BRNO UNIVERSITY OF TECHNOLOGY

Faculty of Electrical Engineering
and Communication

SEMESTRAL THESIS

Brno, 2018 Bc. Martin Kačmarčík

BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
FAKULTA ELEKTROTECHNIKY
A KOMUNIKAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF TELECOMMUNICATIONS
ÚSTAV TELEKOMUNIKACÍ

APPLICATION FOR MONITORING OF LINUX SERVERS
APLIKACE PRO MONITOROVÁNÍ SERVERŮ S OPERAČNÍM SYSTÉMEM LINUX

SEMESTRAL THESIS
SEMESTRÁLNÍ PRÁCE

AUTHOR — **Bc. Martin Kačmarčík**
AUTOR PRÁCE

SUPERVISOR — **doc. Ing. Dan Komosný, Ph.D.**
VEDOUCÍ PRÁCE

BRNO 2018

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Semestrální práce

magisterský navazující studijní obor **Telekomunikační a informační technika**
Ústav telekomunikací

Student: Bc. Martin Kačmarčík ***ID:*** 165394
Ročník: 2 ***Akademický rok:*** 2018/19

NÁZEV TÉMATU:

Aplikace pro monitorování serverů s operačním systémem Linux

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se s aplikací vyvíjenou na Ústavu telekomunikací pro vzdálenou práci se servery sítě PlanetLab (www.planet-lab.eu). Tato aplikace je dostupná na adrese pypi.org/project/plbmng/. V rámci semestrálního projektu aplikaci převeďte do jazyka Python 3. Dále proveďte její aktualizaci na repositáři PyPI. V rámci diplomové práce aplikaci rozšiřte o možnost vyhledávaní serverů podle jejich aktuálního stavu činnosti. Vytvořený kód vystavte pod licencí MIT a umístěte jej na repositář PyPI. Aktualizujte popis aplikace v anglickém jazyce.

DOPORUČENÁ LITERATURA:

[1] Linux Dokumentační projekt. 4. vyd. Computer Press, 2008. 1336 s. ISBN: 978-80-251-1525-1.

[2] PILGRIM, M. Ponořme se do Python(u) 3. CZ.NIC, 2010. 435 s. ISBN: 978-80-904248-2-1.

Termín zadání: 1.10.2018 ***Termín odevzdání:*** 14.12.2018

Vedoucí práce: doc. Ing. Dan Komosný, Ph.D.
Konzultant:

prof. Ing. Jiří Mišurec, CSc.
předseda oborové rady

Contents

Introduction

Task of developing a network project can become a challenging task. Internet is a huge worldwide network and to properly simulate the usage and architecture of the internet requires at least several servers on different locations at best. PlanetLab Network offers a global research network that enables development of new network services. The goal of this semestral is to improve existing tool, make it easier to use and publish the changes by updating the PyPi repositories. PlanetLab Server Manager is an existing tool that allows users to get information about nodes in the PlanetLab network and creates an user interface that helps interact with them. The current state of the application, which will be described later, can be a barrier for more extensive usage of the application and community driven improvements. Semestral thesis aims to re-write the application into Python; a popular community supported multi-platform object oriented programming language [11]. This thesis extends existing tools developed by Ivan Andrašov [3] and Filip Šuba [31].

The approach to achieve the goals of this thesis is to take existing Bash functions and re-write them to Python 3. During this process each functions is evaluated whether the used implementation is correct or not. To achieve easier usage of the application, main focus is applied onto removing system package dependencies and scrapping the necessity to localize the installation folder. To achieve better readability improvements to the implementation of functions are added by using best coding practices. Special emphasis is laid on logical structure and good programming practices to empower later community improvements to the tool.

Since this thesis uses already existing tool created by previous students, in Chapter 2 the tool and summary of previous work is reviewed. In the Chapter 1 the PlanetLab project will be introduced and characterized. As Linux is the main operating system nodes uses, in the Chapter 3 it will be described along with virtualization as it is the technology used for provisioning the PlanetLab nodes [17]. In the Chapter 5 the improvements made to the Plbmng tool will be explained.

1 PlanetLab Network

PlanetLab is a global research network that enables the development of new network services. According to the PlanetLab project main page it was used by more than 1000 researches at top academic institutions and industrial research labs to develop a new technologies for distributed storage, network mapping, peer-to-peer systems, distributed hash tables, and query processing since it launch at 2003 [18]. The main description also states that PlanetLab currently consists of 1353 nodes at 717 sites [1]. The tool's internal database created from PlanetLab API (Application Programming Interface) consists of 1001 nodes where; as shown in Figure 1.1; only 196 are responding to ICMP (Internet Control Message Protocol) packets and 805 are not, which is only around 19.58% reponding nodes. That means 1.62% nodes stopped responding since the last measurement done by Filip Šuba in his thesis [31]. The current committee of the project consists of members like Princeton University, Cambridge University, Intel, Google and many more [18].

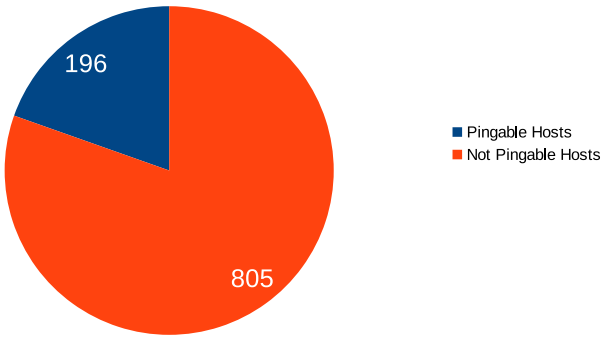

Fig. 1.1: Pie chart displaying number of nodes in PlanetLab network responding to ICMP packets.

1.1 Terminology

During the initial planning of `PlanetLab network` the authors agreed on using common terminology for aspects of the network and defined them in the `Phase 0 document` [1] as follows:

- **Node:** A server machine capable of running components of PlanetLab services.

[1] Important aspect to mention is that not all nodes are accessible. The `plbmng` tool can monitor accessibility of the nodes so its users have always overview which nodes can be actually used for their projects.

- **Site:** A physical geographical location where PlanetLab nodes are located.
- **Cluster:** The set of PlanetLab nodes located at a given site.
- **User:** An authorized human being wishing to deploy or run service over PlanetLab network.
- **Client:** A client of a service running over PlanetLab network.
- **Service:** An application running over PlanetLab network.
- **Application:** A PlanetLab service not being part of PlanetLab infrastructure.
- **Capsule:** A component of a PlanetLab service that runs on a single node.
- **Slice:** A distributed set of resources allocated to a service in PlanetLab.

1.2 Selected Projects Based on PlanetLab

In this section we will shortly describe various projects that PlanetLab network enabled to create. All these projects wouldn't be possible without the resources PlanetLab brings. On PlanetLab site there is partial bibliography of research enabled by PlanetLab and it consist of over two hundred projects [18]. Having over two hundred projects enabled by PlanetLab network shows that PlanetLab had succeeded in their initial goals which was to provide a useful platform for networking and system research [1]. Example of projects enabled by PlanetLab are described in the following subsections.

1.2.1 Securing Web Service by Automatic Robot Detection

This project is focusing on detection of automatic robots by implementing a special form of Touring test. Detection is done by comparing human versus robot behavior on the websites. According to the authors, 95% of the human users can be detected within the first 57 requests [15].

1.2.2 The Design and Implementation of Next Generation Name Service for the Internet

Project that is aiming to solve the vulnerability of the current DNS (Domain Name System) and slow delivery of updates to the system. Project paper describes design and implementation of the Cooperative Domain Name System (CoDoNS), a novel name service, which provides high lookup performance through proactive caching, resilience to denial of service attacks through automatic load-balancing, and fast propagation of updates [19].

1.2.3 Slurpie: a cooperative bulk data transfer protocol

Big data transfers can become problematic during peaks when huge amount of clients starts downloading the data at one point. This can occur for example during a launch of a new game or a new operating system. Slurpie is is a a peer-to-peer protocol for bulk data transfer that aims to reduce client download times of large popular files and to reduce load on the providing servers [26].

2 Present State Of Application Development

Plbmng application called `Data miner for PlanetLab` is available at public PyPi repository[1]. The tool allows managing `PlanetLab` nodes, gathering information about them and pulling the latest data from the `PlanetLab` API service. Its core is written in Bash and additional modules are written in Python 3 [31]. At the moment, it is depended on both Bash and Python modules and its installation consists of several steps:

- Installing the application from PyPi repository or downloading the source codes from GitHub.
- Installing additional system packages like dialog,pssh and fping.
- Locating installation folder and putting symlink into `$PATH` directory.

2.1 Description of Current Tool

First menu option is `Search nodes` for retrieving a node from internal database. This options allows user to either search by DNS (Domain Name System), IP (Internet Protocol) address or by node location. Second option is `Measure Menu` that allows user to schedule gathering of data about the nodes using `crontab`, select elements to monitor or start the data gathering right now. In the `Map Menu` option user has option to generate map showing location of the nodes and select map element. After the first start of the application user is required to fill credentials and `SSH` public key details to be able to access `PlanetLab` API and nodes using the menu option `Settings`. Menu is created using bash library `dialog` and can be seen in Figure 2.1 and can be run directly from terminal making it available even through `ssh` client without setting up any graphical tools.

[1]Link to PyPi repistory containing Data miner for PlanetLab tool: https://pypi.org/project/plbmng/

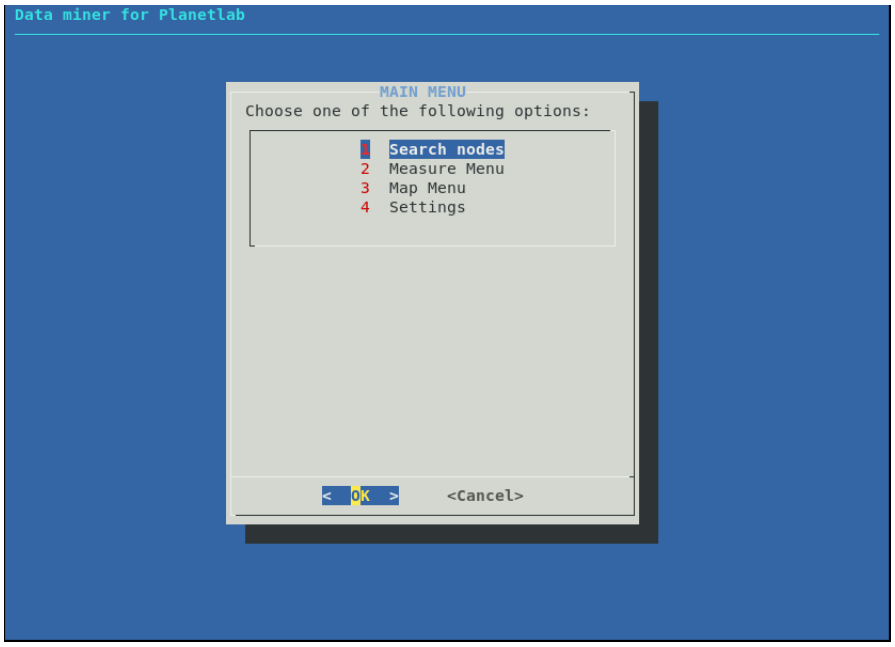

Fig. 2.1: Data miner for PlanetLab menu.

2.2 Current Problems

The first problem of the existing tool is language disparity having half of the functionality in Bash and half of the functionality in Python 3. This makes it difficult to make adjustment to the tool as one needs to study a vast amount of scripts that are in several different folders. Since some of the functionality is done in Python 3, which is according to portal StackOverflow fastest-growing major programming language [27], and because it is available at **zkPYPI!** (**zkPYPI!**) Python is an ideal candidate as a main language of the project. As a part of the semestral thesis the existing code will be re-written into Python 3. Another great advantage of Python 3 is that it is multi-platform. As Mark Pilgrim mentions in his book [16], Python 3 is available on many platform such as `Windows`, `MacOS`, `Linux`, `BSD` and `Solaris` and their derivatives.

Second area of improvement is installation of the tool and post-installation steps. At the moment, it is required to install additional packages and tool is not automatically put into `$PATH` folders forcing its users to locate the installation folder and run the script from there. Because of the single programming language being `Python 3` the dependencies for system packages will be removed and their `Python 3` counter-parts will be added as dependency for the PyPi package. Pypi installer takes care of these dependencies automatically during the installation procedure. To remove post-installation steps the tool will be written as library allowing us to

create an easy `Python` script in `bin` folder which is put into `$PATH` folders by the PyPi installer during the installation.
Another possible improvement is renaming certain menu components and adding more information to the tool itself. This change is not significant and is purely cosmetic but can make it easier for new users to get familiar with the tool. The specific rename details will be later described in Subsection 5.1.3.
The tool currently contains a lot of bugs and bad coding practices. Example of bugs is whole application crashing because of missing file when returning back from `Search nodes` menu. During the rewriting into `Python 3` there is space to improve certain controls to avoid these crashes and needs to restart the application. As for bad code practices, as an example the tool currently calls functions recursively during returning from child menu page into parent one. This means the previous function menu is stored in the `stack` waiting for the application to end before released. During rewriting of the tool these implementation details can be changed to stick with the good coding practices.

3 Linux

In this section the operating system Linux that PlanetLab nodes are running on, and that `plbmng` tool is developed for, will be reviewed and described. Operating system is a connecting layer between hardware and software. It provides interface to work with system resources such as disk, processor or memory and at the same time it provides service layer for client software to run at. Linux is an open-source operating system founded by Linus Torvalds who wrote its kernel using `C language` and began the history of Linux operating system. It was originally developed for personal computers based on the Intel x86 architecture but since its creation it has been ported to many other platforms such as mobile devices, television chips and many others. A package containing Linux operating system is called Linux distribution. The defining component of each distribution is the Linux kernel [6]. Original Linux kernel has been created by Linus in 1991 [12] and since them many other forks of this kernel has emerged. Some of the most famous are `Red Hat Enterprise Linux`, `CentOS`, `Fedora`, `Ubuntu`, `Debian` or `SUSE Linux`. More information about mentioned distributions can be found in at the end of this Chapter. The main advantages of Linux are:

- Almost all Linux distribtuins are free and available for everyone.
- Linux is open-sourced and everyone can contribute and review what his/her machine is running.
- Linux can run on various platforms from personal computers to televisions.
- Linux is considered to be secure. Security model of Linux is based on UNIX security principals which are considered to be robust and verified [2].
- Linux quickly adapt to changes. Since Linux has vast community behind it, it quickly adapt to security threads and new technologies.

On other hand, Linux has some disadvantages which will be summarized in the below list:

- Requires more technical knowledge then other systems like `Windows` or `MacOS`.
- Huge amount of distributions can be confusing for new users to choose.
- Many well known applications are primarily developed for `Windows` or `MacOS` (though many open source variants of these applications exists on Linux).
- Compatibility problems. Proprietary hardware can have issues with driver compatibility. Many hardware vendors are primarily focusing on `Windows` or `MacOS`.

asd

Linux kernel can be found in majority of devices at the moment. According to StatCounter Global Stats, 41.63% of the machines are running on Linux kernel while 36.23% are running on Windows-based kernel as of September 2018 [28]. The

rise of Linux kernel is conditioned by raising Android popularity and in 2016 the StatCounter states Linux kernel based system had only 28.44% market share while Windows had 48.42% market share [28]. Linux can be found also in great amount of devices from phones, desktops, servers to television or cars. Server `opensource.com` states that more then half of all SmartTV devices runs on Linux [13]. Because Linux kernel is originally open-sourced, most of the devices worldwide runs on an open-sourced operating system which shows an interesting trend switching from proprietary software. In the following paragraphs, various popular Linux distributions will be described. Their key features will be reviewed and analyzed.

Red Hat Enterprise Linux Red Hat Enterprise Linux is a Linux based distribution developed by Red Hat Inc. In 1994 the first version of Red Hat Linux has been released [22]. Since it launch Red Hat became a popular enterprise Linux distribution and based on Red Hat's June 2017 statistic 100% of all airlines, telcos and commercial banks in Fortune 500 runs on their software [23]. Red Hat advertise the system being robust and stable which is its biggest advantage for the enterprise companies. Even though Red Hat is open-sourced, it is not free as it is impossible to run it without having active subscription. Red Hat uses its own packaging system called yum (Yellowdog Updater, Modified).

CentOS The CentOS (Community Enterprise Operating System) is a Linux distribution that has been originally forked from Red Hat Enterprise Linux version 2.1 and released under name `CentOS-2` in May, 2014 [14] by John Newbigin. Since then it became a community-driven project that offers and free alternative to the Red Hat Enterprise Linux. Currently, CentOS Project is part of Red Hat while declaring to stay independent of the development of the Red Hat's Linux. Due to the distribution being free and for enterprises many companies and academic institutions choose the CentOS as the primary operating system for their servers.

Fedora Fedora is a Linux distribution developed by the communOriginally created in 1993 by Ian Murdock, Debian is a free Linux kernel based Unix like operating system developed by The Debian Project and its community. Debian in oposite to Red Hat Linux based systems uses different packaging system called APT (Advanced Packaging Tool).ity and is sponsored by Red Hat. Currently, Fedora provides several types of system images such as Workstation, Server and Atomic. Fedora is distribution primarily used as a desktop operating system and shares many technologies with Red Hat Enterprise Linux and CentOS making packages compatible between these systems. Fedora is often considered to be somehow beta environment for changes before they make it to the more conservative Red Hat Enterprise Linux.

Debian Originally created in 1993 by Ian Murdock, Debian is a free Linux kernel based Unix like operating system developed by The Debian Project and its community. Debian in oposite to Red Hat Linux based systems uses different packaging system called APT (Advanced Packaging Tool). In February 2016 the repositories of Debian consisted of 50007 binary packages [29]. Debian is popular server distribution and in 2012 it was named by PCWorld magazine the most popular distribution for web servers [9].

Ubuntu Ubuntu is a free and open-source Linux distribution based on Debian operating system. It is being developed by Canonical and Ubuntu community. Currently Ubuntu offers three version of the system: Ubuntu Desktop for personal computers, Ubutnu Server for server and cloud and lastly Ubuntu Atomic created for IoT (Internet of Things). It is seen that Ubuntu is making the Linux available for broader audience by having its graphical user interface more similar to other operating system like `MacOS` than other Linux distributions. Ubuntu also achieved great success when Microsoft ported official Ubuntu binaries into its Windows system. Ubuntu also is the most popular Linux distribution based on Google Trends data [10].

4 Virtualization

Virtualization is an alternative to the traditional architecture. Comparison schema between virtualization and traditional architecture can be found in Figure 4.1. The traditional architecture consist of a hardware, operating system on the host and the applications running on it. The applications needs to be compatible with the operating system to be able to run on it. On the other hand, virtualization is a layer over the hosting operating system and offers an interface for the guest operating system to use by emulating various resources such as disk, network usb and many more. Guest operating system are operating systems of the virtual hosts running on the hypervisor. Hypervisor is providing resources to each virtual host. This enables better resource handling since all the virtual machines share same resources and theoretically have access to much higher computing power. Another advantage is backward compatibility as various operating systems of a kind can be running one one host. Example is legacy version of a system that is no more compatible with current hardware but is compatible with the virtualization software. This can be particular useful for institutions that requires extremely stable well tested software running on legacy system. Next feature is high availability. If a virtual host crashes, many virtualization software are capable to migrate the run-time data to another virtual host live time. This can be crucial feature for any critical applications. On bare-metal hosts this can be solved by using for example clustered hosts.

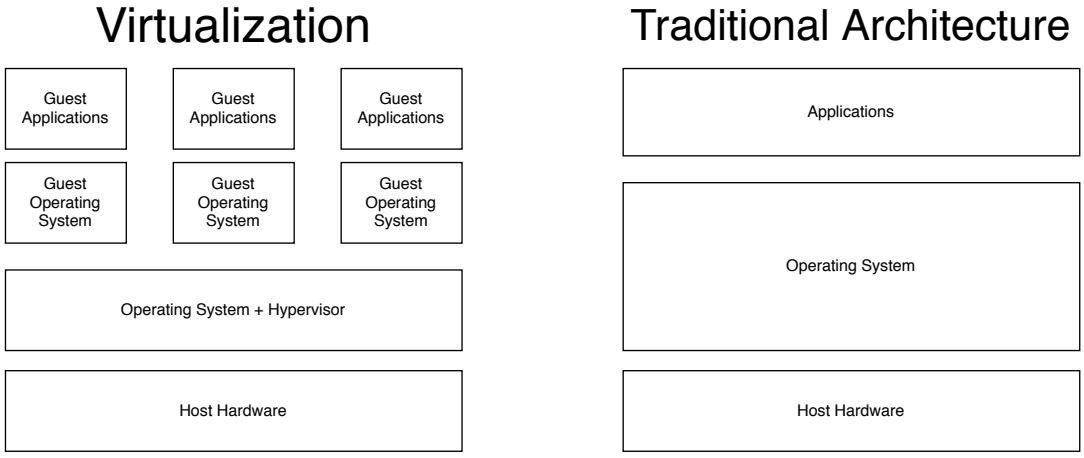

Fig. 4.1: Virtualization vs traditional architecture schema.

Currently there are many virtualization technologies available on the market. Some of the most popular ones are `VMware ESXi`, `Red Hat Enterprise Virtualization`, `Oracle Virtualbox` and `QEMU/KVM` [7]. In the following lines, some of the most popular Virtualization technologies are reviewed.

VMware ESXi VMware is a company providing virtualization software. The company provides several versions of virtualization software including `VMware ESXi` which is an enterprise virtualization solution. `VMware ESXi` was developed to run directly on the bare-metal hardware and doesn't require any other operating system to be installed on the host as it uses its own operating system called VMkernel [30]. This allows the software direct access to the hardware as it includes its own vital OS components. As VMware states in their architecture document [30], this also allows better hypervisor security, increased reliability, and simplified management.

Red Hat Enterprise Virtualization As Red Hat states on their website, `Red Hat Enterprise Virtualization` is an open, software-defined platform that virtualizes Linux and Microsoft Windows workloads [20]. `Red Hat Enterprise Virtualization` is a software that is developed to run on Red Hat Enterprise Linux, a Linux distribution developed by the same company. As Red Hat states in `Red Hat Enterprise Virtualization` datasheet, the biggest advantage of the software is its integration with other `Red Hat` products [21]. Red Hat also offers a complete operating system designed for creating virtual machines called `Red Hat Enterprise Virtualization Hypervisor`.

Oracle Virtualbox `Oracle Virtualbox` is open source virtualization software available for both enterprise as well as home use and is developed by Oracle Corporation. As stated on the Oracle Virtualbox website: "Presently, VirtualBox runs on Windows, Linux, Macintosh, and Solaris hosts and supports a large number of guest operating systems including but not limited to Windows (NT 4.0, 2000, XP, Server 2003, Vista, Windows 7, Windows 8, Windows 10), DOS/Windows 3.x, Linux (2.4, 2.6, 3.x and 4.x), Solaris and OpenSolaris, OS/2, and OpenBSD." [5] which is a considerable availability. `Oracle Virtualbox` offers many features, which are more described in the User Manual [4], such as:

- VBoxManage, a command-line interface for `Oracle Virtualbox`.
- No hardware virtualization required, supporting even older hardware without build-in virtualization support.
- Guest Additions, which is a software packages that offers features like folder sharing, automatic window focus, 3D virtualization and more.
- Multigeneration branched snapshots, which allows taking snapshot of the virtual host in any point of time completely saving the machine state allowing users to later revert machine to the saved state.

KVM with QEMU KVM (Kernel-based Virtual Machine) is a kernel module that provides a CPU (Central Processing Unit) virtualization through the use of Intel VT

or AMD-V hardware extensions. KVM runs in kernel space and handles elements like processor switching or MMU (Memory Management Unit) registers, which are used to handle the virtual machines. On the other hand, QEMU is a free open source emulator that performs hardware virtualization in the user space part of the operating system. On its own, QEMU provides also CPU emulation through binary translation however the best results are delivered when combined with KVM and it can deliver near native performance [25]. During measurement of virtual machine versus host performance using the KVM technology the Geekbench's GPU test shown only 1.19% score decrease over the bare-metal host [8].

5 Plbmng Tool Improvements

This Chapter will discuss improvements made to the PlanetLab application. The improvements are based on the analysis made in the Section 2.2. The implementation details will be described and shown. The goal of the re-implementation is to make the `plmbng` tool simple to use, easier to contribute into by re-writing it fully into Python 3, using good coding practices and remove any post-installations steps. First, in Section 5.1 the implementation details will be described, then in Section **??** the application behavioral diagram will be shown and explained.

5.1 Implementation Details

In this Section the steps to achieve goal described in this Chapter introduction will be shown more in detail in their own Subsections. For each Subsection the approach, specific steps, code examples and results will be illustrated. Since the new implementation uses `pythondialog` module, at the start of the tool an instance of the `Dialog` class is spawned and will be later described just as the `instance`.

5.1.1 Removing Result Limitation

The previous version of `plbmng` tool has been limited to 10 result when searching for a node. This issue was introduced due to difficulty of creating a menu based on dynamic results since author needed to always add a new argument to the overall command. This also can hit limitation of characters that can be passed in a Bash command line. In Python 3 this problem is non-existing since `pythondialog` module is creating menu functions based on list. During the search of the nodes, results are added to the list which is after completed search passed to the instance which renders the GUI (Graphical User Interface). Example of this functionality is shown in Listing 5.1. Currently, the tool is returning all results found.

Listing 5.1: Removing Result Limitation

```
def searchNodesGui(prepared_choices):
  if not prepared_choices:
    d.msgbox("No results found.", width=0,height=0)
    return None
  while True:
    code, tag = d.menu("These are the results:",
              choices=prepared_choices,
              title="Search results")
```

5.1.2 Writing the Application as Library

For the application to be used in other scripts and reduced the need to re-write certain code parts it is desired to write application to be able to run as a library. During the re-implementation this was considered and application is available both as library and standalone script. This will be later used in the Subsection 2.2.

5.1.3 Increasing Readability

Community is a powerful group that helps develop a tool and to add more functionality to it. To have community contribute to a tool, it should follow good practices and be easily readable. Previous version of the tool have been using main Bash script, calling Python script and creating new Bash scripts on a disk which was merging different pieces of code from pre-created `.dat` files in a `bin` folder. Finding a bug in this structure was difficult and non-intuitive. All these pieces of code were fully re-written into single Python library (which can be run as standalone script if needed) and logically divided into two sections. One section is for GUI functions and other is for logical functions. Each functions is very descriptive in its name as shown in Listing 5.2.

Listing 5.2: Example of Function Names

```
def searchNodes(option,regex=None):
def initInterface():
def plotServersOnMap(mode):
def getPasswd():
def searchNodesGui(prepared_choices):
def printServerInfo(chosenOne):
def setCredentialsGui():
```

Each functions is trying to be as atomic as possible only having one purpose. This is helping to increase modularity of the application. Outside of this functions "categories" application is removing any `magic numbers` by defining constants at the beginning of the source code. This greatly helps to understand what is being passed as an argument and is shown in Listing 5.3 where it is descriptive what option is being passed as a search key to the `searchNodes` function. Also, the application has a block for `Initial settings` at the beginning for one single place where outside of functions definitions can be placed. Application is also honoring the conventions defined in PEP (Python Enhancement Proposal) 8 [24], like naming convention and space usage instead of tabs, as much as possible. All these small items described should increase the overall readability of the application for others

to quickly familiarize with it.

Listing 5.3: Example of Constant Usage

```
code, tag = d.menu("Choose one of the following options:"
    ,
            choices=[("1", "Serach by DNS"),
                     ("2", "Search by IP"),
                     ("3", "Search by location")],
                     title="ACCESS SERVERS")
if code == d.OK:
  #Search by DNS
  if(tag == "1"):
    code, answer = d.inputbox("Search for:",title="Search
       ")
    if code == d.OK:
      searchNodes(OPTION_DNS,answer)
    else:
      continue
```

As mentioned in the Section 2.2, renaming certain parts of the tool can improve the readability. Since the tool is not data mining rather than server manager, the tool is internally renamed from `Data miner or PlanetLab` into `PlanetLab Server Manager`. Version is added next to the name for users to see which they are running immediately. Another example is renaming `Search nodes` to `Access servers` since primary function of this menu item is to access the servers while search is just supporting it. The re-designed application can be seen in Figure 5.1.

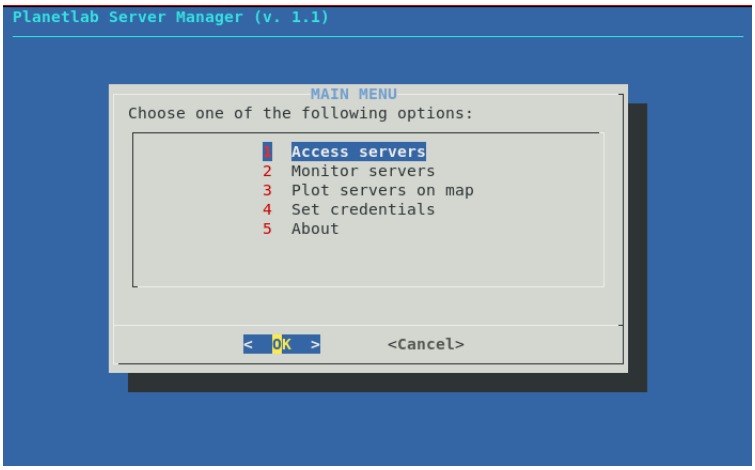

Fig. 5.1: New re-designed menu with added name, version and various name changes.

5.1.4 Removal of Pre and Post Installation Steps

Previous version of application required several Pre and Post installation steps. In the new version developed as part of this Semestral thesis, all these steps were removed. Pre-installation steps were eliminated by completely getting rid of dependencies on additional system packages. All the dependencies were moved into the PyPi package definition and are taken care off PyPi installer during installation of the tool. Post-installation steps were removed by adding the application into `bin` folder in the PyPi package. During installation, the PyPi installer automatically puts any scripts in the `bin` folder into a `$PATH` folder making it accessible directly from command line without the need of accessing installation folder. The contents of the script located in `bin` folder can be seen in Listing 5.4. The duplication of names are created by having the `plbmng.py` script in the `plbmng` folder describing the library. In these names there is definitely still an area for improvement.

Listing 5.4: Source Code of Plbmng Executable Script

```
#!/usr/bin/env python3
import plbmng.plbmng
import sys

if len(sys.argv) > 1:
  if(str(sys.argv[1]) == 'crontab'):
    plbmng.plbmng.crontabScript()
    exit(0)
plbmng.plbmng.initInterface()
```

5.1.5 Set Credentials Improvement

In the previous version of the tool the credentials were filled using forms. When typing the credentials, nothing was shown, like stars, so user was unaware where is the position of the cursor. Also saving of the credentials was not working properly resulting into the need to adjust the configuration file itself which required locating the file first by inspecting the source code. In the new version settings credential is improved by creating a virtual editor in the graphical interface itself, as shown in Figure 5.2, that allows the user transparently set the credentials. One of the disadvantage of this approach is plain text visible password in the editor as it is not hidden and user needs to be careful about setting the credentials in a safe environment.

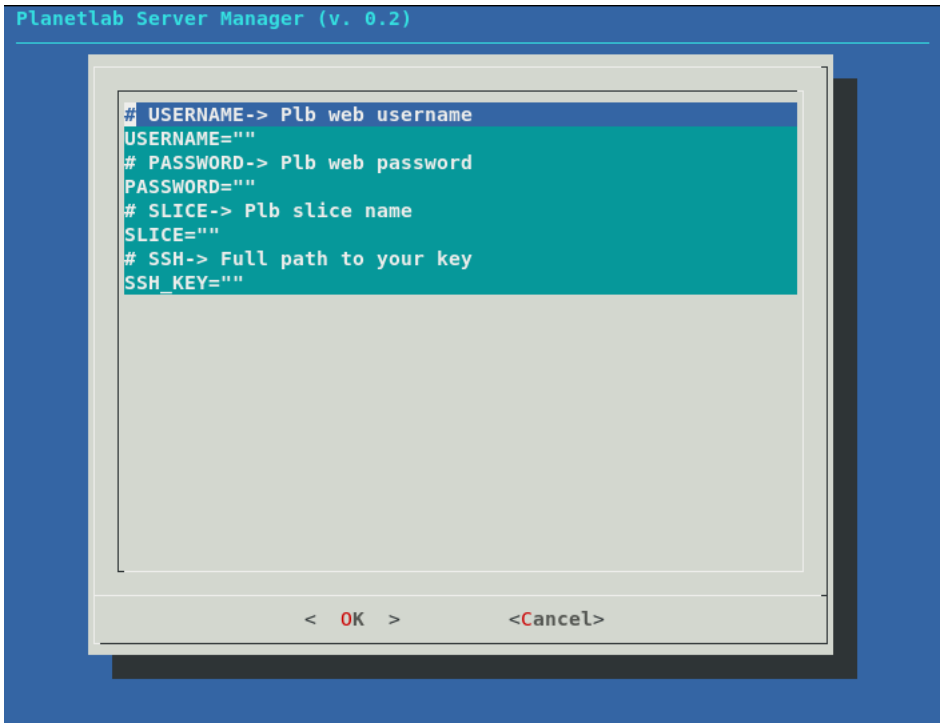

Fig. 5.2: New window for setting up credentials.

5.1.6 Windows Support Preparation

As was mentioned Python is multi-platform language and brings possibility of porting the application also to other operating systems. During the re-implementation of the application this was taken into considerations and new functions were written to be able to run on both Linux and Windows. Example of this dual implementation can be seen in Listing 5.5 showing function `testPing` supporting both mentioned operating systems. There is still a lot of functions to be re-written in this dual approach to fully support the Windows however this is the first step to achieve this compatibility.

Listing 5.5: Function testPing

```
def testPing(target):
  if system().lower()=='windows':
    pingParam='-n'
  else:
    pingParam='-c'
  command = ['ping', pingParam, '1', target]
  p = subprocess.Popen(command, stdout = subprocess.PIPE)
  if system().lower()=='windows':
    avg=re.compile('Average = ([0-9]+)ms')
  else:
    avg=re.compile('min/avg/max/mdev = [0-9.]+/([0-9.]+)
       /[0-9.]+/[0-9.]+')
  avgStr=avg.findall(str(p.communicate()[0]))
  if(p.returncode != 0):
    return "N/A"
  p.kill()
  return avgStr[0]
```

5.1.7 Minor Bug Fixes

In this Subsection, minor bug fixes will be described which were not considered as enough improving to have their own Subsection.

Removing headers from the searches During the search, previous version of the script has also included header of the file containing information about nodes resulting into false search results. Header is now skipped and these false results are removed.

Application crashes during return When returning from a child menu window to a parent page, the application tend to crash on `grep` tool not being able to find file. During re-implementation this bug was fixed and returning now fully works.

5.1.8 Minor Improvements

In this Subsection, minor improvements done during the re-implementation are described. All these improvements were considered minor hence these are not having their own Subsection. More improvements to the tool will be done in the Diploma thesis following this Semestral thesis.

Clearing of the screen after cancel When signal was send to the application using `CTRL + C` key combination, the previous version of the application was not clearing the current terminal window and the GUI. To use the same terminal user was forced to clear the window manually. In the new version, when signal is send to the application, signal handler will catch it and clean after itself.

Recursion removal for return When returning from child window to a parent page, the previous version of application was recursively calling the GUI function. This is not following good coding habits as each recursive call means storing the previous function details into the system stack, unnecessarily filling it. In the new version, while cycle is used instead and returning from a function results into new iteration of the while cycle not storing anything onto the system stack.

About is added to the menu About section is added to the menu displaying version, authors and the license.

Crontab mode created Application is possible to run with `crontab` argument which will trigger just monitoring of the nodes. This is in particular useful when setting the `crontab` since the call can be simply `plbmng crontab`.

5.2 Behavioral Diagram

In this Section, behavioral diagram can be seen in Figure 5.3. The behavioral diagram describes current structure of the tool and its functions. For clarity the Figure is missing returning arrows however each node in the diagram is able to return to its parent. As seen in the Figure, first step is when menu is being initialized. From there user has option to either open `Access server menu`, `Monitor servers`, `Plot servers on map`, `Set credentials` or select `About`. `About` is a single window option that will show information about the software, its authors and license. `Set credentials` option will open an interactive editor where user can fill the credentials for PlanetLab network. Next option is `Plot servers on map` which offers user to choose which map elements should be rendered. After confirmation the map is generated and opened in the system default browser. `Monitor servers` option divides into three different options. First enables user to setup `crontab` to periodically scan for nodes, second option is for setting up monitoring elements and third triggers the scanning immediately. At last, `Access servers` allows users to access the nodes by searching for them either using DNS name, IP address or location. After the search key is inserted, the available results are shown. When specific node

is chosen, information about the node are displayed and user has options to connect to the node using ssh, Midnight Commander or show the node on map.

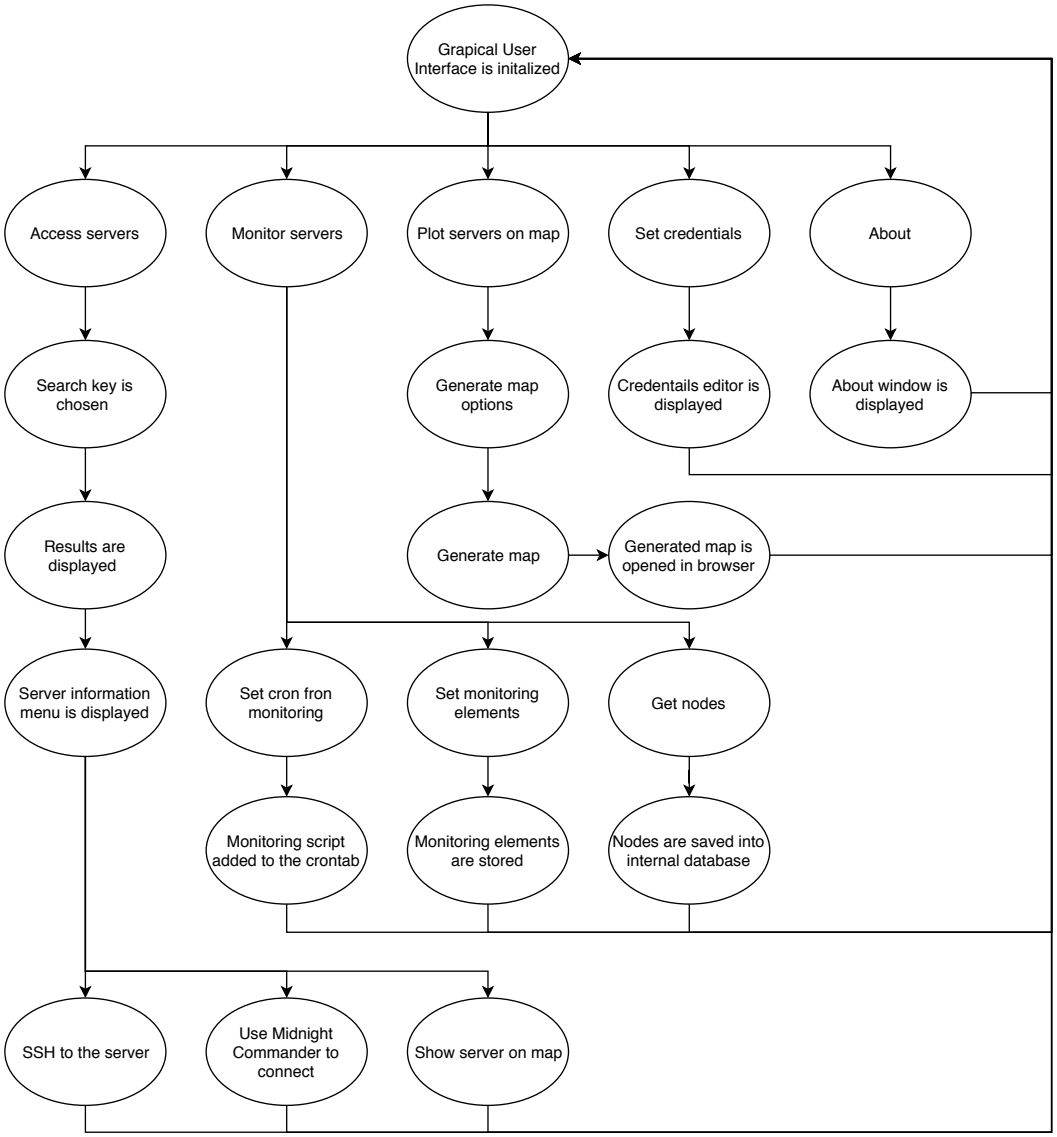

Fig. 5.3: Plbmng tool behavioral diagram.

6 Conclusion

The goal of this thesis was to get familiarized with the `plbmng` tool, re-implement the application into Python 3 and update the PyPi repository. Familiarization with the tool has been a pre-requisite of the other goals. In this thesis, the plbmng tool is, in Chapter 2, reviewed, analyzed and described. This thesis also contains discussion over state of the application and possible improvements. As tool uses PlanetLab network, it was introduced, analyzed and described in Chapter 1. Since PlanetLab network is primarily using Linux and virtualization, these topics were covered in Chapter 3.
Re-implementation is described in Chapter 5. The re-implementation was used to also include improvements to the each functions and their logic. Python usage has enabled various improvements such as removing result limitation, having application available as library, increased readability by having functions logically divided in one file instead of being composed from different files into script saved and run from disk, pre and post installation steps were removed, set credentials function has been improved, functions were written with Windows support in mind whenever possible, few minor bugs were fixed and minor improvements were added. The newly re-implemented tool's behavioral diagram has been described in Section 5.2. Last goal, which was to update the application's PyPi repository[1] with new code and information. The repository was successfully updated with the newest code changing from version `0.1.10` to version `0.2.1`. The description has been updated containing latest information regarding installation process of the tool. Dependencies were removed to reflect this change. With the update all the goals that this Semestral thesis aimed to accomplish were successfully achieved. Overall, all these changes were prerequisite for further development of the tool and increasing its usability for the PlanetLab users.

Future Development As mentioned in this Chapter introduction, the changes done in this Semestral thesis are prerequisite for the future work which will be done in the following Diploma thesis. The Diploma thesis will aim to document use cases of the application and provide documentation using UML (Unified Model Language) diagrams. Based on this documentation, the tool functionality will be revised and improved to reflect the use cases. Various existing functions will be improved such as map generation currently doesn't provide much information. Data to the map nodes, like IP address and country can be added. Also, as the tool was developed in two thesis, there are still Python scripts in different folders. This structure will be reviewed and improved to match one solid library to use. Windows support will

[1] The `plbmng` tool is available at: https://pypi.org/project/plbmng/

be further investigated and if possible, various functions will be ported to support both Linux-based and Windows operating systems. Another possible improvement is to add command line interface support so the script can be usable outside the GUI. Work on this improvement has been already started by adding `crontab` mode to the script which starts monitoring of the available servers.

Bibliography

[1] *PlanetLab Phase 0: Technical Specification.* [b.m.]: PlanetLab Consortium, August 2002. PDN–02–002.

[2] *Linux: dokumentační projekt.* 4., aktualiz. vyd. Brno: Computer Press, 2007. ISBN 978-80-251-1525-1.

[3] ANDRAŠOV, I. *Měření experimentální sítě PlanetLab.* Brno: Brno, University of Technology, 2017. Bachelor thesis. Available at: `https://www.vutbr.cz/studenti/zav-prace/detail/110277`.

[4] CORPORATION, O. *Oracle VM VirtualBox User Manual* [online]. Revised: 09, November, 2018 [cit. 25. November 2018]. Available at: `https://download.virtualbox.org/virtualbox/5.2.22/UserManual.pdf`.

[5] CORPORATION, O. *Welcome to VirtualBox.org!* [online]. Revised: 15, November, 2018 [cit. 25. November 2018]. Available at: `https://www.virtualbox.org/`.

[6] ECKERT, J. *Linux+ Guide to Linux Certification.* [b.m.]: Cengage Learning, 2012. Networking (Course Technology, Inc.). Available at: `https://books.google.cz/books?id=EHLH4S78LmsC`. ISBN 9781418837211.

[7] EDITORS, T. *Top Server Virtualization Software in 2018* [online]. Revised: 2018 [cit. 25. November 2018]. Available at: `https://www.trustradius.com/server-virtualization`.

[8] GRAYBOLTWOLF. *How fast is KVM? Host vs virtual machine performance* [online]. Revised: 1, December, 2016 [cit. 25. November 2018]. Available at: `https://goo.gl/AUWfuH`.

[9] KATHERINE NOYES, P. *Debian Linux Named Most Popular Distro for Web Servers* [online]. Revised: 11, Jannuary, 2012 [cit. 25. November 2018]. Available at: `https://goo.gl/6ZXLZU`.

[10] LLC, G. I. *Google Trends* [online]. Revised: 24, November, 2018 [cit. 25. November 2018]. Available at: `https://trends.google.com`.

[11] LUTZ, M. *Learning Python: Powerful Object-Oriented Programming.* [b.m.]: O'Reilly Media, 2013. Safari Books Online. Available at: `https://books.google.cz/books?id=4pgQfXQvekcC`. ISBN 9781449355692.

[12] Magklaras, D. G. *Introduction to Linux* [online]. [cit. 24. November 2018]. Available at: https://folk.uio.no/georgios/other/IntroductiontoLinux.pdf.

[13] Neary, D. *Did you know Linux is in your TV?* [online]. Revised: 08, May, 2018 [cit. 25. November 2018]. Available at: https://opensource.com/article/18/5/places-find-linux.

[14] Newbigin, J. *CentOS-2 Final finally released* [online]. Revised: 14, May, 2004 [cit. 25. November 2018]. Available at: https://goo.gl/7HsZUP.

[15] Park, K., Pai, V. S., Lee, K.-W. et al. Securing Web Service by Automatic Robot Detection. *Proceedings of the Annual Conference on USENIX '06 Annual Technical Conference.* Berkeley, CA, USA: USENIX Association. S. 23–23. ATEC '06. Available at: http://dl.acm.org/citation.cfm?id=1267359.1267382.

[16] Pilgrim, M. *Ponořme se do Python(u) 3: Dive into Python 3*. Praha: CZ.NIC, c2010. ISBN 978-80-904248-2-1.

[17] Princeton University, T. T. of. *About PlanetLab project* [online]. Revised: 29, May, 2017 [cit. 24. November 2018]. Available at: https://www.planet-lab.eu/about.

[18] Princeton University, T. T. of. *PlanetLab Main Page* [online]. Revised: 2017 [cit. 24. November 2018]. Available at: https://www.planet-lab.org/.

[19] Ramasubramanian, V. a Sirer, E. G. The Design and Implementation of a Next Generation Name Service for the Internet. *SIGCOMM Comput. Commun. Rev.* Srpen 2004, vol. 34, Issue 4, s. 331–342. Available at: http://doi.acm.org/10.1145/1030194.1015504. ISSN 0146-4833.

[20] Red Hat, I. *Red Hat Virtualization* [online]. Revised: 2018 [cit. 25. November 2018]. Available at: https://goo.gl/YAQ9Eb.

[21] Red Hat, I. *Red Hat Virtualization Datasheet* [online]. Revised: January, 2018 [cit. 25. November 2018]. Available at: https://www.redhat.com/en/resources/virtualization-datasheet.

[22] Red Hat, I. *Timeline: Red Hat's history* [online]. Revised: 2016 [cit. 25. November 2018]. Available at: https://goo.gl/H5w6q7.

[23] Red Hat, I. *Trusted* [online]. Revised: June, 2017 [cit. 25. November 2018]. Available at: https://www.redhat.com/en/about/trusted.

[24] Rossum, N. C. Guido van. *PEP 8 – Style Guide for Python Code* [online]. Revised: 1, August, 2013 [cit. 25. November 2018]. Available at: https://www.python.org/dev/peps/pep-0008/.

[25] SAS, V. O. S. *The Virtual Open Systems video demos to virtualize ARM multicore platforms* [online]. Revised: 16, August, 2016 [cit. 25. November 2018]. Available at: https://goo.gl/Me4ikn.

[26] Sherwood, R., Braud, R. a Bhattacharjee, B. Slurpie: a cooperative bulk data transfer protocol. *IEEE INFOCOM 2004*. S. 941–951 vol.2. ISSN 0743-166X.

[27] StackOverflow. *The Incredible Growth of Python* [online]. Revised: 6, September, 2017 [cit. 24. November 2018]. Available at: https://stackoverflow.blog/2017/09/06/incredible-growth-python/.

[28] StatCounter. *Operating System Market Share Worldwide 2018* [online]. Revised: October, 2018 [cit. 25. November 2018]. Available at: http://gs.statcounter.com/os-market-share.

[29] Treinen, R. *50.000 binary packages* [online]. Revised: 8, Februray, 2016 [cit. 25. November 2018]. Available at: https://lists.debian.org/debian-devel/2016/02/msg00122.html.

[30] VMware, I. *Architecture of VMware ESXi* [online]. Revised: 15, October, 2008 [cit. 25. November 2018]. Available at: https://goo.gl/Gskpvf.

[31] Šuba, F. *Monitorování serverů s OS Linux*. Brno: Brno, University of Technology, 2018. Bachelor thesis. Available at: https://www.vutbr.cz/studenti/zav-prace/detail/110178.

List of symbols, physical constants and abbreviations

DNS	Domain Name System
IP	Internet Protocol
KVM	Kernel-based Virtual Machine
MMU	Memory Management Unit
CPU	Central Processing Unit
CentOS	Community Enterprise Operating System
yum	Yellowdog Updater, Modified
APT	Advanced Packaging Tool
IoT	Internet of Things
GUI	Graphical User Interface
PEP	Python Enhancement Proposal
API	Application Programming Interface
ICMP	Internet Control Message Protocol
UML	Unified Model Language