Abdelmalek Essaadi University
Faculty of Sciences and Techniques of Tangier
Department of Computer Engineering

End of module project: research report

Installation and configuration of application services AND Digital investigation DFIR Chkrootkit

Directed by:

Alae eddine El Andaloussi
Chaimae Boussab

Supervised by:

Pr. Zouhair Abdelhamid

Program:

Master's Degree in IT Systems Security and Big Data

Table of Contents

Table of Figures

1 Introduction

1.1 Preface

Digital Forensics and Incident Response (DFIR) are two common terms in cybersecurity initially developed for Information Technology (IT) systems, based on technical steps including preparation, detection, containment, eradication, recovery, and post-incident activity [1]. Each step can be detailed to many technical actions which require high skills to perform. Operational Technology (OT) describes a family of systems used to manage, monitor, and control industrial operations focusing on the physical devices and processes they use (e.g., electricity, water, pharmaceutical). Implementing DFIR methods for these systems requires a series of dedicated extensions of procedures and techniques due to some unique aspects of OT.

1.1.1 Main Incident Response challenge in OT

Traditional Incident Response steps mainly focus on the technical aspects of the process and not on the procedure that leads the process. This issue is not a problem while dealing with an event in IT environments because, in most cases, the IT system is under the full responsibility and control of one department, the IT department. However, this is not the case in OT [2]. A primary aspect that affects incident handling in OT is the number of stakeholders (e.g., operators, maintenance teams, engineers, cyber analysts) doing some level of analysis in the system. As a result, a lack of clear procedures, lack of coordination, and lack of clear definitions of responsibility and authority can lead to failures when managing a real-time cyber incident.

Moreover, although OT systems are considered reliable, non-cyber incidents such as technical malfunctions or process anomalies are not rare. Therefore, there is always a challenge to

1.1.2 Main Digital Forensic challenge in OT

At a very fundamental meaning, Digital Forensics is about analyzing data with tools and techniques to answer questions such as what happened in the system, how it occurred, and what the impact was. The levels of analysis differ in the types of data and tools, the level of needed skill, and the time invested.

Unique devices and data types in OT require unique knowledge and skills to accomplish Digital Forensics. In addition, OT is a type of system that has a strong connection to the physical world. There is great importance in understanding the possible implications for safety and operation resulting from any forensic-related action performed in the system. Therefore, there must be a strong connection between technical OT personnel and Digital Forensics analysts.

1.2 Purpose and Scope

The purpose of this document is to provide an OT Digital Forensics and Incident Response (DFIR) framework. This framework expands the traditional technical steps by giving an Incident Response procedure based on the event escalation and provides additional techniques for OT Digital Forensics.

The scope of this document includes an overview of DFIR and its implementation within OT environments. Its goal is to provide the whole picture as a starting point for the organizations to establish their own OT DFIR capabilities.

2 Overview of OT DFIR

The first part of this section will describe three main terms which will be used in this document: Active Defense (AD), Incident Response (IR), and Digital Forensics (DF). The second part of this section will discuss some of the unique properties of OT systems, which base the motivation for the suggested OT DFIR framework in this document. Each paragraph in this section presents the different challenges and opportunities with any unique property and an essential strategy to deal with them

2.1 DFIR in general

2.1.1 Active Defense

According to the Sliding Scale of Cyber Security, a cybersecurity program is built on five levels [3][4]. The first two levels are Architecture and Passive Defense. These two levels include all the fundamental design considerations and security controls needed to eliminate most of the low-level and traditional cyber-attacks. In comparison to physical defense, these levels are the walls and gates around the property. But walls and gates are not enough. To achieve a satisfactory level of protection against sophisticated adversaries, guardians (or analysts in cybersecurity) must constantly walk around (or analyze the environment) looking for the next attack. In addition, they need to collect intelligence and use their monitoring systems to hunt the adversary before they can gain a foothold. Accordingly, the following two levels in the cybersecurity program are Active Defense and Threat Intelligence. The last level deals with Offensive activity to subdue the adversary while still in his environment. As shown in Fig. 1, the foremost part of the active defense approach related to this document is the Incident Response

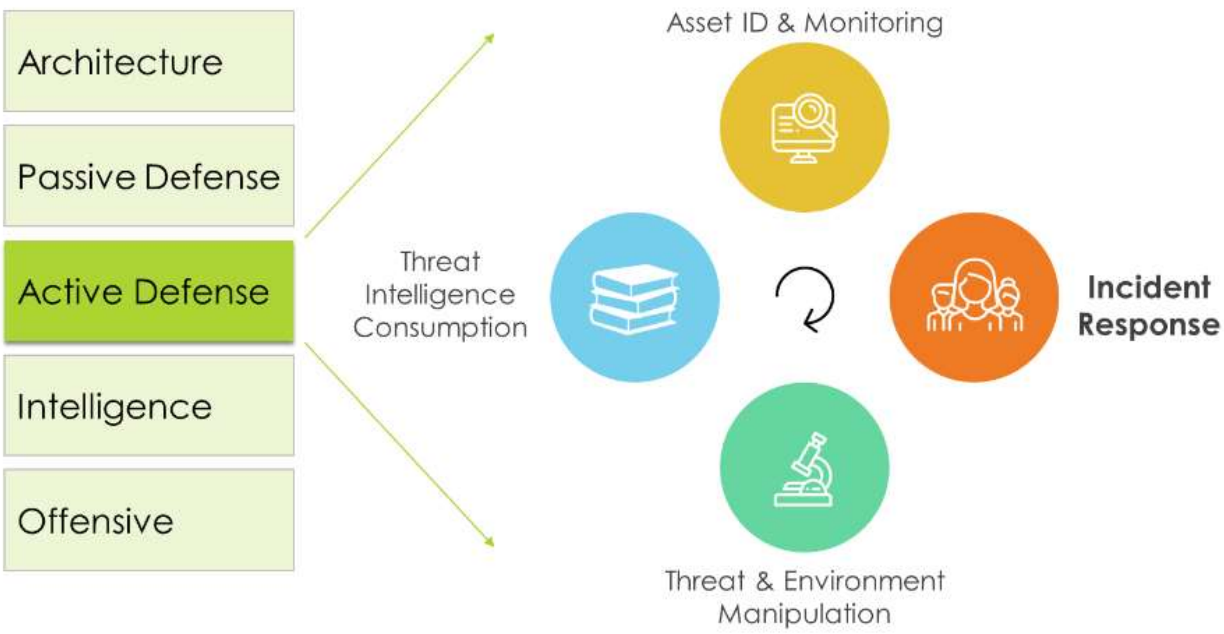


Figure 1 The Sliding Scale of Cyber Security: Active Defense

2.1.2 Incident Response

OT Incident Response is an organized approach to handling and managing the aftereffects of an incident with the primary goal of gathering enough information to contain and recover the system to operate safely. This is sometimes contrary to IT Incident Response which may focus on recovering the system to its pre-incident state immediately rather than examining every piece of forensic data.

NIST SP 800-61 [1] provides guidance for establishing and operating an IT Incident Response process in organizations. Although initially designed for IT environments, it can be adopted for OT systems with some adjustments. The general process includes preparation activities needed to be done before an incident occurs, a set of activities during an incident handling (detection & analysis, containment, eradication & recovery), and post-incident activities.

2.1.3 Digital Forensics

Digital Forensics is a subset of forensic science. It is considered the application of science to the identification, collection, examination, and analysis of data while preserving the integrity of the information and maintaining a strict chain of custody for the data [5]. Digital forensic techniques can be used for many purposes, such as investigating crimes and internal policy violations, reconstructing security incidents, troubleshooting operational problems, and recovering from accidental system damage. This guide presents Digital Forensics from a system point of view, not a law enforcement view.

Digital Forensics is both a field of science and a field of art. It has no deterministic technical procedure to provide a step-by-step guide that will lead the analyst directly to the answer. A lot of training and practice are needed to reach a proficient skill level. Therefore, the goal of this document is to provide the fundamental tools and techniques that will help to build specific OT Digital Forensics capabilities, while a high level of skill can be achieved after continued practice and training.

NIST SP 800-86 [6] is a guide for integrating Forensic Techniques into an Incident Response process. It includes four main phases:

- **Collection** – Relevant data is identified, labelled, recorded, and collected.
- **Examination** – Forensic tools and techniques are applied to identify and extract the relevant information from the collected data.
- **Analysis** – Information is analysed to achieve the proof that will explain the incident’s root cause.
- **Reporting** – Summarize the results and further recommendations

2.2 OT DFIR Unique Properties

2.2.1 Analysis levels during Incident Response

The primary property that affects the Incident Response process in OT is the number of stakeholders performing some level of analysis in the system. During normal operations, operators use sensors and actuators to perform process measurement and control. In addition, Security Operation Center (SOC) analysts are examining alerts and anomalies in the network and other IT components.

During a technical event, the maintenance teams perform a technical inspection to identify and solve mechanical (such as valve mechanism or pump transmission failure) and electrical malfunctions (such as valve actuator or pump motor failure). In addition, the control system engineers perform software analysis to provide technical support.

During a cyber incident, the Incident Response Team uses response methods and Digital Forensics to understand and resolve the incident. An after-event team can perform deep forensics (such as malware reverse engineering) to understand the incident.

The basic principles of these teams are similar. They all use data, tools, and skills to understand an issue in the system and respond correspondingly. The main differences are how much resources and attention each team invests during the Incident Response.

This property raises a major challenge in OT Incident Response. Any lack of clear procedure, lack of coordination, and clear definitions of responsibility and authority can lead to failures when managing a real-time cyber incident. One of the primary goals of the OT DFIR framework in this document is to address that challenge.

2.2.2 Is it a Malfunction or an Attack?

Technical malfunctions and process anomalies may occur more frequently than cyber incidents in OT systems. While cyber incidents are rare, they will usually start with similar symptoms. There are two edge strategies to deal with these malfunctions. One option is to address any technical issue as a potential cyber incident. On the one hand, this approach will ensure that no case is missed, but on the other hand, the Response Team may be desensitized due to many false positives and alert fatigue.

Another option is to provide regular technical support with zero cyber considerations. In this way, the Response Team will only be activated during complex events, but on the other hand, the initial stages of actual cyber events may be missed due to false negatives. The challenge is to balance these two edge strategies. The OT DFIR Framework in this guide suggests a balanced approach

3 OT DFIR Preparation

This section will discuss some of the preparation that should be done before an incident occurs. The section is focused on three main topics. First is the Incident Response Team's structure, tools, facilities, and training. The second is the Digital Forensic hardware and software tools and resources. And the third is the OT field systems preparations for forensics

3.1 Incident Response Team

3.1.1 Roles and Responsibilities

An Incident Response Team (IRT) can be a dedicated team that handles incidents as a full-time job. In most cases, this team is based on existing personnel defined and overseen by the management to be IRT members when needed, in addition to their day-to-day job.

As described above, many groups should be involved in the Incident Response process in OT systems. Here are some of the prominent roles that should be included in the team. Notice that it is a Role-Based list and not a Human Resource one. In a small organization, part of these roles could be applied by the same personnel or outsourcing.

- **Cyber OT Engineers** – Engineers familiar with both fields: OT/ICS Systems and OT Cybersecurity. This group should consist of three types of roles in a hierarchical order:
  - Team Leader – Responsible for establishing and maintaining the team and interfaces with management and POCs at other organizations.
  - Incident Leader – Coordinates all handlers, guides personnel, focuses the efforts, timeline the evens, maintains situational awareness, etc.
  - Data Handler – Responsible for digital data collection, performing timely analysis, containing and cleaning the infected systems, inform the leader. These members should have strong communication skills, work well in a team and under high stress, have patience, work by the plan, and ask for help when needed.
- **SOC Analysts** – Security Operation Center Analysts routinely monitor the IT and OT environments. This group is familiar with the system's baseline and should analyze the data and alerts from the monitoring systems.
- **Control System Engineers** – Experts in control systems development who also serve as a Technical Support Team. This group is most familiar with the PLC and HMI codes of the specific systems and should help with explaining the expected system behavior.
- **Process Operators** – The Facility Control Room operators are most familiar with the process behavior. Together with the Control System Engineers, the control system can be well understood.

- **IT Security** – Technology personnel within the IT security department should help with the event investigation and interpret regulatory requirements.
- **Physical Security** – Physical security staff can assist in gaining access to locations and physically securing forensic data and the area.
- **IT Professional** – Usually, this group is not in charge of the OT systems but can help with their networking and operation systems knowledge.
- **Management** – During an OT Digital Forensics process, it is essential to have management representatives be involved and informed.
- **Other Internal or External third-parties** – There might be a need to rely on Internal or External third parties to help with unique activities such as data recovery from damaged media, Legal advisors, Policy, Private aspects, etc.

3.1.2 Tool Kit

Here are some recommendations for additional tools (not a list of software collection and analysis tools) to include in an IR jump kit:

- **Safety –**
  - Personal protection equipment such as hard hats.
  - Out-of-band communication methods.
- **IR Workstations –**
  - A laptop with imaging/data collection tools.
  - Approved scripts and software for timely analysis.
- **Network tools –**
  - Hub or tap that can support the expected bandwidth.
  - Physical-layer converters.

- **Storage –**
    - o A few high-volume Hard Drives to store digital images and network traffic.
    - o Blank CDs and USB Drives.
- **Auxiliaries –**
    - o Cables/Connectors (USB, Serial, SATA, etc.)
    - o Digital camera for scene photography.
    - o Screwdrivers, power strips, flashlights.
    - o Notebooks and labels.
    - o Procedure checklists.
- **Digital Forensic Data list –**
    - o Fundamental data types (Section 4.2.1, Step A.2).
    - o Supplementary data types (Section 4.2.4, Step D.4).

3.1.3 Situation Room

- One of the uniqueness of dealing with the OT system is the number of stakeholders in the system. For example, Operators, local support teams, control systems engineers, Incident Response Teams, management, etc. During the incident handling, there is a risk that someone will do some action in the system without informing all the team members. Therefore, everyone must be in sync and up to date throughout the process.

- The Situation Room should be used for situation assessments and should serve only the directly related personnel to the incident. The Situation Room is the place for managing the incident, answering questions from the management, contacting POCs, updating timelines, etc. This is not the place for technical analysis to get done; that would occur in the Digital Forensics Lab

3.1.4 Training and practice

- **Training –** The OT Incident Response Team must have an education roadmap to earn knowledge and skills in various fields of profession. These fields include network architecture, industrial protocols, control systems development, industrial plants and processes, operating systems, virtual machines, forensic tools and methods, cybersecurity components, data analysis, scripting, incident handling, malware analysis, etc. This knowledge can be achieved through commercial and academic courses and self-research. An educational framework, for example, could be the NIST National Initiative for Cybersecurity Education (NICE) which provides training for various specialty areas such as Exploitation Analysis, Threat Analysis, and more.
- **Practice –** Theoretical knowledge in Digital Forensic fields is crucial but not enough. While actual OT Cyber incidents are fortunately not daily, the Forensic teams must practice their skills through exercises and incident simulations. These simulations can vary from practicing forensic methods while dealing with regular technical malfunctions to comprehensive organizational Cyber exercises.
- **Testing Lab –** The IRT members need a lab to test their tools and training. They must have experience with the tools they are using and understand each tool's capability and limitation. This lab should include Workstations and Hardware with an identical configuration as those in the OT systems, virtual environments for practice and running malware samples, IR off-network workstations, and of course, IR tools.

4 OT DFIR Framework

This section will introduce the OT DFIR Framework. It will describe the framework in general and then dive deep into every step of the process. The motivation for this framework is based on the unique properties of OT systems which are widely discussed in Sections 1 and 2

4.1 The Framework

Fig below describes the main Phases of the framework, while Fig on the next page, describes the detailed activities during each Step of each phase.

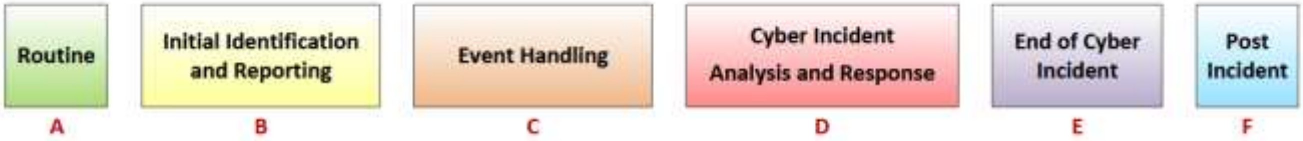

Routine
A
Initial Identification and Reporting
B
Event Handling
C
Cyber Incident Analysis and Response
D
End of Cyber Incident
E
Post Incident
F

Figure 2 The OT DFIR Framework

A. The **Routine** phase focuses on preparing the system for incident response, with activities like Asset Identification and Data Collection. During this phase, the SOC conducts Cyber Monitoring while the Facility Control Room (FCR) manages the operational Events Log. Data collection is integrated into the system routine, ensuring seamless operation

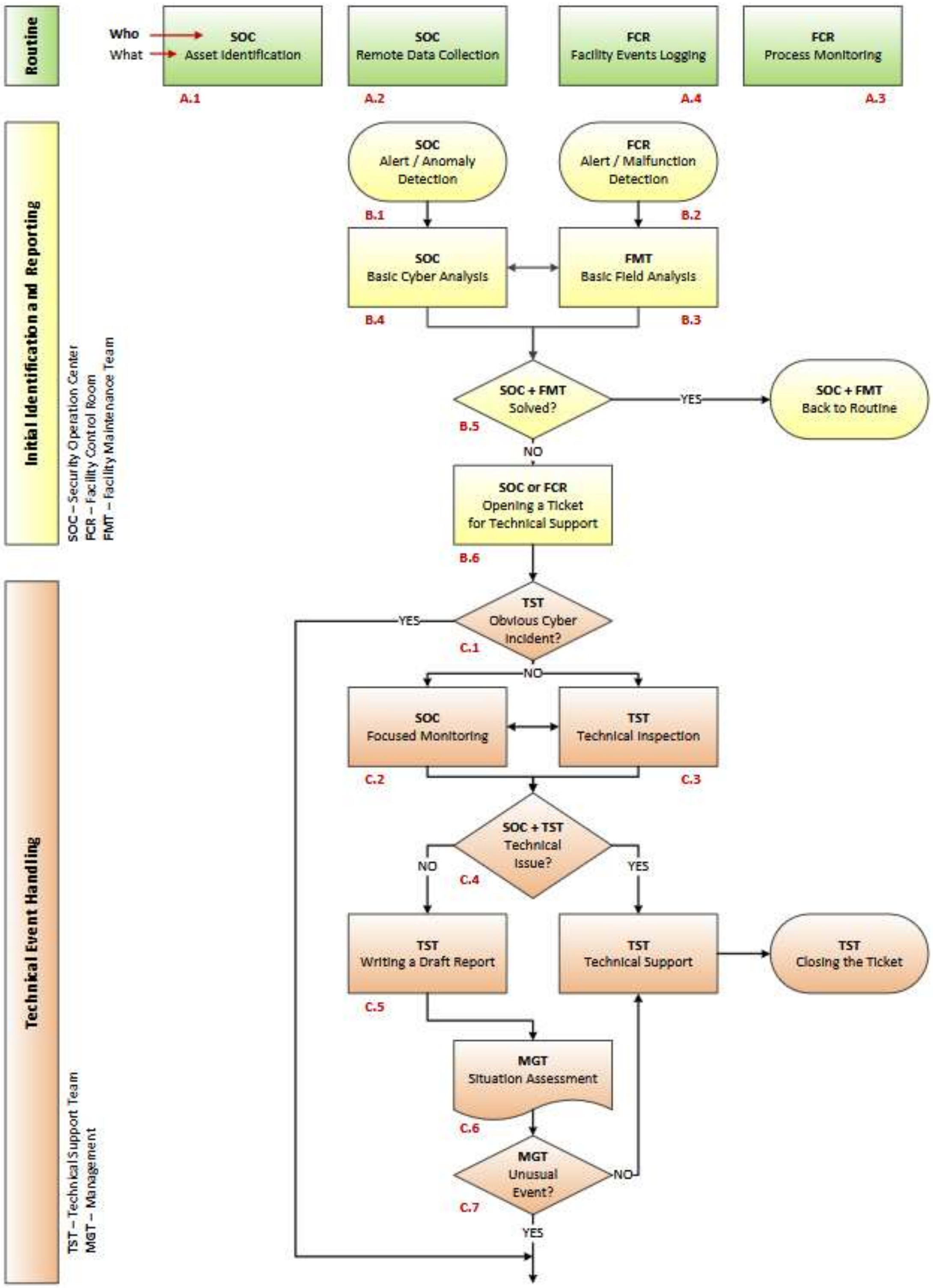
B. **Initial Identification and Reporting** phase begins when operational alerts or anomalies are detected at the SOC or FCR. A brief analysis is conducted to determine whether it is a typical malfunction or a potential incident. If the event is deemed a malfunction, it is resolved and the system returns to routine; if uncertain, a technical support ticket is opened.

C. **Technical Event Handling** phase, more significant technical issues are addressed with the support of the Technical Support Team (TST), who collaborate with the SOC for further escalation. The TST evaluates the event and provides a situation report to management. If the issue remains suspicious, the event moves to cyber incident analysis. During the

D. **Cyber Incident Analysis and Response** phase, management declares the incident status, and stakeholders are informed. The Incident Response Team (IRT) collects data, leveraging information from the routine phase, and performs Digital Forensics to understand the incident.

E. **End of Cyber Incident** phase marks the conclusion of the incident response process, which is determined by management even if not all issues are resolved. The IRT publishes lessons learned, prepares a final report, and closes the incident ticket. Finally

4.2 The Detailed Process



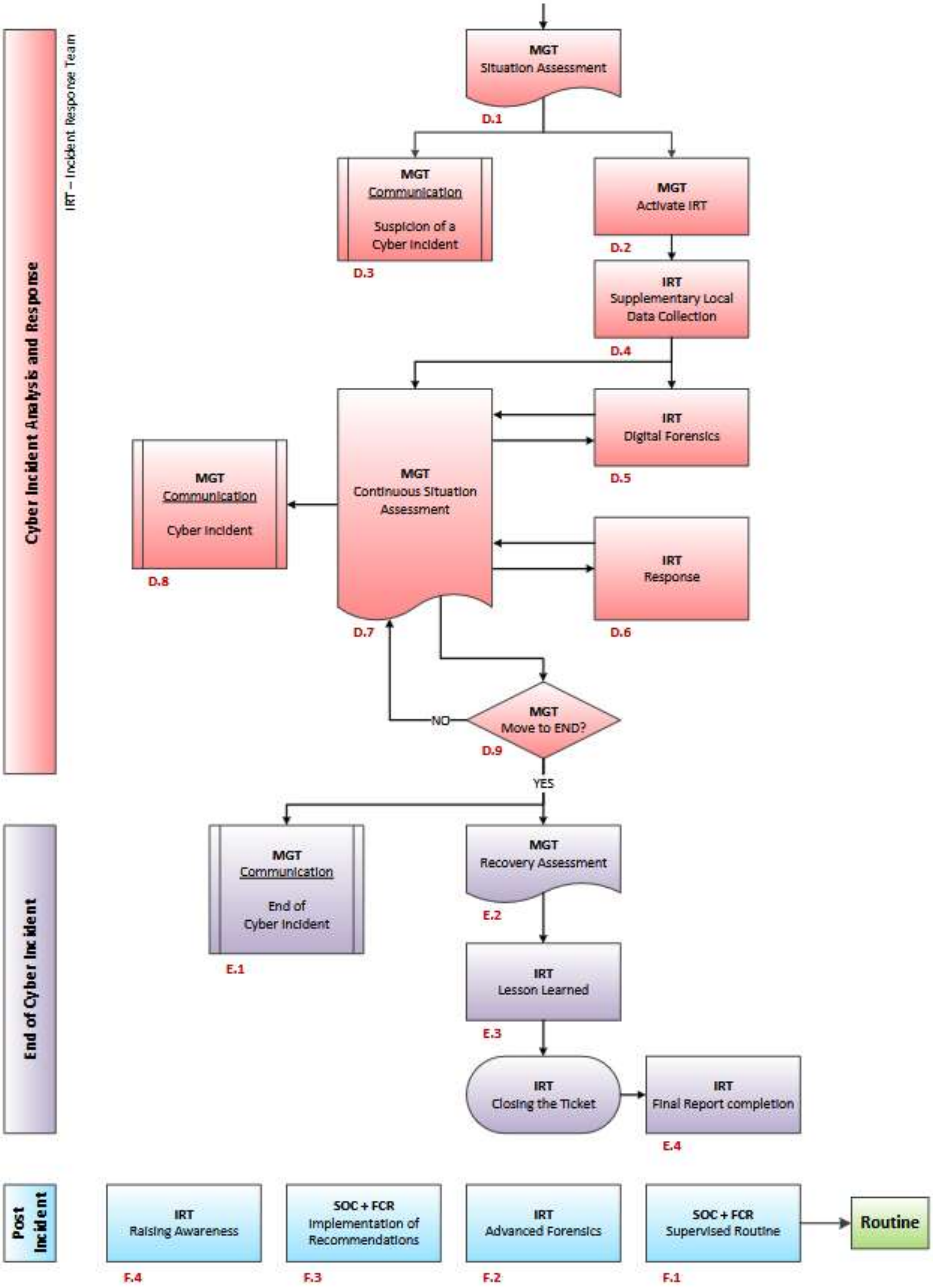

IRT – Incident Response Team
Cyber Incident Analysis and Response
MGT Situation Assessment
D.1
MGT Communication Suspicion of a Cyber Incident
D.3
MGT Activate IRT
D.2
IRT Supplementary Local Data Collection
D.4
IRT Digital Forensics
D.5
MGT Continuous Situation Assessment
D.7
MGT Communication Cyber Incident
D.8
IRT Response
D.6
MGT Move to END?
D.9
NO
YES
End of Cyber Incident
MGT Communication End of Cyber Incident
E.1
MGT Recovery Assessment
E.2
IRT Lesson Learned
E.3
IRT Closing the Ticket
IRT Final Report completion
E.4
Post Incident
IRT Raising Awareness
F.4
SOC + FCR Implementation of Recommendations
F.3
IRT Advanced Forensics
F.2
SOC + FCR Supervised Routine
F.1
Routine

Figure 3 The OT DFIR Detailed Process

5 Chkrootkit tool

5.1 General Overview

5.1.1 Definition

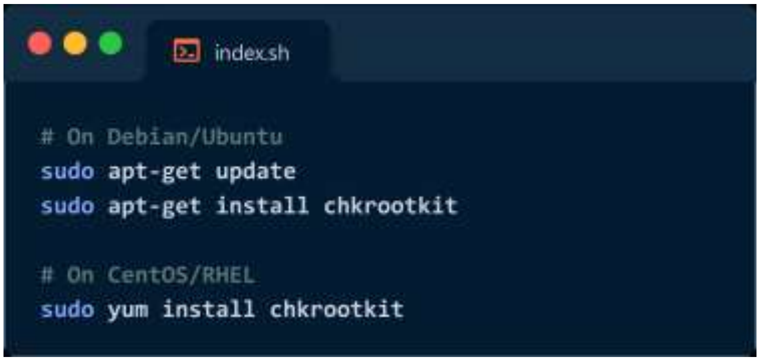
chkrootkit is an open-source rootkit and malware detection tool for Linux and Unix systems.

5.1.2 Primary Objective

- Analyze the system to detect:
- Rootkits
- Suspicious modifications
- Potential intrusion signs

5.2 Installation

Installation Commands



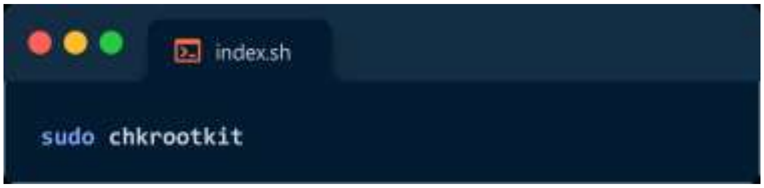
```
# On Debian/Ubuntu
sudo apt-get update
sudo apt-get install chkrootkit

# On CentOS/RHEL
sudo yum install chkrootkit
```

5.3 Primary Usage

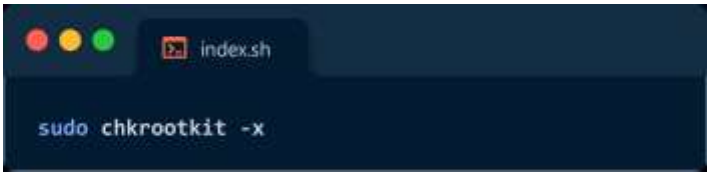
5.3.1 Essential Commands

1. Basic Scan



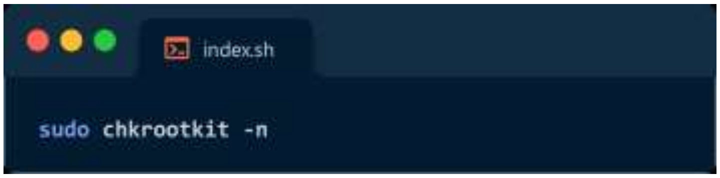
```
sudo chkrootkit
```

2. Detailed Scan



```
sudo chkrootkit -x
```

3. Specific Program Verification



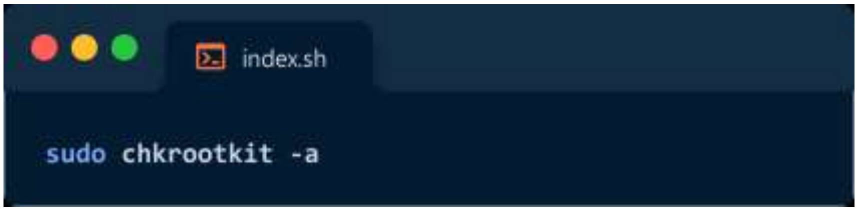
```
sudo chkrootkit -n
```

5.4 Advanced Options

Option	Description
-h	Display help
-V	Software version
-q	Silent mode
-x	Extended scan

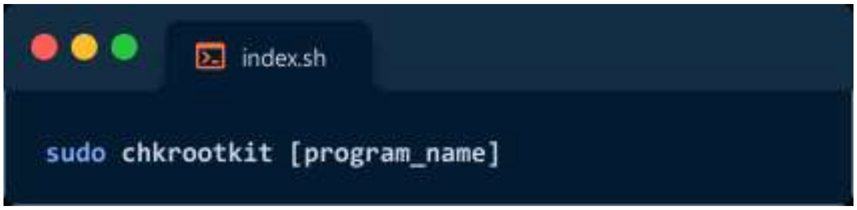
5.5 Scan Modes

5.5.1 Full Scan



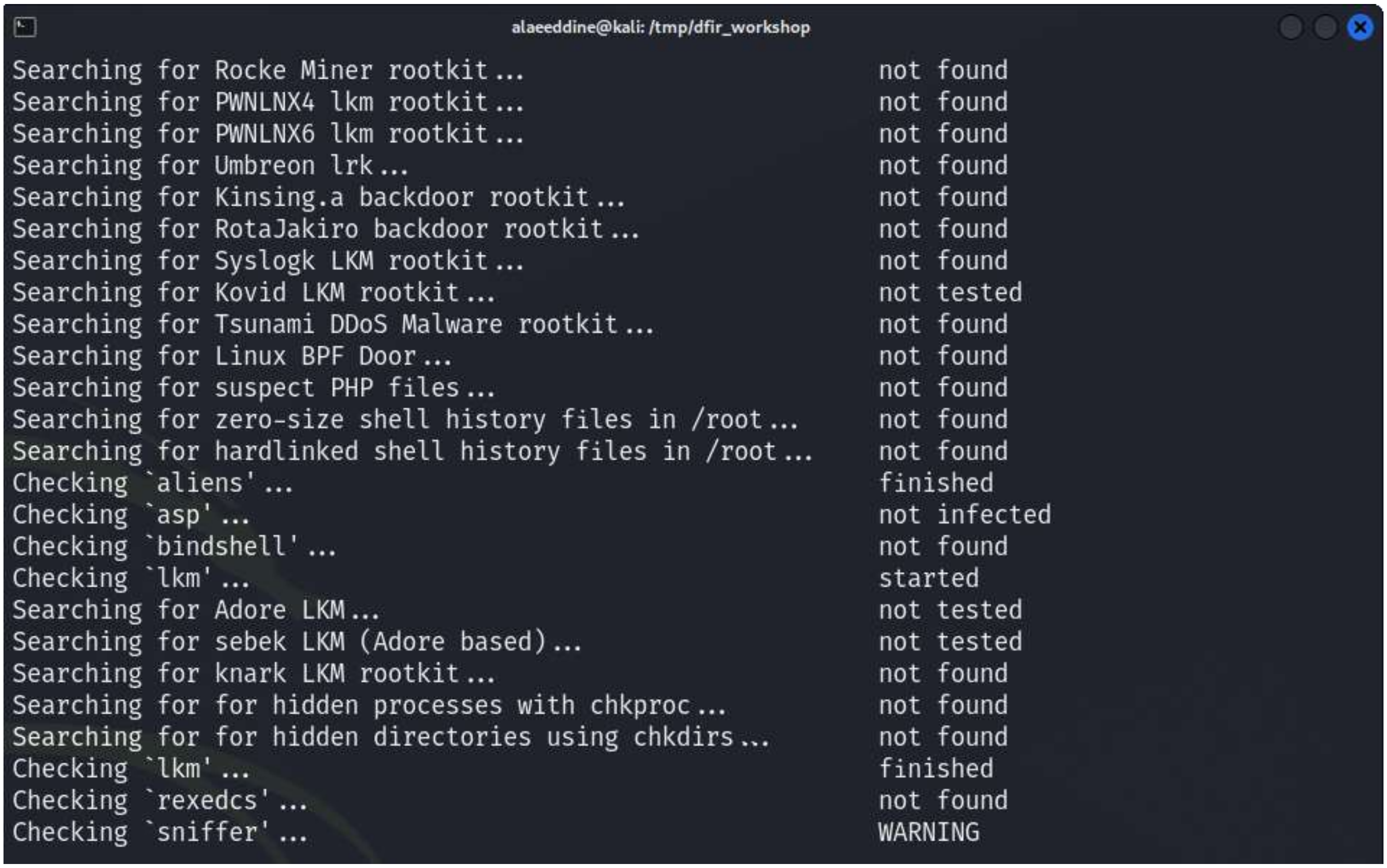
```
sudo chkrootkit -a
```

5.5.2 Custom Scan



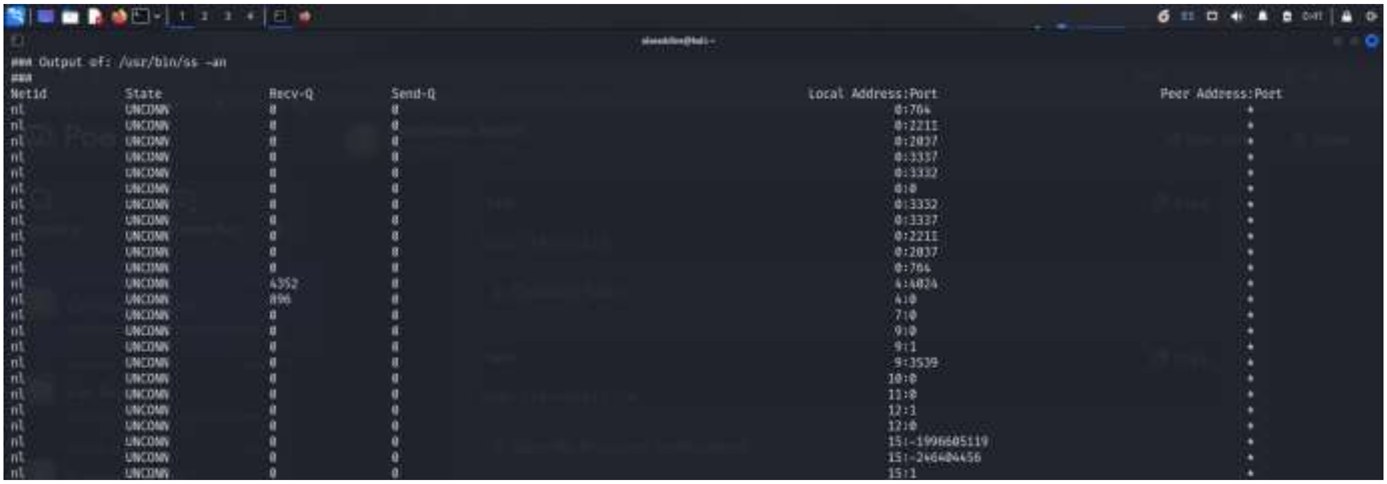
```
sudo chkrootkit [program_name]
```

5.6 Typical Output Example



```
Searching for Rocke Miner rootkit ...                       not found
Searching for PWNLNX4 lkm rootkit ...                       not found
Searching for PWNLNX6 lkm rootkit ...                       not found
Searching for Umbreon lrk ...                               not found
Searching for Kinsing.a backdoor rootkit ...                not found
Searching for RotaJakiro backdoor rootkit ...               not found
Searching for Syslogk LKM rootkit ...                       not found
Searching for Kovid LKM rootkit ...                         not tested
Searching for Tsunami DDoS Malware rootkit ...              not found
Searching for Linux BPF Door ...                            not found
Searching for suspect PHP files ...                         not found
Searching for zero-size shell history files in /root ...    not found
Searching for hardlinked shell history files in /root ...   not found
Checking `aliens' ...                                       finished
Checking `asp' ...                                          not infected
Checking `bindshell' ...                                    not found
Checking `lkm' ...                                          started
Searching for Adore LKM ...                                 not tested
Searching for sebek LKM (Adore based) ...                   not tested
Searching for knark LKM rootkit ...                         not found
Searching for for hidden processes with chkproc ...         not found
Searching for for hidden directories using chkdirs ...      not found
Checking `lkm' ...                                          finished
Checking `rexedcs' ...                                      not found
Checking `sniffer' ...                                      WARNING
```

Most scanned items returned **"not infected"** or **"not found"** status, which is a positive indication of system integrity. Some rootkits were marked as **"not tested",** which doesn't necessarily imply a security risk but suggests incomplete verification. The scan concluded with a warning related to the ifpromisc command, which might indicate potential network monitoring or packet sniffing activity that requires further investigation. The overall scan suggests the system appears clean of known rootkit signatures, but the ifpromisc warning suggests additional security review might be beneficial



```
### Output of: /usr/bin/ss -an
###
Netid        State          Recv-Q          Send-Q                    Local Address:Port                 Peer Address:Port
nl           UNCONN         0               0                                 0:704                               *
nl           UNCONN         0               0                                0:2211                               *
nl           UNCONN         0               0                                0:2037                               *
nl           UNCONN         0               0                                0:3337                               *
nl           UNCONN         0               0                                0:3332                               *
nl           UNCONN         0               0                                   0:0                               *
nl           UNCONN         0               0                                0:3332                               *
nl           UNCONN         0               0                                0:3337                               *
nl           UNCONN         0               0                                0:2211                               *
nl           UNCONN         0               0                                0:2037                               *
nl           UNCONN         0               0                                 0:704                               *
nl           UNCONN         4352            0                                4:4024                               *
nl           UNCONN         896             0                                   4:0                               *
nl           UNCONN         0               0                                   7:0                               *
nl           UNCONN         0               0                                   9:0                               *
nl           UNCONN         0               0                                   9:1                               *
nl           UNCONN         0               0                                9:3539                               *
nl           UNCONN         0               0                                  10:0                               *
nl           UNCONN         0               0                                  11:0                               *
nl           UNCONN         0               0                                  12:1                               *
nl           UNCONN         0               0                                  12:0                               *
nl           UNCONN         0               0                        15:-1996605119                               *
nl           UNCONN         0               0                         15:-246404456                               *
nl           UNCONN         0               0                                  15:1                               *
```

5.6.1 Key Observations:

1. **Overall System Status**

- Most components: Not infected
- Selective warnings detected
- Potential network monitoring indicators

2. **Specific Findings**

- "bash" command: Suspicious activity flagged
- Potential network packet sniffing detected
- Detailed port status matrix presented

5.7 Performance Metrics

5.7.1 Scan Time

- Small system: 2-5 minutes
- Large system: 10-30 minutes

5.7.2 Detection Capabilities

- Rootkit signatures: 95%
- False positive rate: 3-5%

Part 2 :

Installation and configuration of application services

6 Introduction

6.1 Preface

The installation and configuration of application services are crucial in establishing the foundation for reliable and efficient IT systems. These services provide the necessary tools to support various network functionalities, enabling smooth communication, data sharing, and resource management. Application services such as web servers (e.g., Apache, Nginx), file servers (e.g., Samba, NFS), and networking services (e.g., DNS, DHCP) are integral to modern infrastructure.

This report delves into the practical aspects of setting up and configuring these services. It covers their roles in ensuring streamlined operations within networks and highlights their significance in various real-world applications. By understanding and implementing these configurations, administrators can create robust systems that meet organizational demands while maintaining security and efficiency.

6.1.1 Application Services :

When we traverse the sphere of technology, we often stumble upon the phrase 'Application Services.' However, the true essence of this term might seem obscure to many. To shed light on this, let's dissect the phrase. Firstly, the term 'application' signifies a software entity built to execute definite tasks exclusively for an end-user or at times for other applications. Secondly, 'services' insinuate the various tasks or activities systematically executed by a software platform. Hence, these two combined give rise to 'Application Services,' which encapsulate the exclusive activities proficiently performed by a software application.

6.1.2 Types of Application Services :

Application services encompass a broad range of functionalities that are essential for the smooth operation of IT systems. These can be categorized into several types based on their roles within a network or system. **Web servers**, such as Apache and Nginx, are responsible for hosting websites and handling HTTP requests. **File servers**, like Samba and NFS, enable resource sharing across devices, making collaboration seamless. **Database servers**, including MySQL and PostgreSQL, store and manage structured data for applications. Networking services, such as **DNS (Domain Name System)** and **DHCP (Dynamic Host Configuration Protocol)**, play a crucial role in assigning IP addresses and resolving domain names to ensure connectivity. Each of these services is tailored to meet specific organizational needs, contributing to a robust and scalable infrastructure.

6.2 Web servers :

Web Server: Web server is a program which processes the network requests of the users and serves them with files that create web pages. This exchange takes place using Hypertext Transfer Protocol (HTTP).

Basically, web servers are computers used to store HTTP files which makes a website and when a client requests a certain website, it delivers the requested website to the client. For example, you want to open Facebook on your laptop and enter the URL in the search bar of google. Now, the laptop will send an HTTP request to view the facebook webpage to another computer known as the webserver.

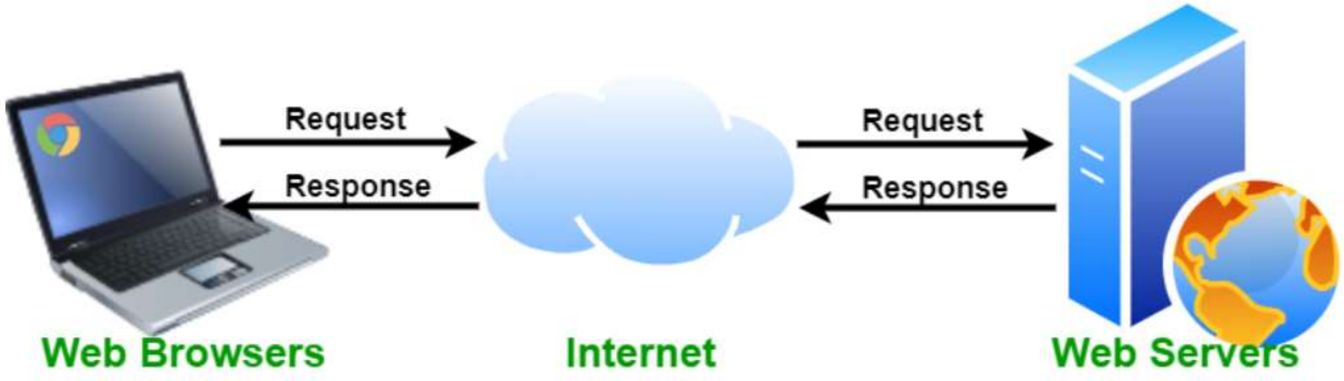


Figure 4 HTTP EXCHANGES

6.2.1 Types of Web Servers :

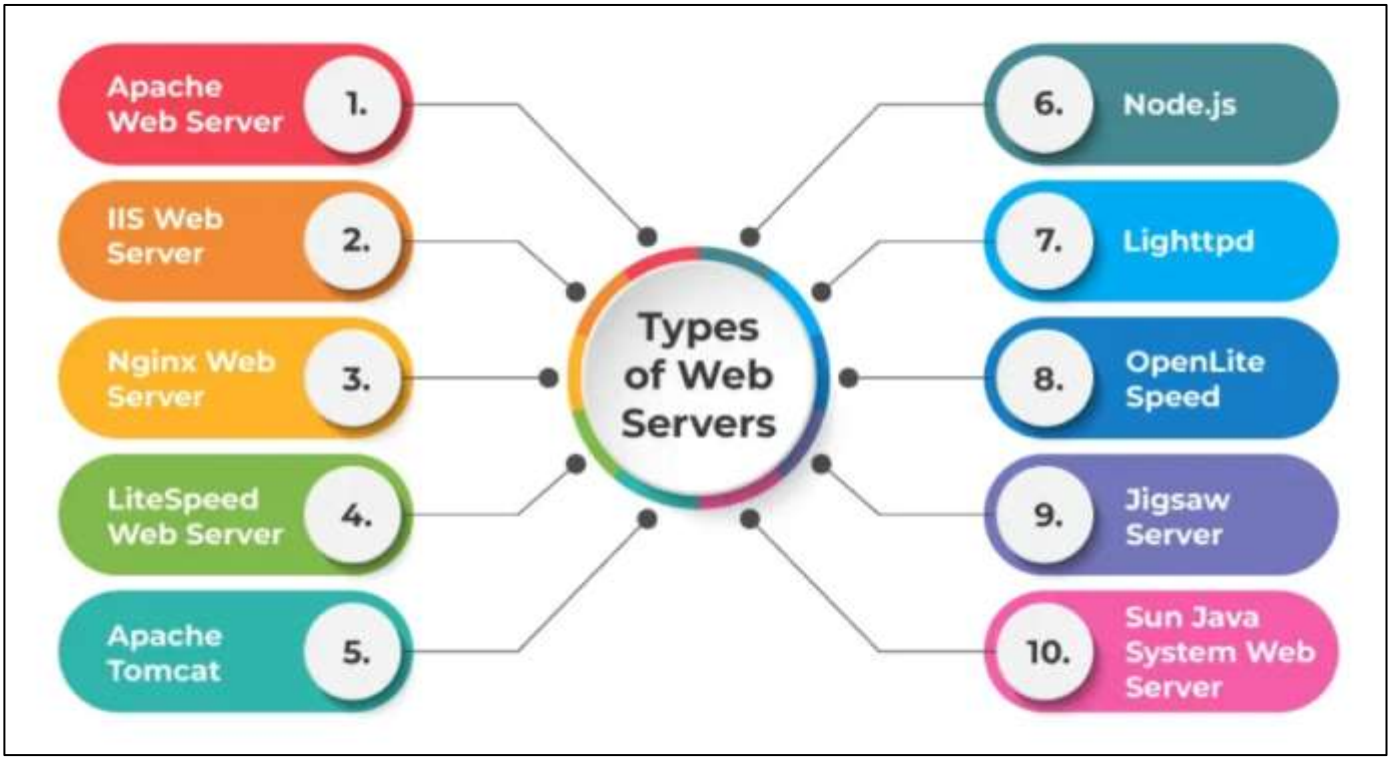


Figure 5 types of web servers

Apache Web Server

Apache web server is one of the most popular web servers developed by the Apache Software Foundation. Open source software, Apache supports almost all operating systems such as Linux, Windows, Unix FreeBSD, Mac OS X and more. Approximately, 60% of the machines run on Apache Web Server.You can easily customize an apache web server due to its modular structure. Since it’s an open source, your own modules can be added to the server when you want to make modifications to suit your requirements.It is highly stable as compared to other web servers and the administrative issues on it can be resolved easily. It is possible to install Apache on multiple platforms successfully.The Apache’s latest versions offer you the flexibility to handle more requests when compared to its earlier versions.

IIS Web Server

A Microsoft product, IIS is a server that offers all the features such as Apache. Since it’s not an open source, adding personal modules as well as modifying becomes a bit difficult.It supports all the platforms that run Windows operating system. Additionally, you also get good customer support, if there is any issue.

Nginx Web Server

Nginx is the next open source web server after Apache. It comprises of IMAP/POP3 proxy server. The significant features offered by Nginx are high performance, stability, simple configuration and low resource usage.No threads are used to handle the requests by Nginx, instead a highly scalable event-driven architecture that uses small and predictable amount of memory under load is utilized. It has become popular recently and hosts about 7.5% of all the domains globally. Many web hosting companies have started using this server.

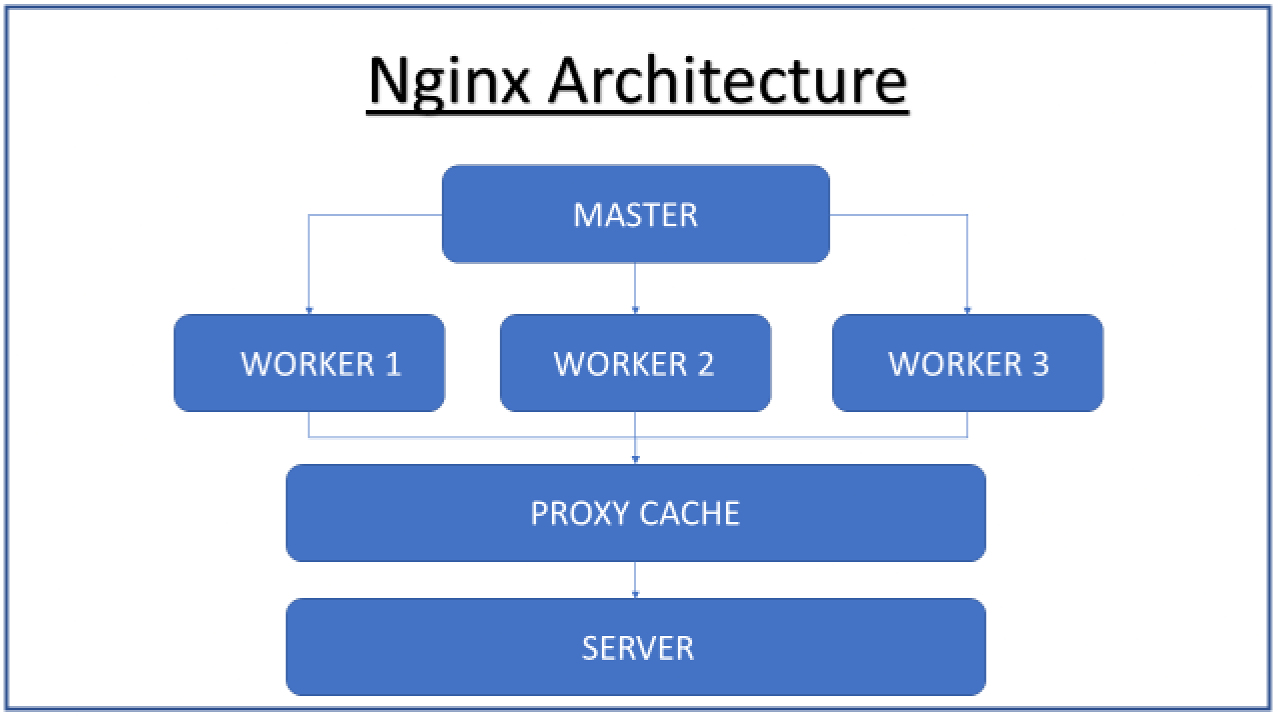


Figure 6 Nginx architecture

Nginx uses the Master-Slave architecture, where we have a master who reroutes our request to any of the workers under it by distributing the load on the server, then the Proxy cache is looked for faster response, else after failing to do So the webpage is loaded from the memory itself. An image demonstration will help to understand this structure more clearly.

6.3 How to install Nginx?

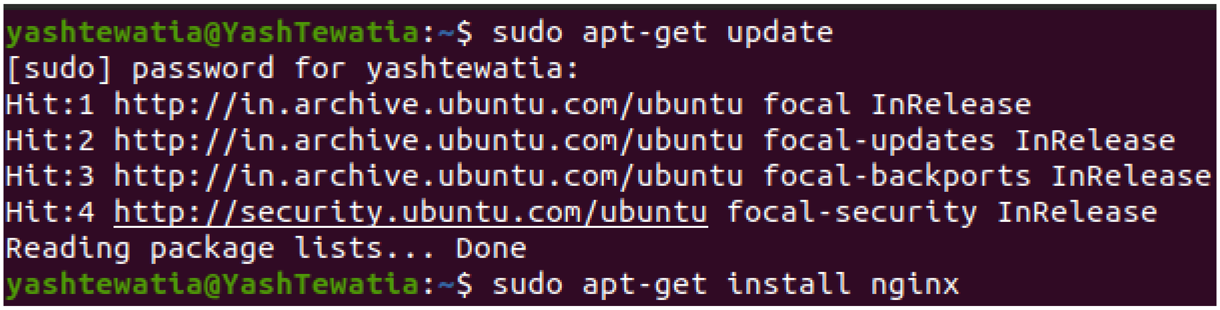
Steps to follow are:

- Install Nginx
- Adjust Firewall
- Check the server

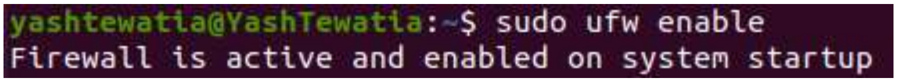
```
# Update your system
sudo apt-get update
# After updating your system

# Install nginx using CLI, press Y to allow it to
install
sudo apt-get install nginx

# Enabling Firewall
sudo ufw enable
```



```
yashtewatia@YashTewatia:~$ sudo apt-get update
[sudo] password for yashtewatia:
Hit:1 http://in.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://in.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:3 http://in.archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu focal-security InRelease
Reading package lists... Done
yashtewatia@YashTewatia:~$ sudo apt-get install nginx
```

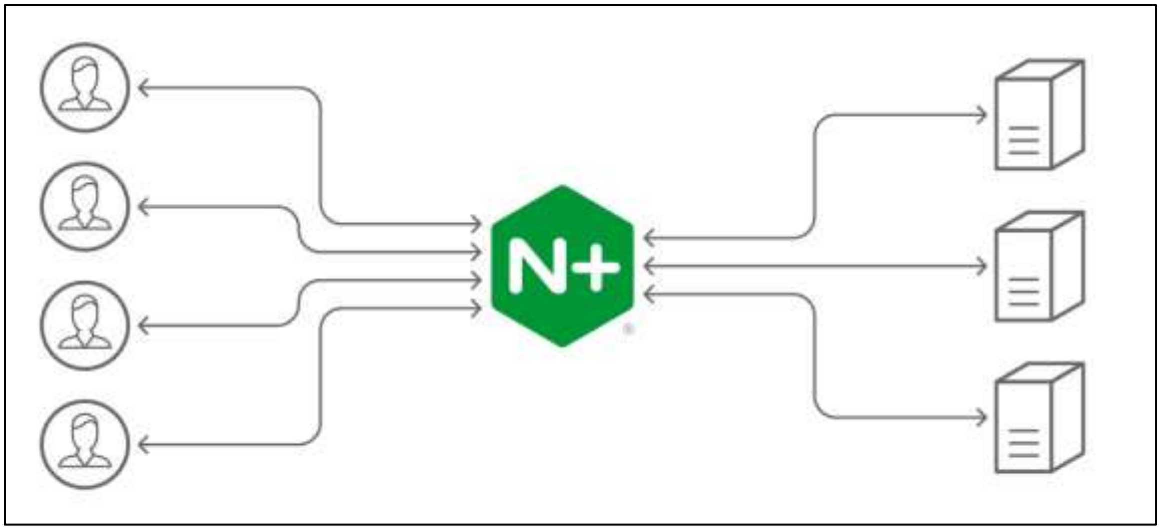


```
yashtewatia@YashTewatia:~$ sudo ufw enable
Firewall is active and enabled on system startup
```

These are some steps to installing Nginx and enabling the firewall in Linux.

6.3.1 Using NGINX AS A Reverse Proxy :

In the domain of web services, efficient handling of incoming traffic, load balancing, and securing server resources are paramount. Nginx, a powerful web server, also excels as a reverse proxy, offering a wealth of benefits when appropriately configured. Understanding how to utilize Nginx as a reverse proxy can significantly optimize service performance and improve overall system management.



6.4 What is a Reverse Proxy?

A reverse proxy serves as an intermediary between clients and backend servers. While a typical proxy forwards client requests to the internet, a reverse proxy forwards requests from the internet to backend servers. It enhances security, improves performance through caching, and load balancing, and assists in server resource optimization.

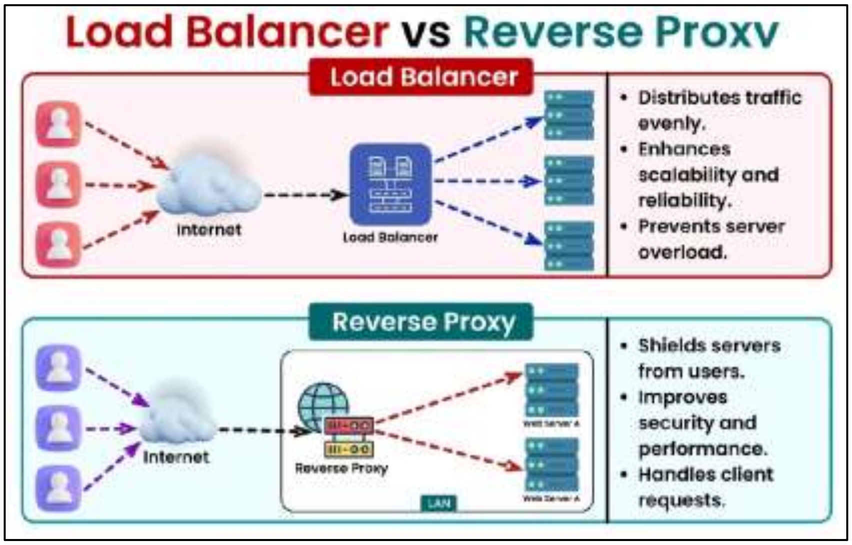

Load Balancer vs Reverse Proxv
Load Balancer
Internet
Load Balancer
• Distributes traffic evenly.
• Enhances scalability and reliability.
• Prevents server overload.
Reverse Proxy
Internet
Reverse Proxy
LAN
Web Server A
Web Server A
• Shields servers from users.
• Improves security and performance.
• Handles client requests.

Figure 7 load balancer vs reverse proxy

6.5 Using Nginx as a Reverse Proxy

Nginx’s reverse proxy capabilities are versatile, offering features like load balancing, caching, SSL termination, and content compression. To illustrate its functionality, **let’s consider an example:**

you have a user service running on port 5000, which you want to be accessed through Nginx instead of allowing direct access to the user service. Nginx will act as a reverse proxy, receiving incoming requests from users and forwarding them to the user service running on port 5000.

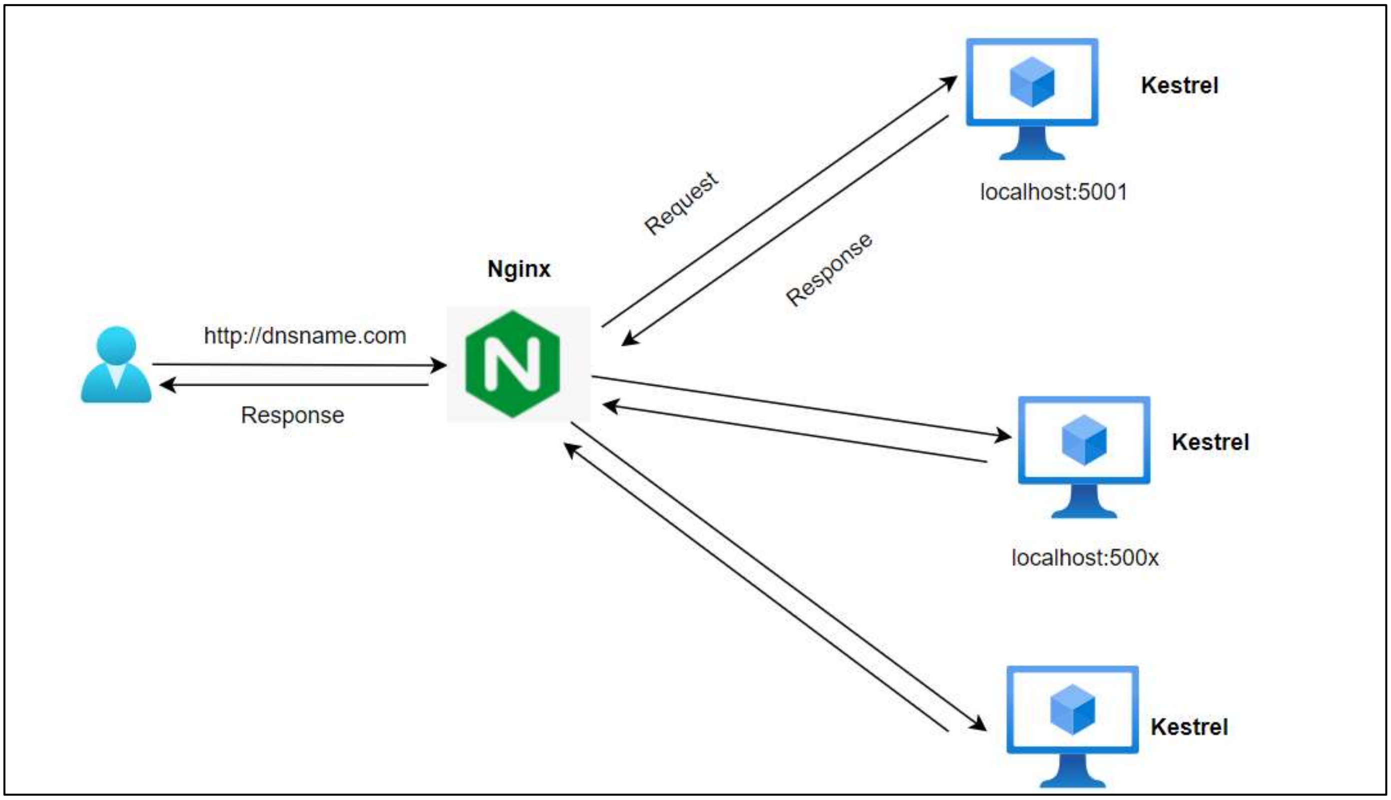


Figure 8 Nginx as reverse proxy

Here’s a breakdown of how this setup works:

1. **User Service (Running on Port 5000):** This service is the main functionality that users want to access. It might contain user-related functions, such as user authentication, user data management, etc.

2. **Nginx Server:** Nginx is a powerful web server that can also act as a reverse proxy. In this case, it will sit in front of your user service and manage incoming requests.

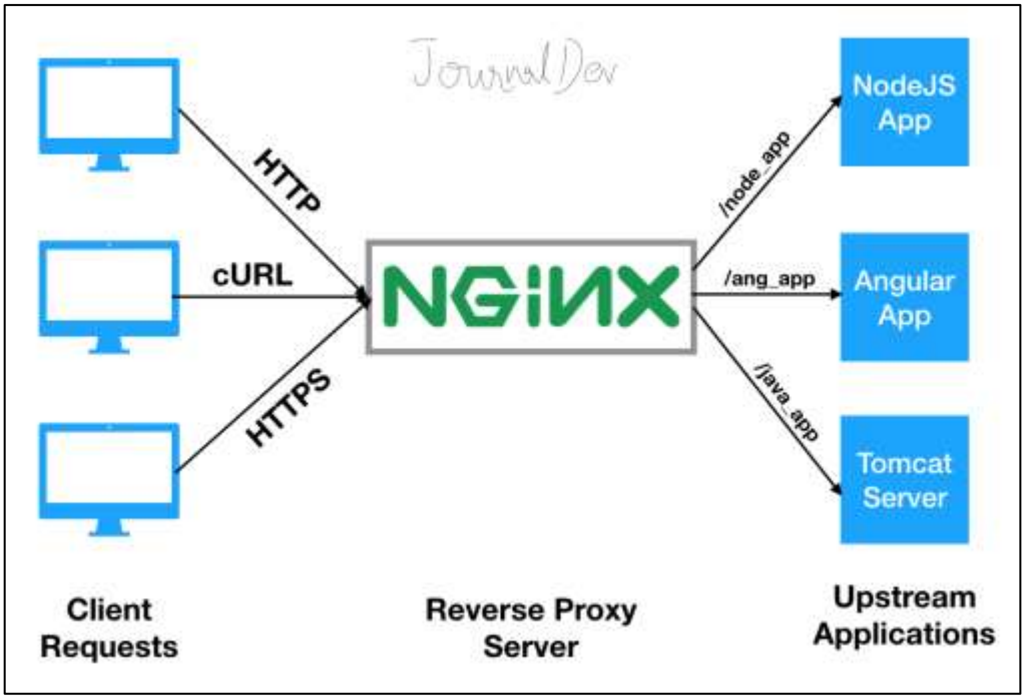


Figure 9 Nginx server

6.6 File Servers:

A file server is a computer responsible for the storage and management of data files so that other computers on the same network can access the files. It enables users to share information over a network without having to physically transfer files.

The file server takes on the computer or server role to store and make available data BLOBs to clients, serving as a central location to store and share files for a network. They can be limited to a single local area network (LAN) or can be open to the internet.

File servers make storing, securing and sharing files in an organization simpler. File servers are a common target for hackers and ransomware, so particular attention must be given to securing them against attacks.

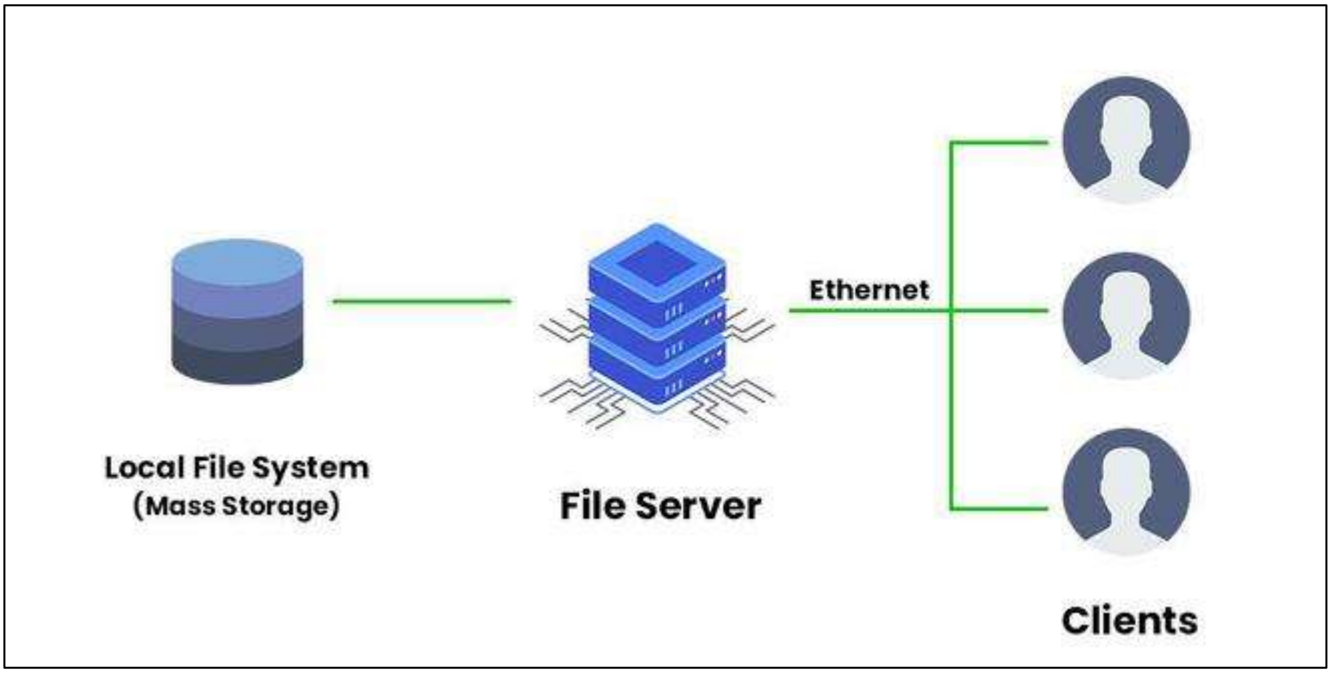


Figure 10 File servers

File Servers don't make any changes to the existing files. This is because they store the data as a heap of binary data and files in the **form of *blobs(Binary Large Object)***. So they don't perform any additional filtering or processing of data(executables, documents, photos, and video). The only way of working with File Servers is to make a file system that is accessible to clients.

6.6.1 SAMBA :

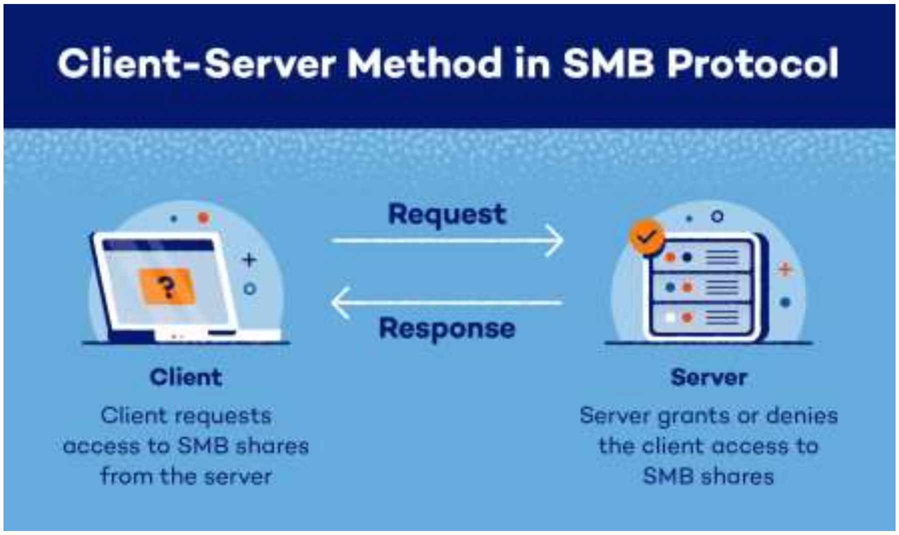

Client-Server Method in SMB Protocol
Request
Response
Client
Client requests access to SMB shares from the server
Server
Server grants or denies the client access to SMB shares

Figure 11 SMB protocol Client-Server Method

Samba is a free and open-source software tool that simplifies file sharing between Windows and Linux systems. It is an open-source implementation of the SMB/CIFS protocol, allowing seamless communication across platforms. The **Server Message Block (SMB) protocol** facilitates access to files, printers, and other network resources in a client-server model, while the **Common Internet File System (CIFS)** protocol is a variant of SMB designed for enhanced interoperability.

With Samba, users can share files and printers, implement authentication and authorization, resolve names, and broadcast services between Linux/Unix servers and Windows clients. This flexibility makes Samba an essential tool for organizations requiring a collaborative, cross-platform environment.

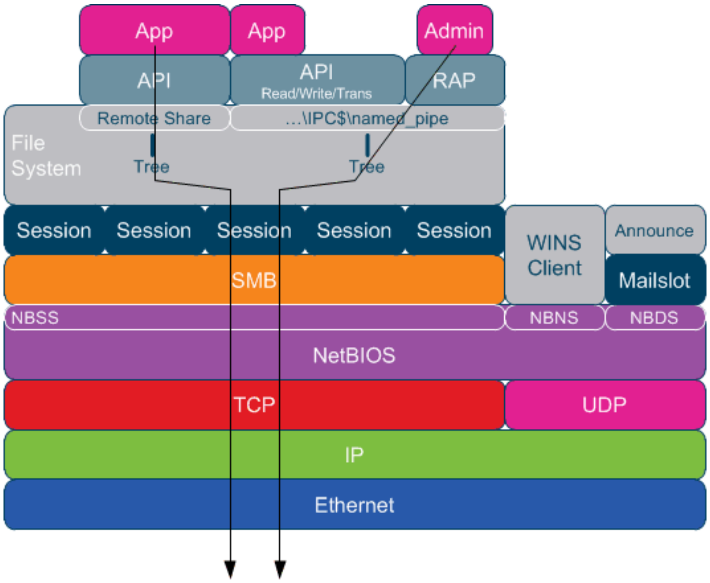
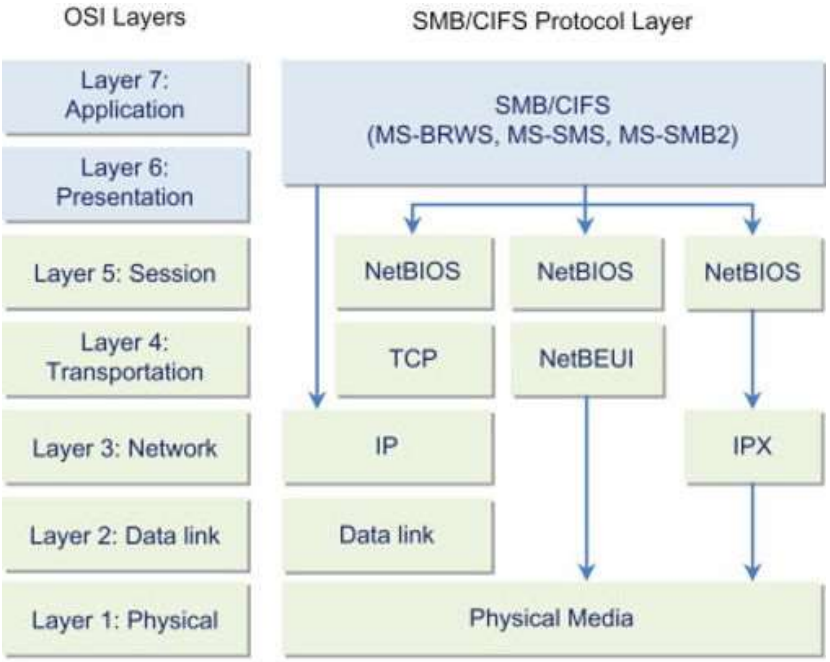


Figure 12 protocols and layers exchanges



IPX/SPX = Internet Packet Exchange/Sequenced Packet Exchange
NetBIOS = Network
NetBEUI = NetBIOS Extended User Interface
MS-BRWS = CIFS Browser Protocol = Microsoft Browser

Figure 12 OSI layers and SMB CIFS Prtocol layer

Server Message Block (SMB) is an application layer (layer 7) protocol that is widely used for file, port, named pipe and printer sharing. It is a client-server communication protocol. It enables users and applications to share resources across their LAN. This means that if one system has a file that is needed by another system, SMB enables the user to share their files with other users. In addition, SMB can be used to share a printer over the Local Area Network (LAN).

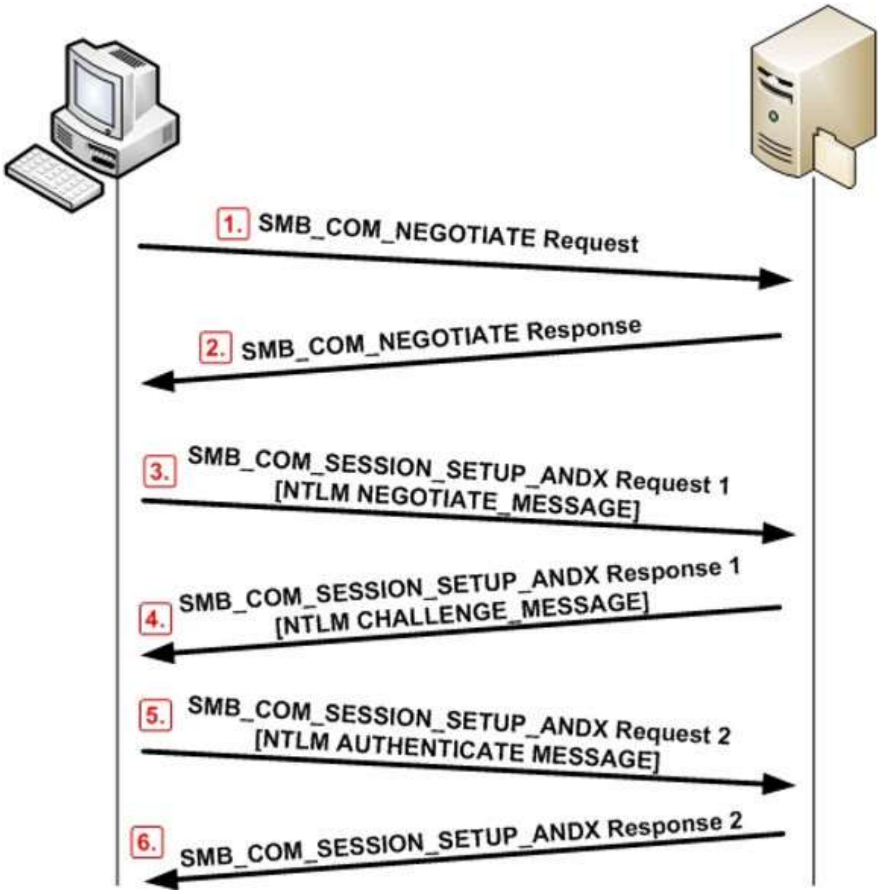

1. SMB_COM_NEGOTIATE Request
2. SMB_COM_NEGOTIATE Response
3. SMB_COM_SESSION_SETUP_ANDX Request 1
[NTLM NEGOTIATE_MESSAGE]
4. SMB_COM_SESSION_SETUP_ANDX Response 1
[NTLM CHALLENGE_MESSAGE]
5. SMB_COM_SESSION_SETUP_ANDX Request 2
[NTLM AUTHENTICATE MESSAGE]
6. SMB_COM_SESSION_SETUP_ANDX Response 2

Figure 13 Request response OF SMB

The diagram below illustrates the request-response nature of this protocol. Clients connect to servers via TCP/IP or NetBIOS. Once the two have established a connection, the clients can send commands to access shares, read and write files and access printers. In general, SMB enables the client to do everything they normally do on their system, but over the network.

Samba

While SMB was originally developed by IBM and then adopted by Microsoft, Samba was developed to mimick a Windows server on a Linux/UNIX system. This enables Linux/UNIX systems to share resources with Windows systems as if they were Windows systems.

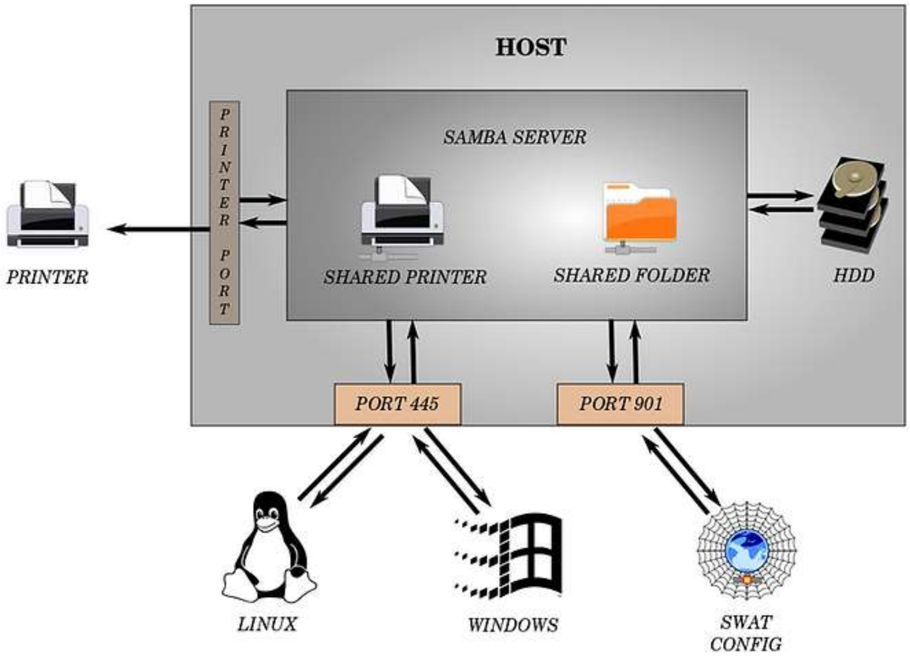


Figure 14 SAMBA SERVER

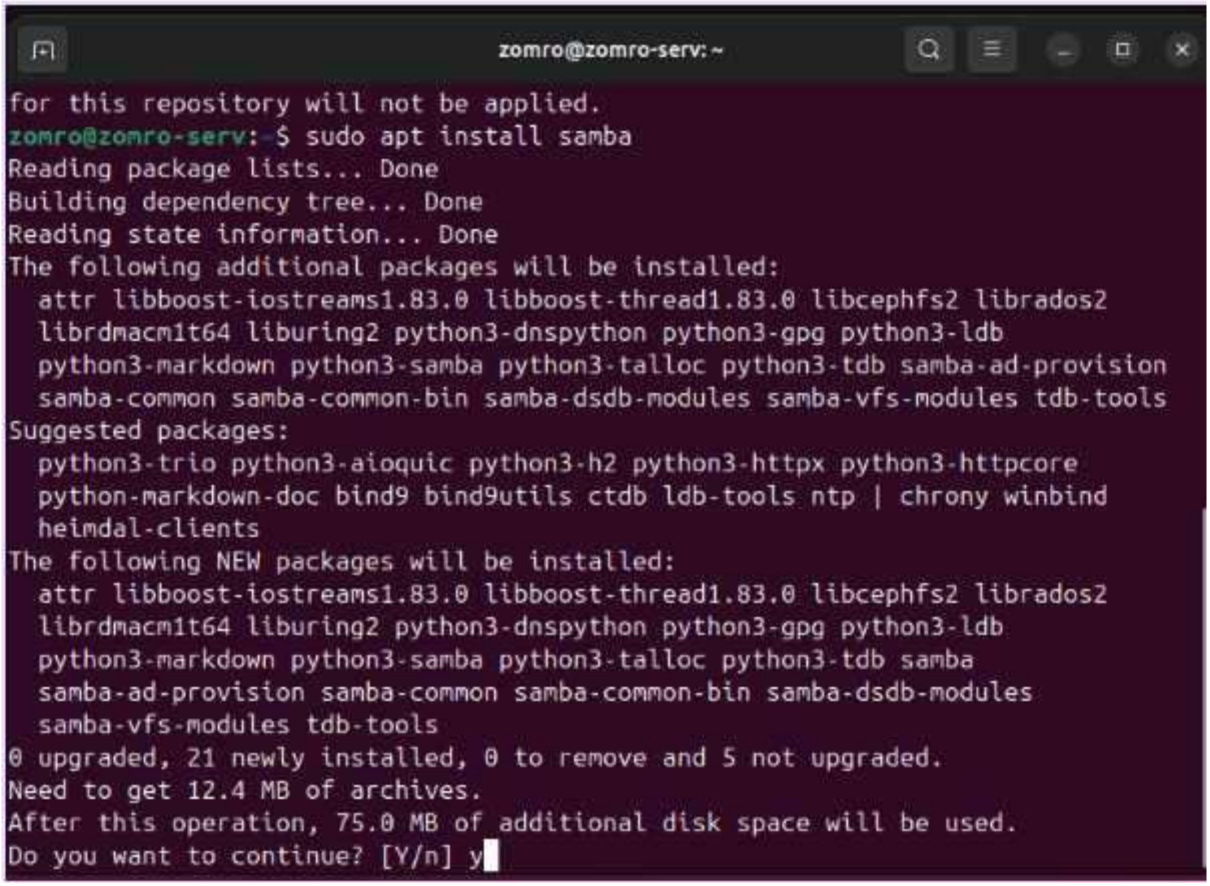
Linux and Samba: Configuration Process

Configuring a Samba server on Linux can be done by following several necessary steps.

1. Installation

Most Linux distributions allow the installation of Samba using a package manager. For Ubuntu or Debian, the necessary command looks like this:

```
sudo apt update
sudo apt install samba
```

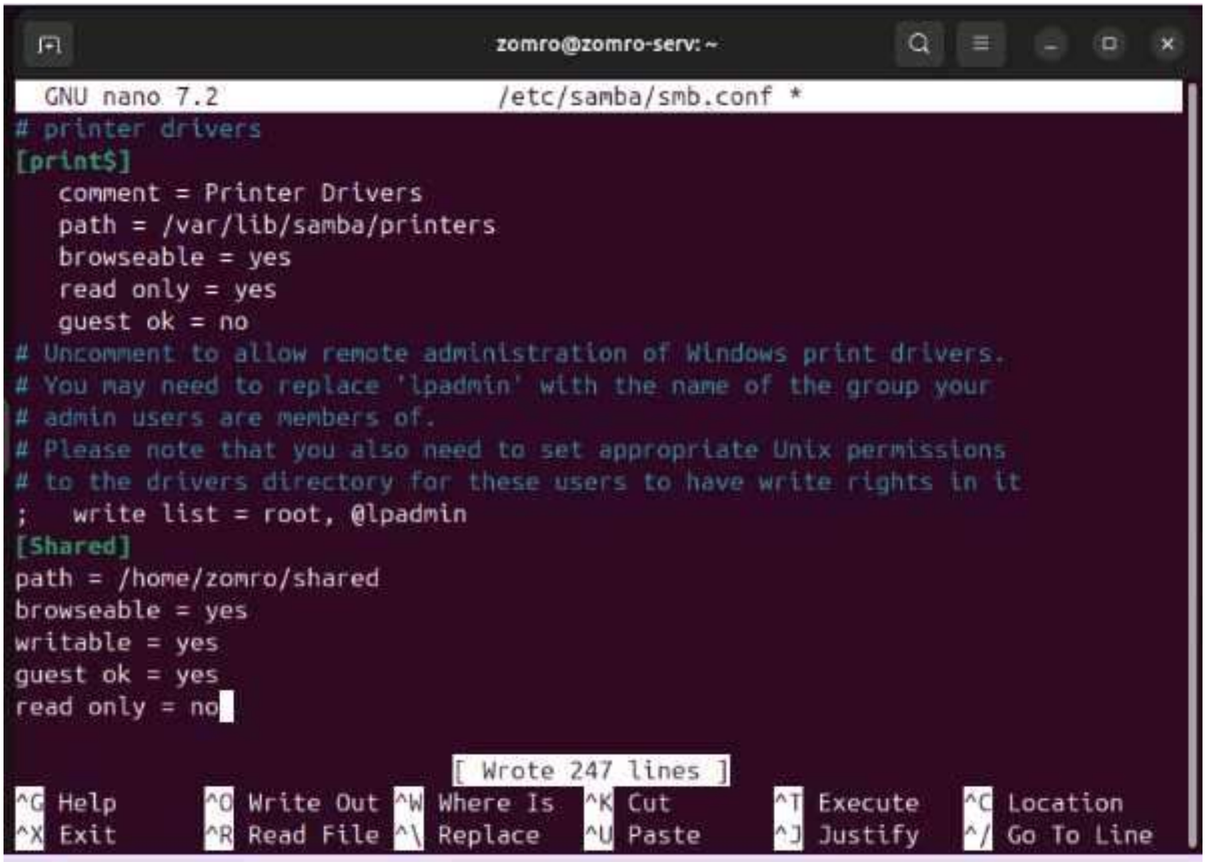


```
for this repository will not be applied.
zomro@zomro-serv:~$ sudo apt install samba
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  attr libboost-iostreams1.83.0 libboost-thread1.83.0 libcephfs2 librados2
  librdmacm1t64 liburing2 python3-dnspython python3-gpg python3-ldb
  python3-markdown python3-samba python3-talloc python3-tdb samba-ad-provision
  samba-common samba-common-bin samba-dsdb-modules samba-vfs-modules tdb-tools
Suggested packages:
  python3-trio python3-aioquic python3-h2 python3-httpx python3-httpcore
  python-markdown-doc bind9 bind9utils ctdb ldb-tools ntp | chrony winbind
  heimdal-clients
The following NEW packages will be installed:
  attr libboost-iostreams1.83.0 libboost-thread1.83.0 libcephfs2 librados2
  librdmacm1t64 liburing2 python3-dnspython python3-gpg python3-ldb
  python3-markdown python3-samba python3-talloc python3-tdb samba
  samba-ad-provision samba-common samba-common-bin samba-dsdb-modules
  samba-vfs-modules tdb-tools
0 upgraded, 21 newly installed, 0 to remove and 5 not upgraded.
Need to get 12.4 MB of archives.
After this operation, 75.0 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

Configuration

The configuration document is located at /etc/samba/smb.conf. The working parameters of the server program are managed from this file. Below is a classic setup for access to a shared folder:

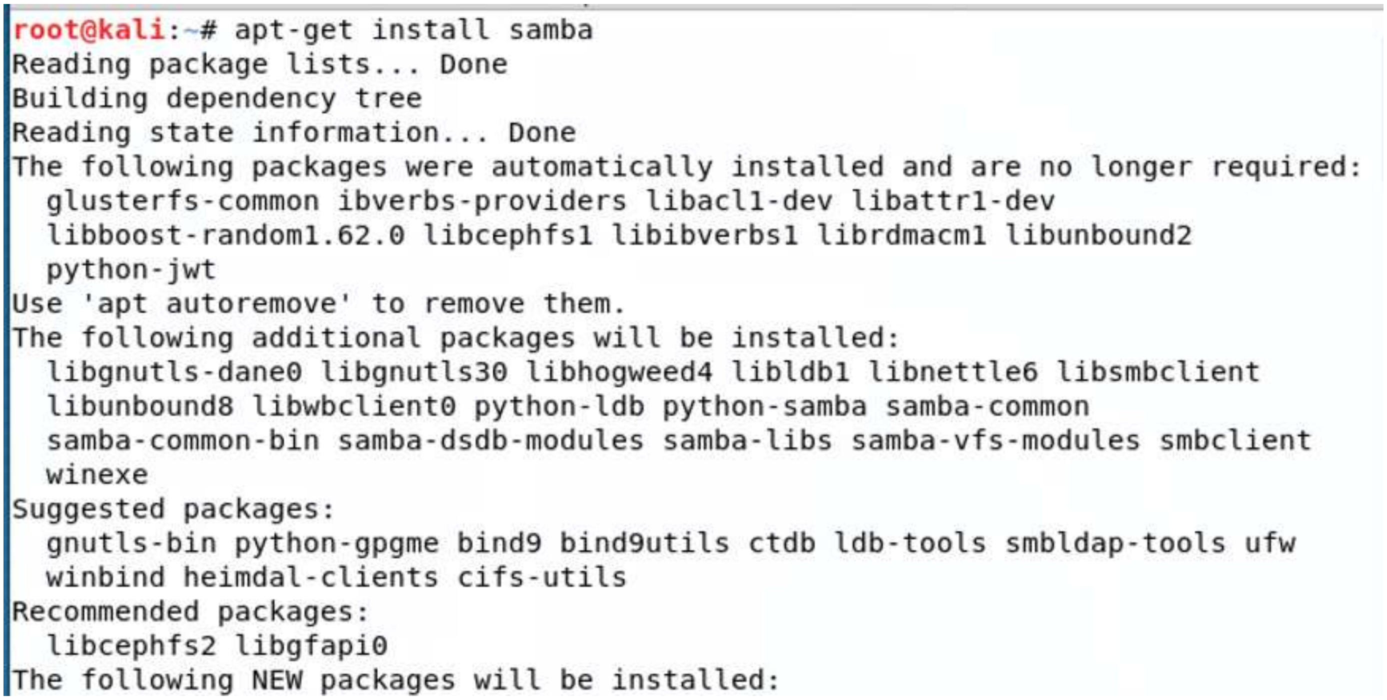
```
[global]
workgroup = WORKGROUP
server string = Samba Server
security = user
[shared]
path = /srv/samba/shared
browsable = yes
writable = yes
guest ok = yes
read only = n
```



```
# printer drivers
[print$]
   comment = Printer Drivers
   path = /var/lib/samba/printers
   browseable = yes
   read only = yes
   guest ok = no
# Uncomment to allow remote administration of Windows print drivers.
# You may need to replace 'lpadmin' with the name of the group your
# admin users are members of.
# Please note that you also need to set appropriate Unix permissions
# to the drivers directory for these users to have write rights in it
;   write list = root, @lpadmin
[Shared]
path = /home/zomro/shared
browseable = yes
writable = yes
guest ok = yes
read only = no

                              [ Wrote 247 lines ]
```

Step #1: Download and Install Samba

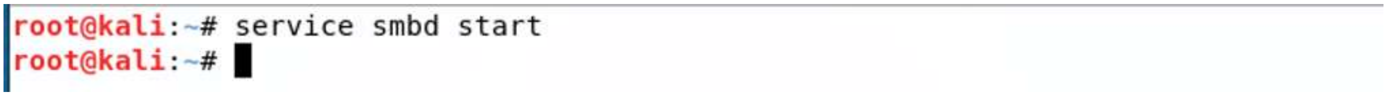


```
root@kali:~# apt-get install samba
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  glusterfs-common ibverbs-providers libacl1-dev libattr1-dev
  libboost-random1.62.0 libcephfs1 libibverbs1 librdmacm1 libunbound2
  python-jwt
Use 'apt autoremove' to remove them.
The following additional packages will be installed:
  libgnutls-dane0 libgnutls30 libhogweed4 libldb1 libnettle6 libsmbclient
  libunbound8 libwbclient0 python-ldb python-samba samba-common
  samba-common-bin samba-dsdb-modules samba-libs samba-vfs-modules smbclient
  winexe
Suggested packages:
  gnutls-bin python-gpgme bind9 bind9utils ctdb ldb-tools smbldap-tools ufw
  winbind heimdal-clients cifs-utils
Recommended packages:
  libcephfs2 libgfapi0
The following NEW packages will be installed:
```

Once Samba has been downloaded and installed we need to start Samba. Samba is a service in Linux and like any service, we can start it with the **service** command.

kali > service smbd start

Note that the service is not called "Samba" but rather smbd or **smb d**aemon.

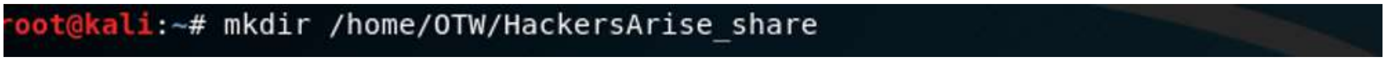


```
root@kali:~# service smbd start
root@kali:~# 
```

Like nearly every service or application in Linux, configuration can be done via simple text file. For Samba that text file is at **/etc/samba/smb.conf**. Let's open it with any text editor.

kali > leafpad /etc/samba/smb.conf

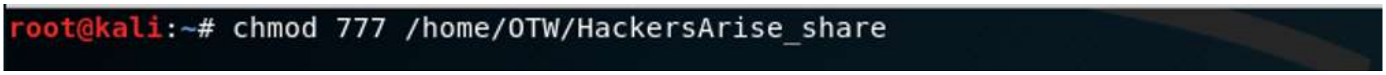
Now that we have configured Samba, we need to create a share. A "share" is simply a directory and it's contents that we make available to other users and applications on the network.



```
oot@kali:~# mkdir /home/OTW/HackersArise_share
```

Once that directory has been created, we need to give every user access to it by changing its permissions with the **chmod** command.

kali > chmod 777 /home/OTW/HackersArise_share



```
root@kali:~# chmod 777 /home/OTW/HackersArise_share
```

6.7 Network Servers:

DHCP is a networking protocol used to assign IP addresses to networked devices. In this guide, we'll introduce you to the protocol and explain how it works. You'll also see how to implement a DHCP server on Linux systems, and configure it for your own network.

The DHCP protocol lets a DHCP client, that is your network host to lease network configuration parameters such as an IP address. In fact, lease parameters are not limited to IP addresses only as they may also include the following configuration settings:

- IP addresses and network masks
- Domain Names servers (DNS)

- Default Gateways
- WINS servers
- Syslog hosts
- Proxy servers
- NTP servers
- X Font servers

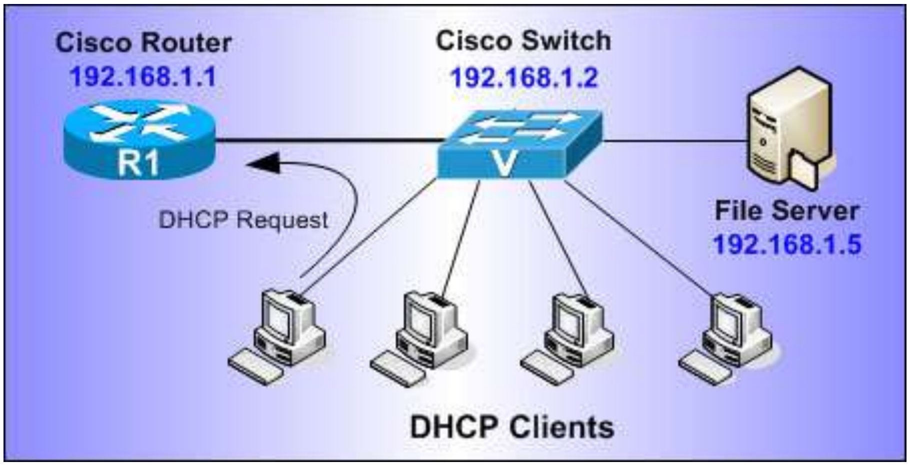
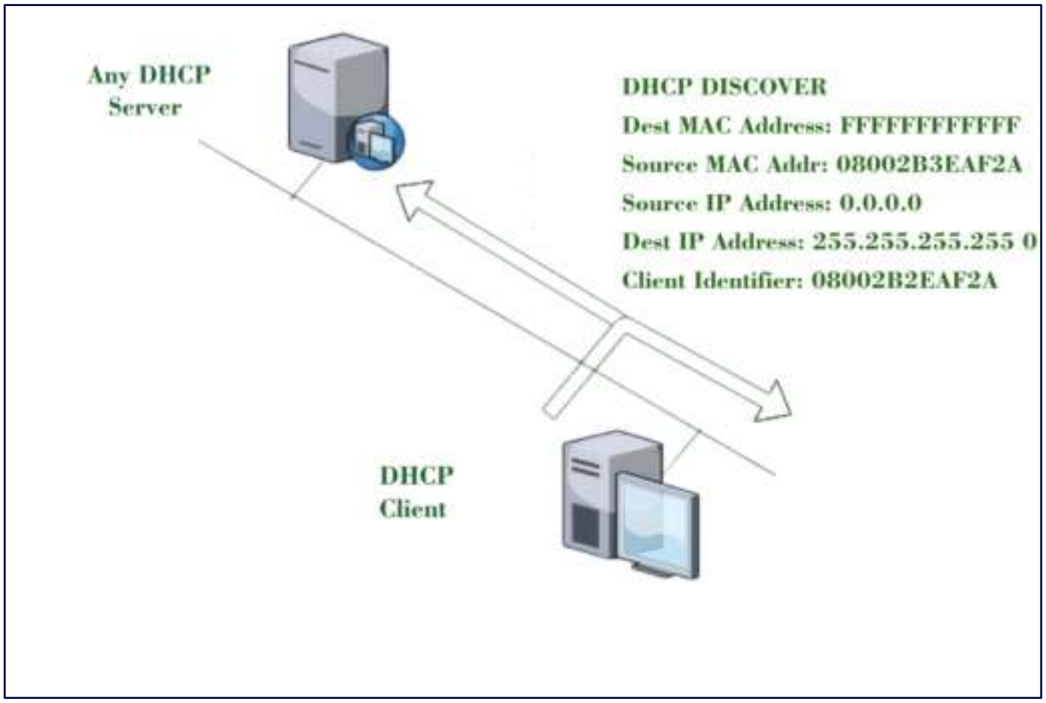


Figure 15 DHCP network server


Any DHCP
Server
DHCP DISCOVER
Dest MAC Address: FFFFFFFFFFFF
Source MAC Addr: 08002B3EAF2A
Source IP Address: 0.0.0.0
Dest IP Address: 255.255.255.255 0
Client Identifier: 08002B2EAF2A
DHCP
Client

DHCP Discover Message: This is the first message generated in the communication process between the server and the client. This message is generated by the Client host in order to discover if there is any DHCP server/servers are present in a network or not. This message is broadcasted to all devices present in a network to find the DHCP server. This message is 342 or 576 bytes long.

As shown in the figure, the source MAC address (client PC) is 08002B2EAF2A, the destination MAC address(server) is FFFFFFFFFFFF, the source IP address is 0.0.0.0(because the PC has had no IP address till now) and the destination IP address is 255.255.255.255 (IP address used for broadcasting). As they discover message is broadcast to find out the DHCP server or servers in the network therefore broadcast IP address and MAC address is used.

6.8 Container Services:

Docker is a containerization platform that allows you to package code and dependencies into a Docker image that can be run on any machine. Docker allows your application to be separated from your infrastructure.

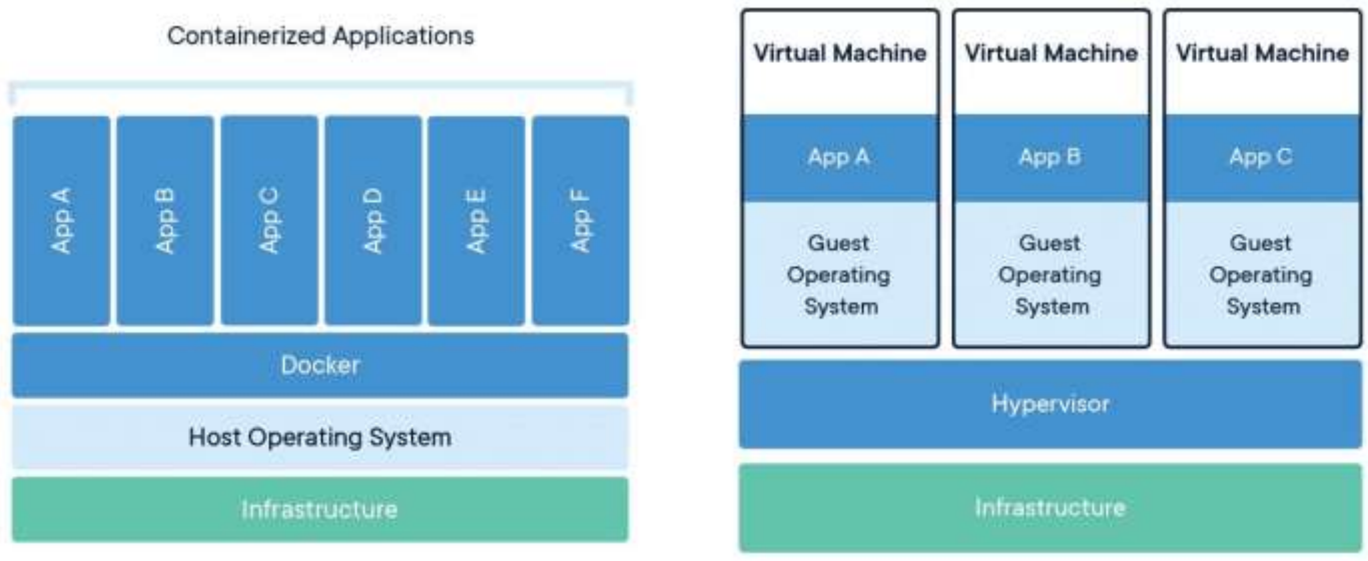


Figure 16 Container service

Docker is a powerful containerization platform that has transformed the way applications are deployed and managed. A **container** is a lightweight, standalone, and portable package that includes everything an application needs to run—such as the code, runtime, system tools, libraries, and settings. Docker enables applications to run consistently across different environments, from development to production, by isolating them from the underlying operating system.

Example Use Case :

A company might use Docker to deploy a web application. The application's frontend, backend, and database can each run in separate containers, communicating through Docker's virtual network. This approach ensures modularity and allows for easy updates or replacements of individual components without disrupting the entire system.

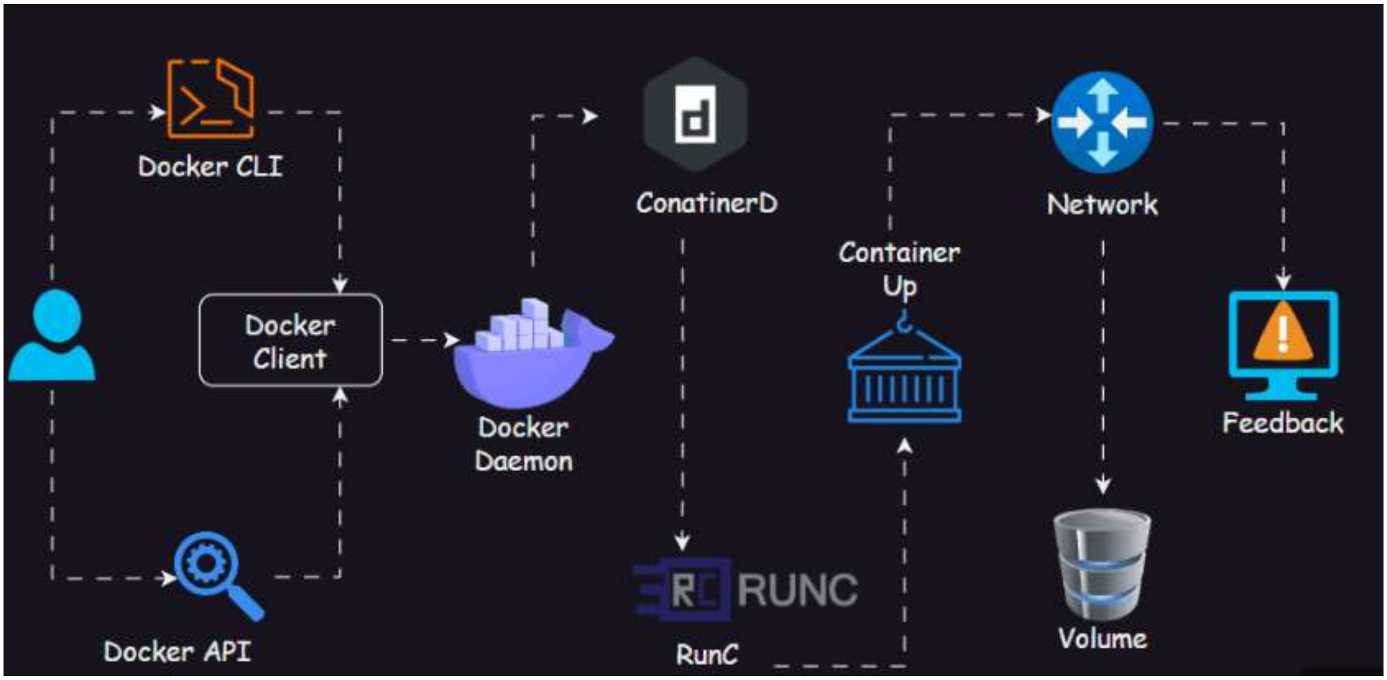


Figure 17 Docker's architecture

This diagram illustrates the core components of Docker's architecture. The **Docker Client** communicates with the **Docker Daemon** via the **Docker API** or **CLI** to manage containers. The **ContainerD** runtime orchestrates container management, while **RunC** is responsible for creating and running containers. Containers interact with the **network**, store data in **volumes**, and provide real-time **feedback** for system optimization. This streamlined architecture ensures efficient containerization and resource management across environments.

Conclusion

In this report, we have explored the crucial role of application services and their configuration in establishing reliable and efficient IT systems. Together, we focused on the installation and management of tools such as Nginx, Apache, Samba, and Docker, demonstrating their significance in creating scalable and robust infrastructures. Each service plays a unique role—whether it's hosting websites, sharing resources, or deploying containerized applications—while collectively contributing to the seamless operation and adaptability of modern systems.

From a Digital Forensics and Incident Response (DFIR) perspective, the analysis of network traffic patterns, particularly the TCP connection termination sequence, highlighted the significance of understanding network protocols for security monitoring and incident investigation. The ability to decode and interpret network communications is crucial for identifying potential security incidents, reconstructing network events, understanding normal versus abnormal behavior, and supporting incident response and threat hunting activities.

This project has strengthened our technical expertise and highlighted the importance of a proactive and collaborative approach to system administration. By working together, we have contributed to laying the foundation of reliable web application services, ensuring the systems we deploy are both functional and future-proof.

Bibliography

1. Geeks for Geeks - Application Services Overview
2. Wikipedia - Server Message Block (SMB)
3. LinkedIn - Diving Into Docker Containerization
4. Docker Official Documentation
5. Samba Official Documentation
6. Nginx Official Documentation
7. Apache HTTP Server Project
8. https://zomro.com/blog/articles/528-what-is-a-samba-server-for
9. SANS Institute
10. National Institute of Standards and Technology
11. Chkrootkit