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 AUTOR PRÁCE	Bc. Martin Kačmarčík
SUPERVISOR VEDOUCÍ PRÁCE	doc. Ing. Dan Komosný, Ph.D.

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 [9]. This thesis extends existing tools developed by Ivan Andrašov [3] and Filip Šuba [24].

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 1 the tool and summary of previous work is reviewed. In the Chapter 2 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 [14]. In the Chapter 4 the improvements made to the Plbmng tool will be explained.

1 Plbmng Tool

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 [24]. 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.

1.1 Description of Tool's Funcionality

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 1.1 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/

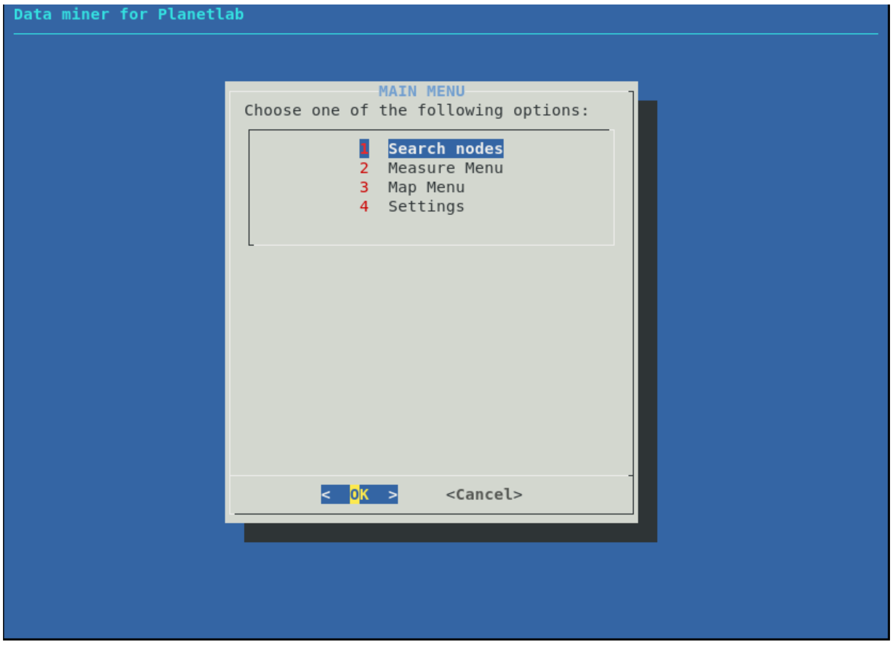

Fig. 1.1: Data miner for PlanetLab menu.

1.2 Areas of improvement

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 [21], and because it is available at PyPi (Python Package Index) 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 [13], 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 improvements is renaming certain menu components. This change is not significant and is purely cosmetic but can make it easier for new users to get familiar with the tool. Since the tool is not data mining rather then using servers and managing them, 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 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 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.

2 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 [15]. The main description also states t currently consists of 1353 nodes at 717 sites [1]. The current committee of the project consists of members like Princeton University, Cambridge University, Intel, Google and many more [15].

2.1 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.
- **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.

2.2 PlanetLab Enabled Projects

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 [15]. Having over

[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.

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.

2.2.1 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 [12].

2.2.2 The Design and Implementation of a 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 [16].

2.2.3 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 [20].

3 Linux and Virtualization

3.1 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 [10] 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 Enterprise Linux`. More information about mentioned distributions can be found in Subsection 3.1.1. 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 [22]. 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 [22]. 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 [11]. 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. Pouzij citace ze zadani!

3.1.1 Linux Distributions

In this subsection the various popular Linux distributions will be described. Their key features will be reviewed and analyzed.

Red Hat Enterprise Linux

CentOS

Fedora

Ubuntu

Debian

SUSE Enterprise Linux

3.2 Virtualization

Virtualization is an alternative to the traditional architecture. Comparison schema between virtualization and traditional architecture can be found in Figure 3.1. 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.

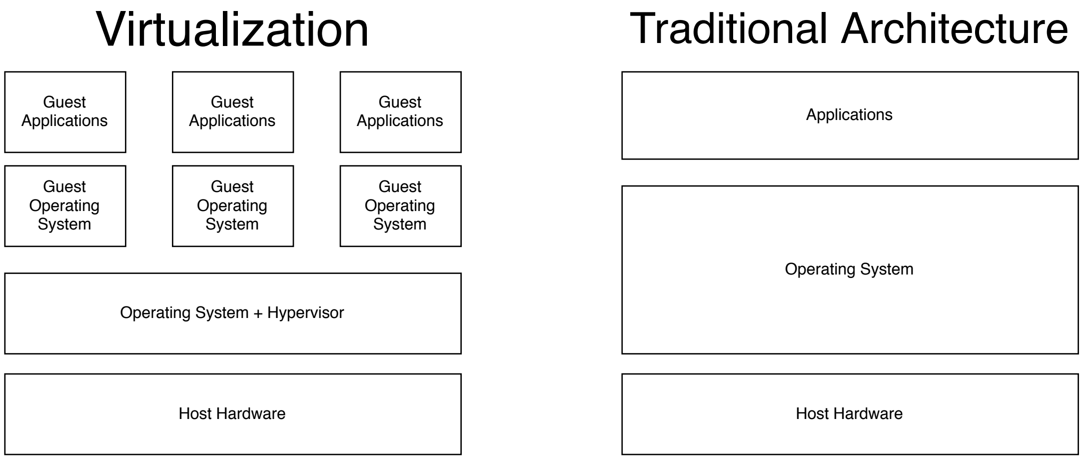

Fig. 3.1: Virtualization vs traditional architecture schema.

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].

3.2.1 Virtualization Technologies

In this subsection, examples of various popular virtualization providers and their software is described. This subsection offers overview and description with crucial attributes of each technology.

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 [23].

This allows the software direct access to the hardware as it includes its own vital OS components. As VMware states in their architecture document [23], 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 [17]. `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 [18]. 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 [19]. 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].

4 Plbmng Tool Improvements

5 Conclusion

Share the results.

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] 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.

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

[11] 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.

[12] 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.

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

[14] 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.

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

[16] 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.

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

[18] 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.

[19] 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.

[20] 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.

[21] 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/.

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

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

[24] 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
PyPi	Python Package Index