NETDOG - CONFIGURATION MANAGEMENT AND MONITORING SYSTEM

A PROJECT REPORT

Submitted by

ASWIN BABU K
TVE16MCA14

to

the APJ Abdul Kalam Technological University
in partial fullfilment of the requirements for the award of the degree

of

Master of Computer Applications

Department of Computer Applications
College of Engineering
Trivandrum-695016

JUNE 2019

DECLARATION

I undersigned hereby declare that the project report (Netdog - Configuration Management and Monitoring System), submitted for partial fulfillment of the requirements for the award of degree of Master of Computer Applications of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by me under supervision of Prof. Baby Syla L. This submission represents my ideas in my own words and where ideas or words of others have been included, I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title

Place : Trivandrum
Date : 13/05/2019 Aswin Babu K

DEPARTMENT OF COMPUTER APPLICATIONS

COLLEGE OF ENGINEERING TRIVANDRUM

CERTIFICATE

This is to certify that the report entitled **Netdog - Configuration Management and Monitoring System** submitted by **Aswin Babu K** to the APJ Abdul Kalam Technological University in partial fulfillment of the requirements for the award of the Degree of Master of Computer Applications is a bonafide record of the project work carried out by him under my guidance and supervision.. This report in any form has not been submitted to any University or Institute for any purpose.

Internal Supervisor External Supervisor

Head of the Dept

ACKNOWLEDGEMENT

If words are considered as symbols of approval and tokens of acknowledgment, then let words play the heralding role of expressing our gratitude.

First of all I would like to thank God almighty for bestowing us with wisdom,courage and perseverance which had helped us to complete this project ***NET-DOG - CONFIGURATION MANAGEMENT AND MONITORING SYSTEM***. This project has been a reality as a result of the help given by a large number of personalities.

I would like to remember with gratitude **Prof. Jose T Joseph**, Head Of Department Department of Computer Applications, College of Engineering, Trivandrum for the encouragement and guidance rendered.

I express our sincere thanks to **Prof. Baby Syla**,Assistant Professor, Department of Computer Applications, College of Engineering Trivandrum for his valuable guidance, support and advices that aided in the successful completion of my project.

Finally, We wish to express our sincere gratitude to all our friends, who directly or indirectly contributed in this venture.

ASWIN BABU K

ABSTRACT

Netdog is a simple to use configuration management and monitoring system. It was developed with schools, colleges and small businesses in mind. Netdog allows people to easily manage the computers on a network. This includes execution of commands across all computers, shutting down the computers and even view various hardware stats across the machines. The current version is the result of over a thousand commits of code and as a result is a ready to deploy, mature product.

Contents

List of Tables **i**

List of Figures **i**

1 Introduction **1**

1.1 Configuration 1
1.2 Interface 2
1.3 Executing commands 2
1.4 Broadcasting files 2
1.5 Convenience features 3

2 Requirement Analysis **4**

2.1 Purpose 4
2.2 Overall Description 4
2.2.1 Product Functions 5
2.2.2 Hardware Requirements 5
2.2.3 Software Requirements 5
2.3 Functional Requirements 5
2.4 Performance Requirements 6

3 Design And Implementation **7**

3.1 Overall Design 7
3.2 User Interfaces 7
3.3 Hardware Interfaces 7
3.4 Communications Interfaces 8
3.5 System Design 8
3.5.1 NetDog Server 8
3.5.2 NetDog client 9
3.5.3 Web Server 9
3.5.4 NetDog protocol 9
3.5.5 DataFlow Diagram 10
3.5.6 Screenshots 11

4 Coding **16**

5 Testing and Implementation **17**
5.1 Testing methods done for the project 17
5.2 Unit Testing 18
5.3 Integration Testing 18
5.4 System Testing 19

6 Results and Inferences **20**
6.1 Advantages and Limitations 20
6.1.1 Advantages 20
6.1.2 Limitations 20
6.2 Future Extensions 20

7 Conclusion **21**

REFERENCES **22**

List of Tables

5.1 Unit test cases and results 18
5.2 Unit test cases and results 18
5.3 Unit test cases and results 19

List of Figures

3.1 NetDog protocol message structure 9
3.2 Level 0 Data Flow 10
3.3 Level 1 Data Flow 11
3.4 First time run screen 11
3.5 First time run screen 12
3.6 Account creation 12
3.7 Login 13
3.8 NetDog Home Page 13
3.9 Executing commands 14
3.10 Selecting target clients 14
3.11 View client statistics 15

Chapter 1

Introduction

The Netdog project initially started as a simple tool to shutdown computers in a network. In the course of development, several features were added into it and it morphed into a easy to use configuration management system.

Configuration Management systems are a class of automation software which allows easy management of computers on a network. Imagine a network where a specific version of a software needs to be installed across all computers. They can be achieved through configuration management system.

One thing about configuration management systems is that, they are designed for computer professionals who are not afraid to get their hands dirty and spend considerable amount of time learning and debugging. This makes these tools not very accessible to non tech-savvy people. For example, Redhat's Ansible, a popular configuration management systems expects its users to write the configuration in YAML format. Not something every literate person in the world knows.

Netdog aims to differ here by offering dead simple interface which can be used to administer the computers on a network. Amazingly, if a user knows to use the keyboard and mouse and is aware of the command that is needed to be run across the machines, he is ready to use Netdog. No further training is needed. But equally as important, Netdog provides a number of additional features which makes management of computers on the network a breeze. The following sections take a more intimate look at Netdog and what it offers.

1.1 Configuration

Netdog provides automated installation and configuration. Make sure the clients and server can see each other on the network, install the client part on each client and install the server and you are good to go. This is in stark difference to normal configuration management systems which require a person to spend considerable amount of time figuring out how to get it up and running.

1.2 Interface

Netdog uses an easy to use web interface for users to manage the computers on the network. On the first run of Netdog server, the interface prompts the user to select the network interface which should be used by the server for communicating with the clients. The interface is protected by a username and password which can be chosen by the user at the time of first run.

Once logged in, the homepage is presented to the user. From here, Netdog allows the users to execute commands, Shutdown, Suspend, broadcast files and monitor all the computers on the network.

1.3 Executing commands

Netdog allows easy execution of commands across all the machines. The commands are executed on the client machine with super user permission. This means you can perform system level tasks easily such as installing and updating the packages on the client as well as make changes to system files.

When the user clicks on the Execute command button, he/she is taken first to a page where the command can be entered. Once that is done, he has to choose the clients on which the command would get executed. By default, all the clients are selected. The user is free to deselect the clients by unchecking check boxes right next to them.

1.4 Broadcasting files

A very useful feature provided by Netdog is the ability to broadcast files across clients. This too can be done from the home page. Clicking on the Broadcast button takens the user to the upload page where the source file can be selected. Once that is done, select the target clients and the file will get broadcasted to all of them. The broadcasted files will be in "\share" directory in the clients.

The interesting thing about broadcast feature is that, it can broadcast any file, no matter what type or how large. This makes it very convinient as documents, pictures, videos and even software packages can be broadcasted.

1.5 Convenience features

Netdog provides a couple of convenience features, which allows the users to shutdown and suspend the clients without having to type the commands for them. These works very similar to above features. The shutdown and suspend features have their own buttons in the home page. The user can simply click the button, select the target machines and the said systems will get affected.

Chapter 2

Requirement Analysis

2.1 Purpose

The purpose is to build a configuration management system that is very simple to use and at the same time providing enough features so that it is power enough and can distinguish itself from the rest of the applications in the market. A second aim was to integrate a monitoring system into the product so that the system administrators do not have to learn and configure a separate monitoring application such as Nagios.

Though feature rich configuration management systems exist in the market, all of them are plagued by learning curve. None of them can be used by pointing and clicking. Netdog aims to fix that and prove that you don't have to be a computer savvy person to manage computers on a network.

2.2 Overall Description

NetDog uses client-server architecture. The server provides the web interface to be used by the system administrator. It also provides interfaces for clients to connect to and exchange information. The server can push information to the clients and vice versa. Server usually pushes files and execution information while clients push out status information.

After installing NetDog server or client on a machine, a unique public-private key pair for the machine is generated which is then used for uniquely identifying the machine and securing data transmission between the client and server. The advantage of this method is that, even if the IP addresses of the client or server changes, they can identify each other and re-start functioning as if nothing happened.

Once the server program is up and running, it listens on the port 1337 for connections from clients. Once the client program is up, it starts listening on port 1994. These ports serve dual purpose of facilitating communication and allowing

the identification of server and clients from the rest of the machines on the network. Both NetDog server and client are daemons. They are system services which remain in memory and automatically starts during system boot. They start even before a user logs into the system, ie. Netdog remains fully functional as long as the computer is powered on.

When a client starts for the first time, it looks for active servers on the network. When it finds one, it starts the pairing procedure. During the pairing process, the client sends its hostname and public-key. The server in turn provides the client with it's public key. These keys are then used for identification and encrypted communication between machines.

2.2.1 Product Functions

- Execute commands/scripts remotely on machines
- Broadcast files to Netdog clients
- Track and identify clients through IP changes
- Secure client-server communication using public key encryption
- Monitor all the clients on the network
- Easy to use web interface

2.2.2 Hardware Requirements

- Intel Pentium IV or equivalent CPU
- 512 MB or more RAM
- 100 mbps Network Interface Card

2.2.3 Software Requirements

- Linux (preferably Debian GNU/Linux)
- Python 3.5

2.3 Functional Requirements

The system should be designed to accept communication requests from many clients at once. For this, a multi threaded server is necessary. Also, network outages can occur during the operation. The server should be resilient enough

so that, it checks the status of the connection every once in a while and restarts the communication process once the network is up and running again. Also, the functions should extensively log themselves so that in case of a system failure, the culprit can be easily found.

2.4 Performance Requirements

The system would need a gigabit ethernet controller for the server to make sure that it can properly handle connections from multiple clients on the network. The client can work satisfactorily well even on old hardware such as a 10 mbit network card. There are no specific requirements for the CPU or the rest of the hardware. The machine must be powerful enough to run a recent version of Linux, which means any usable computing hardware would do.

Chapter 3

Design And Implementation

3.1 Overall Design

NetDog follows the client-server architecture. What this means is that, NetDog has both a client part and a server part. The server part is meant to be run on the machine used by the system administrator, while the client part runs on the rest of the machines on the network.

Both the server and client listens for connections from each other, allowing the server to execute commands on clients and the client to report back any progress. But the server has an additional capability that the clients lack, an easy to use web interface that can be used by the administrator to control the machines and configure the server.

3.2 User Interfaces

NetDog is a very easy to use system. One of the main aims while designing the system was to abstract as much lower level details of the system as possible from the user. There is no user interface for the client machines. All of the system is controlled from the server. The server provides a command line interface and a very convinient web interface for administration. The whole system can be operated and configured from these interfaces.

3.3 Hardware Interfaces

The system tries hard not to reinvent the wheel. Thus, it uses existing Linux subsystems as much as possible. As a side effect, NetDog can work with any networking hardware supported by Linux. If the clients on the network can succesfully communicate over the network using their network interface cards, you are good to go.

3.4 Communications Interfaces

The system uses the TCP/IP system for communication between server and clients. On top of TCP, the system uses a home grown higher level ”NetDog” protocol for coordinating the client machine and servers. It is the NetDog protocol that ensures efficient communication and security. The protocol designed to evolve as per changing requirements and is still under heavy development, but the current specification is as follows.

```
{
      "encrypted": ENCRYPT_STATUS
      "sender_id": SENDER_HOSTNAME
      "message": {
           "hostname": SENDER_HOSTNAME
           "data": {
                "command": NETDOG_PROTOCOL_COMMAND
                "payload": RAW_DATA
           }
      }
}
```

3.5 System Design

On a lower level, the design of NetDog is a bit different. NetDog is splitted into three modules, the server module, the client module and the web server module. The three of them has distinct functions. But the interesting fact is, both the client module and the server module share the same underpinnings. All the functions they use are from a library called libgreen, that was built specifically for use in NetDog.

3.5.1 NetDog Server

The NetDog server module acts as the server. The server module resides in the file “netdog_server.py” It uses functions part of the libgreen library to setup a server working on the netdog protocol.

It listens for connections from clients on port 1337 and also provides the web interface. The server is completeley multithreaded and this allows it to perform tasks involving hundreds of machines at a surprisingly quick rate.

3.5.2 NetDog client

The NetDog client module is the part of NetDog that is run on the client machines. The client module listens for connections from the server at port 1994. Once a connection is received, it is handled as per the command it contains.

3.5.3 Web Server

The webserver module provides the web interface for use by the administrator. It resides in the file "web_interface.py".The web server also uses the libgreen library extensively, to serve its clients. It is not directly invoked from the file, rather the contents of "web_server.py" is imported into "netdog_server.py" and started as a separate thread. The web server was originally written in bottle, but has been completely re-written using Flask, giving it better performance and lot lesser bugs. The interface itself was originally designed using the Google Material Design Lite CSS library, but now has been redesigned using the lightweight Milligram library.

3.5.4 NetDog protocol

The NetDog client and server operates using the NetDog protocol. NetDog protocol is a custom built protocol, that has been developed on top of the Sockets library part of standard Python installation. The mechanism as well as the policies for the protocol is defined inside the libgreen library.

The NetDog protocol defines the structure of messages passed between the server and the clients. The message has primarily three parts:

- Hostname, which is the hostname of the sender
- encrypted flag, whether the message is encrypted or not
- data, which is further divided into command and payload

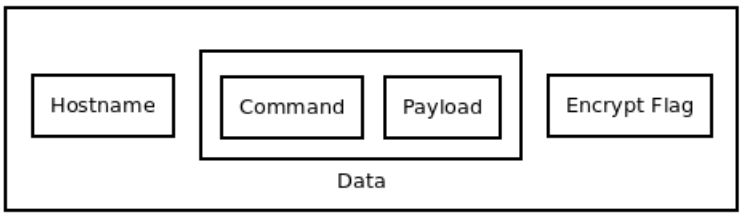

Figure 3.1: NetDog protocol message structure

3.5.5 DataFlow Diagram

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 without going into great detail, 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, how the data will advance through the system, and where the data will be stored. It does not show information about process timing or whether processes will operate in sequence or in parallel, unlike a traditional structured flowchart which focuses on control flow, or a UML activity workflow diagram, which presents both control and data flows as a unified model.

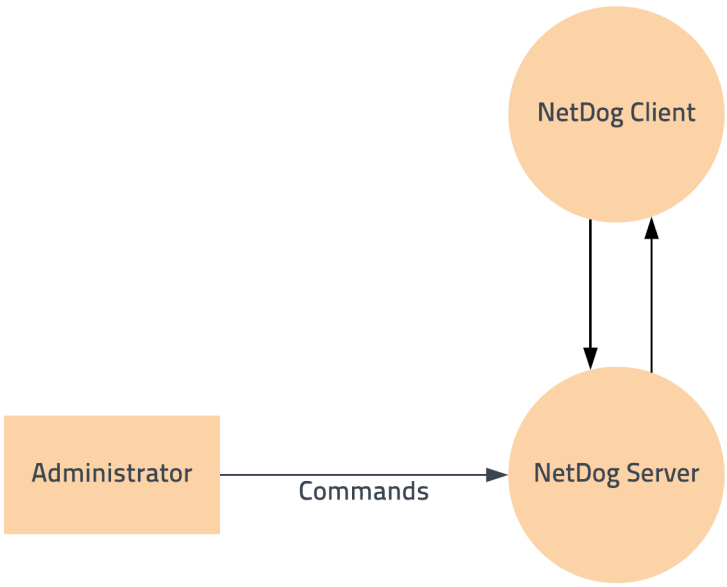

Figure 3.2: Level 0 Data Flow

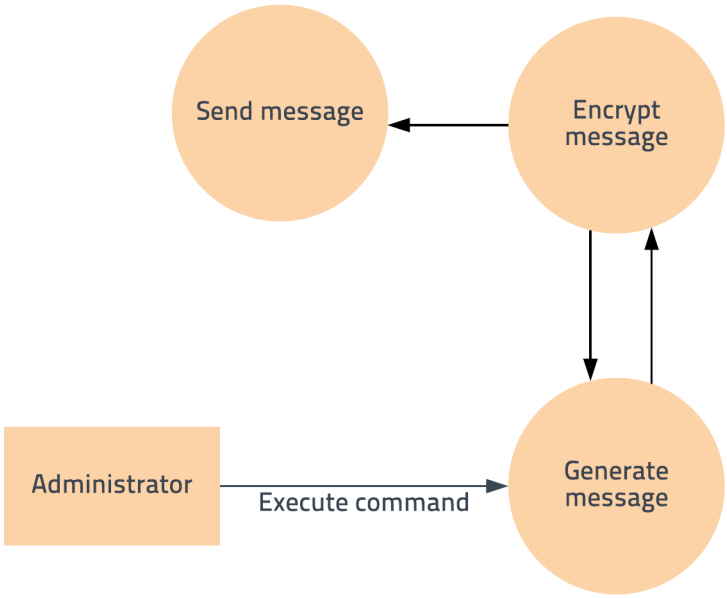

Figure 3.3: Level 1 Data Flow

3.5.6 Screenshots

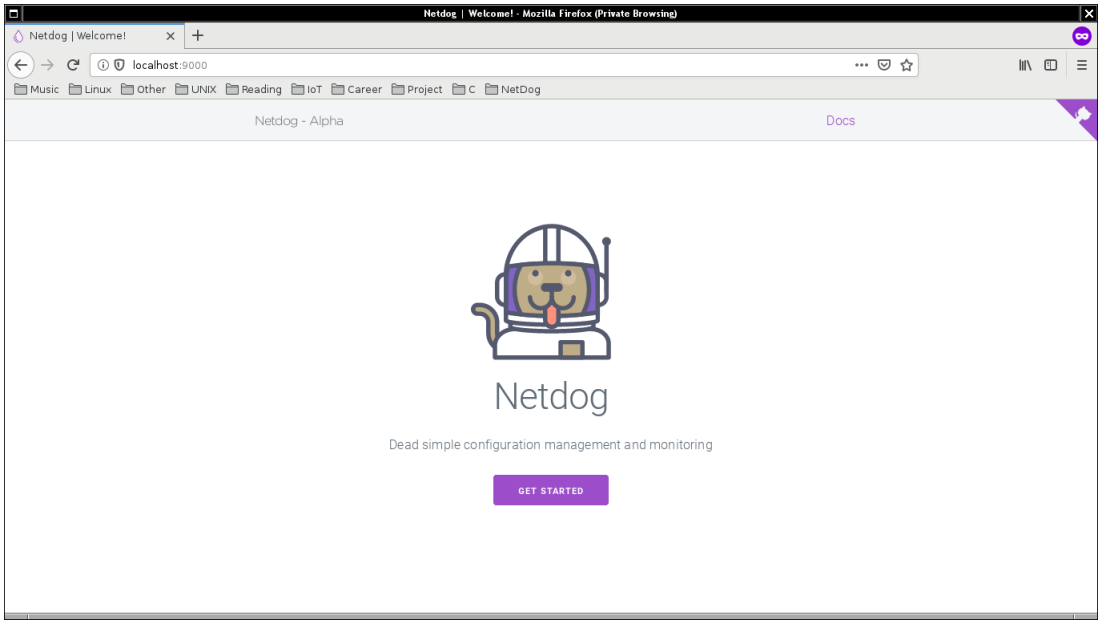

Figure 3.4: First time run screen

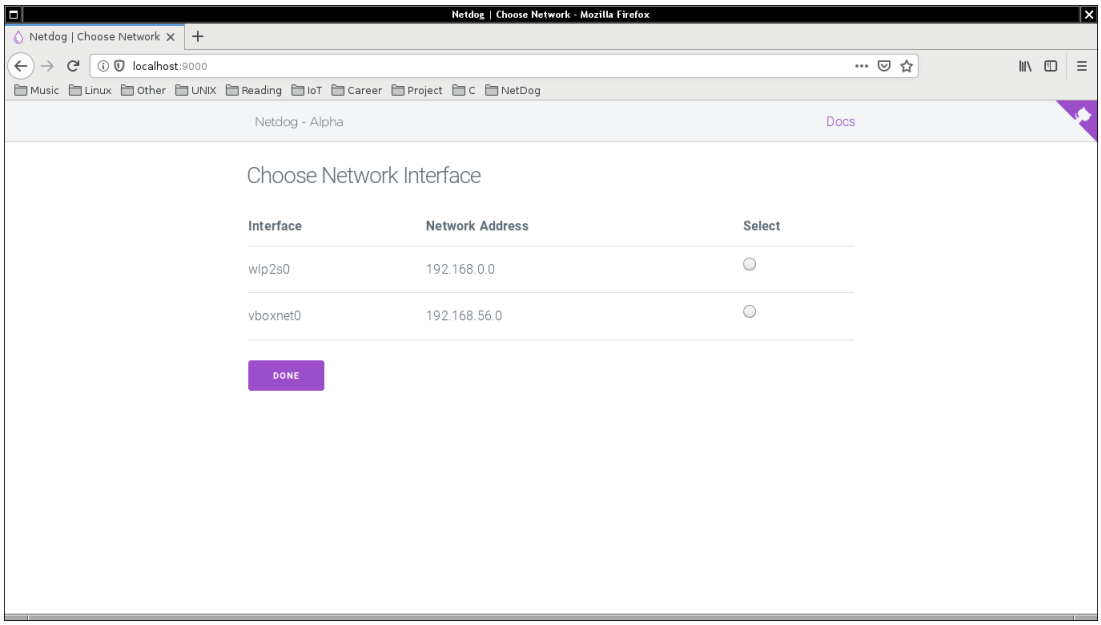

Figure 3.5: First time run screen

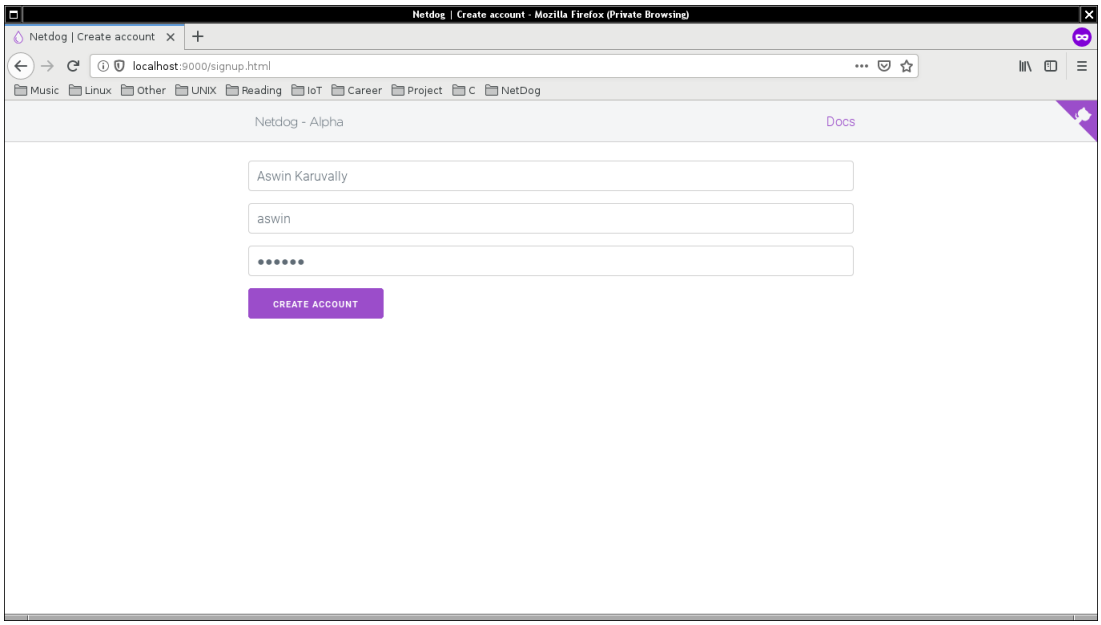

Figure 3.6: Account creation

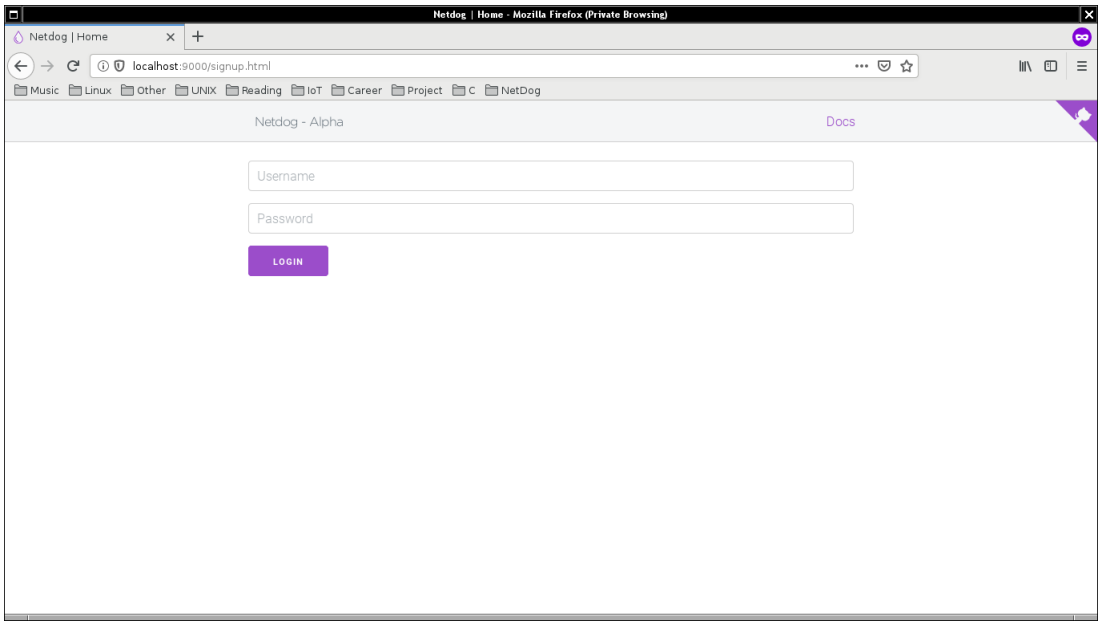

Figure 3.7: Login

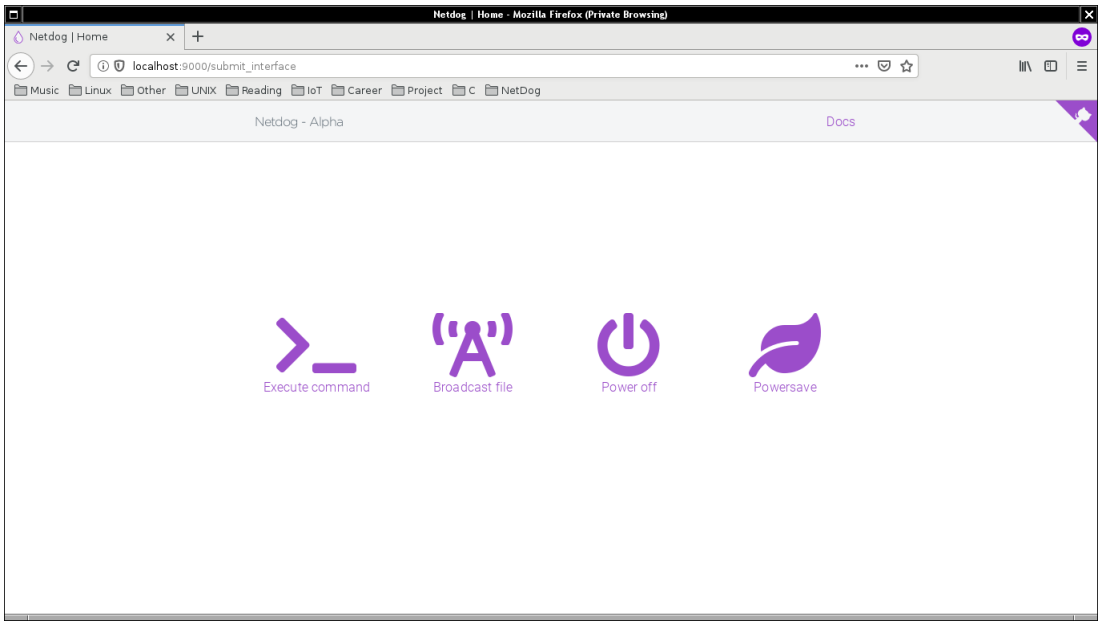

Figure 3.8: NetDog Home Page

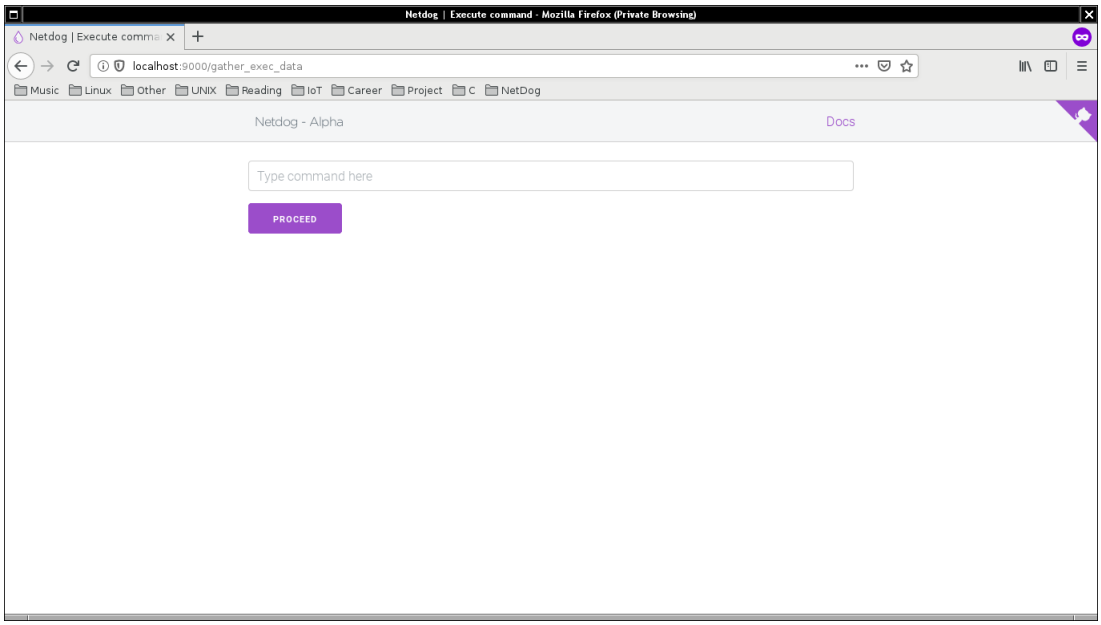

Figure 3.9: Executing commands

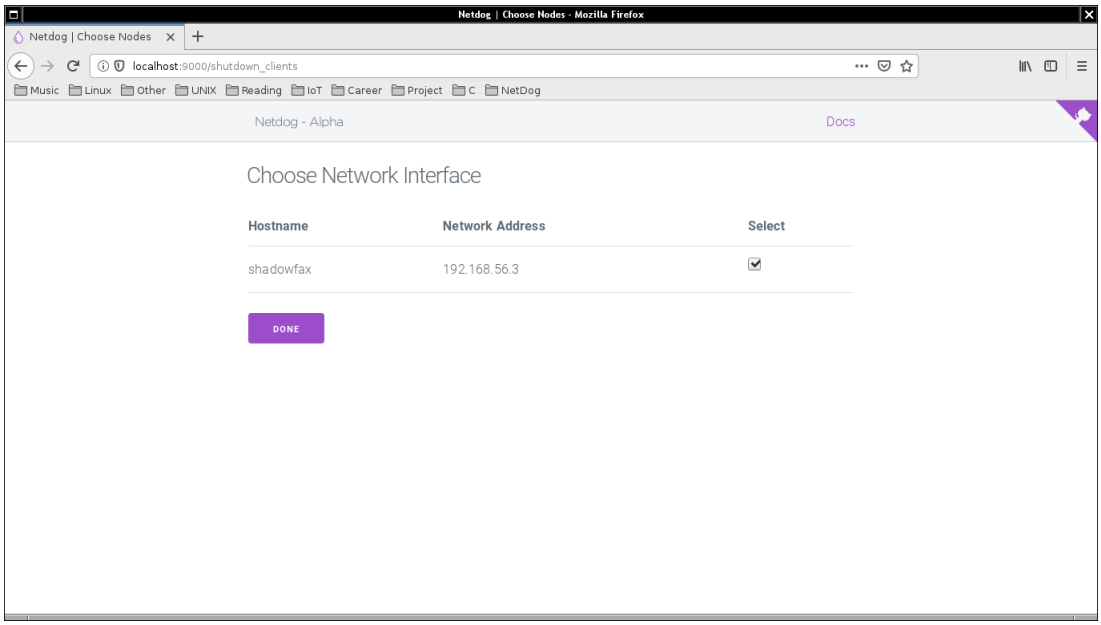

Figure 3.10: Selecting target clients

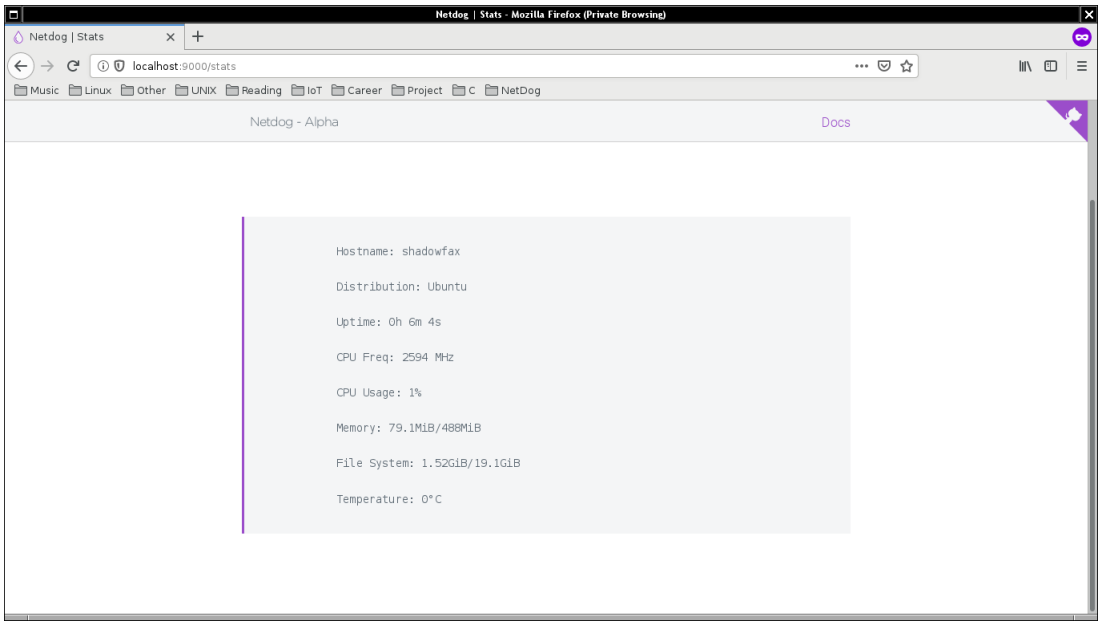

Figure 3.11: View client statistics

Chapter 4

Coding

Algorithm 1 NetDog Server

1: Initialize the system
2: Setup Network with Server value as True
3: Start the server thread
4: Start web server process

Algorithm 2 NetDog Client

1: Initialize the system
2: Start the server thread
3: Start automatic setup of network
4: Start beacon thread

Algorithm 3 NetDog Client

1: Initialize the system
2: Start the server thread
3: Setup the network

Chapter 5

Testing and Implementation

5.1 Testing methods done for the project

System testing is the stage of implementation which is aimed at ensuring that the system works accurately and efficiently before live operation commences. Testing is the process of executing the program with the intent of finding errors and missing operations and also complete verification to determine whether the objective are met and the user requirements are satisfied.

The ultimate aim is quality assurance. Tests are carried and the results are compared with the expected document. In that case of erroneous results,debugging is done. Using detailed testing strategies a test plan is carried out on each module. The test plan defines the unit,integration and system testing approach. The test scope includes the following: A primary objective of testing application systems is to assure that the system meets the full functional requirements, including quality requirements(Non functional requirements).

At the end of the project development cycle, the user should find that the project has met or exceeded all of their expectations as detailed in requirements. Any changes, additions or deletions to the requirements document, functional specification or design specification will be documented and tested at the highest level of quality allowed within the remaining time of the iproject and within the ability of the test team.

The secondary objective of testing application systems will be do:identify and expose all issues and associated risks, communicate all known issues are addressed in an appropriate matter before release. This test approach document describes the appropriate strategies, process, work flows and methodologies used to plan, organize, execute and manage testing of software project ”Real-time Traffic Congestion Analyzer for Road Safety”

5.2 Unit Testing

Sl No	Procedures	Expected result	Actual result	Pass or Fail
1	Generate signature	Signature generated	Same as expected	Pass
2	Verify signature	Signature verified	Same as expected	Pass
3	Encrypt message	Message encrypted	Same as expected	Pass
4	Decrypt message	message decrypted	Same as expected	Pass
5	Read configuration	Configuration read	Same as expected	pass

Table 5.1: Unit test cases and results

5.3 Integration Testing

Sl No	Procedures	Expected result	Actual result	Pass or Fail
1	Client-Server pairing	Pairing works	Same as expected	Pass
2	Web server start	Web server started	Same as expected	Pass
3	Server send commands	Commands sent	Same as expected	Pass
3	Client receives commands	Command received	Same as expected	Pass

Table 5.2: Unit test cases and results

5.4 System Testing

Sl No	Procedures	Expected result	Actual result	Pass or Fail
1	Server startup	Server starts	Same as expected	Pass
2	Client startup	Client starts	Same as expected	Pass

Table 5.3: Unit test cases and results

Chapter 6

Results and Inferences

Netdog has been developed through test driven development and has been proven to work properly in alll types of network configurations. The different modules operate as separate processes and functions spawning multiple threads increase parallellism and further enhances the performance of the system.

6.1 Advantages and Limitations

The proposed system consists of several advantages over existing systems, they have been explained in detail below.

6.1.1 Advantages

- Virtually zero configuration during installation
- Easy to use web interface
- Ability to broadcast files
- Ability to monitor health of clients
- Easy execution of commands across clients

6.1.2 Limitations

- Limited configurability of the system
- System performance is unknown on really large networks (>100 nodes)

6.2 Future Extensions

Netdog has been written in standards compliant Python 3. All the important functions are defined in the libgreen library. What this means is that, the main program is mostly devoid of function definitions and extending the application is really easy. Further, the whole application is available on Github and has been released under MIT license. This means any interested person can suggest, implement and improve the existing code. The possibilities are endless.

Chapter 7

Conclusion

NetDog is a configuration management and monitoring system, that has been developed with naive users in mind. The server and clients will automatically find and pair with each other over the network without the intervention of the users.

The system provides considerable decrease in the learning curve compared to existing configuration management systems such as Ansible, Puppet and provides zero configuration monitoring compared to monitoring tools such as Nagios which require initial setup.

The system is released under MIT license so that anyone can modify, adapt or even sell the source part as part of their product. Anyone can contribute to Netdog by forking the Netdog source code at https://github.com/karuvally/netdog.git

.

REFERENCES

1. **Python Software Foundation**,(2019), *Socket Programming HOWTO*, [online]: https://docs.python.org/3/howto/sockets.html

2. **Python Software Foundation**,(2019), *Low level networking interface*, [online]: https://docs.python.org/3/howto/sockets.html

3. **Darsey Litzenberger**,(2018), *PyCrypto - The Python Cryptography Toolkit*, [online]: https://www.dlitz.net/software/pycrypto

4. **Pallets Team**,(2010), *Flask - Quickstart*, [online]: http://flask.pocoo.org/docs/1.0/quickstart

5. **C J Patoilo**,(2017), *Milligram - CSS Framework*, [online]: https://milligram.io

6. **Python Software Foundation**,(2019), *Subprocess Managmement*, [online]: https://docs.python.org/3.5/library/subprocess.html

7. **Sybren A. Stüvel**,(2011), *Python-RSA*, [online]: https://stuvel.eu/files/python-rsa-doc/index.html

8. **Donald Stufft Et al.**,(2013), *Public Key Encryption - PyNaCl*, [online]: https://pynacl.readthedocs.io/en/stable/public

9. **Mariia Yakimova**,(2017), *Guide to async programming in Python*, [online]: https://medium.freecodecamp.org/a-guide-to-asynchronous-programming-in-python-with-asyncio-232e2afa44f6)

10. **Doug Hellman**,(2018), *Multiprocessing Basics*, [online]: https://pymotw.com/3/multiprocessing/basics.html

11. **Helder Ejis**,(2017), *Crypto.Signautre package guide*, [online]: https://pycryptodome.readthedocs.io/en/latest/src/signature/signature.html

12. **Legrandin**,(2019), *Pycryptodome documentation*, [online]: https://pycryptodome.readthedocs.io/en/latest/index.html