SOFTWARE REQUIREMENTS SPECIFICATION

for

NetDog

Version 1.0

Prepared by Aswin Babu K

College of Engineering Trivandrum

September 4, 2018

Contents

Revision History

Name	Date	Reason For Changes	Version
Initial Release	04-09-18	Initial release of document	1.0

1 Introduction

1.1 Purpose

The purpose of this document is to describe in detail, the requirements for the "NetDog" Project. The document explains various features of the system and its requirements. The document is intented for the end user to determine if the software covers the required features and for the development team for implementing the first version of the system. This document is a work in progress and many sections are yet to be added.

1.2 Project Scope

The NetDog project aims to make it easy for administrators to manage local PCs on a network. It allows remote powering up and down of PCs without worrying about the IP address of client machines. The system natively supports encrypted transfer of files over the network without third party protocols. Futher, the system will be completely pluggable, allowing administrators to easily extend the system by using third party extensions or writing their own.

1.3 References

The following are the main off the shelf components that has been used to implement the project. When trying to extend the system or fixing up the existing bugs, referring to the documentation of these libraies might turn out to be essential. Click on the link to direct the browser to each project's homepage

Python 3 language
Sockets Library
PyCrypto Library
Python Standard logging
Netifaces library
Bottle Web framework

1.4 Licence Agreement

NetDog is licensed under the MIT License. A copy is attached below

MIT License

2 Overall Description

2.1 Product Perspective

The netdog project started with the aim of designing a program which can easily bring computers up and down remotely. The idea came from the realization that, quite a few computers were left powered up when the college lab closes for the day. Thus the initial name "Project Green".

NetDog has a client server architecture. The server is responsible for issuing commands to the clients and is to be used by the administrator of the network. The client application is to be run on machines on the network to be administered.

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.

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.

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.

The client and server uses public key encryption to identify and secure the communication between them. The network admininstrator can issue commands from the server machine which will then be sent to all the clients on the network. A client can also contact the server occasionally, for example if the client detects undseriable network traffic or if the system is overheating.

NetDog is capable of shutting down all the clients on the network at once by remotely executing the shutdown command. It is also able to power up systems which support remote Wake-On-LAN feature, by sending magic packets.

2.2 Product Functions

The initial release of NetDog will have the following features

- Bring all computers on the network up and down remotely
- Execute commands/scripts remotely on machines
- Copy files to remote machines without third party protocols
- Track and identify clients through IP changes
- Secure client-server communication using public key encryption
- Alert if daemon on a client is not running
- Centralized logging of all data regarding clients
- Track power-on power-off times and user login history
- command line and web interface
- Plugin interface for extending features

2.3 Operating Environment

NetDog requires that the server and clients be running on a recent version of Linux. Support for other Operating Systems maybe added in the future. The system is written on Python 3 and thus requires the standard Python package for it to run. Most Linux distributions ships it with default installation. In case Python is not found, the installer is capable of automatically pulling the latest version of Python available from the distribution's repoistories.

2.4 Assumptions and Dependencies

The system will not work properly on Python 2 without extensive modification. The system might have trouble working, if your Linux distribution is considerably old and you have an old release of Python 3 (eg. 3.1/3.2)

3 External Interface Requirements

3.1 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.

3.2 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.

3.3 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" protocl for coordinating the client machine and servers. It is the NetDog protocol that ensures efficient communication and security.

3.4 Functional Requirements

The system is 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.

3.5 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.