.swin.edu.au"
#make sure the above two match
rootdn="Manager"
rootpw=secret
port=389

#control statement for options \\
(help, password, name, shell, home)
if [ $1 == "help" ]
then
        echo "Command usage: ./editusr.sh [option]"
        echo "password - change a user's password"
        echo "name - change a user's name \\
        (first name and last name)"
        echo "shell - change a user's shell (bash, csh and sh)"
        echo "home - change a user's home directory"
        retVal=1
        return ${retVal} 2>/dev/null # this will attempt to return
        exit "${retVal}" # this will get \\
        executed if the above failed.
elif [ $1 == "password" ]
        then
        echo "Enter user id to change password for"
        read -p "User ID:" uid
        ldappasswd -D cn=$rootdn,$domain -w $rootpw -h $host \\
        -p $port -A -S cn="u_"$uid,ou=People,$domain
        #check error code to see if password was actually changed
        if [ $? -eq 0 ]
        then
                echo 'User password changed successfully'
        else
                echo 'There was an error in changing the password'
        fi
elif [ $1 == "name" ]
        then
        echo "Enter user id of the account to change the name of"
        read -p "User ID:" uid
        rm /tmp/tmp.ldif
        echo "Enter the new first name:"
```

```
        read -p "First name:" fname
        echo "Enter the new surname:"
        read -p "Surname:" lname
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        echo 'changetype: modify' >> /tmp/tmp.ldif
        echo 'replace: givenName' >> /tmp/tmp.ldif
        echo 'givenName: '$fname'' >> /tmp/tmp.ldif
        #modify first name first, done in two steps because \\
        ldapmodify doesnt like it otherwise
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        echo 'changetype: modify' >> /tmp/tmp.ldif
        echo 'replace: sn' >> /tmp/tmp.ldif
        echo 'sn: '$lname'' >> /tmp/tmp.ldif
        #then modify last name
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
elif [ $1 == "shell" ]
        then
        echo "Enter user id of the account to change the shell of"
        read -p "User ID:" uid
        echo "Type name of the shell you wish \\
        to change the user shell to:"
        echo "Options: bash, sh, csh"
        read -p "Shell:" shell
        rm /tmp/tmp.ldif
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        #control statement for shells, reject if invalid
        if [ $shell == "bash" ]
        then
                echo 'changetype: modify' >> /tmp/tmp.ldif
                echo 'replace: loginShell' >> /tmp/tmp.ldif
                echo 'loginShell: /usr/local/bin/bash' >> /tmp/tmp.ldif
        elif [ $shell = "sh" ]
        then
                echo 'changetype: modify' >> /tmp/tmp.ldif
                echo 'replace: loginShell' >> /tmp/tmp.ldif
                echo 'loginShell: /bin/sh' >> /tmp/tmp.ldif
        elif [ $shell = "csh" ]
        then
                echo 'changetype: modify' >> /tmp/tmp.ldif
                echo 'replace: loginShell' >> /tmp/tmp.ldif
                echo 'loginShell: /bin/csh' >> /tmp/tmp.ldif
```

```
        else
                echo "Invalid option, won't change shell type"
                echo "Exiting..."
                retVal=1
                return ${retVal} 2>/dev/null # this will \\
                attempt to return
                exit "${retVal}" # this will get executed \\
                if the above failed.
        fi
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
elif [ $1 == "home" ]
then
        echo "Enter user id of the account to \\
        change the home directory of"
        read -p "User ID:" uid
        echo "Enter the new directory (absolute reference):"
        echo "Example: /newlocation/"
        read -p "Home directory:" home
        rm /tmp/tmp.ldif
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        echo 'changetype: modify' >> /tmp/tmp.ldif
        echo 'replace: homeDirectory' >> /tmp/tmp.ldif
        echo 'homeDirectory: '$home'' >> /tmp/tmp.ldif
        #modify home directory here
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
else
        echo "Option invalid, type \\
        './editusr.sh help' to see all options."
fi
```

8.4 Change Passwords

Listing 14: changepasswd.sh

```
#!/usr/local/bin/bash

# changepasswd
# Author: Joshua Redolfi
# Edit a user's password on the LDAP \\
server on rule241.caia.swin.edu.au
# Takes no arguments
```

```
name=$(whoami)

domain='dc=rule241,dc=caia,dc=swin,dc=edu,dc=au'
host=rule241.caia.swin.edu.au
#make sure the above two match
rootdn=Manager
rootpw=secret
port=389

ldappasswd -D cn=$rootdn,$domain -w $rootpw -h \\
$host -p $port -A -S cn="u_"$name,ou=People,$domain

#check error code to see if password was actually changed
if [ $? -eq 0 ]
then
        echo 'User_'$name'_password_successfully'
else
        echo 'There_was_an_error_in_changing_the_password'
fi


#ldappasswd -D cn=Manager,dc=rule241,dc=caia,dc=swin, \\
dc=edu,dc=au -w secret -h rule241.caia.swin.edu.au \\
-p 389 -A -S cn=asmithee,ou=People,dc=rule241, \\
dc=caia,dc=swin,dc=edu,dc=au
```

LDAP Project Documentation

Joshua Redolfi
Faculty of Science,
Engineering and, Technology
Swinburne University of Technology
Email: 101601886@student.swin.edu.au

Semester 2 2019

Abstract

This document describes the operation of the LDAP (Light-weight Directory Access Protocol) on the RULE host rule241.caia.swin.edu.au. It contains information on how the server was configured and how to configure clients to connect to it. Thereafter, account and script details are provided to show how the system has been automated. Finally, the operation of an LDAP web monitor is provided. **This document contains all the documentation required for the project**.

Contents

1 Introduction

This documentation contains information of the LDAP server set up on the following hosts:

Server Address: 136.186.230.241 (rule241.caia.swin.edu.au)
Client Address: 136.186.230.240 (rule240.caia.swin.edu.au)

2 Location of Files and CLI Scripts

This contains the locations of the files that were edited and created for the use of LDAP. It also has the locations of all the scripts for adding, editing and removing user accounts, along with the ability for users to change their passwords.

2.1 Server Locations

SLAPD config: `/usr/local/etc/openldap/slapd.conf`
Initial .ldif file: `/usr/local/etc/openldap/twoacc.ldif`
LDAP config: `/usr/local/etc/openldap/ldap.conf`
PAM-LDAP config: `/usr/local/etc/ldap.conf`
NSS-LDAP config: `/usr/local/etc/nss_ldap.conf`
NSS config: `/etc/nsswitch.conf`
PAM config (system): `/etc/pam.d/system`
PAM config (ssh): `/etc/pam.d/sshd`
PAM config (su): `/etc/pam.d/su`
Apache config: `/usr/local/etc/apache24/http.conf`
Web Monitor: `/usr/local/www/apache24/data/index.cgi`

2.2 Client Locations

LDAP config: `/usr/local/etc/openldap/ldap.conf`
PAM-LDAP config: `/usr/local/etc/ldap.conf`
NSS-LDAP config: `/usr/local/etc/nss_ldap.conf`
NSS config: `/etc/nsswitch.conf`
PAM config (system): `/etc/pam.d/system`
PAM config (ssh): `/etc/pam.d/sshd`
PAM config (su): `/etc/pam.d/su`

2.3 Script Locations

Adding users (root): `/root/addusr.sh`
Removing users (root): `/root/rmusr.sh`
Editing users (root): `/root/editusr.sh`
Changing passwords (user): `/home/changepasswd.sh`

3 Script Commands

To assist in the operating LDAP, several script commands have been provided to save time on tasks. The code for these scripts can be found in the Appendix.

3.1 Adding Accounts

To add an account, use `addusr.sh`, it is solely for root and located in `/root/`. It can be executed as follows: `/root/addusr.sh [uid] [First Name] [Last Name] \\ [Group Number/User Number]` (remove the \\ when in the console). Thereafter it will ask for a password.

3.2 Removing Accounts

To remove an account, use `rmusr.sh`, which is located in `/root/` and can only be used by root. To execute it, enter the command like the following:

```
/root/rmusr.sh [uid] [First Name] [Last Name]
```

3.3 Editing Accounts

Root is also able to modify account details using the script `editusr.sh`. Like the other scripts intended for use by `root`, it can be found in `/root/`. It can be executed with `editusr.sh [option]` and the script will prompt root for the `uid` of the user and other needed information. The options and what they change are shown below:

- `name` - changes the user's `givenName` and `sn`, not their `uid`.
- `home` - changes the user home directory.
- `password` - chnages the user password.
- `shell` - changes the shell between `bash`, `csh` and `sh`.
- `help` - shows how to use the command.

3.4 Changing User Passwords

This script is for the use of a non-root user to change their own password. It can be executed by the use of `./changepasswd`; it will return an error if a local account attempts to use it.

4 Instructions

Below are the instructions to use root to modify accounts and how a user can change their account password.

4.1 Using root to add user accounts

To create a account via `root`, firstly a `.ldif` file must created with the details of the account. This file should contain both a group and user identity, but the former can be ignored if you are using a preexisting group. The following fields must configured:

- `gidNumber`, group number
- `uid`, user ID
- `uidNunmber`, user number
- `cn`, entry/full name
- `givenName`, first name
- `sn`, surname

Below an example configuration is shown. This will create an account with the user ID `jsmith` and a member of the group `jsmith`. It is notable that the group ID of an account can be set to 0, meaning it will be a member of the `wheel` group, which means it can use the `su` command.

To work with the scripts provided, the `cn` of an account **must be equal** to the uid appended with a '`u_`'. For example: `u_[uid]`. For groups the `cn` is equal to `g_[uid]`.

Listing 1: /usr/local/etc/openldap/test.ldif

```
# John Smith, Group, rule241.caia.swin.edu.au
dn: cn=jsmith,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
cn: g_jsmith
gidNumber: 6002
objectClass: posixGroup
objectClass: top

# Johm Smith, People, rule241.caia.swin.edu.au
dn: cn=John Smith,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
givenName: John
sn: Smith
cn: u_jsmith
uid: jsmith
objectClass: inetOrgPerson
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
uidNumber: 6002
gidNumber: 6002
```

```
homeDirectory: /home/jsmith
loginShell: /usr/local/bin/bash
userPassword: test
```

Thereafter, this account can be added by the use of `slapadd -l test.ldif`. Thereafter check that the account has been added by the use of `ldapsearch`.

4.2 Using root to remove user accounts

Root can delete an LDAP account if the `rootdn` username and password are known. The command is `ldapdelete` and the format is the like the following:

```
ldapdelete -D "[rootdn user]" -w [rootdn password] -h [hostname of server] \\
-p [ldap port] "[user1 to be deleted]" "[user2 to be deleted]" ..
```

Going back to the previous example with user `John Smith`, the following command can be used to delete an LDAP account.

```
ldapdelete -D "cn=Manager,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au" \\
-w secret -h rule241.caia.swin.edu.au \\
-p 389 "cn=jsmith,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au" \\
"cn=John Smith,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
```

Verify that the account has been deleted via `ldapsearch`.

4.3 Using root to change account passwords

Similar to the above section, the command to change a user's password (`ldappasswd`) requires the details of the `rootdn`. There are several different options that can be used that achieve the same result.

- `-a` allows for the old password to be received in the command itself.
- `-A` will make the console prompt root for the old password.
- `-s` allows for the new password to be received in the command itself.
- `-S` will make the console prompt root for the new password.

Overall, the command follows this format:

```
ldappasswd -D "[rootdn user]" -w [rootdn password] \\
-h [hostname of server] -p [ldap port] -A -S "[user]"
```

Once again, changing the password of user `jsmith` is as follows.

```
ldappasswd -D "cn=Manager,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au" \\
-w secret -h rule241.caia.swin.edu.au -p 389 -A -S \\
"cn=John Smith,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
```

4.4 How a user can change their own password

There are two methods provided to allow for a user to change their own password. The first is identical to the above method, but expects that the user has knowledge of the manager password (an ACL prevents the user from accessing any other accounts or details) and other server details. There is no method to change passwords without the use of the manager password.

4.4.1 Direct CLI Method

Below a method using `ldappasswd` to change the user passwords is shown. The user needs to know the domain of the server, the manager password and their `uid`. See the above section for the different password changing options in `ldappasswd`.

1. The user should find their `uid` using `whoami`. The account, if created by script, should have a `cn` that looks like `u_[uid]`.
2. The command below should be then executed, and the prompts it has should be filled in correctly (old and new passwords).

```
ldappasswd -D "[rootdn user]" -w [rootdn password] \\
-h [hostname of server] -p [ldap port] -A -S "[user]"
```

4.4.2 Script CLI Method

Running the script `./changepasswd` in the `/home/` directory will allow for a prompt to come up. The user should be enter their old and new passwords to change it.

5 Server Configuration Documentation

The server was configured with the following details. `/usr/local/etc/openldap/slapd.conf` was modified like below to allow for the operation of LDAP server. For brevity, only some configuration options are shown.

Listing 2: /usr/local/etc/openldap/slapd.conf

```
include /usr/local/etc/openldap/schema/core.schema
include /usr/local/etc/openldap/schema/cosine.schema
include /usr/local/etc/openldap/schema/inetorgperson.schema
include /usr/local/etc/openldap/schema/misc.schema
include /usr/local/etc/openldap/schema/nis.schema

moduleload      back_mdb

database        mdb
```

```
maxsize         1073741824
suffix          "dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
rootdn          "cn=Manager,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
```

To ensure `slapd` can be started via `service slapd start` the line `slapd_enable="YES"` was added to `/etc/rc.conf`. The two demonstration accounts were added via the following `.ldif` file and the `slapadd` command.

Listing 3: /usr/local/etc/openldap/twoacc.ldif

```
# rule241.caia.swin.edu.au
dn: dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
dc: rule241
objectClass: top
objectClass: domain
objectClass: domainRelatedObject
associatedDomain: rule241.caia.swin.edu.au

# People, rule241.caia.swin.edu.au
dn: ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
ou: People
objectClass: top
objectClass: organizationalUnit
objectClass: domainRelatedObject
associatedDomain: rule241.caia.swin.edu.au

# Group, rule241.caia.swin.edu.au
dn: ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
ou: Group
objectClass: top
objectClass: organizationalUnit
objectClass: domainRelatedObject
associatedDomain: rule241.caia.swin.edu.au

# Alan Smithee, Group, rule241.caia.swin.edu.au
dn: cn=g_asmithee,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
cn: g_asmithee
gidNumber: 6000
objectClass: posixGroup
objectClass: top

# gspelvin, Group, rule241.caia.swin.edu.au
dn: cn=g_gspelvin,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
cn: g_gspelvin
gidNumber: 6001
```

```
objectClass: posixGroup
objectClass: top

# Alan Smithee, People, rule241.caia.swin.edu.au
dn: cn=u_asmithee,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
givenName: Alan
sn: Smithee
cn: u_asmithee
uid: asmithee
objectClass: inetOrgPerson
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
uidNumber: 6000
gidNumber: 6000
homeDirectory: /home/asmithee
loginShell: /usr/local/bin/bash
userPassword: dune

# George Spelvin, People, rule241.caia.swin.edu.au
dn: cn=u_gspelvin,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
givenName: George
sn: Spelvin
cn: u_gspelvin
uid: gspelvin
objectClass: inetOrgPerson
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
uidNumber: 6001
gidNumber: 6001
homeDirectory: /home/gspelvin
loginShell: /usr/local/bin/bash
userPassword: broadway
```

To enable LDAP users to edit their own passwords, the following access control configuration should be placed in `slapd.conf`.

Listing 4: /usr/local/etc/openldap/slapd.conf

```
access to attrs=userPassword
        by self write
        by anonymous auth
        by users none

access to * by * read
```

5.1 Demonstration Account Details

As a demonstration, two accounts have been added to ensure the operation of the LDAP server, which are shown in Table **LOOK AT TABLE NUMBER**.

Table 1: Demonstration Accounts

Full Name	Alan Smithee	George Spelvin
Account Name	asmithee	gspelvin
Password	dune	broadway

6 Client Configuration Documentation

To install another work station with `root`, the following software is required to ensure LDAP can run. They should all be installed via `pkg install [package name]`

- `openldap-server-2.4.46_4`
- `pam_ldap-1.8.6_3`
- `nss_ldap-1.265_13`

After this, the first file that should be configured is `/usr/local/etc/openldap/ldap.conf`. Three lines should then be added to this file, which are listed below.

Listing 5: /usr/local/etc/openldap/ldap.conf

```
BASE    dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
URI     ldap://rule241.caia.swin.edu.au

pam_login_attribute uid
```

Save this file and then confirm that it was successful by running `ldapsearch`. If successful, it should show that the accounts `asmithee` and `gspelvin` exist and return an error code of 0.

Next, NSS LDAP must be configured, this can be done by opening the file `/usr/local/etc/nss_ldap.conf` and adding the lines as shown in Listing 2. Change the `host` and `base` fields to suit. If you are configuring the LDAP server to receive LDAP requests, also add the line that is commented out and remove the line `host 136.186.230.241`. Furthermore, to ensure the `slapd` server starts instantly (rather than taking about five minutes, add the `bind_policy` line (seen commented below).

Listing 6: /usr/local/etc/nss ldap.conf

```
#host    127.0.0.1
host     136.186.230.241
base     dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
```

```
uri       ldap://rule241.caia.swin.edu.au

#bind_policy soft
```

Moreover, the file `/etc/nsswitch.conf` must have the following lines edited. Ensure you can log in after changing the file and before logging out.

Listing 7: /etc/nsswitch.conf

```
group: files ldap
passwd: files ldap
```

Create the home directory for both users and use `ls -l` to ensure that the group and user names are appearing correctly (instead of their user IDs).

Then we must set up PAM, to do this several configuration files must be changed. It is advised that after each change is made you attempt to log into the client using another PuTTy session to ensure you have not lost access to the system. Firstly, copy `/usr/local/etc/openldap/ldap.conf` to `/usr/local/etc/`. This can easily be done in the `openldap` directory by entering `cp ldap.conf ..`, this act as the configuration file for PAM LDAP.

Thereafter move to the directory `/etc/pam.d/` and first open `system`, then `sshd` and `su`. Note that \\ is used where the line splits on the page.

The configuration for the `auth` and `account` settings must be as follows:

Listing 8: /etc/pam.d/system

```
auth sufficient pam_opie.so no_warn no_fake_prompts
auth requisite pam_opieaccess.so no_warn allow_local
auth sufficient /usr/local/lib/pam_ldap.so no_warn
auth required pam_unix.so  no_warn try_first_pass nullok

account required pam_login_access.so
account required pam_unix.so
account required /usr/local/lib/pam_ldap.so \\
    no_warn ignore_authinfo_unavail ignore_unknown_user
```

Listing 9: /etc/pam.d/sshd

```
auth sufficient pam_opie.so no_warn no_fake_prompts
auth requisite  pam_opieaccess.so no_warn allow_local
auth  sufficient /usr/local/lib/pam_ldap.so  no_warn
auth required pam_unix.so no_warn try_first_pass

account required pam_nologin.so
account required pam_login_access.so
account required pam_unix.so
account  required /usr/local/lib/pam_ldap.so \\
    no_warn ignore_authinfo_unavail ignore_unknown_user
```

Listing 10: /etc/pam.d/su

```
auth sufficient pam_rootok.so no_warn
auth sufficient pam_self.so no_warn
auth requisite pam_group.so no_warn group=wheel root
_only fail_safe ruser
auth include system

account include system
account required /usr/local/lib/pam_ldap.so \\
    no_warn ignore_authinfo_unavail ignore_unknown_user
```

Thereafter, the LDAP client should be setup. Make sure to log into all accounts (both local and LDAP based) via `ssh` and `su`.

7 Additional Features and Configuration

While the above sections deal with the creation and operation of an LDAP server, below are modifications that were made to the server to benefit the troubleshooting process.

7.1 LDAP Web Monitor

The web monitor provides an alternative method of checking the status of the LDAP server. It provides information on if the server is online and what performance it has (in essence: CPU and memory). This means that any user can check if the LDAP is reachable and if connection issues are due to the LDAP server being offline or due to another network fault. An example is a firewall misconfiguration, which could could allow HTTP through but not LDAP packets.

This page can be accessed via `http://rule241.caia.swin.edu.au`

7.1.1 Configuring the Web Monitor

To configure the `.cgi` file, it was placed in `/usr/local/www/apache24/data`. The following steps were then undertaken to make sure it works:

1. The address of the web server was set to `rule241.caia.swin.edu.au:80`.
2. `Options +ExecCGI` was added to `/usr/local/etc/apache24/http.conf` directory.
3. `AddHandler cgi-script .cgi` was added to `/usr/local/etc/apache24/http.conf` for the main directory.
4. `DirectoryIndex` was edited to `index.cgi` from `index.html`.
5. `mpm-prefork modules` was uncommented.

Thereafter the web monitor was functional.

8 Appendix (Code)

Below is the code of the four scripts; \\ is used when a line is too long for the page.

8.1 Add Users

Listing 11: addusr.sh

```
#!/usr/local/bin/bash

# addusr
# Author: Joshua Redolfi
# Add a user to the LDAP server on rule241.caia.swin.edu.au
# Takes the following arguments
# [uid] [First name] [Last name] [Password] OPTIONAL: [uidNumber]

#Get our args
uid=$1
fname=$2
lname=$3
idnum=$4

domain="dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
host="rule241.caia.swin.edu.au"
#make sure the above two match
rootdn="Manager"
rootpw=secret
port=389

if [ $1 == "help" ];
then
        echo "Command usage: ./addusr.sh [uid] [First name] \\
        [Last name] [uid number]"
        retVal=1
        return ${retVal} 2>/dev/null # this will attempt to return
        exit "${retVal}" # this will get executed if the above failed.
fi

#enter password
read -p "Enter the password: " pass

#create file tmp.ldif (and remove old one if it exists)
rm /tmp/tmp.ldif
echo 'dn: cn='g_$uid',ou=Group,'$domain'' >> /tmp/tmp.ldif
echo 'cn: 'g_$uid'' >> /tmp/tmp.ldif
```

```
echo 'gidNumber: '$idnum'' >> /tmp/tmp.ldif
echo 'objectClass: posixGroup' >> /tmp/tmp.ldif
echo 'objectClass: top' >> /tmp/tmp.ldif
echo '' >> /tmp/tmp.ldif
#now let's make the user
echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
echo 'givenName: '$fname'' >> /tmp/tmp.ldif
echo 'sn: '$lname'' >> /tmp/tmp.ldif
echo 'cn: 'u_$uid'' >> /tmp/tmp.ldif
echo 'uid: '$uid'' >> /tmp/tmp.ldif
echo 'objectClass: inetOrgPerson' >> /tmp/tmp.ldif
echo 'objectClass: top' >> /tmp/tmp.ldif
echo 'objectClass: posixAccount' >> /tmp/tmp.ldif
echo 'objectClass: shadowAccount' >> /tmp/tmp.ldif
echo 'uidNumber: '$idnum'' >> /tmp/tmp.ldif
echo 'gidNumber: '$idnum'' >> /tmp/tmp.ldif
echo 'homeDirectory: /home/'$uid'' >> /tmp/tmp.ldif
echo 'loginShell: /usr/local/bin/bash' >> /tmp/tmp.ldif
echo 'userPassword: '$pass'' >> /tmp/tmp.ldif

#add this ldif file to slapd
#slapadd -l /tmp/tmp.ldif

#adding to slapd via ldapadd
ldapadd -D cn=$rootdn,$domain -w $rootpw -h $host:$port -f /tmp/tmp.ldif

#check error code to see if user was actually added
if [ $? -eq 1 ]
then
        echo 'There was an error in adding the .ldif file'
else
        echo 'User '$uid'/'$fname' '$lname' added successfully'
fi

rm /tmp/tmp.ldif
if [ $? -eq 1 ]
then
        echo 'tmp.ldif could not be removed!'
else
        echo 'tmp.ldif removed from /tmp/'
fi
```

8.2 Remove Users

Listing 12: rmusr.sh

```
#!/usr/local/bin/bash

# rmusr
# Author: Joshua Redolfi
# Remove a user off the LDAP server on rule241.caia.swin.edu.au
# Takes the following arguments
# [uid] [First name] [Last name]

#Get our args
uid=$1

if [ $1 == "help" ];
then
        echo "Command usage: ./rmusr.sh [uid]"
        retVal=1
        return ${retVal} 2>/dev/null # this will attempt to return
        exit "${retVal}" # this will get executed if the above failed.
fi

domain="dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
host="rule241.caia.swin.edu.au"
#make sure the above two match
rootdn="Manager"
rootpw=secret
port=389

ldapdelete -D cn=$rootdn,$domain -w $rootpw -h \\
$host -p $port "cn=g_$uid,ou=Group,$domain" \\
"cn=u_$uid,ou=People,$domain"

#check error code to see if user was actually removed
if [ $? -eq 0 ]
then
        echo 'User '$uid' removed successfully'
else
        echo 'There was an error in removing the user'
fi
```

8.3 Edit Users

Listing 13: editusr.sh

```
#!/usr/local/bin/bash
```

```
# editusr
# Author: Joshua Redolfi
# Edit a user's details on the LDAP server \\
on rule241.caia.swin.edu.au

cname="u_"$name
domain="dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
host="rule241.caia.swin.edu.au"
#make sure the above two match
rootdn="Manager"
rootpw=secret
port=389

#control statement for options \\
(help, password, name, shell, home)
if [ $1 == "help" ]
then
        echo "Command usage: ./editusr.sh [option]"
        echo "password - change a user's password"
        echo "name - change a user's name \\
        (first name and last name)"
        echo "shell - change a user's shell (bash, csh and sh)"
        echo "home - change a user's home directory"
        retVal=1
        return ${retVal} 2>/dev/null # this will attempt to return
        exit "${retVal}" # this will get \\
        executed if the above failed.
elif [ $1 == "password" ]
        then
        echo "Enter user id to change password for"
        read -p "User ID:" uid
        ldappasswd -D cn=$rootdn,$domain -w $rootpw -h $host \\
        -p $port -A -S cn="u_"$uid,ou=People,$domain
        #check error code to see if password was actually changed
        if [ $? -eq 0 ]
        then
                echo 'User password changed successfully'
        else
                echo 'There was an error in changing the password'
        fi
elif [ $1 == "name" ]
        then
        echo "Enter user id of the account to change the name of"
        read -p "User ID:" uid
        rm /tmp/tmp.ldif
        echo "Enter the new first name:"
```

```
        read -p "First name:" fname
        echo "Enter the new surname:"
        read -p "Surname:" lname
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        echo 'changetype: modify' >> /tmp/tmp.ldif
        echo 'replace: givenName' >> /tmp/tmp.ldif
        echo 'givenName: '$fname'' >> /tmp/tmp.ldif
        #modify first name first, done in two steps because \\
        ldapmodify doesnt like it otherwise
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        echo 'changetype: modify' >> /tmp/tmp.ldif
        echo 'replace: sn' >> /tmp/tmp.ldif
        echo 'sn: '$lname'' >> /tmp/tmp.ldif
        #then modify last name
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
elif [ $1 == "shell" ]
        then
        echo "Enter user id of the account to change the shell of"
        read -p "User ID:" uid
        echo "Type name of the shell you wish \\
        to change the user shell to:"
        echo "Options: bash, sh, csh"
        read -p "Shell:" shell
        rm /tmp/tmp.ldif
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        #control statement for shells, reject if invalid
        if [ $shell == "bash" ]
        then
                echo 'changetype: modify' >> /tmp/tmp.ldif
                echo 'replace: loginShell' >> /tmp/tmp.ldif
                echo 'loginShell: /usr/local/bin/bash' >> /tmp/tmp.ldif
        elif [ $shell = "sh" ]
        then
                echo 'changetype: modify' >> /tmp/tmp.ldif
                echo 'replace: loginShell' >> /tmp/tmp.ldif
                echo 'loginShell: /bin/sh' >> /tmp/tmp.ldif
        elif [ $shell = "csh" ]
        then
                echo 'changetype: modify' >> /tmp/tmp.ldif
                echo 'replace: loginShell' >> /tmp/tmp.ldif
                echo 'loginShell: /bin/csh' >> /tmp/tmp.ldif
```

```
        else
                echo "Invalid option, won't change shell type"
                echo "Exiting..."
                retVal=1
                return ${retVal} 2>/dev/null # this will \\
                attempt to return
                exit "${retVal}" # this will get executed \\
                if the above failed.
        fi
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
elif [ $1 == "home" ]
then
        echo "Enter user id of the account to \\
        change the home directory of"
        read -p "User ID:" uid
        echo "Enter the new directory (absolute reference):"
        echo "Example: /newlocation/"
        read -p "Home directory:" home
        rm /tmp/tmp.ldif
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        echo 'changetype: modify' >> /tmp/tmp.ldif
        echo 'replace: homeDirectory' >> /tmp/tmp.ldif
        echo 'homeDirectory: '$home'' >> /tmp/tmp.ldif
        #modify home directory here
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
else
        echo "Option invalid, type \\
        './editusr.sh help' to see all options."
fi
```

8.4 Change Passwords

Listing 14: changepasswd.sh

```
#!/usr/local/bin/bash

# changepasswd
# Author: Joshua Redolfi
# Edit a user's password on the LDAP \\
server on rule241.caia.swin.edu.au
# Takes no arguments
```

```
name=$(whoami)

domain='dc=rule241,dc=caia,dc=swin,dc=edu,dc=au'
host=rule241.caia.swin.edu.au
#make sure the above two match
rootdn=Manager
rootpw=secret
port=389

ldappasswd -D cn=$rootdn,$domain -w $rootpw -h \\
$host -p $port -A -S cn="u_"$name,ou=People,$domain

#check error code to see if password was actually changed
if [ $? -eq 0 ]
then
        echo 'User_'$name'_password_successfully'
else
        echo 'There_was_an_error_in_changing_the_password'
fi


#ldappasswd -D cn=Manager,dc=rule241,dc=caia,dc=swin, \\
dc=edu,dc=au -w secret -h rule241.caia.swin.edu.au \\
-p 389 -A -S cn=asmithee,ou=People,dc=rule241, \\
dc=caia,dc=swin,dc=edu,dc=au
```

LDAP Project Documentation

Joshua Redolfi
Faculty of Science,
Engineering and, Technology
Swinburne University of Technology
Email: 101601886@student.swin.edu.au

Semester 2 2019

Abstract

This document describes the operation of the LDAP (Light-weight Directory Access Protocol) on the RULE host rule241.caia.swin.edu.au. It contains information on how the server was configured and how to configure clients to connect to it. Thereafter, account and script details are provided to show how the system has been automated. Finally, the operation of an LDAP web monitor is provided. **This document contains all the documentation required for the project**.

Contents

1 Introduction

This documentation contains information of the LDAP server set up on the following hosts:

Server Address: 136.186.230.241 (rule241.caia.swin.edu.au)
Client Address: 136.186.230.240 (rule240.caia.swin.edu.au)

2 Location of Files and CLI Scripts

This contains the locations of the files that were edited and created for the use of LDAP. It also has the locations of all the scripts for adding, editing and removing user accounts, along with the ability for users to change their passwords.

2.1 Server Locations

SLAPD config: `/usr/local/etc/openldap/slapd.conf`
Initial .ldif file: `/usr/local/etc/openldap/twoacc.ldif`
LDAP config: `/usr/local/etc/openldap/ldap.conf`
PAM-LDAP config: `/usr/local/etc/ldap.conf`
NSS-LDAP config: `/usr/local/etc/nss_ldap.conf`
NSS config: `/etc/nsswitch.conf`
PAM config (system): `/etc/pam.d/system`
PAM config (ssh): `/etc/pam.d/sshd`
PAM config (su): `/etc/pam.d/su`
Apache config: `/usr/local/etc/apache24/http.conf`
Web Monitor: `/usr/local/www/apache24/data/index.cgi`

2.2 Client Locations

LDAP config: `/usr/local/etc/openldap/ldap.conf`
PAM-LDAP config: `/usr/local/etc/ldap.conf`
NSS-LDAP config: `/usr/local/etc/nss_ldap.conf`
NSS config: `/etc/nsswitch.conf`
PAM config (system): `/etc/pam.d/system`
PAM config (ssh): `/etc/pam.d/sshd`
PAM config (su): `/etc/pam.d/su`

2.3 Script Locations

Adding users (root): `/root/addusr.sh`
Removing users (root): `/root/rmusr.sh`
Editing users (root): `/root/editusr.sh`
Changing passwords (user): `/home/changepasswd.sh`

3 Script Commands

To assist in the operating LDAP, several script commands have been provided to save time on tasks. The code for these scripts can be found in the Appendix.

3.1 Adding Accounts

To add an account, use `addusr.sh`, it is solely for root and located in `/root/`. It can be executed as follows: `/root/addusr.sh [uid] [First Name] [Last Name] \\ [Group Number/User Number]` (remove the \\ when in the console). Thereafter it will ask for a password.

3.2 Removing Accounts

To remove an account, use `rmusr.sh`, which is located in `/root/` and can only be used by root. To execute it, enter the command like the following:

```
/root/rmusr.sh [uid] [First Name] [Last Name]
```

3.3 Editing Accounts

Root is also able to modify account details using the script `editusr.sh`. Like the other scripts intended for use by `root`, it can be found in `/root/`. It can be executed with `editusr.sh [option]` and the script will prompt root for the `uid` of the user and other needed information. The options and what they change are shown below:

- `name` - changes the user's `givenName` and `sn`, not their `uid`.
- `home` - changes the user home directory.
- `password` - chnages the user password.
- `shell` - changes the shell between `bash`, `csh` and `sh`.
- `help` - shows how to use the command.

3.4 Changing User Passwords

This script is for the use of a non-root user to change their own password. It can be executed by the use of `./changepasswd`; it will return an error if a local account attempts to use it.

4 Instructions

Below are the instructions to use root to modify accounts and how a user can change their account password.

4.1 Using root to add user accounts

To create a account via `root`, firstly a `.ldif` file must created with the details of the account. This file should contain both a group and user identity, but the former can be ignored if you are using a preexisting group. The following fields must configured:

- `gidNumber`, group number
- `uid`, user ID
- `uidNunmber`, user number
- `cn`, entry/full name
- `givenName`, first name
- `sn`, surname

Below an example configuration is shown. This will create an account with the user ID `jsmith` and a member of the group `jsmith`. It is notable that the group ID of an account can be set to 0, meaning it will be a member of the `wheel` group, which means it can use the `su` command.

To work with the scripts provided, the `cn` of an account **must be equal** to the uid appended with a '`u_`'. For example: `u_[uid]`. For groups the `cn` is equal to `g_[uid]`.

Listing 1: /usr/local/etc/openldap/test.ldif

```
# John Smith, Group, rule241.caia.swin.edu.au
dn: cn=jsmith,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
cn: g_jsmith
gidNumber: 6002
objectClass: posixGroup
objectClass: top

# Johm Smith, People, rule241.caia.swin.edu.au
dn: cn=John Smith,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
givenName: John
sn: Smith
cn: u_jsmith
uid: jsmith
objectClass: inetOrgPerson
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
uidNumber: 6002
gidNumber: 6002
```

```
homeDirectory: /home/jsmith
loginShell: /usr/local/bin/bash
userPassword: test
```

Thereafter, this account can be added by the use of `slapadd -l test.ldif`. Thereafter check that the account has been added by the use of `ldapsearch`.

4.2 Using root to remove user accounts

Root can delete an LDAP account if the `rootdn` username and password are known. The command is `ldapdelete` and the format is the like the following:

```
ldapdelete -D "[rootdn user]" -w [rootdn password] -h [hostname of server] \\
-p [ldap port] "[user1 to be deleted]" "[user2 to be deleted]" ..
```

Going back to the previous example with user `John Smith`, the following command can be used to delete an LDAP account.

```
ldapdelete -D "cn=Manager,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au" \\
-w secret -h rule241.caia.swin.edu.au \\
-p 389 "cn=jsmith,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au" \\
"cn=John Smith,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
```

Verify that the account has been deleted via `ldapsearch`.

4.3 Using root to change account passwords

Similar to the above section, the command to change a user's password (`ldappasswd`) requires the details of the `rootdn`. There are several different options that can be used that achieve the same result.

- `-a` allows for the old password to be received in the command itself.
- `-A` will make the console prompt root for the old password.
- `-s` allows for the new password to be received in the command itself.
- `-S` will make the console prompt root for the new password.

Overall, the command follows this format:

```
ldappasswd -D "[rootdn user]" -w [rootdn password] \\
-h [hostname of server] -p [ldap port] -A -S "[user]"
```

Once again, changing the password of user `jsmith` is as follows.

```
ldappasswd -D "cn=Manager,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au" \\
-w secret -h rule241.caia.swin.edu.au -p 389 -A -S \\
"cn=John Smith,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
```

4.4 How a user can change their own password

There are two methods provided to allow for a user to change their own password. The first is identical to the above method, but expects that the user has knowledge of the manager password (an ACL prevents the user from accessing any other accounts or details) and other server details. There is no method to change passwords without the use of the manager password.

4.4.1 Direct CLI Method

Below a method using `ldappasswd` to change the user passwords is shown. The user needs to know the domain of the server, the manager password and their `uid`. See the above section for the different password changing options in `ldappasswd`.

1. The user should find their `uid` using `whoami`. The account, if created by script, should have a `cn` that looks like `u_[uid]`.
2. The command below should be then executed, and the prompts it has should be filled in correctly (old and new passwords).

```
ldappasswd -D "[rootdn user]" -w [rootdn password] \\
-h [hostname of server] -p [ldap port] -A -S "[user]"
```

4.4.2 Script CLI Method

Running the script `./changepasswd` in the `/home/` directory will allow for a prompt to come up. The user should be enter their old and new passwords to change it.

5 Server Configuration Documentation

The server was configured with the following details. `/usr/local/etc/openldap/slapd.conf` was modified like below to allow for the operation of LDAP server. For brevity, only some configuration options are shown.

Listing 2: /usr/local/etc/openldap/slapd.conf

```
include /usr/local/etc/openldap/schema/core.schema
include /usr/local/etc/openldap/schema/cosine.schema
include /usr/local/etc/openldap/schema/inetorgperson.schema
include /usr/local/etc/openldap/schema/misc.schema
include /usr/local/etc/openldap/schema/nis.schema

moduleload      back_mdb

database        mdb
```

```
maxsize         1073741824
suffix          "dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
rootdn          "cn=Manager,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
```

To ensure `slapd` can be started via `service slapd start` the line `slapd_enable="YES"` was added to `/etc/rc.conf`. The two demonstration accounts were added via the following `.ldif` file and the `slapadd` command.

Listing 3: /usr/local/etc/openldap/twoacc.ldif

```
# rule241.caia.swin.edu.au
dn: dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
dc: rule241
objectClass: top
objectClass: domain
objectClass: domainRelatedObject
associatedDomain: rule241.caia.swin.edu.au

# People, rule241.caia.swin.edu.au
dn: ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
ou: People
objectClass: top
objectClass: organizationalUnit
objectClass: domainRelatedObject
associatedDomain: rule241.caia.swin.edu.au

# Group, rule241.caia.swin.edu.au
dn: ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
ou: Group
objectClass: top
objectClass: organizationalUnit
objectClass: domainRelatedObject
associatedDomain: rule241.caia.swin.edu.au

# Alan Smithee, Group, rule241.caia.swin.edu.au
dn: cn=g_asmithee,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
cn: g_asmithee
gidNumber: 6000
objectClass: posixGroup
objectClass: top

# gspelvin, Group, rule241.caia.swin.edu.au
dn: cn=g_gspelvin,ou=Group,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
cn: g_gspelvin
gidNumber: 6001
```

```
objectClass: posixGroup
objectClass: top

# Alan Smithee, People, rule241.caia.swin.edu.au
dn: cn=u_asmithee,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
givenName: Alan
sn: Smithee
cn: u_asmithee
uid: asmithee
objectClass: inetOrgPerson
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
uidNumber: 6000
gidNumber: 6000
homeDirectory: /home/asmithee
loginShell: /usr/local/bin/bash
userPassword: dune

# George Spelvin, People, rule241.caia.swin.edu.au
dn: cn=u_gspelvin,ou=People,dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
givenName: George
sn: Spelvin
cn: u_gspelvin
uid: gspelvin
objectClass: inetOrgPerson
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
uidNumber: 6001
gidNumber: 6001
homeDirectory: /home/gspelvin
loginShell: /usr/local/bin/bash
userPassword: broadway
```

To enable LDAP users to edit their own passwords, the following access control configuration should be placed in `slapd.conf`.

Listing 4: /usr/local/etc/openldap/slapd.conf

```
access to attrs=userPassword
        by self write
        by anonymous auth
        by users none

access to * by * read
```

5.1 Demonstration Account Details

As a demonstration, two accounts have been added to ensure the operation of the LDAP server, which are shown in Table **LOOK AT TABLE NUMBER**.

Table 1: Demonstration Accounts

Full Name	Alan Smithee	George Spelvin
Account Name	asmithee	gspelvin
Password	dune	broadway

6 Client Configuration Documentation

To install another work station with `root`, the following software is required to ensure LDAP can run. They should all be installed via `pkg install [package name]`

- `openldap-server-2.4.46_4`
- `pam_ldap-1.8.6_3`
- `nss_ldap-1.265_13`

After this, the first file that should be configured is `/usr/local/etc/openldap/ldap.conf`. Three lines should then be added to this file, which are listed below.

Listing 5: /usr/local/etc/openldap/ldap.conf

```
BASE    dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
URI     ldap://rule241.caia.swin.edu.au

pam_login_attribute uid
```

Save this file and then confirm that it was successful by running `ldapsearch`. If successful, it should show that the accounts `asmithee` and `gspelvin` exist and return an error code of 0.

Next, NSS LDAP must be configured, this can be done by opening the file `/usr/local/etc/nss_ldap.conf` and adding the lines as shown in Listing 2. Change the `host` and `base` fields to suit. If you are configuring the LDAP server to receive LDAP requests, also add the line that is commented out and remove the line `host 136.186.230.241`. Furthermore, to ensure the `slapd` server starts instantly (rather than taking about five minutes, add the `bind_policy` line (seen commented below).

Listing 6: /usr/local/etc/nss ldap.conf

```
#host    127.0.0.1
host     136.186.230.241
base     dc=rule241,dc=caia,dc=swin,dc=edu,dc=au
```

```
uri       ldap://rule241.caia.swin.edu.au

#bind_policy soft
```

Moreover, the file `/etc/nsswitch.conf` must have the following lines edited. Ensure you can log in after changing the file and before logging out.

Listing 7: /etc/nsswitch.conf

```
group: files ldap
passwd: files ldap
```

Create the home directory for both users and use `ls -l` to ensure that the group and user names are appearing correctly (instead of their user IDs).

Then we must set up PAM, to do this several configuration files must be changed. It is advised that after each change is made you attempt to log into the client using another PuTTy session to ensure you have not lost access to the system. Firstly, copy `/usr/local/etc/openldap/ldap.conf` to `/usr/local/etc/`. This can easily be done in the `openldap` directory by entering `cp ldap.conf ..`, this act as the configuration file for PAM LDAP.

Thereafter move to the directory `/etc/pam.d/` and first open `system`, then `sshd` and `su`. Note that \\ is used where the line splits on the page.

The configuration for the `auth` and `account` settings must be as follows:

Listing 8: /etc/pam.d/system

```
auth sufficient pam_opie.so no_warn no_fake_prompts
auth requisite pam_opieaccess.so no_warn allow_local
auth sufficient /usr/local/lib/pam_ldap.so no_warn
auth required pam_unix.so  no_warn try_first_pass nullok

account required pam_login_access.so
account required pam_unix.so
account required /usr/local/lib/pam_ldap.so \\
    no_warn ignore_authinfo_unavail ignore_unknown_user
```

Listing 9: /etc/pam.d/sshd

```
auth sufficient pam_opie.so no_warn no_fake_prompts
auth requisite  pam_opieaccess.so no_warn allow_local
auth  sufficient /usr/local/lib/pam_ldap.so  no_warn
auth required pam_unix.so no_warn try_first_pass

account required pam_nologin.so
account required pam_login_access.so
account required pam_unix.so
account  required /usr/local/lib/pam_ldap.so \\
    no_warn ignore_authinfo_unavail ignore_unknown_user
```

Listing 10: /etc/pam.d/su

```
auth sufficient pam_rootok.so no_warn
auth sufficient pam_self.so no_warn
auth requisite pam_group.so no_warn group=wheel root
_only fail_safe ruser
auth include system

account include system
account required /usr/local/lib/pam_ldap.so \\
    no_warn ignore_authinfo_unavail ignore_unknown_user
```

Thereafter, the LDAP client should be setup. Make sure to log into all accounts (both local and LDAP based) via `ssh` and `su`.

7 Additional Features and Configuration

While the above sections deal with the creation and operation of an LDAP server, below are modifications that were made to the server to benefit the troubleshooting process.

7.1 LDAP Web Monitor

The web monitor provides an alternative method of checking the status of the LDAP server. It provides information on if the server is online and what performance it has (in essence: CPU and memory). This means that any user can check if the LDAP is reachable and if connection issues are due to the LDAP server being offline or due to another network fault. An example is a firewall misconfiguration, which could could allow HTTP through but not LDAP packets.

This page can be accessed via `http://rule241.caia.swin.edu.au`

7.1.1 Configuring the Web Monitor

To configure the `.cgi` file, it was placed in `/usr/local/www/apache24/data`. The following steps were then undertaken to make sure it works:

1. The address of the web server was set to `rule241.caia.swin.edu.au:80`.
2. `Options +ExecCGI` was added to `/usr/local/etc/apache24/http.conf` directory.
3. `AddHandler cgi-script .cgi` was added to `/usr/local/etc/apache24/http.conf` for the main directory.
4. `DirectoryIndex` was edited to `index.cgi` from `index.html`.
5. `mpm-prefork modules` was uncommented.

Thereafter the web monitor was functional.

8 Appendix (Code)

Below is the code of the four scripts; \\ is used when a line is too long for the page.

8.1 Add Users

Listing 11: addusr.sh

```
#!/usr/local/bin/bash

# addusr
# Author: Joshua Redolfi
# Add a user to the LDAP server on rule241.caia.swin.edu.au
# Takes the following arguments
# [uid] [First name] [Last name] [Password] OPTIONAL: [uidNumber]

#Get our args
uid=$1
fname=$2
lname=$3
idnum=$4

domain="dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
host="rule241.caia.swin.edu.au"
#make sure the above two match
rootdn="Manager"
rootpw=secret
port=389

if [ $1 == "help" ];
then
        echo "Command usage: ./addusr.sh [uid] [First name] \\
        [Last name] [uid number]"
        retVal=1
        return ${retVal} 2>/dev/null # this will attempt to return
        exit "${retVal}" # this will get executed if the above failed.
fi

#enter password
read -p "Enter the password: " pass

#create file tmp.ldif (and remove old one if it exists)
rm /tmp/tmp.ldif
echo 'dn: cn='g_$uid',ou=Group,'$domain'' >> /tmp/tmp.ldif
echo 'cn: 'g_$uid'' >> /tmp/tmp.ldif
```

```
echo 'gidNumber: '$idnum'' >> /tmp/tmp.ldif
echo 'objectClass: posixGroup' >> /tmp/tmp.ldif
echo 'objectClass: top' >> /tmp/tmp.ldif
echo '' >> /tmp/tmp.ldif
#now let's make the user
echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
echo 'givenName: '$fname'' >> /tmp/tmp.ldif
echo 'sn: '$lname'' >> /tmp/tmp.ldif
echo 'cn: 'u_$uid'' >> /tmp/tmp.ldif
echo 'uid: '$uid'' >> /tmp/tmp.ldif
echo 'objectClass: inetOrgPerson' >> /tmp/tmp.ldif
echo 'objectClass: top' >> /tmp/tmp.ldif
echo 'objectClass: posixAccount' >> /tmp/tmp.ldif
echo 'objectClass: shadowAccount' >> /tmp/tmp.ldif
echo 'uidNumber: '$idnum'' >> /tmp/tmp.ldif
echo 'gidNumber: '$idnum'' >> /tmp/tmp.ldif
echo 'homeDirectory: /home/'$uid'' >> /tmp/tmp.ldif
echo 'loginShell: /usr/local/bin/bash' >> /tmp/tmp.ldif
echo 'userPassword: '$pass'' >> /tmp/tmp.ldif

#add this ldif file to slapd
#slapadd -l /tmp/tmp.ldif

#adding to slapd via ldapadd
ldapadd -D cn=$rootdn,$domain -w $rootpw -h $host:$port -f /tmp/tmp.ldif

#check error code to see if user was actually added
if [ $? -eq 1 ]
then
        echo 'There was an error in adding the .ldif file'
else
        echo 'User '$uid'/'$fname' '$lname' added successfully'
fi

rm /tmp/tmp.ldif
if [ $? -eq 1 ]
then
        echo 'tmp.ldif could not be removed!'
else
        echo 'tmp.ldif removed from /tmp/'
fi
```

8.2 Remove Users

Listing 12: rmusr.sh

```
#!/usr/local/bin/bash

# rmusr
# Author: Joshua Redolfi
# Remove a user off the LDAP server on rule241.caia.swin.edu.au
# Takes the following arguments
# [uid] [First name] [Last name]

#Get our args
uid=$1

if [ $1 == "help" ];
then
        echo "Command usage: ./rmusr.sh [uid]"
        retVal=1
        return ${retVal} 2>/dev/null # this will attempt to return
        exit "${retVal}" # this will get executed if the above failed.
fi

domain="dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
host="rule241.caia.swin.edu.au"
#make sure the above two match
rootdn="Manager"
rootpw=secret
port=389

ldapdelete -D cn=$rootdn,$domain -w $rootpw -h \\
$host -p $port "cn=g_$uid,ou=Group,$domain" \\
"cn=u_$uid,ou=People,$domain"

#check error code to see if user was actually removed
if [ $? -eq 0 ]
then
        echo 'User '$uid' removed successfully'
else
        echo 'There was an error in removing the user'
fi
```

8.3 Edit Users

Listing 13: editusr.sh

```
#!/usr/local/bin/bash
```

```
# editusr
# Author: Joshua Redolfi
# Edit a user's details on the LDAP server \\
on rule241.caia.swin.edu.au

cname="u_"$name
domain="dc=rule241,dc=caia,dc=swin,dc=edu,dc=au"
host="rule241.caia.swin.edu.au"
#make sure the above two match
rootdn="Manager"
rootpw=secret
port=389

#control statement for options \\
(help, password, name, shell, home)
if [ $1 == "help" ]
then
        echo "Command usage: ./editusr.sh [option]"
        echo "password - change a user's password"
        echo "name - change a user's name \\
        (first name and last name)"
        echo "shell - change a user's shell (bash, csh and sh)"
        echo "home - change a user's home directory"
        retVal=1
        return ${retVal} 2>/dev/null # this will attempt to return
        exit "${retVal}" # this will get \\
        executed if the above failed.
elif [ $1 == "password" ]
        then
        echo "Enter user id to change password for"
        read -p "User ID:" uid
        ldappasswd -D cn=$rootdn,$domain -w $rootpw -h $host \\
        -p $port -A -S cn="u_"$uid,ou=People,$domain
        #check error code to see if password was actually changed
        if [ $? -eq 0 ]
        then
                echo 'User password changed successfully'
        else
                echo 'There was an error in changing the password'
        fi
elif [ $1 == "name" ]
        then
        echo "Enter user id of the account to change the name of"
        read -p "User ID:" uid
        rm /tmp/tmp.ldif
        echo "Enter the new first name:"
```

```
        read -p "First name:" fname
        echo "Enter the new surname:"
        read -p "Surname:" lname
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        echo 'changetype: modify' >> /tmp/tmp.ldif
        echo 'replace: givenName' >> /tmp/tmp.ldif
        echo 'givenName: '$fname'' >> /tmp/tmp.ldif
        #modify first name first, done in two steps because \\
        ldapmodify doesnt like it otherwise
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        echo 'changetype: modify' >> /tmp/tmp.ldif
        echo 'replace: sn' >> /tmp/tmp.ldif
        echo 'sn: '$lname'' >> /tmp/tmp.ldif
        #then modify last name
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
elif [ $1 == "shell" ]
        then
        echo "Enter user id of the account to change the shell of"
        read -p "User ID:" uid
        echo "Type name of the shell you wish \\
        to change the user shell to:"
        echo "Options: bash, sh, csh"
        read -p "Shell:" shell
        rm /tmp/tmp.ldif
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        #control statement for shells, reject if invalid
        if [ $shell == "bash" ]
        then
                echo 'changetype: modify' >> /tmp/tmp.ldif
                echo 'replace: loginShell' >> /tmp/tmp.ldif
                echo 'loginShell: /usr/local/bin/bash' >> /tmp/tmp.ldif
        elif [ $shell = "sh" ]
        then
                echo 'changetype: modify' >> /tmp/tmp.ldif
                echo 'replace: loginShell' >> /tmp/tmp.ldif
                echo 'loginShell: /bin/sh' >> /tmp/tmp.ldif
        elif [ $shell = "csh" ]
        then
                echo 'changetype: modify' >> /tmp/tmp.ldif
                echo 'replace: loginShell' >> /tmp/tmp.ldif
                echo 'loginShell: /bin/csh' >> /tmp/tmp.ldif
```

```
        else
                echo "Invalid option, won't change shell type"
                echo "Exiting..."
                retVal=1
                return ${retVal} 2>/dev/null # this will \\
                attempt to return
                exit "${retVal}" # this will get executed \\
                if the above failed.
        fi
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
elif [ $1 == "home" ]
then
        echo "Enter user id of the account to \\
        change the home directory of"
        read -p "User ID:" uid
        echo "Enter the new directory (absolute reference):"
        echo "Example: /newlocation/"
        read -p "Home directory:" home
        rm /tmp/tmp.ldif
        echo 'dn: cn='u_$uid',ou=People,'$domain'' >> /tmp/tmp.ldif
        echo 'changetype: modify' >> /tmp/tmp.ldif
        echo 'replace: homeDirectory' >> /tmp/tmp.ldif
        echo 'homeDirectory: '$home'' >> /tmp/tmp.ldif
        #modify home directory here
        ldapmodify -D cn=$rootdn,$domain -w $rootpw \\
        -h $host:$port -f /tmp/tmp.ldif
        rm /tmp/tmp.ldif
else
        echo "Option invalid, type \\
        './editusr.sh help' to see all options."
fi
```

8.4 Change Passwords

Listing 14: changepasswd.sh

```
#!/usr/local/bin/bash

# changepasswd
# Author: Joshua Redolfi
# Edit a user's password on the LDAP \\
server on rule241.caia.swin.edu.au
# Takes no arguments
```

```
name=$(whoami)

domain='dc=rule241,dc=caia,dc=swin,dc=edu,dc=au'
host=rule241.caia.swin.edu.au
#make sure the above two match
rootdn=Manager
rootpw=secret
port=389

ldappasswd -D cn=$rootdn,$domain -w $rootpw -h \\
$host -p $port -A -S cn="u_"$name,ou=People,$domain

#check error code to see if password was actually changed
if [ $? -eq 0 ]
then
        echo 'User_'$name'_password_successfully'
else
        echo 'There_was_an_error_in_changing_the_password'
fi


#ldappasswd -D cn=Manager,dc=rule241,dc=caia,dc=swin, \\
dc=edu,dc=au -w secret -h rule241.caia.swin.edu.au \\
-p 389 -A -S cn=asmithee,ou=People,dc=rule241, \\
dc=caia,dc=swin,dc=edu,dc=au
```