

# **SUMMER TRAINING REPORT**

On

## **MASTER SESSION FOR SSH-CONSOLE SERVER**

Submitted to Guru Gobind Singh Indraprastha University, Delhi (India)  
in partial fulfilment of the requirement for the award of the degree of

**B.TECH**

in

**INFORMATION TECHNOLOGY**

**Submitted By**

**PRAGYA TRIPATHI**

**Roll. No. 00596303122**



**DEPTT. OF INFORMATION TECHNOLOGY**

**MAHARAJA SURAJMAL INSTITUTE OF TECHNOLOGY,  
NEW DELHI-110058**

**November 2024**

## ACKNOWLEDGEMENT

A research project owes its success, from commencement to completion, to the individuals who are passionate about the work at various stages. On this page, I would like to express my gratitude to all those who helped me throughout this study.

First, I would like to extend my sincere gratitude to **Dr. Sunesh Malik** (HOD, Department of Information Technology, Maharaja Surajmal Institute of Technology, New Delhi) for allowing me to undergo a 60-day summer training program at Ciena India Private Limited.

I am grateful to my guide, Mr. Nilabh Shant, for his assistance in completing the project assigned to me. Without his friendly help and guidance, developing this project would have been difficult.

I also want to thank Mr. Arun Kumar and Mr. Anand Shivhare for their genuine help and inspiration in preparing the final report and presentation.

Last but not least, I sincerely thank all the staff members of Ciena India Private Limited for their support, who made my training valuable and fruitful.

Submitted By:

Pragya Tripathi (00596303122)

## CERTIFICATE BY COMPANY



### CERTIFICATE OF COMPLETION OF INTERNSHIP

This is to certify that Candidate Name has successfully participated in Internship program with Ciena India Private Limited. His/Her service details are as follows:

**Name** : Pragya Tripathi  
**Reference Code** : 33292  
**Position/Service Title** : Summer Student  
**Duration of training** : 17 June 2024 – 16 Aug 2024

We wish you all the best in your future endeavors.

Yours Sincerely,  
for Ciena India Pvt. Ltd.

*Kyadav*



Kanika Yadav  
Associate, People Services

\*This certificate is not an accreditation, and thus Ciena disclaims from any legal obligation or for interpretations thereof, arising out of its use"

## **CANDIDATE’S DECLARATION**

I, **PRAGYA TRIPATHI**, Roll No. 00596303122, B. Tech (Semester- 5<sup>th</sup>) of the Maharaja Surajmal Institute of Technology, New Delhi hereby declare that the Internship Report entitled “**MASTER SESSION FOR SSH-CONSOLE SERVER**” is an original work and data provided in the study is authentic to the best of my knowledge. This report has not been submitted to any other Institute for the award of any other degree.

**Pragya Tripathi**

(00596303122)

**Place:** Maharaja Surajmal Institute of Technology, Affiliated to GGSIPU

**Date:** 8<sup>th</sup> November’ 2024

## ORGANIZATION INTRODUCTION

**Organization Name:** Ciena India Private Limited

**Industry:** Telecommunications

**Headquarters:** 7035 Ridge Road, Hanover, Maryland 21076, United States

**Website:** <https://www.ciena.com/>



Ciena Corporation is an American networking systems and software. It was established in 1992 and headquartered in Hanover, Maryland, USA. Ciena is a global leader in networking systems, services, and software, primarily focused on providing solutions for telecommunications and data networks. The company specializes in optical and routing systems, services, and automation software. In fact, Ciena supports more than 85 percent of the world's largest service providers as well as cloud and regional service providers, enterprise networks, financial services, healthcare, utilities, media and entertainment, retail, public sector, and all levels of education.

Ciena India is the company's largest R&D facility outside of North America, and has consistently been ranked India's Top Optical Networking Company in 2019 by Voice & Data magazine.

The company reported revenues of \$3.63 billion and more than 8,000 employees, as of October 2022. Gary Smith serves as president and chief executive officer.

Customers include AT&T, Meta, Reliance, Deutsche Telekom, KT Corporation and Verizon Communications.

## **ABSTRACT**

In the evolving landscape of data centre management, the need for robust and efficient systems has become paramount. To address this demand, Meta partnered with Ciena to implement an advanced XGSPON network infrastructure. Ciena's design encompassed multiple Optical Network Units (ONUs), each tailored to optimize network performance and reliability. My specific contribution was centred on the ONU 3803-MTL, a critical component in this sophisticated network.

The primary objective of my internship was to design a Master session for the SSH-Console server associated with the 3803-MTL. This Master session is pivotal for monitoring and managing all active sessions and connections within the ONU. It provides administrators with the capability to list active sessions by port number and username, which is essential for tracking and auditing network activity. Additionally, the system allows for the checking of administrative states, indicating whether an administrator is enabled or disabled for specific operations, thus ensuring proper control and oversight.

A significant feature of this Master session is its ability to identify and terminate suspicious or problematic sessions, thereby enhancing network security and operational integrity. This functionality is crucial for preventing unauthorized access and maintaining the stability of the data centre environment.

This report details the design and functionality of the Master session, highlighting its role in streamlining session management and improving the overall efficiency of data centre operations. By integrating advanced monitoring and control mechanisms, the Master session contributes to a more secure and reliable data centre infrastructure. The project represents a noteworthy advancement in data centre management, emphasizing the importance of effective session oversight in maintaining operational excellence and security in large-scale environments.

## LIST OF FIGURES

Fig 1.1: Bash Logo	3
Fig 1.2: AWK Logo	5
Fig 1.3: Linux Logo	6
Fig 1.4: TigerVNC Logo	8
Fig 1.5: SSH-Protocol Logo	10
Fig 1.6: MobaXterm Logo	12
Fig 1.7: Telnet Logo	13
Fig 1.8: Bitbucket Logo	14
Fig 1.9: Confluence Logo	15
Fig 3.1: Basic PON Architecture	20
Fig 3.1: Ciena XGSPON Architecture (Complex)	22
Fig 3.3: Ciena XGSPON Architecture (Simplified)	23
Fig 3.4: ONU 3803 MTL connected to Console Box	24
Fig 3.5: SSH-Console Server	25
Fig 3.6: Flowchart of Master Session script	28
Fig 3.7: Master Session 1	30
Fig 3.8: Master Session 2	31
Fig 3.9: Master Session 3	31
Fig 3.10: Master Session 4	32
Fig 3.11: Master Session 5	32

## **TABLE OF CONTENT**

<b>TOPIC</b>	<b>PAGE NO</b>
Title Page	I
Acknowledgement	II
Certificate By Company	III
Candidate's Declaration	IV
Organization Introduction	V
Abstract	VI
List of Figures	VII
Chapter – 1 Introduction	1
1.1 Need & Objective	1
1.2 Project Overview	1-2
1.3 Software Used	3-14
Chapter – 2 Project Design	15
2.1 Objective	15-16
2.2 Project Strategy	15-17
Chapter - 3 Implementation	18
3.1 PON and XGSPON Overview	18-19
3.2 Ciena PON Architecture	20-22
3.3 Design and Working of SSH-Console Server	22-24
3.4 Introduction to Master Session	24
3.5 Development and Implementation Approach	25
3.6 Algorithm and Data Handling	25-28



3.7 System Integration Testing	29
3.8 Deployment and Monitoring	30-32
3.9 Maintenance and Future Improvements	33
Chapter - 4 Result & Discussion	34
4.1 Performance Metrics and Evaluation	34
4.2 Case Studies and Analysis	34
4.3 Challenges and Solution	35
Chapter – 5 Future Scope & Conclusion	36
5.1 Future Scope	36
5.2 Conclusion`	37
References	38-40



# **CHAPTER 1**

## **INTRODUCTION**

### **1.1 Need & Objective**

In today's digital age, data is the new gold, driving innovation and transforming industries. The exponential growth of data has led to the rise of data-centric companies, harnessing information for insights and competitive advantage. Hyperscale companies like Amazon, Google, Meta, and Microsoft are at the forefront, providing scalable cloud solutions to manage vast data volumes. This surge in data has spurred an unprecedented demand for data centres, which form the backbone of modern digital infrastructure, ensuring seamless storage, processing, and transmission of information.

Meta, a hyperscale company and a client of Ciena, requested them to design an XGSPON system for their personal data centre. In response, Ciena began developing multiple Optical Network Units (ONUs) as part of the larger XGSPON solution, such as the ONU 3802, 3801, 3803, 3806, and 3807. I specifically worked on the ONU 3803-MTL. This ONU, connected to a console (includes an SSH-Console Server), is to be mounted on top of the data centre racks. The server is responsible for monitoring and managing the equipment and servers housed on each rack, ensuring efficient performance and control.

My role and primary objective in this project were to design the Master Session for the SSH-Console, which allows the operator (master) to monitor and control all active sessions on the console. This script plays a crucial role in managing session activity and maintaining the infrastructure's integrity across the data centre's servers.

### **1.2 Project Overview**

Passive Optical Networks (PON) are a type of telecommunications network that utilize optical fibre to deliver high-speed internet services to multiple users. A typical PON consists of three main components: the Optical Line Terminal (OLT) located at the service provider's central office, the Optical Network Unit (ONU) situated at the customer's premises, and a passive optical splitter that connects the OLT to multiple ONUs.

Among the various PON technologies, XGS-PON (10 Gigabit-capable Symmetric Passive Optical Network) stands out as an advanced solution, offering symmetrical bandwidth of up to 10 Gbps. Hyperscale companies require XGSPON for its ability to provide high-capacity, cost-effective broadband services, ensuring efficient data transmission across vast networks. Additionally, XGSPON's support for multiple services and scalability aligns with the growing demand for reliable, high-speed connectivity in data centres.

Ciena was requested by its client, Meta, to develop an XGSPON solution for its data centre, leading to the creation of various ONUs, including the ONU 3803-MTL that I specifically worked on. This ONU, mounted on top of data centre racks and connected to a console, is designed to monitor and manage rack equipment, ensuring high performance. Key components of the XGSPON setup include the Micro OLT, SFP, passive splitter, ONU and variety of cables, all crucial for effective connectivity and signal conversion.

The SSH-Console Server running on the 3803-MTL facilitates secure and remote management of devices in the network. It includes components like the SSH daemon, OMCI for service configuration, USB ports, plug, logger and bus attachments. This server allows seamless communication between clients and the rack-mounted ONU, enabling efficient control and monitoring through multiple sessions per USB connection.

The Master Session is a critical component of the SSH-Console Server, designed to enhance session management by allowing administrators to handle multiple active sessions efficiently. Its functionality includes filtering sessions, managing connectivity, and ensuring real-time updates on active sessions, thereby improving the overall performance and reliability of the network management process.

## 1.3 Software Used

During the internship, I came across many various software, protocols, tools and technologies. We will discuss many of those below:

### 1.3.1 Bash

#### 1. Key Features of Bash:



**Figure 1.1:** Bash Logo

**Source:** <https://bashlogo.com/>

- i. **Broad Compatibility:** Bash is the default shell on most Linux distributions and is widely supported across Unix-like systems. Scripts written in Bash can run on almost any Linux-based system without additional dependencies, making it ideal for networking components like ONU consoles.
- ii. **Simplicity and Readability:** Bash syntax is relatively simple and readable, which is helpful for scripts that require configuration and automation in networking environments. You can handle tasks like setting up connections, parsing output, managing processes, and logging with concise, readable code.
- iii. **Automation and Scripting Efficiency:** Bash is a powerful scripting language ideal for automation in server management. Its capability to handle complex scripts simplifies repetitive tasks in an SSH-console environment, such as session management, logging, and automation of regular server maintenance activities.
- iv. **Command Line Control:** Bash provides direct access to command-line utilities such as netstat and netstat -tbn, making it highly suitable for tasks that require direct interaction with the operating system. Direct command-line control helps in

managing server resources efficiently, as you can write and execute scripts that perform a series of administrative actions in one go.

- v. **Integration with Linux System Utilities:** Bash seamlessly integrates with Linux system utilities (e.g., grep, awk, sed, ps), allowing powerful data manipulation and process control. This capability supports session monitoring, user tracking, and other essential functions within the SSH-console server environment.

## 2. Why Bash for Master Session in SSH-Console Server:

- i. **Real-time Monitoring and Alerts:** With Bash, real-time monitoring scripts can be implemented to alert administrators of specific events within the SSH-console server, such as unauthorized access attempts or system resource thresholds being met. This enhances the security and responsiveness of the server.
- ii. **Streamlined Session Management:** In server management, many tasks are repetitive, such as backups, updates, and monitoring. Bash scripts can automate these tasks, reducing the need for manual intervention.
- iii. **Ease of Implementation and Maintenance:** Bash's simplicity and readability make script maintenance straightforward, especially for long-term server management. This is beneficial for ongoing SSH operations and configuration adjustments.
- iv. **System Interaction:** Bash allows direct interaction with the operating system's command-line utilities such as netstat and netstat -tbn, enabling users to execute system commands quickly and effectively.

### 1.3.2 Awk

#### 1. Key Features of Awk



**Figure 1.2:** Awk Logo

**Source:** <https://en.wikipedia.org/wiki/AWK>

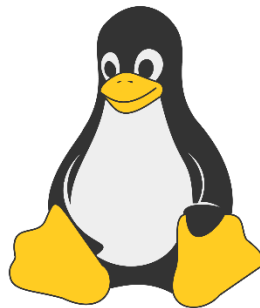
- i. **Powerful Text Processing:** AWK is a specialized tool designed for text processing and data extraction. Its ability to handle and manipulate structured text files (like logs and CSVs) makes it invaluable for analysing session logs and extracting relevant information from server outputs in an SSH-console environment.
- ii. **Pattern Matching and Reporting:** AWK excels in searching for specific patterns within data. This feature allows administrators to quickly filter through logs to identify critical events, such as unauthorized access attempts or errors, which is crucial for maintaining security and performance in an SSH session.
- iii. **Built-in String and Arithmetic Operations:** AWK includes a rich set of built-in functions for string manipulation and arithmetic calculations. This functionality enables complex data analysis and reporting directly within scripts, making it easier to summarize session statistics or resource usage metrics.
- iv. **Integration with Shell Scripts:** AWK can be easily integrated into shell scripts, allowing for the creation of powerful command pipelines. This is beneficial for automating tasks in the SSH-console environment, such as generating reports or filtering session data.
- v. **Portability and Efficiency:** As a part of the Unix toolset, AWK is highly portable and available on virtually all Unix-like systems. Its efficiency in handling large datasets makes it suitable for processing extensive logs generated during SSH sessions.

## 2. Why Awk for Master Session in SSH-Console Server:

- i. **Text Processing:** The language is designed for processing and analysing text files, making it ideal for extracting, filtering, and formatting session data.
- ii. **Field Manipulation:** The bash shell script can use awk to split lines into fields, allowing it to easily extract specific columns (e.g., port numbers and usernames) based on delimiters.
- iii. **Sorting and Uniqueness:** By utilizing AWK, we can generate unique lists of ports and usernames. For instance, the use of sort -u in combination with awk helps in filtering out duplicates from the session data.
- iv. **Integration with Shell Commands:** AWK works seamlessly with other shell commands, allowing for complex data processing pipelines. It can be used alongside commands like sort and grep to enhance data manipulation.

### 1.3.3 Linux

#### 1. Key Features of Linux



**Figure 1.3** Linux Logo

**Source:** <https://www.linux.org/>

- i. **Open-Source and Free:** Linux is an open-source operating system, meaning it's free to use, modify, and distribute. This transparency and collaborative development model make Linux highly customizable and accessible, ideal for both personal and enterprise-level deployments.



- ii. **Robust Security and Stability:** Known for its strong security and stability, Linux is a preferred choice for servers and critical systems. Its user permissions and firewall configurations support provide robust security, making it well-suited for SSH-console environments where secure access and data integrity are paramount.
- iii. **Extensive Command-Line Tools:** Linux offers a powerful command-line interface (CLI) with a vast set of tools and utilities for scripting, process management, and automation. For SSH sessions, Linux's CLI provides the flexibility and efficiency needed to manage remote systems effectively.
- iv. **Wide Hardware Compatibility:** Linux is compatible with a broad range of hardware, from high-performance servers to older legacy systems. This flexibility allows it to be used in various environments, supporting diverse system setups and making it a versatile choice for SSH-console servers.
- v. **Multi-User and Multi-Tasking Capabilities:** Linux's multi-user, multi-tasking capabilities allow multiple users to access and perform tasks on a single system simultaneously. This feature is invaluable in SSH environments where multiple sessions and users need to work concurrently.

## 2. Why Linux for Master Session in SSH-Console Server:

- i. **Stable Environment for Remote Access:** Linux provides a stable and reliable environment for SSH sessions, ensuring minimal downtime and high performance. Its stability makes it ideal for running master sessions where uninterrupted access is crucial for remote management.
- ii. **Efficient Command-Line Operations:** Linux's CLI enables efficient session management, automation, and scripting, which is essential for SSH-console server administration. Its wide range of built-in commands and scripting capabilities streamline tasks like monitoring, reporting, and troubleshooting.
- iii. **Robust Networking Capabilities:** Linux includes advanced networking tools and protocols, such as OpenSSH, which facilitate secure remote connections. Its networking features are essential for maintaining stable and secure SSH sessions, allowing administrators to manage and monitor connections seamlessly.
- iv. **Powerful Shell Scripting Support:** Linux provides robust shell scripting capabilities, enabling automation of routine tasks within SSH sessions. This scripting functionality

allows administrators to automate session management, data processing, and reporting, which improves productivity and reduces manual intervention.

- v. **Optimized for Diverse Hardware:** Linux's lightweight and adaptable architecture allows it to run efficiently on a wide range of hardware, from high-end servers to older, resource-constrained systems. This versatility ensures that SSH-console servers can be deployed on virtually any hardware, making it a flexible and cost-effective choice for various organizational needs.

### 1.3.4 TigerVNC

#### 1. Key Features of TigerVNC



**Figure 1.4** TigerVNC Logo

**Source:** <https://tigervnc.org/>

- i. **High-Performance Remote Access:** TigerVNC is designed for high-performance remote desktop sharing, making it an ideal tool for accessing and controlling remote systems. Its optimized graphics handling allows for smooth rendering, even over slower network connections, making it valuable for users who need responsive remote sessions.
- ii. **Security and Encryption:** TigerVNC supports Secure Socket Layer (SSL) and Transport Layer Security (TLS) encryption, ensuring secure remote sessions. This is essential for safeguarding sensitive data transmitted over remote connections, especially in SSH-console environments where security is a priority.
- iii. **Cross-Platform Compatibility:** TigerVNC is cross-platform and supports multiple operating systems, including Linux, Windows, and macOS. This

compatibility allows seamless remote connections across diverse environments, providing flexibility for users working in mixed OS setups.

- iv. **Virtual Desktop Management:** TigerVNC allows users to create and manage multiple virtual desktops on a remote server. This feature is especially useful in multi-user environments where multiple sessions are required, each with its own unique configuration and permissions.
- v. **Lightweight and Efficient:** TigerVNC is designed to be lightweight and minimizes resource consumption on both the client and server. This makes it suitable for environments where system resources are limited or where remote sessions need to be efficient, such as in virtualized environments.

## 2. Why TigerVNC for Master Session in SSH-Console Server:

- i. **Remote Session Management:** TigerVNC provides an intuitive solution for managing remote desktops, making it ideal for SSH-console servers. It allows administrators to connect to and control multiple sessions, monitor server activity, and troubleshoot issues from a central interface.
- ii. **Secure and Encrypted Connections:** In an SSH-console environment, security is paramount. TigerVNC's support for SSL and TLS encryption ensures that remote sessions remain secure, protecting sensitive session data and minimizing the risk of unauthorized access.
- iii. **Multi-User Access:** TigerVNC allows multiple users to access different virtual desktops simultaneously, making it ideal for collaborative SSH sessions. This feature is beneficial in environments where multiple administrators or users need concurrent access to the server.
- iv. **Persistent Sessions:** One of the standout features of TigerVNC is its ability to maintain session persistence. Unlike some remote desktop tools, TigerVNC allows sessions to keep running on the server even if the client disconnects or the user shuts down their computer. This means any progress, open applications, or configurations within the remote session remain intact and accessible upon reconnecting.

### 1.3.5 SSH- Protocol

#### 1. Key Features of SSH-Protocol

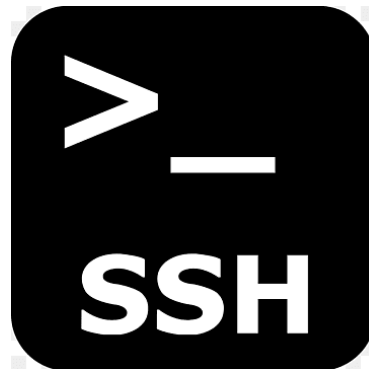


Figure 1.5 SSH- Protocol Logo

Source: <https://www.pngwing.com>

- i. **Secure Data Transmission:** The SSH (Secure Shell) protocol provides a secure channel over an unsecured network by using encryption. This ensures that all data transmitted between the client and the server remains confidential and protected from eavesdropping, making it essential for secure remote access.
- ii. **Authentication Mechanisms:** SSH supports multiple authentication methods, including password-based, public key, and host-based authentication. This flexibility allows administrators to implement robust security measures tailored to their specific needs, enhancing the overall security of SSH sessions.
- iii. **Remote Command Execution:** SSH enables users to execute commands on a remote machine as if they were local. This functionality is vital for system administration, allowing administrators to manage servers efficiently without needing physical access.
- iv. **Session Management:** SSH provides features for session multiplexing and management, allowing multiple terminal sessions over a single connection. This helps to conserve resources and streamline operations, particularly in environments where multiple commands need to be executed concurrently.

## 2. Why SSH-Protocol for Master Session in SSH-Console Server:

- i. **Secure Remote Access:** The primary reason for using SSH in an SSH-console server is its ability to provide secure remote access to systems. This is crucial for maintaining the integrity and confidentiality of session data, especially in environments where sensitive information is handled.
- ii. **Efficient Session Management:** SSH's support for session multiplexing allows administrators to manage multiple sessions effectively without needing to establish new connections each time. This reduces overhead and increases efficiency in handling multiple remote tasks.
- iii. **Versatile Authentication Options:** SSH's flexible authentication mechanisms enable administrators to choose the most appropriate method for their security requirements. Public key authentication, for example, provides enhanced security compared to traditional password methods, reducing the risk of unauthorized access.
- iv. **Widely Supported and Standardized:** SSH is a widely adopted standard across various platforms and systems, ensuring compatibility and ease of use. Its integration into many operating systems makes it a reliable choice for secure communications, simplifying the deployment of secure access solutions in diverse environments.

### 1.3.6 MobaXterm



**Figure 1.6 MobaXterm Logo**

Source: <https://mobaxterm.mobatek.net/>

MobaXterm is a versatile terminal emulator that allows you to manage remote sessions easily. I used it to connect locally and test my scripts efficiently. With support for multiple protocols, including SSH, RDP, and VNC, it simplifies remote access.

MobaXterm also has a user-friendly interface, making it ideal for developers and administrators. Its built-in tools for scripting and file transfer enhance productivity, allowing you to execute commands and transfer files seamlessly between local and remote environments. Overall, MobaXterm is a great solution for testing scripts and managing remote sessions effectively.

### 1.3.7 Telnet Protocol



**Figure 1.7 Telnet Logo**

Source: <https://www.arigo-software.de>

Telnet is a simple and widely used network protocol that enables users to establish remote sessions over a TCP/IP network. I used Telnet to connect to a testing board (ONU) for initial session testing. It provides a straightforward command-line interface, making it easy to send commands and receive responses.

After conducting tests with Telnet, I transitioned to SSH sessions on the ONU-3803 MTL for enhanced security. This workflow allowed me to validate my script effectively in a controlled environment before deploying it in a more secure setup. Telnet's simplicity makes it a useful tool for preliminary testing and diagnostics.

### 1.3.8 Bitbucket



**Figure 1.8 Bitbucket Logo**

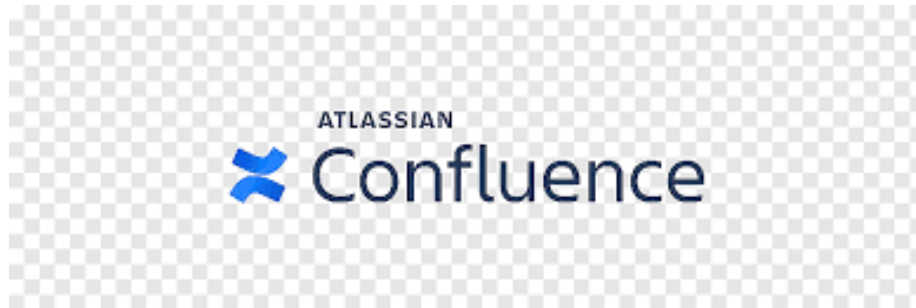
**Source:** <https://bitbucket.org/product/>

Bitbucket is a web-based version control repository designed for collaborative software development. It supports Git and Mercurial, making it easy to manage source code repositories. I used Bitbucket to host and share my scripts related to SSH sessions on the ONU. Its built-in code review and pull request features facilitate collaborative development, allowing team members to provide feedback and suggestions on code changes.

Bitbucket also integrates seamlessly with tools like Jira and Confluence, enhancing project management and documentation workflows. The platform's branching and merging capabilities enable efficient handling of multiple development streams, ensuring that features can be developed concurrently without disrupting the main codebase. Overall, Bitbucket is a powerful tool for version control and collaboration, essential for maintaining high-quality code in software projects.



### 1.3.9 Confluence



**Figure 1.9 Confluence Logo**

**Source:** <https://www.logo.wine/logo/Confluence>

Confluence is a collaborative documentation tool designed for teams to create, share, and manage knowledge efficiently. It allows users to organize content in spaces and pages, making it easy to structure documentation related to projects and technical processes.

I utilized Confluence to document my testing procedures and findings, to ensure that my mentor could access the latest updates on the master session script. Moreover, I also used Confluence to understand the architecture and working of the whole project and specifically the ONU 3803-MTL as well.

The platform's version control enables tracking changes and reverting to previous versions, if necessary, which is crucial for maintaining accurate records. Additionally, its robust search capabilities help quickly locate important information, enhancing team collaboration and knowledge sharing. Integrating with tools like Jira streamlines project management, making Confluence an essential tool for effective communication and documentation within the team.

# CHAPTER 2

## PROJECT DESIGN

### 2.1 Objective

The objective of the Master Session in our SSH-Console Server project is to establish a robust and centralized control mechanism for monitoring and managing active user sessions on the ONU 3803-MTL. This session efficiently handles session listings, checks administrative states, and identifies suspicious connections to maintain system security. It provides administrators with a streamlined interface for managing multiple sessions, including listing details by port number and username. By integrating comprehensive session monitoring, the Master Session enhances operational efficiency and ensures the overall stability of the SSH-Console environment. The design emphasizes reliability, ease of use, and robust performance for optimal network management.

### 2.2 Project Strategy

#### 1. PON and XGSPON Overview

- **Introduction to PON:** This section will provide an overview of fibre technologies, focusing specifically on Passive Optical Networks (PON). Key components such as optical line terminals (OLT), optical network units (ONU), and passive splitters will be discussed.
- **Introduction to XGSPON:** Here, we will delve into the specifics of XGSPON (10 Gigabit Symmetrical Passive Optical Network), which enhances traditional PON technologies.

#### 2. Ciena XGSPON Architecture

- **Introduction and key components:** Outline the architecture of Ciena's XGSPON solution, detailing its key components such as the Micro OLT, Switch, SFP, Passive splitter, Fixed Attenuator, ONU, Console Box, Cables etc.

- **Working of the Ciena XGSPON:** Explain the operational principles of Ciena's XGSPON, including data transmission processes and network efficiency.

### 3. Design and working of the SSH-Console Server

- **Architecture of the SSH-Console Server:** Present the design architecture of the SSH-Console Server, describing its components in detail to gain a basic understanding of its architecture.
- **Working of the SSH-Console Server:** Provide details of the operational processes of the SSH-Console Server, including session initiation, management, and termination. This will showcase how the server facilitates remote management of ONUs and enhances operational efficiency.

### 4. Introduction to the master session

- **Master session functionality:** Define the Master Session concept, outlining its role in managing multiple SSH sessions concurrently. To explain its functionalities, such as session monitoring, filtering, and session termination, highlighting its importance for network administrators.

### 5. Development and Implementation approach

- **Planning:** Detail the initial planning phase of the project, covering requirements gathering, system analysis, and development environment setup. We will emphasize the importance of thorough planning for successful implementation.
- **Implementation:** To discuss the programming tools and technologies used in developing the Master Session, such as Bash scripts, MobaXterm, and TigerVNC. This segment will also cover coding practices and testing strategies employed throughout the development process.

### 6. Algorithm and Data Processing

- **Session Management and Logic:** This section will outline the flow of the script that allows session management, explaining how the script maintains session integrity and

manages multiple user inputs. To discuss the logic that allows for efficient session handling using a flowchart.

- **Data Handling and Processing:** Explore how the script processes data from multiple sessions and commands, including session state management, filtering, and logging. This segment will highlight the importance of effective data handling for overall system performance.

## 7. System Integration and Testing

- **System Integration:** Detail how the Master Session was integrated into the SSH-Console Server, ensuring compatibility and reliability with existing systems. To discuss the challenges faced during integration and the solutions implemented.
- **Testing and Validation:** Describe the testing approach, including unit testing and performance evaluation. This section will emphasize the importance of rigorous testing in validating the Master Session's functionality and reliability in a live network environment.

## 8. Deployment and Monitoring

- **Deployment:** To explain the deployment process of the Master Session in the live network environment, addressing challenges encountered and solutions applied to ensure smooth deployment.
- **Monitoring and Feedback:** Discuss the tools and methods used to monitor system performance and gather user feedback post-deployment. This segment will highlight how monitoring contributes to ongoing improvements and user satisfaction.

## 9. Maintenance and Future Improvements

- **Bug Fixes and Optimization:** Address the post-deployment maintenance phase, detailing how bug fixes were managed and optimizations were implemented to enhance script performance and reliability.
- **Plans for Future Enhancement:** Outline potential future improvements for the Master Session, including advanced features, scalability options, and plans for extending functionality to support more complex network environments. This section will emphasize the importance of continuous development in meeting evolving user needs.

## **CHAPTER 3**

### **IMPLEMENTATION**

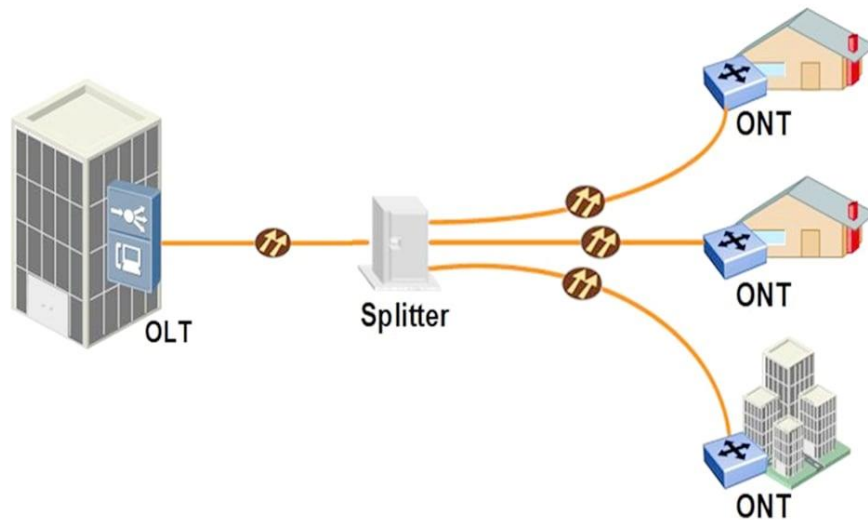
#### **3.1 PON and XGSPON Overview**

**3.1.1 Introduction to PON:** A network is a collection of connected devices that communicate with each other to share information, resources, and services. They can be set up through various mediums, such as wired connections, wireless systems, and fibre optics. Fiber networks use optical fibre cables to transmit data as pulses of light, offering high speeds, low latency, and large bandwidth, which are ideal for modern internet services. Unlike traditional copper cables, fibre networks deliver faster speeds over long distances, making them critical for high-demand applications like streaming, online gaming, and large-scale data transfers.

A Passive Optical Network (PON) is a type of fibre network that delivers internet services to multiple end users without using active electrical components between the provider's central office and the customer.

Key components of a PON network include:

- **Optical Line Terminal (OLT):** The OLT is located at the service provider's central office and serves as the network's main hub, managing traffic between the provider and multiple Optical Network Units (ONUs) at customer locations.
- **Optical Network Unit (ONU):** The ONU, sometimes called Optical Network Terminal (ONT), is the endpoint device at the customer's premises, converting optical signals into electrical signals for use by devices within the home or business.
- **Optical Splitter:** The passive optical splitter divides a single fibre signal from the OLT into multiple signals for delivery to various ONUs. This allows multiple customers to share a single optical fibre infrastructure, which reduces costs while delivering high-speed, reliable internet.



**Figure 3.1** Basic PON Architecture

Source: <https://www.optcore.net/what-is-pon/>

**3.1.2 Introduction to XGSPON:** XGS-PON, or "10-Gigabit Symmetrical PON," is an advanced PON standard that supports data speeds up to 10 Gbps for both upstream and downstream traffic, providing a significant boost over previous PON standards. XGS-PON is ideal for meeting the growing demand for high-speed internet in homes, businesses, and enterprise environments.

This technology enables internet service providers to deliver gigabit-speed connections, supporting modern applications such as cloud computing, remote work, and high-definition video streaming. With its high capacity, reliability, and scalability, XGS-PON is well-suited for future-proofing networks.

## 3.2 Ciena XGSPON Architecture

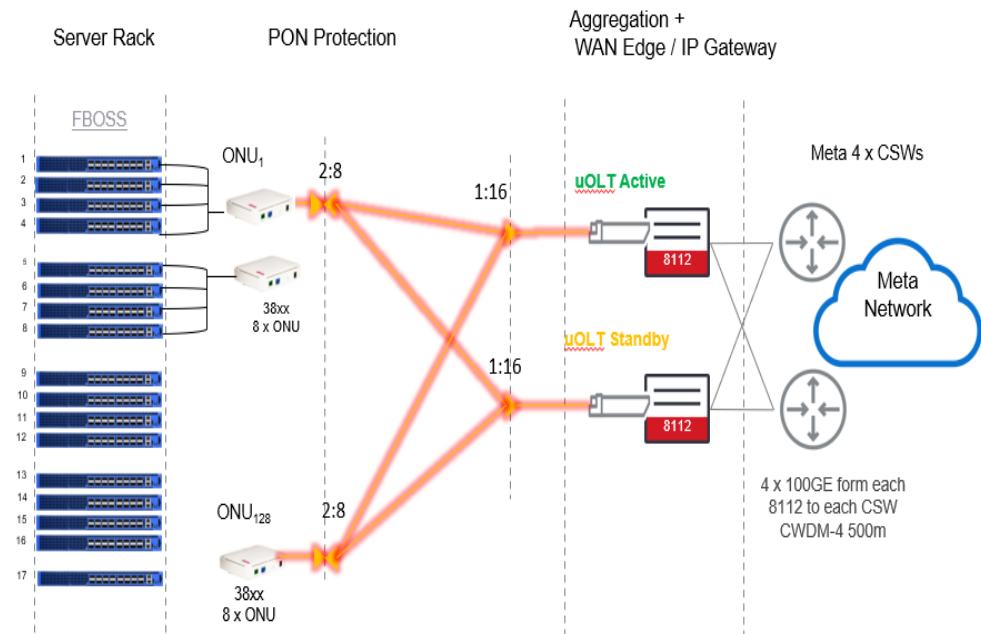
**3.2.1 Introduction and key components:** The Ciena XGS-PON architecture is a high-speed fibre-optic network solution designed to efficiently manage data distribution across large networks, like data centres. Built on the XGS-PON standard, which provides symmetric 10 Gbps data rates, this architecture is ideal for high-demand environments. It consists of various key components that work together to deliver fast and reliable internet to multiple endpoints.

The key components are as follows:

1. **Switch:** It is a device that works in a local network and manages or routes data traffic. This component, equipped with the Micro OLT receives electrical signals from the Meta Network and facilitates their conversion to optical signals via the OLT.
2. **Micro OLT:** Refers to a more compact version of the traditional OLT, which is beneficial for space-constrained environments. It is embedded within the switch. It acts as the central management device and is responsible for controlling, managing, and coordinating communication between the central network (i.e. Meta Network) and multiple ONUs (Optical Network Units). It distributes bandwidth, manages network traffic, and ensures quality of service (QoS) across connected devices. The Micro OLT also has a prebuilt SFP inside it.
3. **SFP:** Located within the OLT, the SFP module, which stands for Small Form-factor Pluggable, enables the physical connection between the fibre cabling and the OLT (or Switch). It is also responsible for signal conversion (electrical to optical signals).
4. **Passive Splitter:** A critical component that divides optical signals from the OLT to multiple ONUs.
5. **Fixed Attenuator:** Reduces signal strength to maintain optimal transmission levels.
6. **ONU (Optical Network Unit):** Devices that receive optical signals from the splitter, located at the user's end. Ciena produced multiple ONU models (3802, 3801, 3803, 3806, 3807), all operating on the OSI model's Layer 2. The ONU 3803-MTL, for instance, connects to the Console for session management.
7. **Console Box:** Connected to the ONU 3803-MTL through a console cable and to end devices via Type-C USB cables. Each console port supports multiple sessions.
8. **Cables:** Includes PON, Console, and Type-C USB cables.

# Data Center Management Architecture

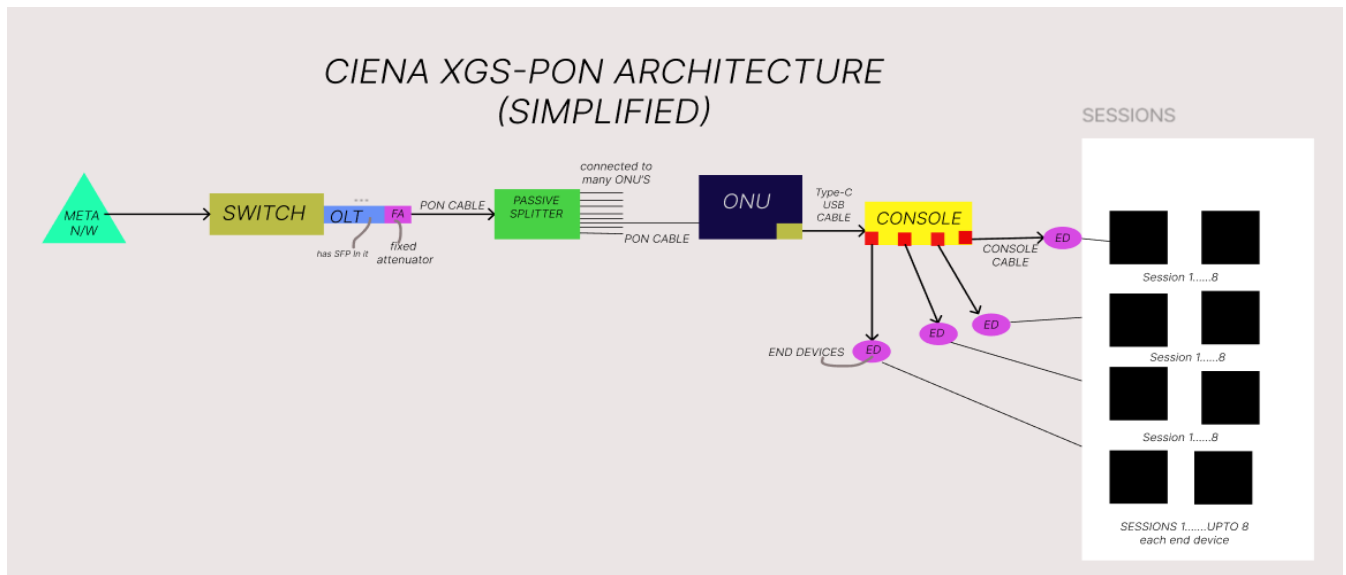
## With PON Protection



**Figure 3.2** Ciena XGSPON Architecture (Complex)

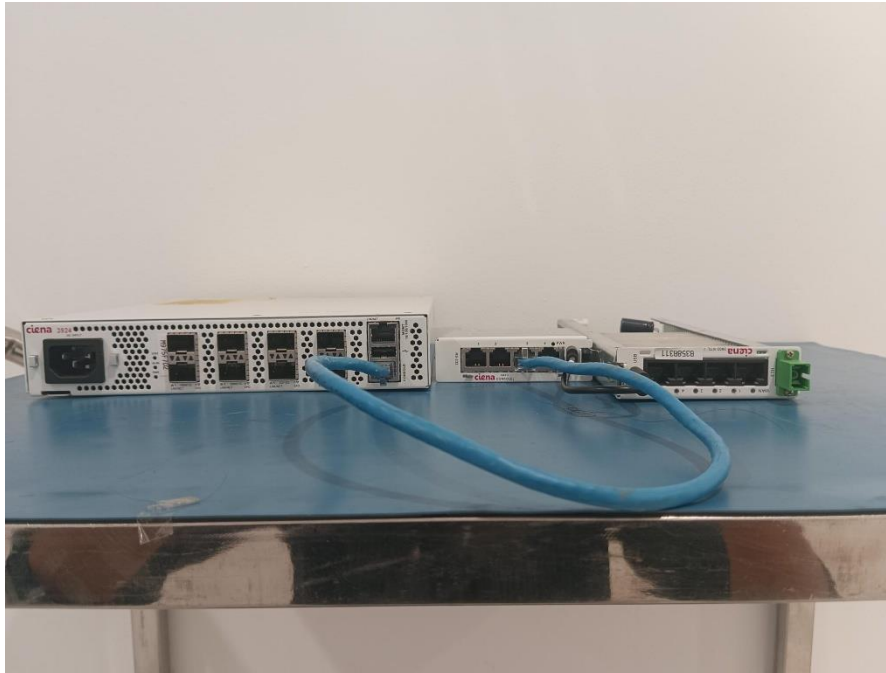
**Source:** Intellectual Property of Ciena India Private Limited





**Figure 3.3** Ciena XGSPON Architecture (Simplified)

**3.2.2 Working of the Ciena XGSPON:** The process begins when the Switch receives signals from the Meta Network. The Micro OLT inside the switch converts these signals to optical form and transmits them to a passive splitter through a PON cable. The splitter divides the signal, connecting multiple ONUs. Each ONU then connects to a Console Box, which links up to four end devices. Each end device can host up to eight sessions, providing up to 16 sessions per ONU.



**Figure 3.4** ONU 3803-MTL connected to Console Box

**Source:** Intellectual Property of Ciena India Private Limited

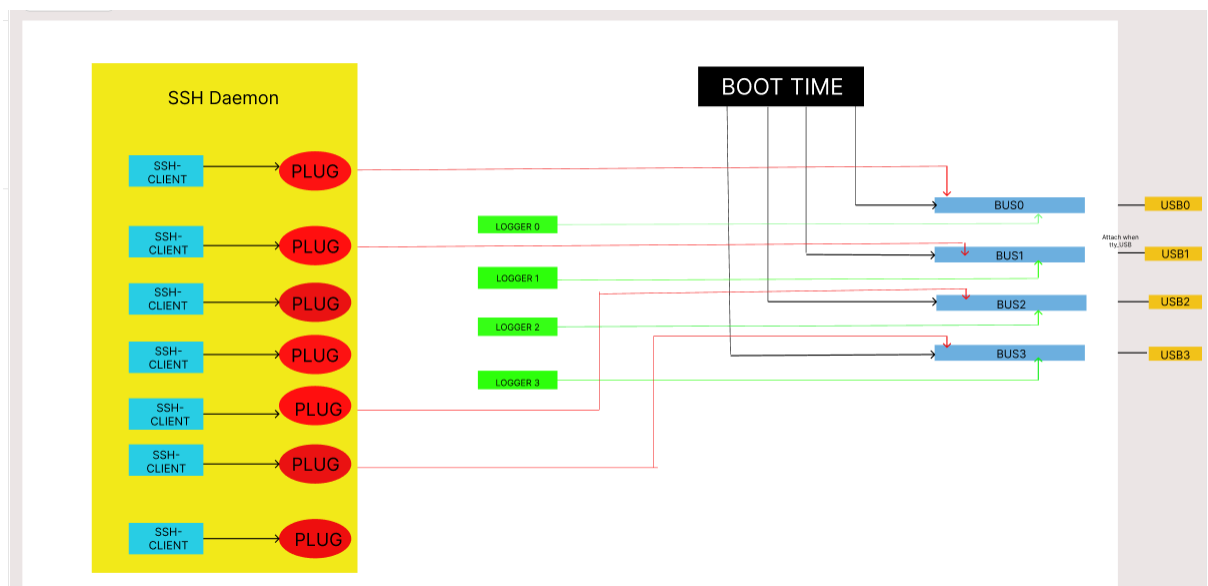
### 3.3 Design and working of the SSH-console server

**3.3.1 SSH-Console Server Architecture:** The SSH Console Server is designed to facilitate secure, remote access to devices connected via USB, providing a seamless method for managing multiple SSH clients through a centralized system. The architecture, as illustrated in the diagram, comprises key components such as SSH Daemon, ttyUSB devices, tty\_bus, tty\_attach, tty\_plug, ConsoleShm and loggers. Each component has a specific function, working together to create a versatile and robust server system for device management.

The key components are as follows:

1. **SSH Daemon (SSHD):** This daemon is essential for enabling encrypted SSH connections over the network. It listens on port 22 for incoming connections from SSH clients and manages authentication and communication.

2. **ttyUSB Devices:** These are physical USB devices connected to the server labelled as USB0, USB1, USB2, and USB3. They are created when the console box is connected.
3. **tty\_bus:** This creates virtual bus paths for each USB connection, allowing devices to plug into the SSH Console Server. The command `tty_bus` initializes a virtual bus path, allowing multiple clients to connect to the console server and allowing data transfer through it.
4. **tty\_attach and tty\_plug:** These utilities manage the attachment and connection of devices. `tty_attach` connects a real ttyUSB to a virtual tty\_bus, enabling communication, while `tty_plug` connects SSH clients to specific bus paths.
5. **ConsoleShm:** This shared memory structure allows virtual session indexing, supporting multiple sessions per port with a maximum of 16 sessions in total.
6. **Logger:** Each tty\_bus has a dedicated logger that records session details, including connection and termination events, providing insights and tracking for each session.



**Figure 3.5** SSH-Console Server Simplified

**3.3.2 Working of SSH-Console Server:** When the console box is connected, ttyUSB entries are created for the USB devices. During boot-up, a bus is established, and attachments are created for the devices. When a client starts a session, they can send data to the bus through a plug, which is then forwarded to the ttyUSB via the `tty_attach`. Similarly, USB devices can send data to the bus, which then transfers it back to the client through the plug. The logger broadcasts all ttyBus paths to ensure monitoring and security. When the boot process is shut down or rebooted, the ttyUSB entries are removed, the attachments are terminated, and the bus is shut down.

## **3.4 Introduction to the master session**

**3.4.1 Master Session Functionality:** The Master Session is a pivotal component for the SSH-Console Server, designed to streamline the management of active SSH sessions within the Ciena XGS-PON architecture. This script provides a user-friendly interface, allowing operators to efficiently list all active sessions, check the administrative states of connections, and identify any suspicious activities.

By integrating commands that facilitate session management, the Master Session plays a crucial role in enhancing network security and operational efficiency. Users can interactively explore various options, such as listing sessions by port number or username, ensuring they have immediate access to pertinent information. Furthermore, the script empowers administrators to terminate sessions based on specific criteria, such as serial number, port number, or username, thereby mitigating potential security risks.

The Master Session not only aids in monitoring active connections but also incorporates features for updating session listings and validating user inputs, which contributes to the robustness of the SSH-Console Server. Overall, this functionality supports effective network management, helping maintain a secure and organized operational environment within the XGS-PON infrastructure.

## 3.5 Development and Implementation Approach

**3.5.1 Planning:** During the initial planning phase, I focused on gathering requirements for session management, analysing system needs, and defining key functionalities, such as session listing, filtering, and termination. This involved understanding the specifics of session interactions and designing a clear workflow. The development environment was set up with essential tools and configurations to streamline coding and testing.

**3.5.2 Implementation:** For implementation, I used Bash as the primary scripting language to manage session logic, taking advantage of its compatibility with the Linux-based environment. MobaXterm and TigerVNC were utilized for remote access and visual session management, enhancing testing capabilities. Each function was coded and tested individually to ensure correct execution of tasks like listing sessions, handling user inputs, and terminating specified sessions. Iterative testing helped identify issues early and allowed for continuous improvements in functionality and performance.

## 3.6 Algorithm and Data Handling

**3.6.1 Session Management Logic:** In the session management logic, algorithms were implemented to list, monitor, and manage active sessions based on user-selected parameters, such as port number or username. This flowchart-driven design enables users to interact through a main menu, where choices lead to functions for viewing active sessions, filtering by specific identifiers, and terminating sessions as needed. The kill options include terminating all sessions for a given serial number or username, or only selected sessions, adding flexibility to the session management process.



**3.6.2 Data Handling and Processing:** For data handling and processing, the focus was on efficient data retrieval and manipulation to minimize resource consumption. The system processes session data to deliver timely responses while avoiding performance bottlenecks. Optimized loops and conditional checks ensure that only necessary resources are utilized, maintaining system responsiveness even under high session loads.

As the script would be running in an infinite loop in background; the same has been optimized to update or refresh all active connections to ensure consistency at all times.

## **3.7 System Integration and Testing**

**3.7.1 Integration:** In the integration phase, the Master Session script was incorporated into the SSH-Console Server, ensuring seamless compatibility and reliable operation within the existing infrastructure. The script was saved permanently and is executable via the Ciena PON manager/controller, allowing for streamlined access and control.

Administrators can utilize OMCI (connected to the switch) to operate the Passive Optical Network (PON) and use the master session script to access specific ONUs through their unique IP addresses. Each session on an ONU shares the same IP address as the ONU, differentiated only by port number and username.

**3.7.1 Testing and Validation:** For testing and validation, a comprehensive approach was adopted. Unit testing was conducted for individual script functions to verify session listing, filtering, and termination.

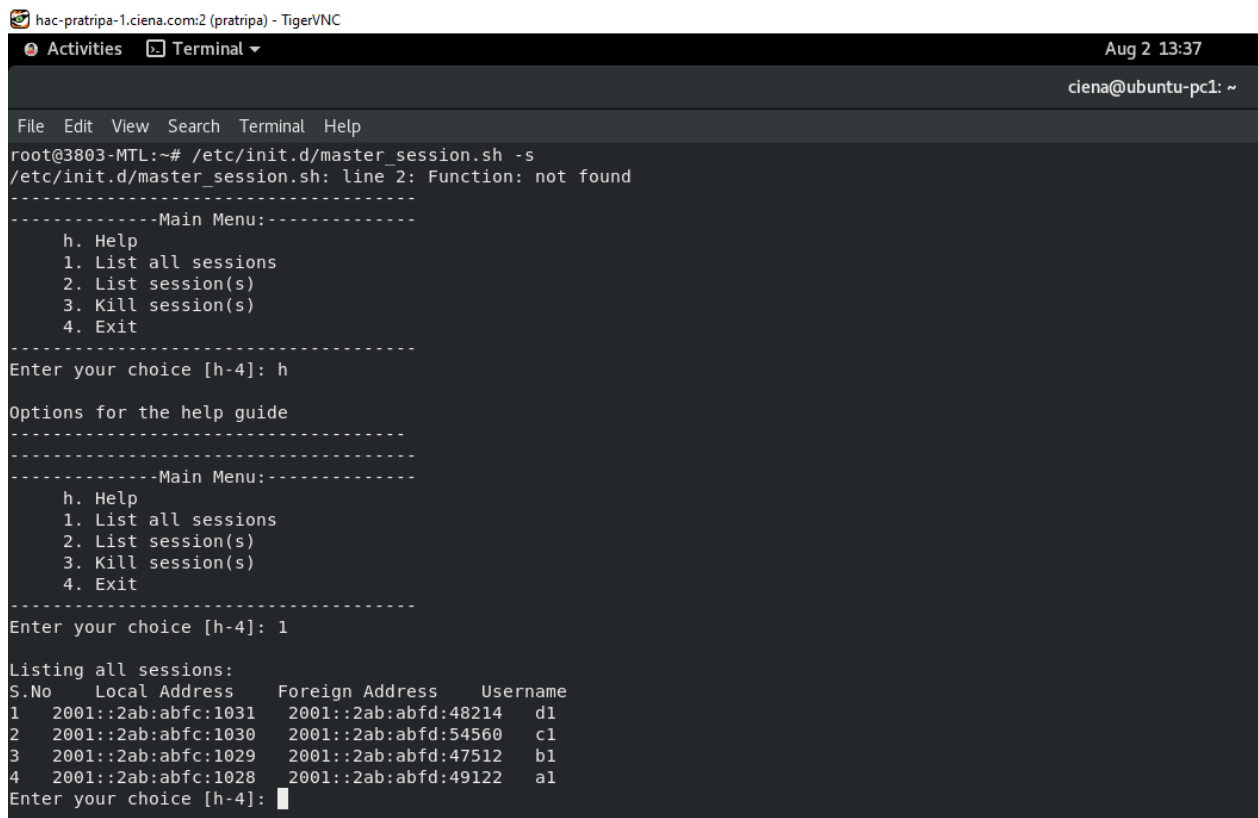
Integration testing ensured the script's compatibility within the PON management system, verifying that the script operated as expected in conjunction with OMCI.

Finally, the script was evaluated against various test cases, including scenarios where no active sessions were present and the user attempted to filter by port number or username, as well as cases where invalid serial numbers, port numbers, or usernames were entered.

Additional test cases included connection loss, session disconnection, and sudden changes in admin state. These issues were resolved by ensuring that the list of active sessions is continuously updated, providing accurate, real-time information and maintaining stability under different conditions.

## 3.8 Deployment and Monitoring

**3.8.1 Deployment:** The Master Session script was deployed in the live network environment, requiring careful coordination to minimize disruption. Challenges included handling real-time network fluctuations and ensuring compatibility with the SSH-Console Server. Solutions involved pre-deployment testing and phased implementation to ensure stability across the network.



```

hac-pratripa-1.ciena.com:2 (pratripa) - TigerVNC
Aug 2 13:37
ciena@ubuntu-pc1: ~
File Edit View Search Terminal Help
root@3803-MTL:~# /etc/init.d/master_session.sh -s
/etc/init.d/master_session.sh: line 2: Function: not found
-----Main Menu:-----
  h. Help
  1. List all sessions
  2. List session(s)
  3. Kill session(s)
  4. Exit
-----
Enter your choice [h-4]: h
Options for the help guide
-----
-----Main Menu:-----
  h. Help
  1. List all sessions
  2. List session(s)
  3. Kill session(s)
  4. Exit
-----
Enter your choice [h-4]: 1
Listing all sessions:
S.No   Local Address   Foreign Address   Username
1    2001::2ab:abfc:1031  2001::2ab:abfd:48214  d1
2    2001::2ab:abfc:1030  2001::2ab:abfd:54560  c1
3    2001::2ab:abfc:1029  2001::2ab:abfd:47512  b1
4    2001::2ab:abfc:1028  2001::2ab:abfd:49122  a1
Enter your choice [h-4]: 

```

Figure 3.7 Master Session 1



```

4  2001::2ab:abfc:1028  2001::2ab:abfd:49122  a1
Enter your choice [h-4]: 2

List session using given below options:
1.Listing sessions by port number
2.Listing sessions by username
Enter the required option: 1

Listing sessions by port number
-----

admin state: UP, port no: 1028
S.No  Local Address  Foreign Address  Username
4  2001::2ab:abfc:1028  2001::2ab:abfd:49122  a1
-----

admin state: UP, port no: 1029
S.No  Local Address  Foreign Address  Username
3  2001::2ab:abfc:1029  2001::2ab:abfd:47512  b1
-----

admin state: UP, port no: 1030
S.No  Local Address  Foreign Address  Username
2  2001::2ab:abfc:1030  2001::2ab:abfd:54560  c1
-----

admin state: UP, port no: 1031
S.No  Local Address  Foreign Address  Username
1  2001::2ab:abfc:1031  2001::2ab:abfd:48214  d1
-----
Enter your choice [h-4]: 

```

Figure 3.8 Master Session 2

```

Enter your choice [h-4]: 2

List session using given below options:
1.Listing sessions by port number
2.Listing sessions by username
Enter the required option: 2

Listing sessions by username
-----

Active sessions for username a1:
S.No  Local Address  Foreign Address  Username
4  2001::2ab:abfc:1028  2001::2ab:abfd:49122  a1
-----

Active sessions for username b1:
S.No  Local Address  Foreign Address  Username
3  2001::2ab:abfc:1029  2001::2ab:abfd:47512  b1
-----

Active sessions for username c1:
S.No  Local Address  Foreign Address  Username
2  2001::2ab:abfc:1030  2001::2ab:abfd:54560  c1
-----

Active sessions for username d1:
S.No  Local Address  Foreign Address  Username
1  2001::2ab:abfc:1031  2001::2ab:abfd:48214  d1
-----
Enter your choice [h-4]: 

```

Figure 3.9 Master Session 3

```

-----
Enter your choice [h-4]: 3

    Kill session using given below options:
        1.Killing session by serial number
        2.Killing session by port number
        3.Killing seesion by username
Enter the required option: 1

*****Killing session by serial number*****
S.No    Local Address    Foreign Address    Username
1    2001::2ab:abfc:1031    2001::2ab:abfd:48214    d1
2    2001::2ab:abfc:1030    2001::2ab:abfd:54560    c1
3    2001::2ab:abfc:1029    2001::2ab:abfd:47512    b1
4    2001::2ab:abfc:1028    2001::2ab:abfd:49122    a1
    Enter the serial number: 1
Session 1  killed succesfully.
Enter your choice [h-4]: █

```

**Figure 3.10** Master Session 4

```

Enter your choice [h-4]: 3

    Kill session using given below options:
        1.Killing session by serial number
        2.Killing session by port number
        3.Killing seesion by username
Enter the required option: 3

*****Killing session by username*****
Available usernames:
b1
c1
Enter the username: b1

Sessions for user b1:
2    2001::2ab:abfc:1029    2001::2ab:abfd:47512    b1
Choose an option:
A. Kill all sessions for user b1
B. Kill few sessions for user b1
Enter your choice (A/B): a
Session 2  killed succesfully.
Enter your choice [h-4]: █

```

**Figure 3.11** Master Session 5

**3.8.2 Monitoring and Feedback:** System performance was monitored using network monitoring tools integrated with the PON manager, providing real-time insights. Administrator feedback was gathered to identify usability issues and performance bottlenecks, enabling immediate adjustments. Continuous monitoring ensured reliability and helped maintain efficient resource allocation.

## **3.9 Maintenance and Future Improvements**

**3.9.1 Bug Fixes and Optimization:** Post-deployment maintenance included addressing bugs and refining the script for optimal performance. To improve efficiency, the active session data from the `netstat -tpn` command was stored in a file, which was continuously updated through a single function. This eliminated the need to repeatedly execute the command, significantly reducing processing time and optimizing resource usage. This enhancement ensured smoother performance, especially when handling multiple sessions.

**3.9.2 Plans for Future Enhancements:** Future improvements include adding advanced filtering options, automated session cleanup, and enhanced reporting features.

## **CHAPTER 4**

### **RESULTS AND DISCUSSION**

#### **4.1 Performance Metrics and Evaluation**

Key performance metrics for evaluating the Master Session script included response time, resource utilization, and session handling efficiency. By centralizing session data processing into a single function that continuously updated session information, the script avoided redundant system calls, resulting in faster response times and reduced CPU load.

With each ONU limited to a maximum of 16 sessions, the script demonstrated effective session handling without latency, ensuring smooth and reliable performance. Compared to previous methods, the Master Session script greatly improved monitoring and management by allowing administrators to control active sessions efficiently and reduce manual processes.

#### **4.2 Case Studies and Analysis**

In deployment scenarios, the Master Session script effectively handled cases where multiple sessions existed for a single ONU, each distinguished by unique port numbers and usernames. For example, when administrators needed to manage up to 16 simultaneous sessions on a specific ONU, the script allowed efficient filtering and control over these sessions without performance degradation.

Expected results, like accurate session listing and timely response, aligned closely with actual outcomes, validating the script's efficiency. Minor issues, such as handling unexpected disconnections, were resolved by ensuring the session list updated continuously for accurate tracking.

### **4.3 Challenges and Solutions**

Challenges during implementation included handling network latency, managing unexpected session drops, and validating user inputs. Latency was minimized by streamlining the script to reduce unnecessary command executions.

To address invalid inputs, such as incorrect serial numbers or usernames, validation checks were added, providing clear feedback to users. Security and compatibility with existing protocols were ensured by restricting session management access. These solutions collectively contributed to the script's reliability, allowing it to operate consistently in live environments with up to 16 active sessions per ONU, enhancing session management capabilities without compromise.

## **CHAPTER 5**

### **FUTURE SCOPE AND CONCLUSION**

#### **1.1 Future Scope**

Future enhancements for the Master Session script could include the integration of AI-based anomaly detection systems designed to identify unusual session patterns or potential security threats in real time. This feature would enable the system to proactively alert administrators about suspicious activities, thus strengthening overall network security.

In addition, implementing improved logging mechanisms will provide more detailed insights into session activities, which can be invaluable for troubleshooting and auditing purposes. A more intuitive user interface would enhance usability, allowing administrators to navigate session monitoring and management tasks more efficiently.

Furthermore, the Master Session script could be scaled to accommodate larger and more complex network environments. This could be achieved by incorporating multi-node management capabilities, allowing administrators to oversee multiple servers and sessions from a centralized interface.

Extended functionalities, such as automated session clean-up based on customizable rules, would enhance performance by ensuring that stale or inactive sessions are regularly purged, thus optimizing system resources. These advancements would transform the script into a versatile tool capable of adapting to the evolving needs of diverse network setups, including those in larger enterprises and service providers.

## **1.2 Conclusion**

The Master Session project has significantly enhanced the SSH-Console Server, contributing to streamlined and efficient network management. By automating session listing, monitoring, and control, the script has reduced administrative overhead and improved the reliability of session handling. This work not only provided a practical solution within the telecommunication industry but also underscored the importance of optimization and adaptability in network management tools.

The project's success highlights the value of continual improvement, demonstrating that even well-established systems can benefit from thoughtful upgrades and innovations.

Ultimately, this experience has emphasized the critical role that automation and scalability play in effectively managing modern networks, paving the way for future enhancements that will further elevate operational efficiency and security.

## REFERENCES

During the process of doing the following project I referred to various online material available as well as private documentation of Ciena.

### **To understand the working and architecture of Ciena XGSPON Solution and ONU 3803-MTL:**

- Ciena Confluence Page
- <https://github.com/danielinux/ttybus>
- [https://github.com/danielinux/ttybus/blob/master/tty\\_attach](https://github.com/danielinux/ttybus/blob/master/tty_attach)
- [https://github.com/danielinux/ttybus/blob/master/tty\\_plug.c](https://github.com/danielinux/ttybus/blob/master/tty_plug.c)

### **To understand PON and XGSPON:**

- <https://ecin.ca/optical-transceivers/xpon/10g-xgspont/#:~:text=What%20is%20XGS%2DPON%3F,10G%20up>
- <https://www.cisco.com/c/en/us/products/switches/what-is-passive-optical-networking.html>

### **To understand the tools used in development of Master Session:**

- <https://www.geeksforgeeks.org/bash-scripting-introduction-to-bash-and-bash-scripting/>
- <https://www.gnu.org/software/bash/manual/bash.html>
- <https://www.redhat.com/en/topics/linux/what-is-linux>
- <https://tigervnc.org/>
- [https://docs.redhat.com/en/documentation/red\\_hat\\_enterprise\\_linux/7/html/system\\_administrators\\_guide/ch-tigervnc](https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/7/html/system_administrators_guide/ch-tigervnc)
- <https://www.bu.edu/tech/services/security/server/vulnerability-management/xprobe/securing-mobaxterm/#:~:text=What%20is%20MobaXterm,on%20the%20M>



[icrosoft%20Windows%20desktop.](#)

- <https://bitbucket.org/product/guides/getting-started/overview#a-brief-overview-of-bitbucket>
- <https://www.geeksforgeeks.org/introduction-to-bitbucket/>
- <https://www.atlassian.com/software/confluence/resources/guides/get-started/overview#about-confluence>
- <https://www.techtarget.com/searchsecurity/definition/Secure-Shell#:~:text=SSH%20uses%20the%20client%2Dserver,terminal%20emulation%20or%20file%20transfers.>
- <https://www.geeksforgeeks.org/introduction-to-telnet/>

**To understand deployment of the product in a data center:**

- [https://youtu.be/q6WlZHLxNKI?si=x3yZf93k\\_PiOq1e8](https://youtu.be/q6WlZHLxNKI?si=x3yZf93k_PiOq1e8)
- [https://youtu.be/XZmGGAbHqa0?si=O3I\\_rVpDr0Y5o0nF](https://youtu.be/XZmGGAbHqa0?si=O3I_rVpDr0Y5o0nF)
- <https://aws.amazon.com/what-is/data-center/#:~:text=A%20data%20center%20is%20a,stores%20any%20company's%20digital%20data.>
- <https://www.cisco.com/c/en/us/solutions/data-center-virtualization/what-is-a-data-center.html>

**Information about Ciena:**

- <https://www.ciena.in/at-a-glance>
- <https://www.ciena.in/>
- <https://www.ciena.com/about>
- <https://en.wikipedia.org/wiki/Ciena>