VISVESVARAYA TECHNOLOGICAL UNIVERSITY
Jnana Sangam, Belagavi-590018

Project Report
on

ENHANCING NETWORKING MONITORING SYSTEMS BY OVERLAYING PROTOCOLS

by

RAJASHREE	**PAVITHRA K**
1VI17CS114	**1VI17CS071**

UNDER THE GUIDANCE OF

Mr. NOOR PASHA
Asst. Professor Department of CSE, Vemana Institute of Technology

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING,
VEMANA INSTITUTE OF TECHNOLOGY
3RD BLOCK, KORAMANGALA, BENGALURU – 560034

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

A computer network is a group of network nodes that use a set of common communication protocols over digital interconnections for the purpose of sharing resources located on or provided by the network nodes. A network node is either a redistribution point or a communication endpoint such as modem, hub, switch, bridge, routers, network file systems, personal computers and so on. Network monitoring system is a scripted tool that administers a computer network for slow or failing components and that notifies the configured users in case of outages, latencies, upgrades, and other events.

With the advent of internet, network monitors can now perform variating services such as fault analysis, performance management, provisioning of networks and maintain quality of service. Internet technologies have become very important in every person's day to day life and proactive monitoring of interconnected devices has become vital in the internet service provision business. The onset of ecommerce has reduced the prices of IoT to great extent and a typical home in suburban consists of Layer 3 router, a Wi-Fi access point, internet connected TV and remotely controlled light bulbs. As a result, security and monitoring have become a critical concern for every person with internet connectivity.

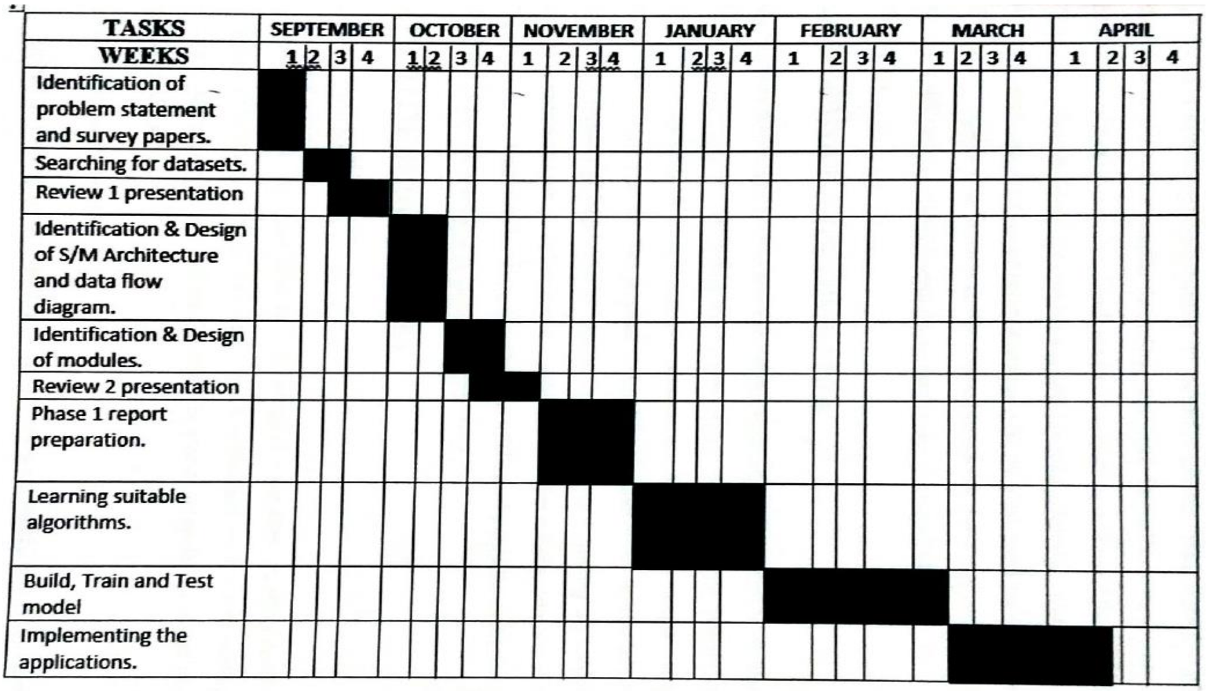
The latest generation of network monitoring systems are designed to support very specific applications. Most of them in general, are devised to monitor enterprise environments that generally consist of 100s of similar network nodes such as a network rack, routers, switches, and personal computers. Further, they are designed to support very specific applications such as identifying the device's connectivity or latency or out-of-band analysis and so on. To obtain results they are homogenous in terms of protocol usage. For example, some systems use ICMP alone to obtain performance analysis of a huge network. Also, due to the lack of a buffer Dataset, they are not generally real-time and are quite torpid. A typical topology that consists of less than 50 network nodes such as a startup environment or a home office environment cannot afford such a stack of tools and maintaining a stable and high performance network is an up-hill task for them. As a result, most of the emerging organizations employ a separate team that act as network administrators.

1.1 Scope

The developed network and server monitoring system addresses the problems of the current monitoring tools. It is an all-round application for network monitoring. The system is designed to monitor network information, server resources information and do basic configurations on network devices. The developed network monitoring system is web based. This allows network engineers and network administrators to remotely monitor the network infrastructure. Therefore with today's technological improvements the system can be monitored by virtually any web enabled mobile device. This feature empowers the network engineers to monitor their systems even on-the-go. Inevitably in a failure of network connectivity or a device the network engineers and administrators can assist the situation while working offsite, while traveling, while at home or even when they are abroad. The configuration facility also vastly improves the flexibility of the system. This reduces the need for another tool to do configurations on network devices. The monitoring system is based on open source technologies as well as cross platform operability. The developed system is a combination of two applications. One is a web interface for the users to view and monitor network and server and also the terminal for viewing configurations of network devices. The other part is a backend injector application, which gathers information and raw data and processes in order to be displayed via the web interface. The entire system is hosted on a Window based server with web services.

1.2 Objective

1.2.1



TASKS	SEPTEMBER				OCTOBER				NOVEMBER				JANUARY				FEBRUARY				MARCH				APRIL			
WEEKS	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Identification of problem statement and survey papers.	■																											
Searching for datasets.		■	■																									
Review 1 presentation			■	■																								
Identification & Design of S/M Architecture and data flow diagram.					■	■																						
Identification & Design of modules.							■	■																				
Review 2 presentation								■	■																			
Phase 1 report preparation.										■	■	■																
Learning suitable algorithms.													■	■	■	■												
Build, Train and Test model																	■	■	■	■	■							
Implementing the applications.																						■	■	■	■	■		

CHAPTER 2

LITERATURE SURVEY

The survey of various papers on Network Monitors is presented in the following section:

1] Network Monitoring System for Network Equipment Availability and Performance Reporting –by Baphumelele Masikisiki, Siyabulela Dyakalashre[2020]

It describes availability and performance of a network. Reporting with UFH being used as a test platform. This system is meant to help network administrators with troubleshooting network devices because it will tell them exactly where to go in order to fix the network. Additionally, the prototype will also have an ability to generate charts that presents the network performance in and untreatable format and also send notifications to the network administrators in cause of any outages.

Advantages and Disadvantages

- It used to test the availability and performances.
- The limitation of memory configurations led to the failure of memory performance

[2] Architecture of a Network Performance Monitor for Application Services on Multi-Clouds – by Young-min Kim, Ki-sung Lee, Jae-cheol Uhm, Si-chang Kim, and Chan-gun Lee 2013

Rudimentary reasons for having a network performance monitor for multi-cloud environments have been illustrated with examples in this research. An architecture for such a network monitor is proposed that explains how external agents are used to connect monitor such an environment. Also, a model to integrate various public clouds is formulated in a flexible manner. In addition, the issues of timely delivery and off-line analysis of measured results are addressed.

Advantages and Disadvantages:

- The model proposed for monitoring multi clouds is flexible and is integrated using external agents. These agents may be network intensive models for logging of network node events.
- This model isn’t tested in real-life scenario and the results mentioned are through simulated environments.
- A DBMS approach to store results for offline analysis has been proposed which would

be good for non-cloud network monitoring systems as well.

[3] Measurement-Aware Monitor Placement and Routing: A Joint Optimization Approach for Network-Wide Measurements – by Guanyao Huang, Chia-Wei Chang, Chen-Nee Chuah, and Bill Lin 2017

A theoretical framework is proposed in this research that jointly optimizes monitor placement and dynamic routing strategy to achieve maximum measurement utility with limited monitoring resources. Through experiments using real traces and topologies, it has been justified that these heuristic solutions can achieve measurement gains that are quite close to the optimal solutions, while reducing the computation times.

Advantages and Disadvantages:

- There are many implementation issues with the proposed framework that need to be addressed especially determining what routing protocols are being used.
- Also, this framework requires that the traffic reaches it in real-time instead of **the** framework polling for the same.
- The major takeaway from this research is the procedure involved in forming the base framework for a network monitoring system.

[4] A Web-based monitor and management system architecture for enterprise virtual private network – by Ruey-Shun Chen *, Change-Jen Hsu, Chan-Chine Chang, S.W. Yeh 2017

Monitoring encrypted connection gateways such as enterprise VPNs is a tedious task. This papers formulates a feasible system that can monitor VPNs and includes essential elements such as system components, operation flow and much more. Also, it explains how a web GUI can be integrated with back-end for real time analysis.

Advantages and Disadvantages:

This research is primarily localized on VPNs and encrypted gateways such as SSLVPN, Client-based VPNs and so on. Also, it is more focused on monitoring at a hardware level than at the end-user level.

It is still dependent on external management tools in order for the enterprises to be rest assured for using the new technologies.

2.1 Comparative Analysis

Ref	Technique	Advantage	Drawback
[1]	QoS and VoIP	Balanced Cost and Quality	Localized to large enterprises
[2]	Multi-Cloud	Flexible and first of its kind	Network intensive and simulated results
[3]	Routing	Method to implement a framework	No Real-time analysis
[4]	VPN	Integration of GUI with back end	Covers only VPNs

CHAPTER 3

SYSTEM REQUIREMENTS

3.1 Functional Requirements

1. To instantly detect device outages or performance reductions and immediately trigger notifications via E-Mail/SMS/GUI alerts to configured users.
2. To perform polling of device statistics in real-time to measure various performance attributes such as reachability, availability, bandwidth, and latency.
3. To provide easy to view and comprehensive web-based application user interface that automatically obtains device info, uptime, real-time statistics, traffic statistics and so on.
4. To provide periodic historical reports of devices on their performance.
5. To facilitate accessing different network nodes via SSH/Telnet/Web access from the monitor itself.
6. To ate custom configuration of notifications to different types of users in various situations through various mediums.
7. To add support monitoring of various network nodes such as Servers, Routers, Switches, Power Distribution Units (Network PDUs), Virtual Machines, IoT devices, Cloud instances, Data Stores, Wireless Access Points, Endpoint PCs and so on.

3.2 Non-Functional Requirements

- **Reliability**: The system should inform user when it operates in degraded mode. It should be able to inform user about any malfunction and be able to handle failures. System should be able to deliver the required service in reliable manner, i.e. for the input or state of the system the output obtained must be correct. Reliability is an attribute of any computer-related component (software, or hardware, or a network, for example) that consistently performs according to its specifications. It has long been considered one of three related attributes that must be considered when making, buying, or using a computer product or component. Reliability, availability, and serviceability - RAS, for short - are important aspects to design into any system. In theory, a reliable product is totally free of technical errors; in practice, however, vendors frequently express a product's reliability quotient as a percentage. Evolutionary products (those that have evolved through numerous versions over a significant period of time) are usually considered to become increasingly reliable, since it is assumed that bug s have been eliminated in earlier releases. For example, IBM's z/OS (an operating system for their S/390 server series), has a reputation for reliability because it evolved from a long line of earlier MVS and OS/390 operating system versions.

- Availability: Availability refers to the frequency at which the service provided by the system is available, if the system can be accessed at any point of time then that system is highly available. Availability of a system is typically measured as a factor of its reliability - as reliability increases, so does availability. Availability of a system may also be increased by the strategy on focusing on increasing testability & maintainability and not on reliability. Improving maintainability is generally easier than reliability. Maintainability estimates (Repair rates) are also generally more accurate. However, because the uncertainties in the reliability estimates are in most cases very large, it is likely to dominate the availability (prediction uncertainty) problem, even while maintainability levels are very high. When reliability is not under control more complicated issues may arise, like manpower (maintainers / customer service capability) shortage, spare part availability, logistic delays, lack of repair facilities, extensive retrofit and complex configuration management costs and others. The problem of unreliability may be increased also due to the "domino effect" of maintenance induced failures after repairs. Only focusing on maintainability is therefore not enough. If failures are prevented, none of the others are of any importance and therefore reliability is generally regarded as the most important part of availability. Reliability needs to be evaluated and improved related to both availability and the cost of ownership (due to cost of spare parts, maintenance man-hours, transport costs, storage cost, part obsolete risks etc.). Often a trade-off is needed between the two. There might be a maximum ratio between availability and cost of ownership. Testability of a system should also be addressed in the availability plan as this is the link between reliability and maintainability. The maintenance strategy can influence the reliability of a system (e.g. by preventive and/or predictive maintenance), although it can never bring it above the inherent reliability. So, Maintainability and Maintenance strategies influences the availability of a system. In theory this can be almost unlimited if one would be able to always repair any fault in an infinitely short time. This is in practice impossible. Repair-ability is always limited due to testability, manpower and logistic considerations.

- **Extensibility**: The system should be extensible to add further information and users for more expansion. In software engineering, extensibility (not to be confused with forward compatibility) is a system design principle where the implementation takes future growth into consideration. It is a systemic measure of the ability to extend a system and the level of effort required to implement the extension. Extensions can be through the addition of new functionality or through modification of existing functionality. The central theme is to provide for change, typically enhancements, while minimizing impact to existing system functions.

  Extensibility is a software design principle defined as a system's ability to have new functionality extended, in which the system's internal structure and data flow are minimally or not affected, particularly that recompiling or changing the original source code is unnecessary when changing a system's behaviour, either by the creator or other programmers. Because software systems are long lived and will be modified for new features and added functionalities demanded by users, extensibility enables developers to expand or add to the software's capabilities and facilitates systematic reuse. Some of its approaches include facilities for allowing users' own program routines to be inserted and the abilities to define new data types as well as to define new formatting mark-up tags.

- **Re-Usability**: The system's code could be reused to add further new features if need to be added in future. In computer science and software engineering, reusability is the use of existing assets in some form within the software product development process. More than just code, assets are products and by-products of the software development life cycle and include software components, test suites, designs and documentation. Leverage is modifying existing assets as needed to meet specific system requirements. Because reuse implies the creation of a separately maintained version of the assets, it is preferred over leverage. Subroutines or functions are the simplest form of reuse. A chunk of code is regularly organized using modules or namespaces into layers. Proponents claim that objects and software components offer a more advanced form of reusability, although it has been tough to objectively measure and define levels or scores of reusability. The ability to reuse relies in an essential way on the ability to build larger things from smaller parts, and being able to identify commonalities among those parts. Reusability is often a required characteristic of platform software. Reusability brings several aspects to software development that do not need to be considered when reusability is not required. Reusability implies some explicit management of build, packaging, distribution, installation, configuration, deployment, and maintenance and upgrade issues. If these issues are not considered, software may appear to be reusable from design point of view, but will not be reused in practice. Software reusability more specifically refers to design features of a software element (or collection of software elements) that enhance its suitability for reuse.

- **Robustness**: The system must be tolerant enough to cope up with minor errors during it run time. In computer science, robustness is the ability of a computer system to cope with errors during execution. Robustness can also be defined as the ability of an algorithm to continue operating despite abnormalities in input, calculations, etc. Robustness can encompass many areas of computer science, such as robust programming, robust machine learning, and Robust Security Network form, the more robust the software. Formal techniques, such as fuzz testing, are essential to showing robustness since this type of testing involves invalid or unexpected inputs. Alternatively, fault injection can be used to test robustness. Various commercial products perform robustness testing of software systems and is a process of failure assessment analysis. In general, building robust systems that encompass every point of possible failure is difficult because of the vast amount of possible inputs and input combinations. Since all inputs and input combinations would require too much time to test, developers cannot run through all cases exhaustively. Instead, the developer will try to generalize such cases. Interoperability: New updates to the system must be easily accommodated in future. Interaction with the user must be minimal. System is to be developed in phases, so it shall be easily upgradeable. Upgrading is the process of replacing a product with a newer version of the same product. In computing and consumer electronics an upgrade is generally a replacement of hardware, software or firmware with a newer or better version, in order to bring the system up to date or to improve its characteristics. Common software upgrades include changing the version of an operating system, of an office suite, of an anti-virus program, or of various other tools.

- **Performance**: Performance requirements define how well the software system accomplishes certain functions under specific conditions. Examples include the software's speed of response, throughput, execution time and storage capacity. The service levels comprising performance requirements are often based on supporting end-user tasks. Like most quality attributes, performance requirements are key elements in the design and testing of a software product. The system should provide real time alerts of any events occurring in the added devices by the customer.

- **Efficiency**: Efficiency tests the amount of resources required by a program to perform a specific function. In software companies, this term is used to show the effort put in to develop the application and to quantify its user-satisfaction. In a company, how much resources used and how much of these are turned in to productive goods are internal. This mainly focuses on resource availability, tools, people, time, type of project, complexity, situation, customer requirement etc. It is a measure of how well the team does the required work to get useful output. Efficiency is one of the parameters. It can never be more than a 100%. By definition, efficiency is the ratio of output to input expressed in percentage. It is not just one single formula but a number of calculations at each step and activity of testing. We can take an example of a team where the manager would want to deliver the product on time, within budget, without compromising with the quality.

- **Maintainability**: Maintainability refers to the ease with which you can repair, improve and understand software code. Software maintenance is a phase in the software development cycle that starts after the customer has received the product. Developers take care of maintainability by continuously adapting software to meet new customer requirements and address problems faced by customers. This includes fixing bugs, optimizing existing functionality and adjusting code to prevent future issues. The longevity of a product depends on a developer's ability to keep up with maintenance requirements. Software maintenance is the most expensive phase of development, typically consuming more than half of development budgets. It is important to plan maintenance into the development lifecycle so you can maintain software efficiently. Maintainability is a long-term aspect that describes how easily software can evolve and change, which is especially important in today's agile environment. Easy Learning of the system is necessary for the maintenance personnel to repair it easily in the events of failure.

- **Safety and Security**: Functional safety is achieved through engineering development to ensure correct execution and behavior of software functions as intended. Safety consistent with mission requirements, is designed into the software in a timely, cost effective manner. On complex systems involving many interactions safety-critical functionality should be identified and thoroughly analyzed before deriving hazards and design safeguards for mitigations. Safety-critical functions lists and preliminary hazards lists should be determined proactively and influence the requirements that will be implemented in software. Contributing factors and root causes of faults and resultant hazards associated with the system and its software are identified, evaluated and eliminated or the risk reduced to an acceptable level, throughout the lifecycle. Reliance on administrative procedures for hazard control is minimized. The number and complexity of safety critical interfaces is minimized.

The number and complexity of safety critical computer software components is minimized. Sound human engineering principles are applied to the design of the software-user interface to minimize the probability of human error. Failure modes, including hardware, software, human and system are addressed in the design of the software. Sound software engineering practices and documentation are used in the development of the software. Safety issues and safety attributes are addressed as part of the software testing effort at all levels. Software is designed for human machine interface, ease of maintenance and modification or enhancement. Software with safety-critical functionality must be thoroughly verified with objective analysis and preferably test evidence that all safety requirements have been met per established criteria.

- **Scalability**: Scalability is a non-functional property of a system that describes the ability to appropriately handle increasing (and decreasing) workloads. A system is described as scalable, if it will remain effective when there is a significant increase in the number of resources and the number of users. Sometimes, scalability is a requirement that necessitates the usage of a distributed system in the first place. Also, scalability is not to be confused with raw speed or performance. Scalability competes with and complements other non-functional requirements such as availability, reliability and performance.

There are two basic strategies for scaling--vertical and horizontal. In case of vertical scaling, additional resources are added to a single node. As a result, the node can then handle more work and provides additional capacities. Additional resources include more or faster CPUs, more memory or in case of virtualized instances, more physical shares of the underlying machine. In contrast, horizontal scaling adds more nodes to the overall system.

Both scaling variants have different impacts on the system. Vertical scaling almost directly speeds up the system and rarely needs special application customizations. However, vertical scaling is obviously limited by factors such as cost effectiveness, physical constraints and availability of specialized hardware. Horizontal scaling again requires some kind of inherent distribution within the system. If the system cannot be extended to multiple machines, it could not benefit from this type of scaling. But if the system does support horizontal scaling, it can be theoretically enlarged to thousands of machines. This is the reason why horizontal scaling is important for large-scale architectures. Here, it is common practice to focus on horizontal scaling by deploying lots of commodity systems. Also, relying on low cost machines and anticipating failure is often more economical than high expenses for specialized hardware.

Considering a web server, we can apply both scaling mechanisms. The allocation of more available system resources to the web server process improves its capacities. Also, new hardware can provide speed ups under heavy load. Following the horizontal approach, we setup additional web servers and distribute incoming requests to one of the servers.

- **Usability**: Usability measures characteristics such as consistency and aesthetics in the user interface. Consistency is the constant use of mechanisms employed in the user interface while aesthetics refers to the artistic, visual quality of the user interface.

It is the ease at which the users operate the system and make productive use of it. Usability is discussed with relation to the system interfaces, but it can just as well be applied to any tool, device, or rich system. This addresses the factors that establish the ability of the software to be understood, used, and learned by its intended users.

The application interfaces must be designed with end users in mind so that they are intuitive to use, are localized, provide access for differently. Usability is a controlled aspect of User Experience design that ensures the end-user doesn’t strain or encounter problems with the use of a product or website’s user interface. A user experience designer can control accessibility, user interface, information architecture and usability to suit the uncontrolled aspects like goals, user lifestyle and habits. UX design uses the controlled aspects of technology to suit the uncontrolled. A usable product does not mean automatically that its interface is simple or easy to use. Usability is not about making everything easy. The three main enhancements to the end-users experience from a usable product are efficiency, effectiveness and ultimately satisfaction. Efficiency is the ability to perform the intended task with desired speed. Effectiveness measures availability and unavailability. It is the comparison of usability with technology and without technology. It is the satisfaction of the end user with what they are capable of doing with the interface.

- **Durability**: Durability means the solution ability of serviceability of software and to meet user's needs for a relatively long time. Software durability is important for user's satisfaction. For a software security to be durable, it must allow an organization to adjust the software to business needs that are constantly evolving, often in impulsive ways. The flexibility of current frameworks encourage system architects to enable reconfiguration mechanisms that refocus the available, safe resources to support the most critical services rather than over-provisioning to build failure-proof system. With the generalisation of networked information systems, accessibility was introduced to give greater importance to users' experience.

3.3 Hardware Requirements

The following section describes the various hardware requirements for a network monitor to be developed and used:

- Processor: Intel® Core i5 ™ CPU and above
- RAM: 8 GB or higher
- Hard Disk: 100 GB or higher

3.4 Software Requirements

The following section describes the various hardware requirements for a network monitor to be developed and used:

- Operating System: Windows 10/Ubuntu 20.04 LTS
- Architecture: 64-bit OS
- Python 3.8 or higher
- PIP Packages: RegEx, Django, Pymysql
- Database: MySQL5.7 or higher
- JavaScript 1.8.5 or higher
- Front End: HTML5, CSS3, Bootstrap4

CHAPTER 4

DESIGN METHODOLOGY

A design document is a primary document of agreement between a customer and a software engineer. It consists of the various functional and non-functional requirements of software in a language that is understood both by the user and the software engineer. Design documents usually have various levels of details. The high-level design document describes the abstract or the outer functionalities of the system. The low-level design documents focus on the minute and internal details of the system.

4.1 System Architecture:

The system architecture of the proposed system is illustrated in the Figure 4.1

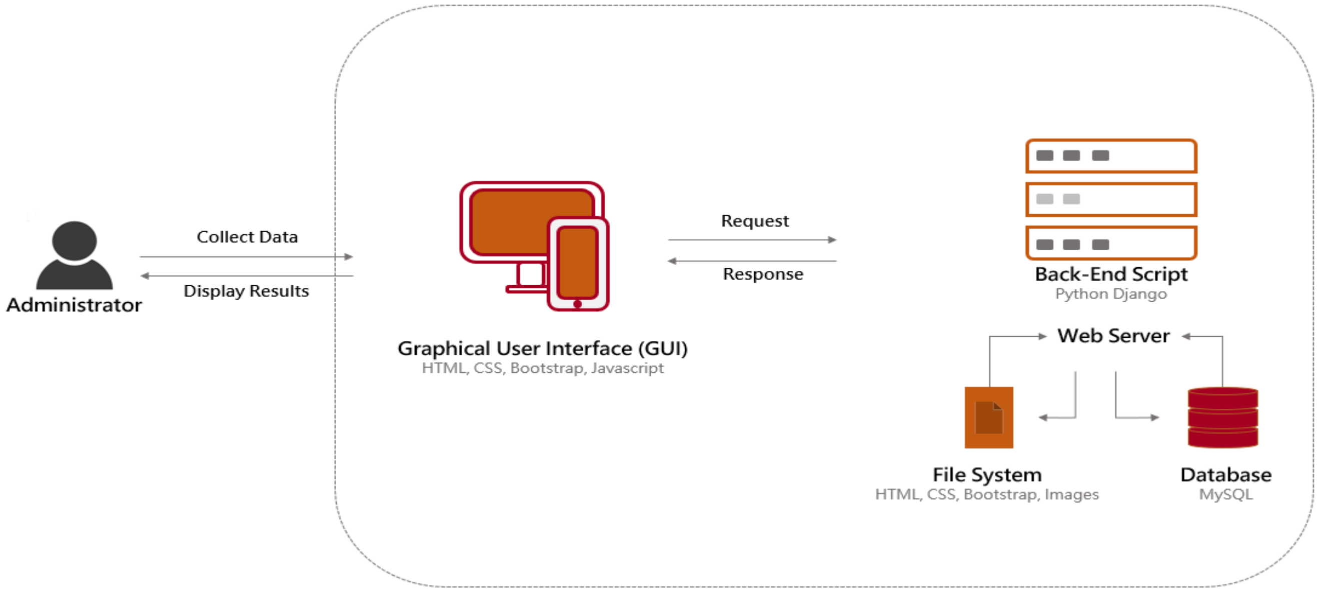


Fig: 4.1 System Architecture

4.2 Design:

The various diagrams that illustrate the design of the network monitor are shown using use case diagrams, class diagrams and sequence diagrams in the following section:

4.2.1 Use Case Diagram:

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams.

The Figure 4.2 shows the Use Case Diagram for Network Monitor that illustrates the relationship between the user and the system in different use cases. Various actors involved in the product are Administrator – user who controls the network monitor system, Back-end – involves a python script running the functions, Database – to store and relay the obtained data from devices and Devices that are monitored as per administrator's instruction.

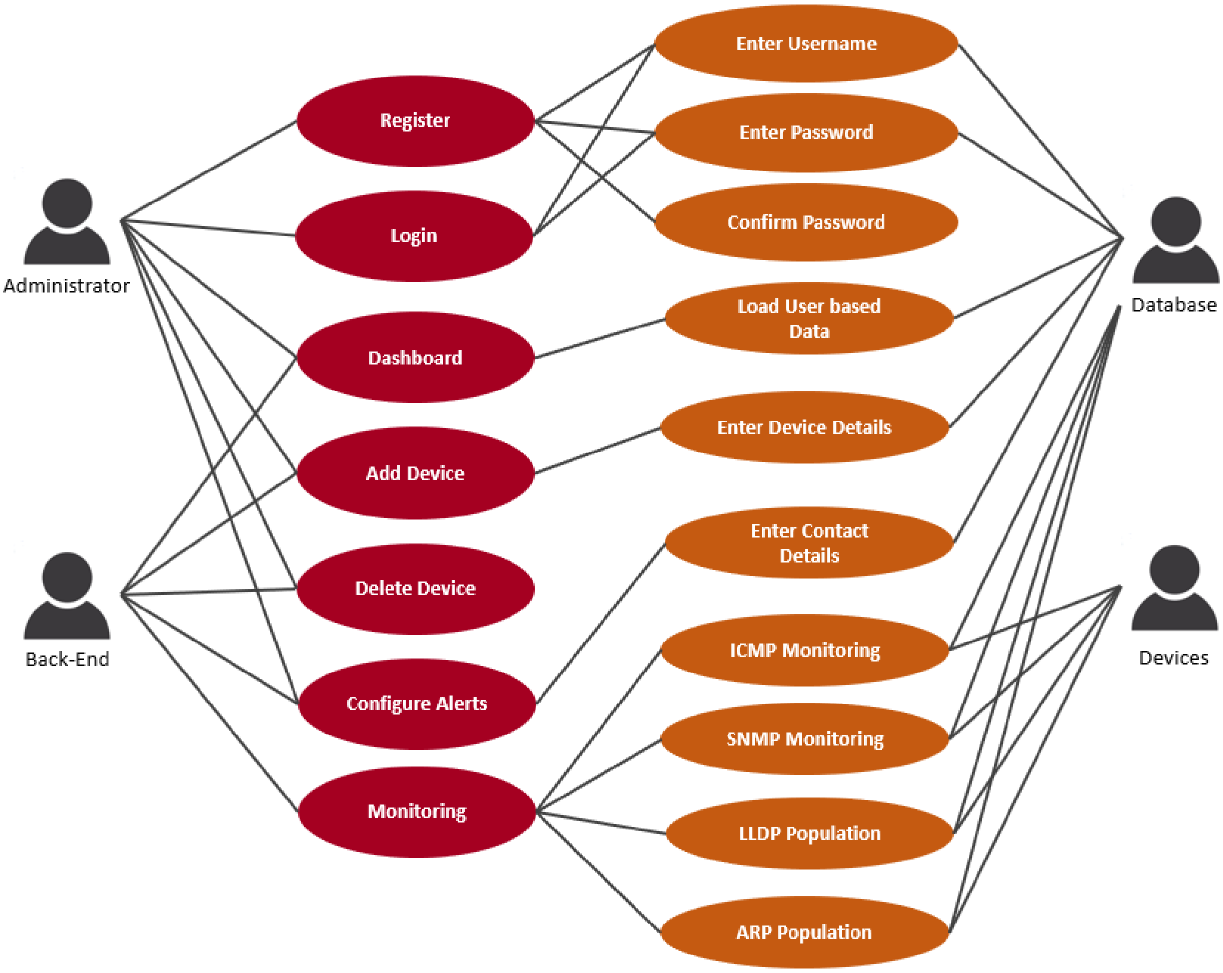


Fig: 4.2 Use Case Diagram for Network Monitor

4.2.2 Class Diagram:

A class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects. The Figure 4.3 shows classes and associations involved in Network Monitor.

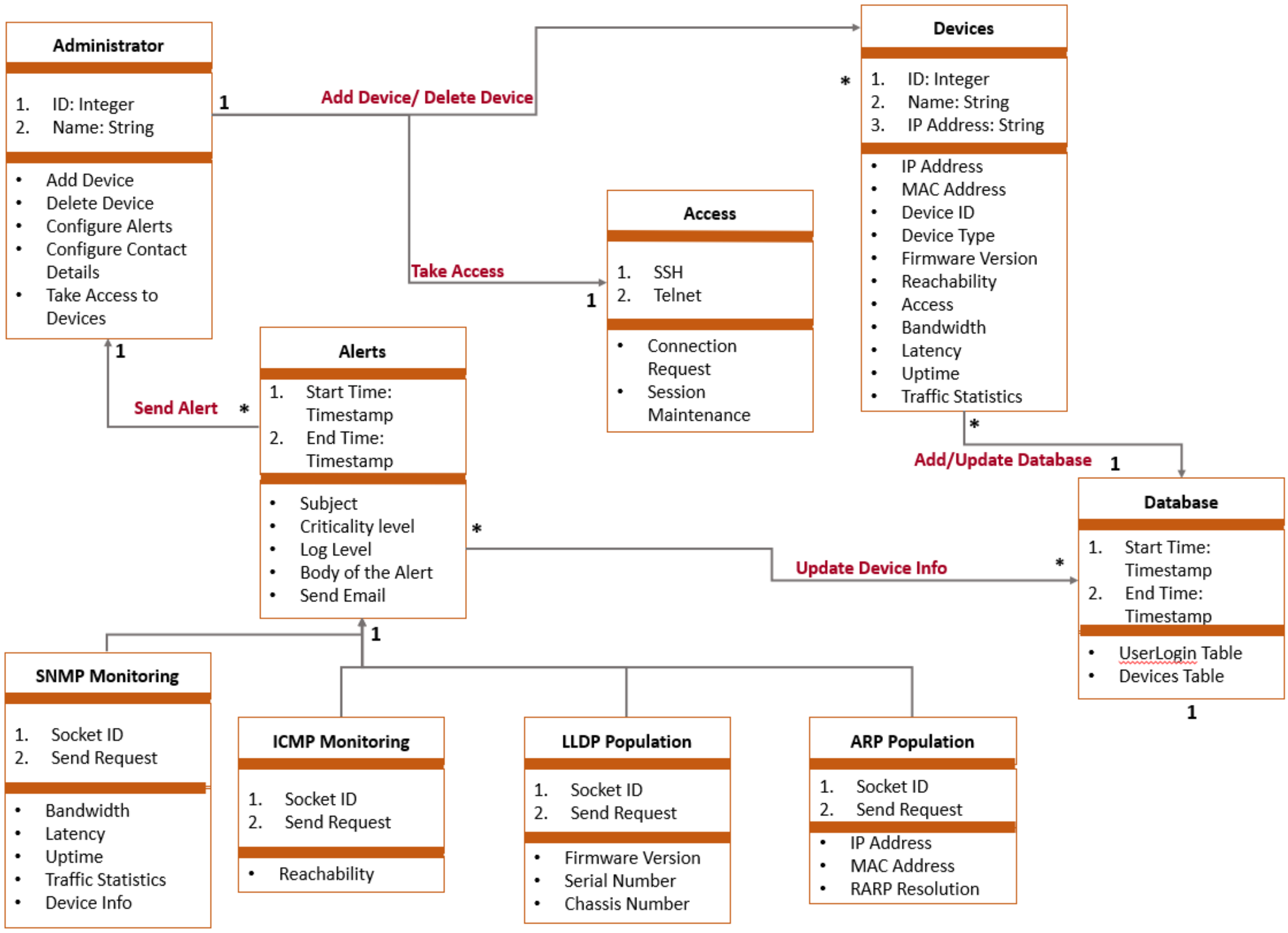


Fig: 4.3 Class Diagram for Network Monitor

The most important classes in this prohect are the Administrator, Devices, Alerts, Database, Access, SNMP Monitoring, ICMP Monitoring, LLDP Population, ARP Population.

1. The **Administrator** class contains various attributes such as ID, Name and so on which are needed to login to the Network Monitor System to perform various functionalities such as Adding a Device, Deleting a Device, Configuring custom Alerts, Configuring contact details such a email and so on and taking access to the device.
2. The **Devices** class containes several attributes such as ID, Name, IP Address and so on that are used to populate database with various parameters of each device such as MAC,

Address, Device ID, Device Type, Firmware Version, Reachability, Access, Bandwidth, Latency, Uptime, Traffic Statistics and so on

3. The **Alerts** class contains two timestamps namely Start Time and End Time that identify every alert event. Information parameters that describe an alert include Subject, Criticality Level, Log Level, body of the alert and Sending an Email to configured mail address by the Administrator.
4. The **Access** class is used to denote enabling the administrator to take access to the devices using protocols such as SSH and Telnet provided the requisite access has been allowed and configured in device.
5. The **SNMP Monitoring** class contains attributes such Socket ID and Send Request that are used to obtain information related to the device including Bandwidth, Latency in reaching the device, Uptime of the device, Traffic Statistics in real time and information about device.
6. The **ICMP Monitoring** class contains attributes such Socket ID and Send Request that are used to determine the reachability of the device.
7. The **LLDP Population** class contains attributes such Socket ID and Send Request that are used to learn more about the device through the LLDP RX packets; firmware versions, serial number and chassis number are of interest in particular.
8. The **ARP Population** class contains attributes such Socket ID and Send Request that are mainly used to populate MAC Address and IP Address of the device into the Database. Occasionally this functionality can also be used to perform a RARP resoultion or even detect Gratiutious ARP packets as a future enhancement.

4.2.3 Sequence Diagrams:

A Sequence diagram is an interaction diagram that shows how processes operate with one another and what is their order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios. Figure 4.4 explains how an administrator logs into the Network Monitor System. Various steps performed for this case are explained below:

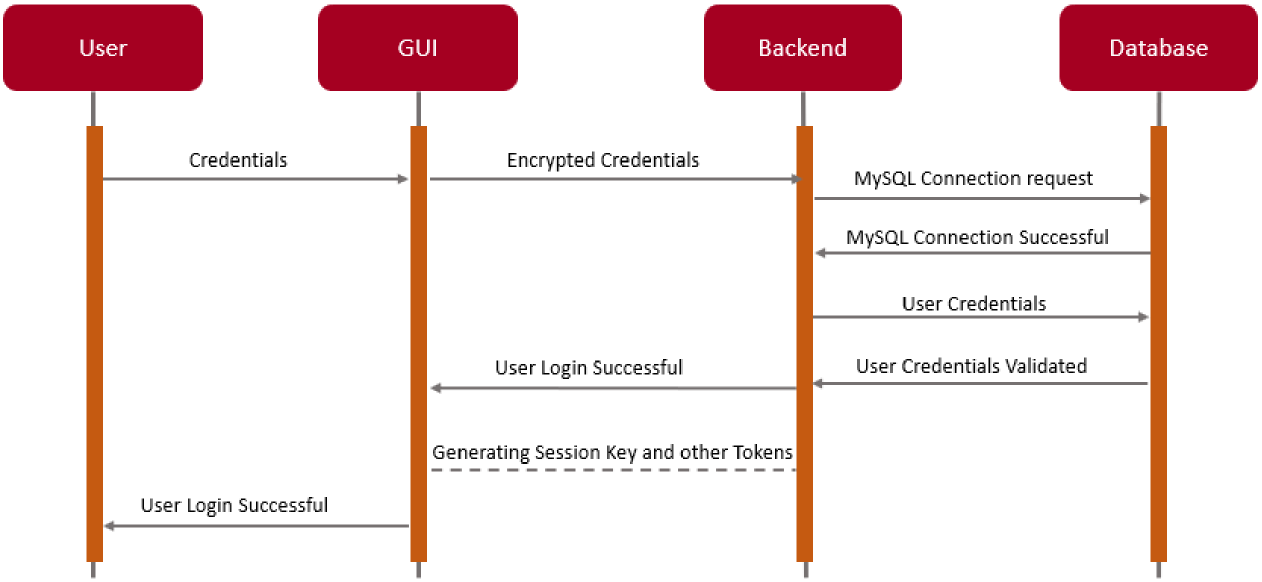


Fig: 4.4 Sequence Diagram for Network Monitor

Firstly, the user clicks on the **"Login"** button and enters the credentials namely username and password in the graphical user interface and clicks on **"Submit"** button. These credentials are then encrypted by the Django framework's session manager and sent to the Backend python script. The Backend control makes connection request to the Database and sends the user credentials upon connection establishment. The Database, MySQL in this case, would verify the credentials and sends a "OK" message if valid as a query response. The Framework then generates session key and other tokens to manage the user session. Upon successful creation of the session, a message is displayed on the GUI that "User Login is Successful" and loads the customer specific dashboard.

4.2.4 Data Flow Diagrams:

A Data Flow Diagram (DFD) is a graphical representation of the "flow" of data through an information system, modelling its process aspects. A DFD is often used as a preliminary step to create an overview of the system, which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design).

A DFD shows what kind of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of process or information about whether processes will operate in sequence or in parallel.

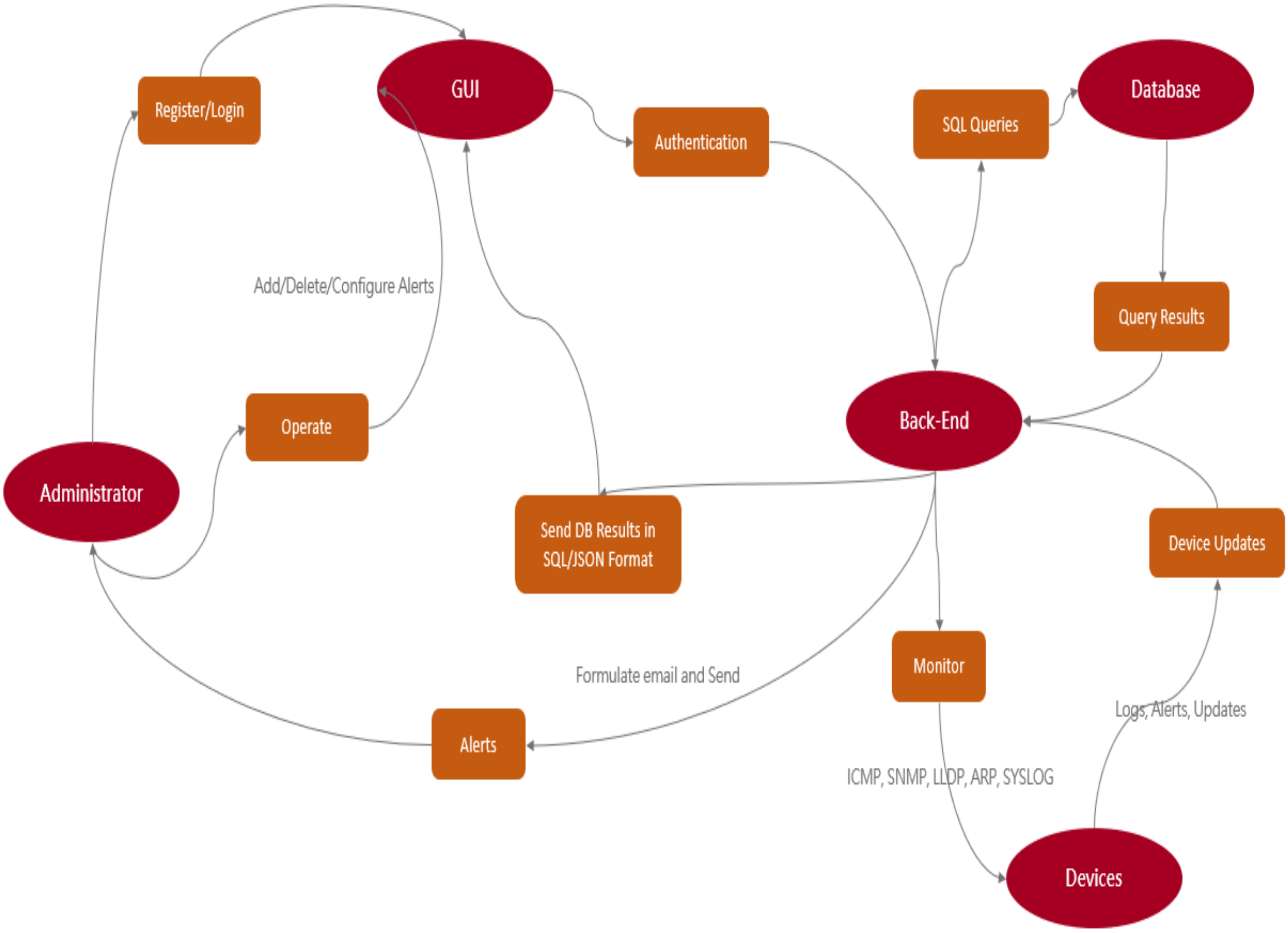


Fig: 4.5 Data Flow Diagram for Network Monitor

The Figure 4.5 shows the various data items flowing for Network Monitor. The various data items involved in this design are: Authentication, addition/deletion of devices, alerts, database updates, query retrievals, device monitoring and device data updates.

Authentication involves registration and login modules of the network monitor. When the user enters credentials into the GUI and clicks on Login, the credentials are encrypted by backend processing and sent to the Database through SQL Query exchange. When the database validates the authentication request, the backend creates a new session along with their tokens. Then appropriate message is displayed on GUI. Adding/Deleting a device basically involves addition/removal of rows in Device Table of the database which are identified using the device id, name, and IP address as parameters.

An added device is monitored using different protocols, namely, ICMP, SNMP, LLDP, ARP and SYSLOG in the initial phase. Each protocol is used to obtain a particular data about various parameters attributing to the device. For example, ICMP provides us the reachability status and TTL values for required to reach the device. Similarly, LLDP is used to obtain Serial number, chassis number, and firmware version and so on from the LLDP RX packets from the device. Administrators are facilitated with various options to configure alerts such as allowing only critical alerts to be sent at certain times. Based on this configuration and criticality of the events, various logs are accumulated and sent in email to the user. Backend processing is basically one constantly running python scripts that updates a database using pymysql framework, which acts as a buffer for GUI to load quickly, every time an update comes up from added devices.

CHAPTER 5

MODULE DESCRIPTION

Different Modules involved in a Network Monitor are explained in detail below:

Graphical User Interface:

- This facilitates the user to login into the system, initiate the utility, whereby detail of the entire network is present in a table.
- This table is automatically filled with details of the network node such as MAC Address, OS and so on using LLDP protocol
- The Traffic Stats will show up eventually which are obtained from the device using ICMP protocol
- The user can also see all the performance attributes such as reachability, latency and so on present on any network node by switching between tabs.
- The user can take SSH/Web/Telnet access to the network node by clicking on Access button in the table.
- The user can update, modify, change, or configure the network nodes. This triggers a background process of history keeping. It also provides the ability to caution any change of an attribute which may not be compatible with the resource at hand.
- The GUI also shows latest alerts on the top with various color codes assigned based on the criticality of the event.
- Types of Alerts and their frequency can be configured under the Profile option.
- Coding Languages used for this Component: HTML, CSS, Bootstrap Templates, JavaScript

Back-end Processing:

- This module handles all functionality from finding the network nodes, attributes, querying with the network nodes and updating the database with information obtained through back end processing
- It also maintains a dynamically updated database which contains all the attributes present on every resource along with compatibility problems. As a result, it can take decisions on issues occurred and report errors accordingly.
- Various protocols are used to obtain different types of data for example LLDP for device information, SNMP for performance statistics, SSH/Telnet/HTTP for device

access, ICMP for traffic statistics and so on.

- Raw IP sockets are formed and sent in various ways to determine the operating system the network nodes are using, packet filters in use and several other attributes.
- Coding Languages used for this Component: Python, Django/Flask Framework, JavaScript, pymysql framework

Database:

- This module handles acts a buffer between the backend functionality and the graphical user interface. It provides synchronization between the activities of the user and the backend
- To avoid top down triggering and querying of the network node data every time the web page is loaded or meta-refreshed, the backend functionality runs infinitely to fill various attributes of the network node by querying it using various network protocols at regular intervals.
- Simultaneously, GUI fetches data from database which results in real-time transmission at any given second.
- MySQL Tables used:
    - UserAuthentication: Deals with user login, logout, and session keys
    - Devices: Deals with devices and their attributes at real time

Languages used for this Component: MySQL Database, MySQL query Language, MySQL Workbench / MySQL CLI for diagnostics

Ping: The word Ping means "packet internet Groper Ping" Ping is a basic Internet program that allows a user to verify that a particular IP address exists and can accept requests. Ping is used diagnostically to ensure that a host computer the user is trying to reach is actually operating. Ping works by sending an Internet Control Message Protocol (ICMP) Echo Request to a specified interface on the network and waiting for a reply. Ping can be used for troubleshooting to test connectivity and deter mine response time Ping command is very useful in debugging and testing network and internet connection.

Every time the user ping a device user get a response back and it is very important to check the reply. Each response will help and guide the user to find the right troubleshooting path and it will be easy to find the exact device which is having problems and troubleshoot that device. Some common response that user can get from ping described below.

- Reply From: by getting this response from the address that the user has pinged it means the connection is working.
- Request Timed out: when the destination host down then pings will fail and the user will get this reply because that host doesn't reply.
- Unknown Host: getting this reply means the user's computer cannot recognize this address. Usually the IP address of the destination host is wrong or the host name.
- Destination Host unreachable: The destination that user is trying to ping is down or some ports may be down. Hardware Error this means that user's network adapter is shut down or the cables are unplugged.

SNMP: The Simple Network Management Protocol (SNMP) is a standard internet protocol for maintaining devices and proactively let you know what's going on your IP Network Monitor network.. It is often used in network management system for situation that warrant the attention of the administration. It is part of the Transmission Control Protocol/Internet Protocol (TCP/IP) protocol, Hence is an Application Layer protocol, the Layer 7 of the OSI model. It uses one or more administrative Computers, called managers.

It has the feature of monitoring or managing a group of devices on a Computer Network, however there is a software component called an Agent installed on the managing system which reports information via SNMP to the manager, thus SNMP work using management Console and SNMP Agent. One of the reasons why it is a very good protocol is that you can get full information about what is happening on your devices, at the exact time. It can see what's going on with the hardware of the computer, take for instance in an organization with over 200 servers, and you as an administration want to keep track on them to make sure that they running fine however, before they can be up and running fine, all the advices should be in good working conditions.

SNMP Agent exposes management data in the form of variable on the managed systems, you also can perform additional task, like modifying and applying new setting or configuration through the remote modification of these variables.

SNMP managed network components:

- Managed Device: this is a network node that contains an SNMP agent on managed network. Management devices include Router, Switch it store the information collected and make it available on Network-Management Systems (NMS).
- Agent: This is a software that is installed on a device that is being managed by SNMP.

Management System: This is software is installed on Manger

SYSLOG is a standard for system logging. In short we can say anything we configure to log from a system will be forwarded and will store like a typical log book. SYSLOG use UDP port 514 for communication between SYSLOG servers and clients. As its UDP we know that it doesn’t do reliable communication. SYSLOG packet contains information about the facility, Severity, Hostname, Timestamp, and Message. SYSLOG messages are categorized widely basis of the source generate them. For example it could be a router, switch, an operating system, process etc. These categories are called by the Facility.

Let see how SYSLOG works and it can help us in monitoring. SYSLOG has three Components by which it performs logging. There are three SYSLOG components described below.

- Log device: devices which create log messages such as router, firewall, server or any device which can create an event message for SYSLOG.
- Log collector: These components collect SYSLOG data. SYSLOG server would be a typical log collector in this case.
- Log relay: sometime message need to forward through a log collector. These component forward messages which they receive from log devices.

Typical applications and OS are created in such a way that they are capable to give clues of their condition or any change of state. These clues help us to understand what is the application is doing? Individual applications and OS deliver message differently but SYSLOG is one of such application which is capable of accepting messages from different vendors at the same time. Logging is a very important tool for network admin to identify unusual activity in the network. This can be used for debugging any problems also. SYSLOG can be used as intrusion detection system. If any system is compromised, intruder tries to cover the footprint but with SYSLOG it’s really hard because it gives real time logging. So network admin actually can find out instantly or later how intruder compromised the system. If any system is compromised, a proper investigation will be started in order to determine of severity and compromise. At that time SYSLOG played a great role in creating a picture of the system of that time. Let say we monitoring one of the routers. What kind of info we can get by SYSLOG. Nay port failure will be logged. So we can instantly fix the problem by watching log message. Some time we need to see who logged remotely in our system. We can get info also form SYSLOG. SYSLOG provides us info of different severity level which helps us to understand how the system is healthy and what steps we need to take based on that severity level.

ICMP: Ping is a basic Internet program that uses ICMP and allows a user to verify that a particular IP address exists and can accept requests. Ping is used diagnostically to ensure that a host computer the user is trying to reach is actually operating. Ping works by sending an Internet Control Message Protocol (ICMP) Echo Request to a specified interface on the network and waiting for a reply. Ping can be used for troubleshooting to test connectivity and determine response time Ping command is very useful in debugging and testing network and internet connection.

ARP: The Address Resolution Protocol (ARP) is a communication protocol used for discovering the link layer address, such as a MAC address, associated with a given internet layer address, typically an IPv4 address. The Address Resolution Protocol uses a simple message format containing one address resolution request or response. The message header specifies the types of network in use at each layer as well as the size of addresses of each.

LLDP: The Link Layer Discovery Protocol (LLDP) is a vendor-neutral link layer protocol used by network devices for advertising their identity, capabilities, and neighbors on a local area network based on IEEE 802 technology, principally wired Ethernet. Media Endpoint Discovery is an enhancement of LLDP, known as LLDP-MED, that provides:

- Auto-discovery of LAN devices
- Device location discovery to allow creation of location databases and, in the case of Voice over Internet Protocol (VoIP)
- Inventory management, allowing network administrators to track their network devices, and determine their characteristics (manufacturer, software and hardware versions, serial or asset number).

Web Interface: Data which is collected by the core software and the syslog is stored in the database. The database is read by a web application which is written using Python. Main objective here is to display the important information which is collected by the software in an attractive manner. Technologies like JavaScript, CSS, and JQuery are used in the client side and Python is used as the server side programming. One of the most important objectives we accomplished in this was that we could be able to run basic configuration commands using the web interface. The administrator can run simple commands like show and debug, then observe the output of those commands in the same web interface. And the most important thing is this process can be done from anywhere in the world.

CHAPTER 6

APPLICATIONS

- Networks serve as the backbone for any enterprise. Any network outage during working hours is huge loss for the organizations.
- As a result, they employ a separate team to look after their labs by constantly logging into several system interfaces and checking their statuses.
- It is a tedious task to login to each of these nodes, check if they are reachable and check their health status constantly.
- As the enterprise grows, the numbers increases exponentially.
- Generating network performance reports
- Deploying new technology and software upgrade successfully
- Monitoring the flow of traffic with netflow
- To track user network activity in an organization

CHAPTER 7

SUMMARY

Network monitoring systems ensure the availability and overall performance of computers and network services. Network admins monitor access, routers, slow or failing components, firewalls, core switches, client systems, and server performance—among other network data. Network monitoring systems are typically employed on large-scale corporate and university IT networks. When choosing a particular tool to use for monitoring, an Admin must first decide if they would like to use a more proven system or a newer system. If the proven system is the direction that feels more comfortable, NetFlow is the most beneficial tool to use since a data analysis package can be used in conjunction with it to present the data in a user friendly environment. Being able to monitor and analyze networks is vital in the job of Network Administrators. They must strive to keep the networks they oversee in good health as to not disrupt productivity within a company and to not disrupt any essential public services. As summarized throughout this paper several router based and non-router based techniques are available to assist Network Administrators in the day to day monitoring and analysis of their networks. SNMP, LLDP, SYSLOG, and NetFlow are a few of the router based techniques that are briefly reviewed. The non-router based techniques that were discussed were Active, Passive, and Combinational monitoring tools.

CONCLUSION

This report illustrates how very basic tools can be used to monitor the network. Knowing that simple tools can be so useful in monitoring our network, it gives details about all those tools work which most network administrators don't really take note, also with the details giving in this report reader can easily see which type of network monitoring to use and what it can monitor. And depends on those tools, companies are creating tailor made monitoring tools. These monitoring tools give us a great flexibility to monitor our network in order to tune the performance measurement in a great extent. Moreover they tell us what to monitor deeper, why should we monitor and what can be done with the result. Monitoring a network is very essential, intelligent management of the business. It helps the business grow and prevent them from un-necessary downtime due to inappropriate use of resources. Implementing the most essential Monitoring tools for BIM logical can prevent their network down age, they can take a decision about their company internet accessibility policy depends on available data. Finally one point can be proven by this report that by monitoring a network we can be benefited in many ways, some of them are tangible and some are intangible. We should know our network well by monitoring it.

REFERENCES

[1] **Dynamic Network Control with QoS and Resource Priority Monitor Based on Active "eM" for Commercial VoIP** – by Melanie Grah and Dr. Peter Radcliffe 2018

[2] **Architecture of a Network Performance Monitor for Application Services on Multi-Clouds** – by Young-min Kim, Ki-sung Lee, Jae-cheol Uhm, Si-chang Kim, and Chan-gun Lee 2016

[3] **Measurement-Aware Monitor Placement and Routing: A Joint Optimization Approach for Network-Wide Measurements** – by Guanyao Huang, Chia-Wei Chang, Chen-Nee Chuah, and Bill Lin 2017

[4] **Study on monitor and control of POL transport by road in IOT** – by Yang Chen, Qidong Yong, Dong Xiang 2017

[5] **Distributed Interplanetary Delay/Disruption Tolerant Network (DTN) Monitor and Control System** – by Shin-Ywan Wang 2012

[6] **Overhead Contact System On-line Monitor Technology Based on Wireless Sensor Network** – by Jiangjian Xie and Yi Wang, Tingting Lu 2011

[7] **The Design and implementation of a UPS Monitor and Control System** – by Lidong Fu and Bin Zhang 2011

[8] **Fault-tolerant Schemes for NoC with a Network Monitor** – by Zhang Ying , Wu Ning , Wan Yu Peng , Ge Fen , Zhou Fang 2010

[9] **Using activity sensitivity and network topology information to monitor project time performance** – by Mario Vanhoucke1. 2010

[10] **A transparent virtual machine monitor level packet compression network service** – by Ali Hamidi, Hadi Salimi and Mohsen Sharifi. [2010]

[11] **On evaluating the differences of TCP and ICMP in network measurement** – by Li Wenwei, Zhange Dafang, Yang Jinmin and Xie Gaogang [2017]