UNIVERZITA PARDUBICE

Fakulta elektrotechniky a informatiky

Využití Raspberry Pi pro cloudové datové centrum

Bc. Čeněk Pozdník

Diplomová práce

2017

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2016/2017**

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Čeněk Pozdník**
Osobní číslo:	**I15230**
Studijní program:	**N2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Využití Raspberry Pi pro cloudové datové centrum**
Zadávající katedra:	**Katedra softwarových technologií**

Zásady pro vypracování:

Cílem diplomové práce bude analyzovat možnost vytvoření cloudového datového centra s využitím Raspberry Pi. Autor v teoretické části analyzuje principy cloudových datových center a na základě této analýzy implementuje vhodné nástroje pro vytvoření takového centra na Raspberry Pi. Datové centrum bude vybaveno monitoringem, vzdálenou správou a bude využívat síťovou infrastrukturu tvořenou SDN zařízeními (softwarové switche běžící na Raspberry Pi).

Rozsah grafických prací:

Rozsah pracovní zprávy: **50**

Forma zpracování diplomové práce: **tištěná**

Seznam odborné literatury:

***MOLLOY, Derek. Exploring raspberry PI. Indianapolis, IN: John Wiley and Sons, 2016. ISBN 9781119188681.**
***DOHERTY, Jim. SDN and NFV simplified: a visual guide to understanding software defined networks and network function virtualization. Upper Saddle River, NJ: Pearson Education, Inc., 2016. ISBN 0134306406.**

Vedoucí diplomové práce: **Ing. Soňa Neradová, Ph.D.**
Katedra informačních technologií

Datum zadání diplomové práce: **31. října 2016**
Termín odevzdání diplomové práce: **17. května 2017**

Ing. Zdeněk Němec, Ph.D.
děkan

L.S.

prof. Ing. Antonín Kavička, Ph.D.
vedoucí katedry

V Pardubicích dne 15. listopadu 2016

Prohlášení autora

Prohlašuji, že jsem tuto práci vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Beru na vědomí, že v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, a směrnicí Univerzity Pardubice č. 9/2012, bude práce zveřejněna v Univerzitní knihovně a prostřednictvím Digitální knihovny Univerzity Pardubice.

V Pardubicích dne 16.5.2017

Bc. Čeněk Pozdník

PODĚKOVÁNÍ

Tímto bych chtěl poděkovat své vedoucí diplomové práce paní Ing. Soně Neradové, Ph.D. za odborné vedení, cenné připomínky, rady a doporučení publikací při vytváření této práce. Dále bych chtěl poděkovat Fakultě elektroniky a informatiky Univerzity Pardubice za umožnění použití laboratoře počítačových sítí pro provedení praktických úloh této práce. Rád bych také poděkoval svým rodičům za podporu při studiu.

ANOTACE

Tato práce popisuje vlastnosti cloudových datových center a prostředků, které jsou v těchto datových centrech využívány. Teoretická část práce představuje vlastnosti cloudu, virtualizace, kontejnerů, NAS a Raspberry Pi. Každá část práce se věnuje podrobnému popisu jednotlivých témat, podle kterých jsou provedeny praktické příklady. Tyto příklady představují podrobný návod, jak vybudovat cloudové datové centrum na zařízeních Raspberry Pi.

KLÍČOVÁ SLOVA

cloud, kontejnery, virtualizace, NAS, Raspberry Pi

TITLE

Using Raspberry Pi for cloud data center.

ANNOTATION

This dissertation describes characteristics of cloud data centers and resources that are in these data centers used. The theoretical part presents characteristics of cloud, virtualization, containers, NAS and Raspberry Pi. Each part of the thesis deals with a detailed description of each topic, according to which practical examples are made. These examples are detailed instructions of how to build a cloud data center on Raspberry Pi devices.

KEYWORDS

Cloud, container, virtualization, NAS, Raspberry Pi

OBSAH

SEZNAM ILUSTRACÍ A TABULEK

SEZNAM ZKRATEK A ZNAČEK

AFP	Apple Filing Protocol
API	Application Programming Interface
APIaaS	API as a Service
BaaS	Backend as a Service
CIFS	Common Internet File Systém
CRM	Customer relationship management
DaaS	Data as a Service
DAS	Direct-attached storage
DbaaS	Database as a Service
DDoS	Distributed Denial of Service
DLNA	Digital Living Network Alliance
EU	Evropská unie
FTP	File Transfer Protocol
GPDR	General Data Protection Regulatio
GUI	Grafické uživatelské rozhraní
HDD	Pevný disk
HTTP	Hypertext Transfer Protocol
HW	Hardware
IaaS	Infrastructure as a service
IPaaS	Integration platform as a Service
MIT	Massachusetts Institute of Technology
NaaS	Network as a Service
NAS	Network Attached Storage

NFS	Network File Systém
NIST	National Institute of Standards and Technology
OS	Operační systém
PaaS	Platform as a service
PC	Osobní počítač
RAID	Redundant Array of Independent Disks
SaaS	Software as a Service
SCSI	Small Computer System Interface
SECaaS	Security as a Service
SHR	Synology Hybrid RAID
SLA	Service Level Agreements
SMB	Server Message Block
SSH	Secure Shell
SSL	Secure Sockets Layer
StaaS	Storage as a Service
TEaaS	Test Environment as a Service
UPnP	Universal Plug and Play
USB	Universal Serial Bus
VPN	Virtual private network
VPS	Virtual Private Server
WebDAV	Web-based Distributed Authoring and Versioning

ÚVOD

Téměř každá firma v dnešní době používá nějaké počítače, které potřebuje spravovat. Ať už to jsou koncové stanice, servery nebo celá datová centra. Dále každá firma produkuje při své činnosti nějaká data, která jsou potřeba ukládat a archivovat. Proto tyto firmy hledají řešení, jak tato data co nejlépe spravovat a ukládat. Prvním řešením tohoto problému, které se pro firmy nabízí je vlastní server nebo datové centrum. Ale vlastní servery a datová centra jsou pro firmy často drahé a náročné na údržbu. Pro některé z firem jsou to náklady navíc, protože jim to nařizuje například jen zákonná legislativa, ale jinak by se bez těchto prostředků obešly. Pro začínající firmy je klíčové při startu své činnosti dát co nejvíce finančních prostředků do oboru své činnosti, a ne do počítačů. Celá řada firem, která by chtěla zahájit svoji činnost v oboru, kde je nezbytně nutné mít servery nebo dokonce už i datová centra, nemůže z důvodů vysokých počátečních investic do těchto zařízení vůbec vzniknout. Jako východisko z těchto situací se firmám nabízí řešení v podobě cloudu. Dokáže firmám ušetřit peníze jak v průběhu své činnosti nebo při jejím zahájení a odebrat jim starosti o správu vlastních zařízení. Proto je cílem této práce seznámit čtenáře s vlastnostmi cloudu, jak vybudovat cloudové datové centrum a jaké prostředky jsou k tomu využívány.

V první části této práce je vysvětlena teorie ohledně cloudu a prostředků se kterými je pracováno v praktické části této práce. První kapitola je věnována přímo cloudu. Na úvod této kapitoly se čtenář dozví něco z historie cloudu. Dále je vymezena hranice mezi tím, co za cloud považovat lze a co ne. Poté jsou uvedeny modely služeb a nasazení cloudu, které určují základní rysy služeb a cloudu samotného. V další části jsou představeni aktéři a nejznámější poskytovatelé cloudového řešení. Nakonec je uvedeno, co se v cloudu musí řešit z hlediska zabezpečení a legislativních aspektů. Druhá a třetí kapitola je věnována prostředkům využívajícím při tvorbě cloudu, což jsou virtualizace a kontejnery, které nově vzrůstají na oblibě. Na závěr teoretické části jsou postupně představeny kapitoly o zařízeních využitých v praktické části práce. Nejdříve se čtenář seznámí se zařízením Raspberry Pi, na kterém je postaveno datové centrum a posléze se zařízením NAS, na které se v příkladu ukládají data z cloudu.

Praktická část je věnována již výstavbě cloudového datového centra. Nejprve se provede praktická ukázka, jak nainstalovat OS, nastavit a zálohovat Raspberry Pi. Dále se nainstaluje a nastaví zařízení NAS. Poté je již vytvořeno první datové centrum, které slouží ke sdílení a zálohování souborů. Datové centrum běží na síti tvořené SDN zařízeními. V této ukázce je

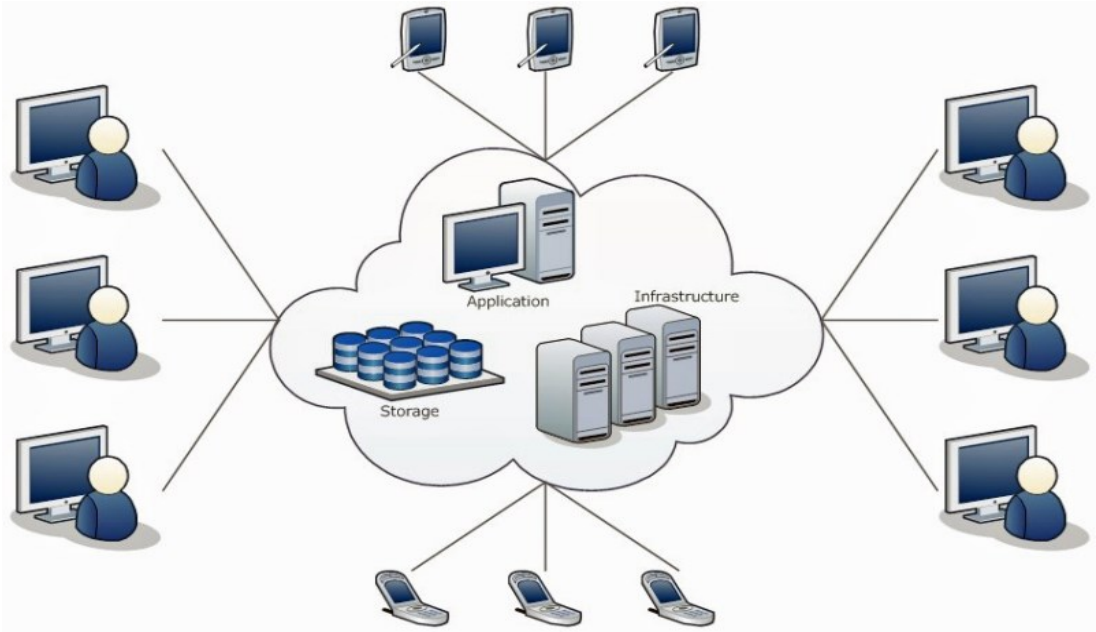
předvedena celková konfigurace tohoto datového centra na Raspberry Pi. Následně je předvedena pokročilejší ukázka, kdy je vytvořen celý cluster zařízeních Raspberry Pi. Na zařízeních v clusteru běží kontejnery pro zajištění dobrého škálování zdrojů. Hlavním úkolem clusteru je spolupráce vícero zařízení, která zajistí vysokou dostupnost a stabilitu služeb v cloudu.

1 CLOUD

První myšlenka ve smyslu cloud computingu je již z roku 1961, kdy profesor John McCarthy z MIT prezentoval myšlenku sdílení počítačových technologií na podobném principu jako je sdílení elektrické sítě. Lidé si pro použití domácích spotřebičů zpravidla nepořizují vlastní elektrárnu, ale platí za elektřinu (službu) poskytovateli, který ji sdílí z elektráren pomocí elektrické sítě uživatelům. Platí se jen částka, která odpovídá spotřebované elektřině. Zákazníka nezajímá, jak se elektřina vyrábí a ani jak je k němu vedena, stačí mu, že když něco zapojí do zásuvky tak to bude fungovat. Aby byla tato myšlenka kompletní, tak je třeba ještě zdůraznit, že elektrárny jsou mezi sebou propojeny elektrickou sítí a v případě výpadku jedné z elektráren poskytují službu další elektrárny a její činnost tak zastoupí. Z dnešního pohledu cloud computingu by se dalo uvažovat, že elektrárna zastává roli datového centra, elektrická síť roli internetu a spotřebič roli počítače. (Mácha, 2012) (Lacko, 2012)

Pojem cloud, česky mrak, vznikl převzetím z telekomunikací a je popisem schematického obrázku infrastruktury. Mrak byl historicky používán při grafickém znázornění telekomunikační sítě, kde byl používán pro vyjádření sítě od uživatele k cíli a znázorňoval pojem internetu. U internetu ikona obláčku zobrazovala další zařízení, o které se zpravidla stará někdo jiný, takže pro funkčnost není potřeba znát její konkrétní strukturu. Do roku 1997 nebyl cloud computing označován tak jak ho známe dnes, ale byl označován jako unity computing. (Mácha, 2012) (Velte a další, 2011)

Obecně řečeno se za cloud dá označit komplexní IT infrastruktura, která poskytuje IT služby na míru. Může se například jednat o úložiště dat, síťovou infrastrukturu, platformy a aplikace. Služby jsou poskytovány v rámci předplatného a předplatitel platí jen za to, co doopravdy používá. Některé služby, ač v omezené míře, mohou být poskytovány i zdarma. Tedy uživatel si může nastavit, jaké prostředky bude doopravdy potřebovat. Cloud zabezpečuje výpočetní zdroje, kde IT prostředky jsou pro uživatele sdíleny. Zdroje a služby jsou zde odděleny od infrastruktury a jsou plně automatizovány. Cloud představuje přenesení zodpovědnosti na někoho jiného, který bude odpovídat za správu, zabezpečení a poskytování zdrojů. (Lacko, 2012) (Velte a další, 2011)



Obrázek 1 − Ukázka cloudu

Zdroj: (Cloud Productivity, 2017)

1.1 Co je a není Cloud

Pod pojmem cloud se dá představit skoro vše co je na internetu, a proto je potřeba si vyjasnit jeho hranice. Cloud může být kdekoliv, není svázaný s geografickou polohou a nemůžeme o něm mluvit jako o místu a nelze o něm uvažovat ve fyzickém kontextu. Cloud je otevřený a nabízí volnost při výběru technologií, společností a HW. Spousta lidí si myslí, že virtualizace je cloud, ale tato představa je mylná, přesto je virtualizace při implementaci cloudu velice hojně využívána. Poskytovatelé virtuálních serverů si konkurují především účtování a možnostmi změn v nastavení.

U Virtual private server (VPS) se změny mohou projevit řádu hodin[1] a platí se měsíční pravidelný poplatek, kdežto cloud se dá měnit téměř okamžitě a účtován je spotřebovaný výkon. Cloud na rozdíl od VPS je plně automatizován a nevyžaduje spolupráci s poskytovatelem při nahrávání nového virtuálního stroje. Také se o něm nedá mluvit jako o webhostingu, ten je také účtován v pravidelných měsíčních poplatcích, zatímco když webová aplikace běží v cloudu tak poplatky závisejí na aktuálním vytížení a dokáží svůj výkon přizpůsobovat podle zátěže aplikace v různých časových úsecích. Z tohoto vyplývá, že cloudové služby nejsou řízené služby, protože jejich zdroje nemusejí být stálé v čase a dají se škálovat.

[1] Ve výjimečných případech i dnů.

Cloud dokáže pracovat na takzvaném samoobslužném principu. Nemusí být vždy privátní či veřejný, ale dokáže tyto principy na míru kombinovat. Nejedná se o outsourcing, protože ten pracuje se stálým množstvím zdrojů, kdežto cloud je škálovatelný a sdílený. Také se nejedná o utility computing[2]. Základní charakteristiky cloudu computingu budou probrány v následující kapitole. (Špička, 2013) (Malý, 2011) (Nieves, 2014)

1.2 Pět základních charakteristik cloudu

Základní charakteristiky cloud computingu podle organizace *National Institute of Standard and Technology* patří: samoobslužný princip, síťový přístup, fond prostředků, elasticita a měření prostředků.

Samoobslužný princip (*On-demand self-service*) dovoluje zákazníkovy měnit parametry bez potřeby spolupráce s poskytovatelem služeb. Jinak řečeno, zákazník může objednat a nastavit služby on-line za pomoci interaktivní aplikace. Výjimkou mohou být velmi velké společnosti, které potřebují enormní zdroje. U těchto společností, i když to není pravidlem, může dojít i k osobnímu setkání.

Síťový přístup (*Broad network access*) umožňuje zákazníkovy přístup odkudkoliv s využitím známých standartních protokolů. Má umožnit přístup z jakéhokoliv PC, chytrého zařízení a tenkého klienta. K přístupu by neměl potřebovat žádné další nestandartní prostředky, jako jsou například kabely. Nejčastěji se ke cloudu přistupuje za pomoci Internetu.

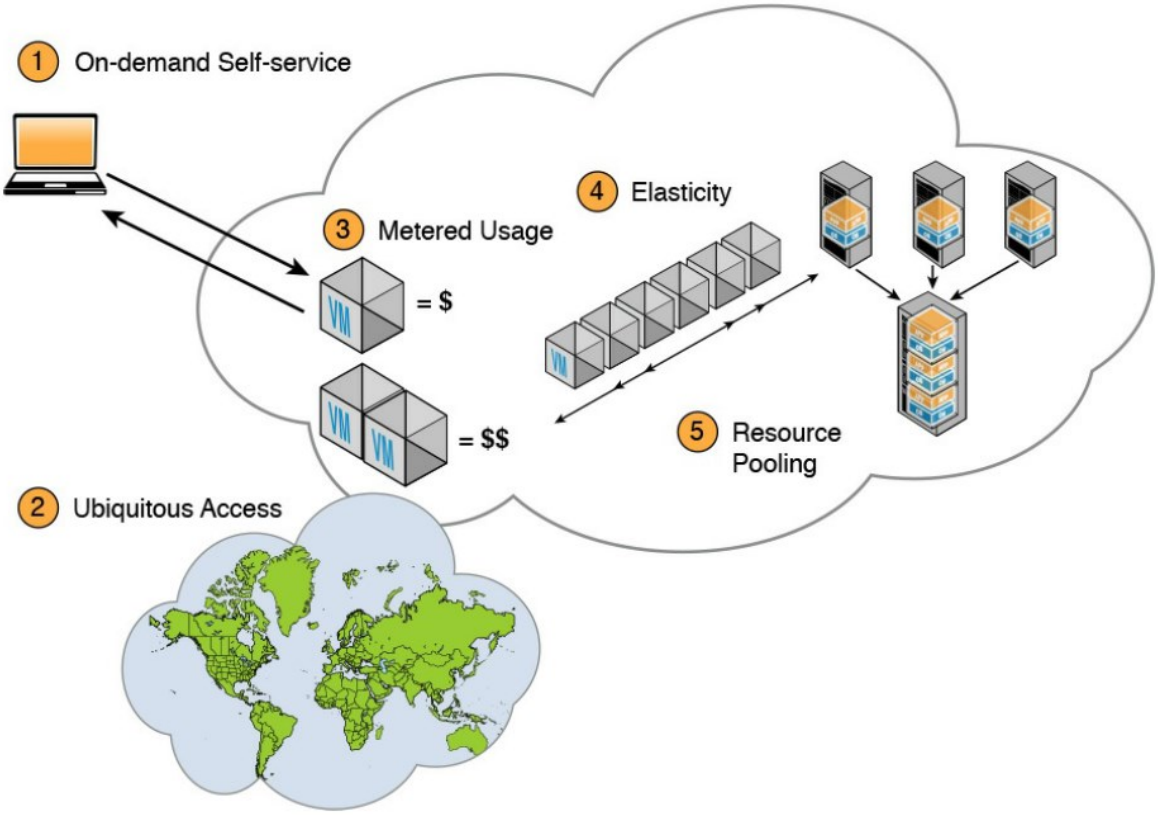
Fond prostředků (*Resource pooling*) umožňuje seskupování zdrojů. Mezi tyto zdroje se může například nejčastěji řadit kapacita úložiště, operační paměť, procesorový čas a šířka pásma. Tyto zdroje jsou přidělovány jednotlivým zákazníkům podle toho, kolik jich požadují. Tedy cloud umožňuje klientům čerpat prostředky z jednotlivých sdílených poolů dle jejich potřeby. Toto sdružování a přidělování prostředků maximalizuje využití a efektivitu prostředků na straně poskytovatele. Díky tomu zákazníci nemusí platit za prostředky, které nevyužijí. V případě privátního cloudu, který bude probrán v jedné z následujících kapitol, nemusí být pool sdílený mezi zákazníky a zákazník má svůj vyhrazený pool, ale z pohledu klienta je pak tento pool sdílen mezi uživateli.

Elasticita (*Rapid elasticity*) je jedna ze základních charakteristik pro definici cloudu. Umožňuje skrze rozhraní dle potřeb rychle, ne-li okamžitě zvýšit výpočetní zdroje a pak je zase dle potřeb nastavit zpátky. Tedy umožní zákazníkovi nastavit zdroje dle jeho potřeb v jakémkoliv

[2] Unitility computing nezahrnuje přístup za pomoci sítě.

okamžiku. Díky tomuto principu je cloud využíván u malých firem, kde jim dovolí si pořídit slušné výpočetní zdroje za zlomek ceny. Velké firmy potřebují velká data centra a díky rychlé pružnosti zdrojů ušetří také nemalé náklady, protože zdroje mohou nastavit podle toho, co aktuálně potřebují.

Měření prostředků (*Measured service*) znamená, že cloud umožňuje monitorovat využívání služeb. Dle tohoto se pak pro zákazníka odvíjí cena jen podle toho, co má aktuálně nastaveno. Tento princip umožní snížit náklady zákazníka. Jednou výjimkou je privátní cloud, kde se platí pevný měsíční poplatek, protože z principu privátního cloudu nemůže zdroje používat někdo jiný, i když zdroje nejsou zrovna využívány. Podrobněji bude privátní cloud popsán v kapitolách níže. (Doherty, 2016) (National Institute of Standards and Technology, 2013)



Obrázek 2 − Pět základních charakteristik cloudu

Zdroj: (Doherty, 2016)

1.3 Termíny v oblasti cloudu

1.3.1 Cluster

Je složen z vícero počítačů, které spolupracují na řešení nějakého problému (úlohy). Komunikují mezi sebou nejčastěji pomocí počítačové sítě. Využívány jsou především pro

zvýšení výkonu pro velké úlohy, zajištění přesných výsledků (korelace chyby), nebo pro velmi vytěžované služby. Clustery se nejčastěji rozdělují na: cluster s vysokou dostupností, výpočetní cluster, cluster s rozložením zátěže a úložný cluster. Jedním z příkladů, kde se využívá clusteru je předpověď počasí. (iDNES.cz, 2010)

1.3.2 Datové centrum

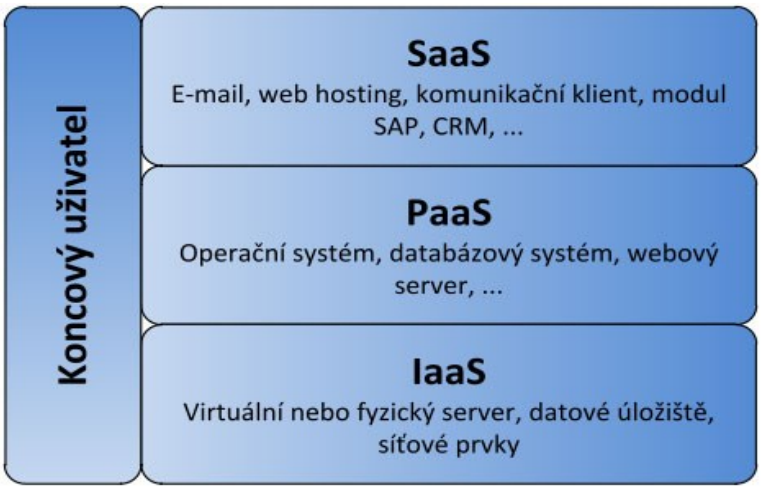
Jedná se o sál nebo místnost, ve které jsou servery a obsahuje speciální infrastrukturu pro jejich obsluhu. Většinou obsahují speciální zařízení pro ochranu a zachování funkčnosti serverů. Například mohou být vybaveny požární ochranou, klimatizaci a záložním zdrojem energie. (Lacko, 2012)

1.4 Modely služeb

Protože cloud je založen na průběžném modelu placení, tak jsou zdroje nabízeny jako služba, na rozdíl od produktu, který se dá koupit. Cloud computing se rozděluje podle toho, jaký model služeb je využíván. Existuje celá řada modelů poskytování služeb zákazníkům. Podle organizace NIST jsou uznávány tyto 3 základní modely služeb:

- *Infrastructure as a Service* (IaaS) - Infrastruktura jako služba.
- *Platform as a Service* (PaaS) - Platforma jako služba.
- *Software as a Service* (SaaS) - Software jako služba.

(Doherty, 2016) (National Institute of Standards and Technology, 2013)



Obrázek 3 − Modely služeb

Zdroj: (Foldyna, 2012)

1.4.1 Infrastructure as a Service

V modelu infrastruktura jako služba si zákazníci pronajímají zdroje, které jim umožňují spouštět jejich vlastní programy a operační systémy, které chtějí. Takže zákazníci mají plnou kontrolu nad operačním systémem a programy. Dále mají kontrolu nad ukládáním dat, sítí a zabezpečením jako je například firewall. Jednoduše řečeno, klient si pronajme server a může do něj nahrát jakékoliv softwarové vybavení.

Velké společnosti mohou tento model využít v případě, že chtějí ušetřit provozní náklady a nepotřebují zdroje celého serveru nebo je jejich spotřeba zdrojů nesouměrná a mají tedy výkyvy ve spotřebě zdrojů. Pro malé organizace může být tento model východiskem, když nemají kapitál na pořízení vlastní infrastruktury. Tento model má, jak už z textu vyplývá největší uplatnění ve firemní sféře.

Společnosti ušetří i náklady za správu HW, o který se stará poskytovatel služby. Společnosti tedy mají ve své správě vrstvu: operačního systému, middleware a aplikací. Poskytovatel poskytuje virtuální stroje, které jsou umístěny na jednom fyzickém stroji za pomoci specializovaných virtualizacích nástrojů zvaných jako hypervizor.

Tento model přináší pro klienty mnoho výhod, mezi které patří výhody finanční, kde klienti pocítí snížení nákladů. Největší snížení nákladů pocítí začínající podniky při pořizování zcela nové infrastruktury. Firmy nemusejí mít speciální sály, které splňují podmínky zabezpečení pro datová centra. Klienti mohou využívat různé operační systémy a přepínat mezi nimi.

Model IaaS má ale i svoje úskalí. Klient neví, kde jsou přesně uložena jeho data. Ale ví, že jsou jen někde v cloudu. Klient tedy ztrácí fyzickou kontrolu nad uložením dat. Pokud se pracuje s citlivými daty v cloudu, kde jsou zdroje sdíleny, je zvýšeno bezpečnostní riziko narušení dat. Nemůže se plně kontrolovat co se přesně děje na síti. (Doherty, 2016) (CESNET, 2016)

1.4.2 Platform as a Service

Při použití platformy jako služby se zákazník nemusí start o hardware a operační systém, protože ten je zcela ve správě poskytovatele. Tím pádem zákazníkovi odpadá starost o infrastrukturu, jako jsou například servery, úložiště, sítě a další. PaaS je především určen pro vývojáře, kteří zde provozují, vyvíjejí a testují vlastní aplikace, nebo programy. Veškeré softwarové vybavení se tedy pro koncové uživatele nachází v cloudu a uživatelé se nemusí starat o instalaci speciálních software a správu jejich licencí.

Zdroje se přidělují podle toho, jaké programové vybavení a prostředí bude zákazníkem používáno. Služba je často využívána pro databázové systémy. Služba je velmi finančně výhodná při provozování programů a operačních systémů, protože je nákladné vše pořádně otestovat a odladit na nové platformě. Rozhodně tato služba přináší nemalé úspory pro zákazníky, kteří nemusejí mít správu infrastruktury ve své režii a byla by to pro ně starost navíc.

Ale služba přináší i svá rizika. První může být například pro vývojáře, když neví, kde přesně mají uložen zdrojový kód svých aplikací a může to pro ně přinést jistá bezpečnostní rizika. Někdy se také potřebují data udržovat zcela v soukromí, ať je to například zdrojový kód aplikací, nebo jejich data, a to je zde problémem. (Doherty, 2016) (Lacko, 2012)

1.4.3 Software as a Service

Je to služba, se kterou se lidé nejběžněji setkávají a nemusí si to vůbec uvědomovat. Základním principem je pronájem aplikace klientovi za určitý pravidelný poplatek, zákazník si nekoupí aplikaci, ale jen přístup k ní. Přístup k aplikaci a datům je umožněn odkudkoliv a v jakýkoliv čas, takže data aplikace jsou umístěna v cloudovém datovém centru. Klient aplikaci využívá nejčastěji za pomoci aplikace na svém zařízení nebo je přístupná z webového prohlížeče. Také může být použit jen tenký klient. Aplikace tak po klientovi požaduje minimum výpočetního výkonu.

SaaS by se dal rozdělit dvou základních kategorií:

- Služby pro firmy – poskytují specializované aplikace pro firmy, například skladový systém.
- Služby pro uživatele – dostupné komukoliv, často i zdarma (například email).

Dá se říci, že u této služby se zákazník o nic nestará a službu jen používá. Odpadá mu starost o správu stroje, jako je například instalace aktualizací operačních systému, software, nebo zálohování a různá nastavování. Z toho vyplývá výhoda pro firmy, protože už nepotřebují další zaměstnance pro správu technického vybavení. Klient má možnost využívat aplikaci na více zařízení a tím ušetří náklady za licence na všechny tyto zařízení.

Služba pro běh aplikace zpravidla vyžaduje připojení k internetu, ale to není v dnešní době zase takový problém. Zákazník zde ztrácí kontrolu nad svými daty a mohlo by to ztížit změnu aplikace či poskytovatele. Pro některé typy aplikací nemusí být tento typ služby vhodný. Nejčastěji se jedná o aplikace, které jsou šité na míru firmě nebo vyžadují velké množství zdrojů. Například se může jednat o aplikace typu Business Inteligence. (Doherty, 2016) (Velte a další, 2011)

1.4.4 Nové modely služeb

Nabídka služeb cloudu se stále rozšiřuje a přibývají další modely služeb. Většinou jsou tyto modely specializovány na jednu konkrétní věc. V následujících podkapitolách budou jednotlivé služby jmenovány a krátce představeny.

Network as a Service (NaaS) – síť jako služba: Za pomoci internetu jsou poskytovány virtuální síťové služby. NaaS je velmi výhodná pro začínající společnosti, protože nemusejí investovat do síťových zařízení. Dále společnosti nemusí mít personál, který by se o síť staral, protože toto řeší poskytovatel služeb.

Storage as a Service (StaaS) – úložiště jako služba: Za předplatné poskytuje uživatelům úložiště. Společnostem ušetří náklady za budování vlastního úložiště a personál, který by musel ho spravovat. Uživatel se také nemusí starat o zálohování svých dat a poruchy zařízení, protože toto také řeší poskytovatel služby.

Security as a Service (SECaaS) – bezpečnost jako služba: Poskytovatel nabízí různá řešení zabezpečení a jejich nasazení do infrastruktury klienta. Zákazník tak získává lepší zabezpečení při snížení finančních nároků. Správa bezpečnostních mechanismů se většinou dá zákazníkem spravovat pomocí webového rozhraní, ale většinu řeší poskytovatel služby. Mezi zabezpečení se může například řadit antivirus.

Data as a Service (DaaS) – data jako služba: Nabízí distribuci dat a informací po síti zákazníkům. Příkladem použití může být způsob rozšíření letáků. Cena služby se může odvíjet od spotřebovaných dat.

Desktop as a Service (DaaS) – desktop jako služba: Počítač je pro zákazníka hostován jako virtuální stroj v cloudu v rámci předplatného. K desktopu mohou klienti přistupovat odkudkoliv po síti a mají vždy k dispozici svoje aplikace a data. Výhoda tkví v ušetření nákladů za správu desktopu především u malých firem.

Database as a Service (DbaaS) – databáze jako služba: Databáze v cloudu mají za cíl jako všechny cloudové služby snížit náklady. Protože pro zákazníka odpadá spravování databáze a může ji jen používat. Poskytovatel služby se stará jak o upgrade databáze, tak i o zálohování databáze.

Test Environment as a Service (TEaaS) – testovací prostředí jako služba: Umožňuje ušetřit peníze společnosti za vlastní testovací prostředí. Obsahuje sady testovacích software, například pro testování zátěže webů. Další výhodou je rychlé zřízení a nastavování použití zdrojů.

Zákazník může službu mít aktivní jen v dobu, kdy ji bude potřebovat, a tím že služba bude neaktivní v období, kdy nebude třeba, tak zákazník ušetří peníze.

API as a Service (APIaaS): Protože programové aplikační rozhraní získává čím dál více na oblibě, vznikla další služba APIaaS, která má své využití na trhu. Služba má za svůj cíl poskytnout rychlé nasazení, budování a zdokumentování API. Aplikační rozhraní slouží jako dokumentované standardní rozhraní pro komunikaci mezi aplikacemi.

Backend as a Service (BaaS): Uplatňuje se při zpracování dat především u mobilních, tabletových a webových aplikací. Má jednoduše propojit cloudové backendové úložiště s aplikací. Vývoj backendu je často náročný a díky této službě může být v některých případech funkční za jednu minutu. Dále usnadňuje například tvorbu notifikací v aplikaci.

Integrated development environment as a Service (IDEaaS): Služba obsahuje kompletní vývojové prostředí a další specializované nástroje určené pro vývoj. Vybavení je cíleno podle požadovaného programovacího jazyka.

Integration platform as a Service (IPaaS) – integrační platforma jako služba: Představuje propojení cloudových služeb. Umožňuje kombinaci aplikací cloudových a firemních aplikací, procesů, dat a služeb. Poskytovatelé dodávají jak serverovou, tak i datovou infrastrukturu. (Hospodářské noviny IHNED.cz, 2015)

Software plus služby: Rozšiřují model SaaS o možnost mít některé aplikace nainstalovány lokálně. Tedy vyhoví zákazníkovi použití lokální a cloudové aplikace a umožní mu pro něj to nejlepší řešení. Data jsou dále umísťována a zálohována formou synchronizace v cloudu. Díky synchronizaci a možnosti mít aplikaci nainstalovanou na svém zařízení je zde možnost pracovat i off-line. Výhodné je to například pro pracovníky na dálku. Při práci s citlivými daty nemusejí být posílány do cloudu a ukládány na lokálním zařízení. (Velte a další, 2011)

1.5 Modely nasazeni

Modely nasazení určují, jak je služba v cloudu nasazena, jak se pracuje s daty v cloudu, jak jsou data uložena a kdo k nim může přistupovat. Některá z dat se totiž musejí uchovávat ve firmě a nesmějí opustit podnik. Základní modely nasazení mohou být: veřejný, privátní, hybridní a komunitní cloud.

1.5.1 Veřejný cloud

Je to nejrozšířenější a nejznámější model cloudu. Prostředky jsou zde poskytovány všem, kdo mají aktivní předplatné. Cloud je celý umístěn u poskytovatele a poskytovatel jej i spravuje.

Tento model je vhodný pro jednotlivce a začínající podniky, ale nemusí být dobrou volnou pro firmu, která má již vybudováno vlastní datové centrum.

1.5.2 Privátní cloud

Je vlastně opakem veřejného cloudu. Cloud je zde provozován firmou samotnou a firma si jej tedy spravuje sama. Je provozována na zařízeních firmy a z toho plyne výhoda pro firmy, které již mají vybudované vlastní datové centrum. Datové centrum je v takovém případě jen přebudováno, aby splnilo znaky cloudu a firma nemusí investovat velké peníze do budování datového centra. Prostředky jsou zde poskytovány jen firmě.

1.5.3 Komunitní cloud

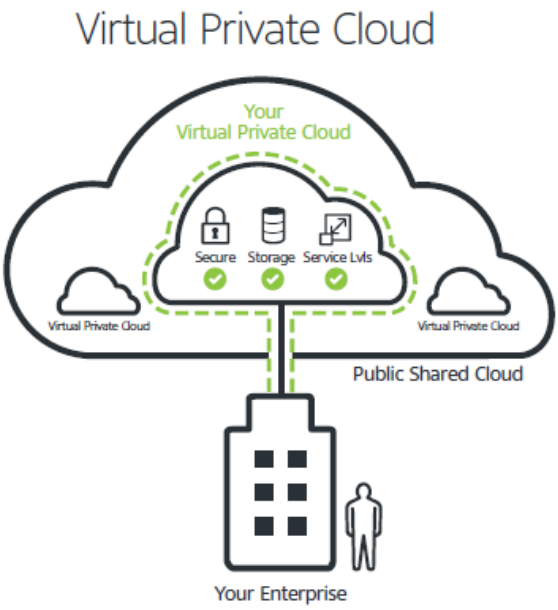
Tento model nasazení je především určen pro skupiny, které mají spolu sdílené zdroje. Jsou to především firmy, které mají stejné zájmy. Takže infrastrukturu cloudu mezi sebou sdílí více firem.

1.5.4 Hybridní cloud

Kombinuje spolu dva a více modelů nasazeni. Navenek se tato skupina tváří jako jeden cloud a mezi sebou komunikují za pomoci síťových technologií API. Jedná se o výhodné řešení, pokud je potřeba kombinovat vícero modelů nasazení. (Doherty, 2016) (National Institute of Standards and Technology, 2013) (Lacko, 2012)

1.5.5 Virtuální privátní cloud

Virtuální privátní cloud je vhodný pro společnost, která chce využít výhod privátního cloudu, ale chce ušetřit finanční prostředky jako tomu je u veřejného cloudu. Tedy společnost využije většího zabezpečení dat a nepotřebují vlastní infrastrukturu a personál, který by ho spravoval. Do tohoto typu cloudu se přistupuje prostřednictvím internetu, ale spojení je zabezpečeno za pomoci VPN a ke cloudu se dostane jen za pomoci něj. VPN vytváří zabezpečené spojení mezi účastníky komunikace a všechna přenášená data šifruje. (Viswanathan, 2012)



Obrázek 4 – Virtuální privátní cloud

Zdroj: (FileCloud, 2015)

1.6 Potenciální aktéři

Potenciální aktéři popisují podle organizace *National Institute of Standards and Technology* odpovědnosti aktérů v cloud computingu. Každý aktér představuje právnickou či fyzickou osobu, která má svoji určitou roli a provádí nějaký proces v cloudu. Podle organizace NIST[3] je pět aktérů: cloud consumer, cloud provider, cloud auditor, cloud broker a cloud carrier.

Cloud consumer je cloudový uživatel, který reprezentuje osobu či firmu používající služby od poskytovatele cloudu. Uživatel si nastavuje služby cloudu dle jeho potřeb. Podle služeb, které má nastavené je prováděno fakturování poskytovatelem.

Cloud provider je poskytovatel služeb cloudu. Může to být buď fyzická osoba, nebo právnická osoba, která cloud provozuje. Má za úkol zprostředkovat požadovaný typ služby na její požadované úrovni, chránit její zabezpečení a starat se o ni. Provádí správu cloudového datového centra.

Cloud auditor provádí nezávislé testy cloudových služeb. Do testů zahrnuje zabezpečení, výkon a soukromí cloudových služeb. Tedy kontroluje, jak se nakládá s daty, jak jsou data zabezpečena a jak je vše správně implementováno.

[3] https://www.nist.gov/

Cloud broker zprostředkovává cloudové služby. Slouží jako mezistupeň mezi cloud provider a cloud consumer, aby zákazník nemusel komunikovat přímo s poskytovatelem. Jednoduše se dá představit jako makléř nabízející různá řešení a správu cloudových služeb od různých poskytovatelů. Dokáže i často zajistit spojení služeb od různých poskytovatelů, aby byly pro zákazníka co nejlepší.

Cloud carrier slouží jako most mezi poskytovatelem cloudu a zákazníkem. Zajišťuje zabezpečené síťové spojení ke cloudu. Obvykle uzavírá smlouvu s poskytovatelem cloudu podle service level agreements. (National Institute of Standards and Technology, 2013)

1.7 Nejznámější poskytovatelé

Cloud provozuje mnoho společností a některé z nich již působí řadu let ve světě počítačů. Společnosti mohou poskytovat rozdílné služby a možnosti nastavení. Každá z firem může být specializována na konkrétní druh služby. Mezi nejznámější poskytovatele patří: Google, EMC, NetApp, Microsoft, Amazon, Salesforce.com, IBM a další. Někteří z nich budou v následujících odstavcích představeny.

Společnost Google nabízí především produkty týkající se webu. Proto se zaměřuje na webové aplikace, jako je webové úložiště nebo webové kanceláře. V jeho nabídce cloudových služeb lze nalézt služby od pronájmu celé platformy až po nástroje pro vývojáře. Specialitou řešení cloudu od této společnosti je jejich cloudová služba *App Engine*, která umožňuje budovat vývojářům celé webové aplikace v nejpoužívanějších jazycích. Umí využívat jak virtualizace, tak kontejnerů. Veškeré nastavení služeb je prováděno pomocí webové aplikace společnosti. (Velte a další, 2011) (Lacko, 2012) (Forrest, 2014)

EMC se zaměřuje na nejrůznější služby spojené se správou dat včetně cloud computingu. Součástí této společnosti je *VMware*, který se jako samostatná divize zaměřuje na virtualizaci. Je jednou z předních společností na trhu ve světě IT.

Organizace *NetApp* se zaměřuje na počítačová data a snižování nákladů za jejich správu. Jako první předvedla síťové úložiště a do cloudu se pustila také jako jedna z prvních společností. Pro zlepšení datových center uzavřeli partnerství se společností Cisco. Dokáže zajistit služby, jako jsou například operační systém, databáze, zabezpečí dat a software.

Microsoft zakládá své cloudové služby na produktech, které lidé používají i bez cloudu. Jedním z příkladů je třeba kancelářský balík *Office365*, který funguje na bázi předplatného jak na PC, tak ve webovém prohlížeči. Microsoft staví své cloudové služby především na platformě Azure,

kde provozuje všemožné modely služeb. Jeho služby jsou zaměřeny i na vývojáře, kteří zde najdou mnoho užitečných služeb pro vývoj a testování. Azure velmi dobře spolupracuje i s vývojářskou sadou Visual Studio. Mezi jeho služby se například může řadit: SQL services, .NET Services, Live Services, Exchange, Sharepoint a další.

Amazon je jeden z nejznámějších společností na poli cloudu. Dokáže poskytovat různé varianty napříč službami. Mezi jeho nejznámější služby například patří: Elastic Compute Cloud, SimpleDB, Simple Storage Service, CloudFront Simple Queue Service, Elastic Block Store.

Další jmenovanou společností je Salesforce.com. Pro poskytování cloudových služeb má platformu Force.com, která je určena především pro PaaS. Poskytuje také cloudové služby určené pro CRM. Společnost se soustředí převážně na tři hlavní produkty: The Sales Cloud, The Service Cloud a Your Cloud.

Poslední jmenovanou společností je IBM. Jako jedna z nejznámějších firem s dlouhou tradicí na poli serverů a datových center se také zaměřuje na cloud computing. Klientům pomáhají s přechodem z tradičních řešení do cloudových a poskytují jim odborné konzultace a řešení. Mezi poskytováním služeb u veřejných cloudů dokáží budovat i privátní, nebo hybridní řešení cloudu. (Lacko, 2012) (Velte a další, 2011)

1.8 Zabezpečení

Cloud pracuje s daty uživatelů a je potřeba tyto data nějak chránit. Při komunikaci s cloudem se musí řešit, jak zabezpečit komunikaci tak, aby ji nemohl přečíst každý. Dále musí být chráněna i samotná datová centra poskytovatele jak před útoky z vnějšku, tak z vnitřku. Jako dále se řeší i zabezpečení hardware před vnějšími vlivy. Bezpečnost v této oblasti je velice přísně řešena, protože je to jedna z konkurenčních výhod poskytovatelů a pokud u jednoho poskytovatele dojde k jejímu narušení, může to znamenat odliv zákazníků. Nástrojů, jak zabezpečit cloudové datové centrum je celá řada a v následujících odstavcích budou vybrané nástroje představeny.

Aby nemohl komunikaci s cloudem přečíst kdokoliv je pro zajištění bezpečné komunikace mezi uživatelem a cloudem využíváno autentizace a šifrování. Při komunikaci pomocí webového prohlížeče může být zabezpečení zajištěno například za pomoci protokolu SSL. Protokol SSL využívá certifikátu, který obsahuje údaje o majiteli a vydavateli. Certifikát musí podepsat jedna z certifikačních autorit. Je zde využíváno veřejného a privátního klíče. Privátní klíč zůstává v držení stanice a veřejný je veřejně známý. Stanice se navzájem ověří a vyjednají parametry, například jaký typ šifrování použijí.

U cloudového datového centra musí být zabezpečena nepřetržitá dodávka elektrické energie a nepřetržitá funkčnost chlazení. Takže každé datové centrum je vybaveno podpůrnou infrastrukturou. Do první skupiny, která se stará o nepřetržitý provoz patří: klimatizační jednotky, diesel agregáty a baterie pro zajištění funkčnosti datového centra, než naběhne diesel agregát. Funkčnost této podpůrné infrastruktury je klíčová, a proto jsou tyto systémy často zdvojené. Další podpůrnou infrastrukturou je samo-hasicí systém, kterým je vybaven každý serverový sál. Většinou se do místnosti, kde hoří, vpustí dusík, aby se zredukovala hladina kyslíku a požár nemohl hořet. Jednou ze samozřejmostí je také zálohování dat v datacentru, aby nedošlo v případě poruchy k jejich ztrátě. (Lacko, 2012) (Velte a další, 2011)

Při přistupování jednotlivých uživatelů do cloudu je nutné spravovat jejich bezpečnostní politiku a pravomoc. Každý uživatel se musí důsledně autentizovat a autorizovat vhodným typem úrovně zabezpečení korespondující s úrovní důležitosti dat. Musejí se řešit přístupová práva (role) jednotlivých uživatelů. Striktně vymezit co uživatel smí, nesmí, kam může a nemůže přistupovat. To vše za důsledkem, aby uživatelé měli přístup jen k vlastním datům.

Data v cloudu musejí být také zabezpečena, a proto se řeší několik bezpečnostních politik. Řeší se oddělení dat, které se stará o izolaci dat mezi uživateli za pomoci virtualizace, šifrování a granulární řízení přístupu. V cloudu se také musí řešit granulární bezpečnost dat u jednotlivých položek dat. Ochrana musí být kladena především na citlivá data, tedy data se musejí klasifikovat. Monitoring a audit funguje všude, kde se pracuje s citlivými informacemi a stará se, aby citlivá data byla auditována za pomoci logů. (Bezpalec, 2014)

Datová centra musí být chráněna i před viry, malware, útoky a dalšími pokusy o prolomení přístupu ke cloudu. Potenciálním rizikem může být i zaměstnanec firmy, který úmyslně nebo z neznalosti, či neopatrnosti může svým působením negativně ovlivnit zabezpečení. Příkladem může být otevření zadních vrátek do systému nebo zneužití dat. Velkou hrozbou pro cloud je *Distributed Denial of Service*, který sice nezcizí žádná data, ale dokáže zpomalit nebo vyřadit značnou část služeb cloudu. DDoS může mít poté pro koncového zákazníka fatální následky z důvodu nefunkčnosti některých z jeho služeb. (Waterford Technologies, 2016)

Cloud má ze své povahy sdílené prostředky a tyto sdílené prostředky mohou být napadeny. Následný dopad této hrozby může být pro cloud velmi velký, protože se může jednat o chybu u několika stovek zákazníků. Toto může způsobit napadení systémů, ale i nechtěná chyba v konfiguraci. Úplná ztráta dat v cloudu je z hlediska chyby HW nulová, jediná možnost

smazání je napadení útočníkem. Poslední možností je napadení API, které mohou sloužit jak ke komunikaci se zákazníkem, tak mezi komponenty cloudu. (Rashid, 2016)

1.9 Předpisy a legislativa

Cloud podléhá spoustě zákonům a nařízením. Nejčastěji se jedná o zákony na ochranu osobních údajů a manipulaci s daty. Poskytovatelé také často garantují dostupnost a kvalitu svých služeb.

Ve většině případů jsou data v cloudu ukládána mimo stát zákazníka. V takovém případě je poskytovatel cloudových služeb povinen tuto skutečnost oznámit zákazníkovi. V rámci států Evropské unie je možnost data ukládat bez informování zákazníka i bez souhlasu Úřadu pro ochranu osobních údajů za předpokladu, že je i datové centrum umístěno v rámci EU. Při práci s citlivými údaji musí zákazník s poskytovatelem uzavřít smlouvu o zpracování osobních údajů. V rámci EU je jasně stanoveno směrnicí[4], co se má v rámci ochrany osobních údajů dodržovat.

Data se mohou ukládat ale i do zemí mimo EU, a to už je rizikovější, protože například v USA mohou podle jejich zákonů vládní subjekty požadovat data ze serverů na svém území. Takzvaně pokud jsou data poskytována do zemích mimo EU, je vyžadován souhlas úřadu pro ochranu osobních údajů. Výjimkou jsou státy, které garantují záruku bezpečnosti dat. Tedy státy, které ratifikovali úmluvu o ochraně osobních dat. V ostatních případech postupuje úřad podle zákonů nebo musí mít vyložený souhlas od subjektu, který data vlastní.

V rámci cloudu je tato skutečnost velice obtížná, protože cloudové datové centrum může být rozložené po celém světě a je proto obtížné zaručit, kde data budou uložena. Proto musí být vybrán poskytovatel, který dokáže zaručit zákonné zpracování dat vzhledem k jejich umístění. Za nedodržení podmínek zpracování osobních údajů hrozí v rámci EU velké pokuty. Takže pokud si firma (zákazník) vybere poskytovatele, který tuto skutečnost neumožňuje, může za něj dostat firma pokutu. Při výběru cloudu, kde se bude pracovat s osobními údaji, je velice dobré si pečlivě vybrat poskytovatele a smluvně tuto skutečnost podchytit. (Kreisl a další, 2016)

Poskytovatel cloudu se zavazuje k určené kvalitě služeb (SLA). Při nedodržení SLA je zákazník nějakou formou kompenzován. Nejčastěji se formuluje garance dostupnosti služeb. SLA definuje hranice dodávané služby, podmínky a odpovědnosti. Skládá se z těchto tří základních částí:

- Základní specifikace, podmínky a pravidla (poskytovatel, měření, popis)

[4] Směrnice EU 95/46/ES

- Tvrdé metriky (dostupnost, doba odezvy)
- Měkké metriky – ostatní

(Učeň, 2002) (Kmoch, 2014)

Některé dokumenty mohou mít u sebe Copyright (autorská práva) a to přináší další řadu legislativních omezení. Při ukládání dat mohou nastat právní problémy při nedodržení právních vztahů. Kolem tohoto tématu jsou vedeny velké soudní spory. (Solmecke, 2013)

Od května 2018 bude pro všechny země Evropské Unie platit nový zákon *General Data Protection Regulation*. V tomto zákonu EU zpřísňuje podmínky nakládání s osobními daty. Vlastní zákony zemí, které nařizují, jak nakládat s osobními údaji budou od nástupu GPDR neplatné. Zákon se týká veškerých společností zpracovávající osobní údaje. Dnešní zákon nepamatuje na zpracovatele osobních údajů, jako jsou například společnosti poskytující cloud. Musí být umožněno úplné vymazání dat, změny dat a další podmínky. Společnosti budou muset kdykoliv doložit GPDR, jak nakládají s údaji a jak jsou zabezpečena (Mika, 2017)

2 VIRTUALIZACE

Virtualizace je velmi hojme používaná při tvorbě cloudového datového centra, přestože to není podmínka cloudu. Díky samotné virtualizaci se dá lépe hospodařit se zdroji, a proto je vhodným kandidátem při použití v cloudu. Virtualizace spolu s dalšími prostředky cloudu dokáže velmi dobře ušetřit zdroje.

2.1 Co je virtualizace

Virtualizace je jedna z často používaných technologií na poli IT. Tato technologie je vytvořena proto, aby umožnila spouštět více různých operačních systémů na jednom PC. To se většinou provádí za účelem snížení nákladů za HW, nebo testováním různých operačních systémů. Díky umístění strojů na jednom místě je také usnadněna správa systémů.

Většinou není jeden server využit na sto procent a jeho prostředky by byly jinak nevyužity, ale díky virtualizaci může jeho prostředky využít virtuální stroj. Virtualizace je vytvořena skupinou spolupracujících společností, kteří vytvořili dohromady jeho standart. Virtuální počítač si lze velice zjednodušeně představit jako aplikaci běžící v počítači.

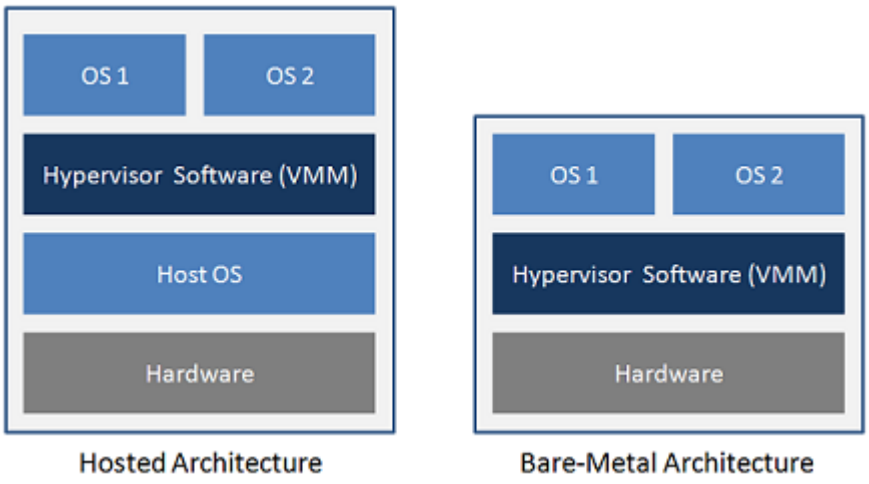
Virtuální počítač zahrnuje všechny vlastnosti fyzického počítače. V podstatě emuluje veškeré hardwarové vybavení počítače a na něm se posléze mohou spouštět operační systémy. Všechny běžící instance virtuálních počítačů jsou od sebe odděleny, jako by operační systém doopravdy běžel na jiném počítači. Díky tomu, že jsou virtuální stroje od sebe odděleny, nezpůsobí výpadek jednoho z nich výpadek ostatních. Protože se jedná o virtuální stroj, tak může být pozastaven, uložen jeho stav na disk, přenesen do jiného počítače, nebo vytvořena jeho kopie. Virtuální stroj se snadno migruje na jiná zařízení a usnadňuje opravy v případě poruchy serveru, na kterém běží virtuální stroje. (Doherty, 2016) (Lacko, 2012) (Velte a další, 2011)

2.2 Základní principy virtualizace

Pro přístup ke zdrojům využívají virtuální počítače virtualizační vrstvu. Tato vrstva představuje virtuální připojení k fyzickým zdrojům. Nazývá se *Hypervizor* a určuje dva základní přístupy k virtualizaci.

Prvním typem je „Bare Metal“, neboli nativní hypervizor. Virtualizační prostředek zde běží přímo nad hardwarem. Je zde možné provádět plnou virtualizaci, to znamená, že hardware je ovládaný přímo hypervisorem. Operační systém se zde spouští až nad hypervisorem.

Druhým typem je „Host Based“ neboli hostovaný hypervizor. Je zde hostitelský operační systém, na kterém teprve běží hypervizor. V tomto typu je o jednu vrstvu více z důvodu existence hostujícího operačního systému. (Doherty, 2016) (Lacko, 2012) (Velte a další, 2011)



Obrázek 5 − Hypervizor

Zdroj: (National Instruments Corporation, 2016)

Pro provádění virtualizace je několik způsobů:

1. Pomocí softwarového emulátoru, který emuluje celý počítač. Emuluje procesor a jeho instrukce, přístup na sběrnice, paměť, tedy vytváří rozhraní a transformace pro skutečný hardware. Protože je zde počítač kompletně emulován, je možné spustit i systém jiné architektury, ale za cenu výkonu.
2. Paravirtualizace vyžaduje zásah do jádra virtualizovaného systému. Musí se v jeho jádře vytvářet speciální rutiny pro přesměrování do hypervizoru. Je to složitější cesta pro nasazení z důvodu úpravy OS.
3. Virtualizace na úrovni operačního systému podporuje pouze jeden operační systém. Je tedy vždy využíváno jen jedno jádro OS. Tohoto principu využívají kontejnery a budou blíže popsány ve vlastní kapitole.
4. Plná virtualizace se chová, jako by operační systém běžel doopravdy na hardware a nevyžaduje po hostovaném počítači žádné modifikace.
5. Virtualizace s podporou hardwaru je vylepšením výkonu díky menší zátěži hypervizoru a přenesení zodpovědnosti přímo na hardware. Pro podporu této virtualizace je vyžadováno využití procesorů podporujících *Intel-VT*, nebo *AMD-V*.

(Doherty, 2016) (Lacko, 2012) (Pomazal, 2010)

2.3 Řešení virtualizace

Virtualizací se zabývá celá řada společností. Každá ze společností nabízí vlastní produkty pro řešení virtualizace. Každý z těchto produktů může být vhodnější při jiném nasazení. V následujících dvou odstavcích budou probrány jedny z nejvýznamnějších produktů na poli virtualizace.

Jako nejznámější společnost zabývající se virtualizací je VMware. Tato společnost poskytuje jak bezplatné produkty, tak i placené produkty, které jsou určené především pro profesionální použití. Mezi jeho nejznámější produkty se mohou řadit: VMware Server, VMware workstation a VMware Infrastructure.

Dalším známým produktem je Hyper-V od společnosti Microsoft. Produkt je nabízen uživatelům systému Windows od verze Windows Server 2008. Hyper-V poskytuje zákazníkům vysoký výkon, spolehlivost a škálovatelnost zdrojů. (Velte a další, 2011)

3 KONTEJNERY

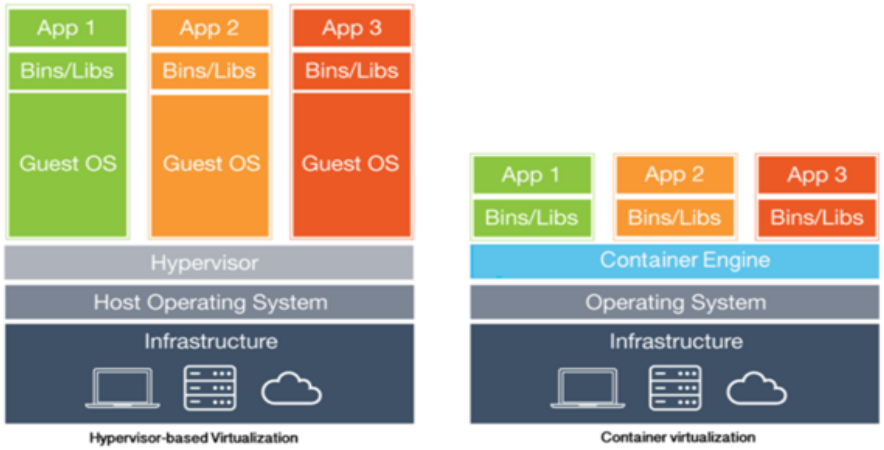
Moderní cloud není budován ve spolupráci s virtuálními stroji, ale jde kupředu a využívá nové možnosti a technologie, které jsou mu nabízeny. Mezi tyto nové technologie můžeme řadit kontejnery.

3.1 Úvod

Kontejnery jsou aplikace běžící v izolaci, a kromě jádra mají jen omezený přístup k systémovým prostředkům. Pracují vždy na jednom OS a využívají jeho jádro pro svoji činnost, tedy jádro hostujícího systému plánuje úlohy pro všechny kontejnery. Pro funkčnost kontejnerů je přítomen kontejner engine. Jako první kontejnery lze označit FreeBSD Jails a Solaris Zones ze kterých ostatní vycházejí. Kontejner je soubor technologií, které mají zajistit izolaci některých procesů od OS v Linuxu za pomoci namespaces a cgroups.

Kontejnery obsahují jen potřebné izolované aplikace a potřebné knihovny. Kontejnery se spouští na jádře stávajícího OS. Protože je kontejner izolován, nedokáže ovlivňovat stávající systém. Aplikace, která se bude v kontejneru spouštět, musí být určena pro jádro OS, na kterém kontejner běží. Příkazy se spouští přes jádro hostitele a jeho procesy jsou v něm viditelné.

Kontejnery si začínají získávat velkou oblibu v prostředí, kde je požadováno optimalizovat přidělování zdrojů. Oproti virtualizaci nemusejí spouštět vlastní jádro OS a nemusí mít hypervizor, takže ušetří značné množství zdrojů. Kontejnery by neměly přistupovat do hlavního systému za pomoci práv superuživatele, protože aplikace kontejneru by pak mohly ovlivnit i jiný kontejner nebo dokonce hostovaný systém. (Wang, 2016) (Computer world, 2014) (Banerjee, 2014)



Obrázek 6 – Porovnání virtualizace a kontejnerů

Zdroj: (Wang, 2016)

První kontejnery byly určeny především pro systém Linux a s příchodem Windows server 2016 se kontejnery dostaly do základu systému i zde. Windows server 2016 nabízí dva typy kontejnerů, které se liší především mírou izolace. Windows server kontejner sdílí jádro hostitelského systém, tedy v dnešní době je to jádro Windows serveru 2016. Zatímco Hyper-V kontejner sdílí jádro systému běžícího ve virtuálním stroji, tedy dokáže zajistit ještě větší izolaci a možnost spuštění kontejneru na jádře rozdílného OS jako je například Linux. Windows využívají především kontejner *Docker*. (Sheldon, 2016)

3.2 Výhody a nevýhody

Kontejnery mají své výhody a nevýhody. Většina z nich je porovnávána s virtualizací a zaměřuje se na hlediska optimalizace a bezpečnosti.

Mezi hlavní výhody kontejnerů můžeme řadit:

- Jsou velmi malé a tím dokáží ušetřit místo (kontejner pár MB a VM řádově GB).
- Nepotřebují pro svůj běh vlastní jádro a tím ušetří systémové zdroje.
- Oproti virtuálním strojům je jejich nasazení několikrát rychlejší.
- Lepší sledování běhu a ladění aplikací.
- Jejich zapnutí je velmi rychlé oproti virtuálnímu stroji.
- Mohou velmi ušetřit náklady.

Mezi hlavní nevýhody kontejnerů patří:

- Oproti virtuálnímu stroji nedosahují takové izolace (společné jádro OS pro kontejnery).
- Dokáží spustit aplikaci určenou jen pro jádro hostitelského systému.
- Obtížnější nastavování politik zabezpečení u sítě.

(flow.ci, 2016) (Computer world, 2014)

3.3 Kontejnery

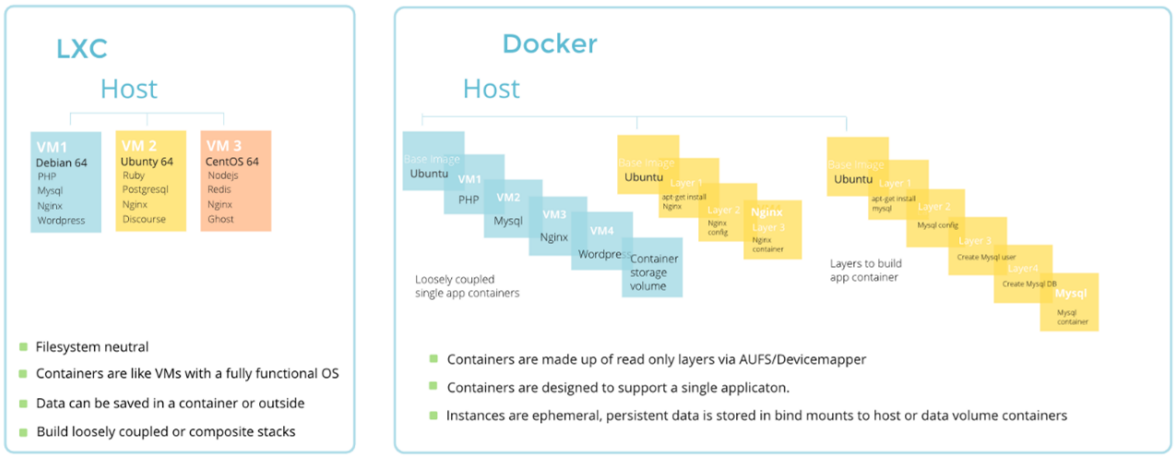
Mez první linuxové kontejnery se může řadit LXC. Umožňuje spustit více systémů Linux na jednom hostujícím OS. Do kontejneru se nainstaluje minimální verze operačního systému a do něj se posléze instalují aplikace. LXC samozřejmě poskytuje nástroje na správu kontejnerů.

Docker[5] jde oproti LXC ještě dál a zvedá úroveň minimalizace a usnadňuje používání. Založen byl na LXC kontejneru. Spojil více technologií dohromady a vytvořil jednotné API pro správu. Umožňuje na sebe stavět jednotlivé komponenty aplikací a nepotřebuje vždy vlastní další

[5] https://www.docker.com/

prostředky. Takže z jednotlivých součástí kontejnerů vytváří vrstvy neboli jemné prostory. Díky jmenným prostorům dokáže vytvořit prostor pro kontejner určený přímo pro jednu aplikaci kontejneru. Je označován jako nejrozšířenější a nejznámější kontejner. (Wang, 2016) (Banerjee, 2014) (Kubica, 2016)

Mezi další kontejnery může patřit LXD, který vylepšuje LXC o lepší zabezpečení a izolaci, dále vylepšuje uživatelské prostředí. Kontejner RKT je určen především pro Linuxové clustery a je součástí CoreOS. Poslední dva kontejnery jsou určeny pro speciální distribuce Unixu a jsou to: FreeBSD Jails a Solaris Zones. (Wang, 2016) (Betz, 2016)



Obrázek 7 − Porovnání LXC a Docker

Zdroj: (Banerjee, 2014)

3.4 Orchestrátory

Orchestrátory se snaží integrovat do sebe vícero aplikací, či služeb. Mají pomoci uživatelům s nasazením, spravováním a škálováním aplikací. Dále dokáží řídit, kam a jak se budou organizovat kontejnery mezi vícero zařízeními infrastruktury. V prostředí kontejnerů jsou velmi využívané, ale s orchestrátory se můžeme setkat i jinde, například i u virtuálních strojů. Mezi nejznámější orchestrátory pro kontejnery patří: Kubernetes, *Docker Swarm*, Amazon ECS, Azure Container Service, Marathon, CoreOS Fleet, Open Stack Magnum, Diego, Hashicorp Nomad a Google Container Engine. (flow.ci, 2016) (Kubica, 2016) (Ankerholz, 2016)

4 RASPBERRY PI

4.1 Cíle Raspberry Pi a jeho historie

Raspberry Pi je miniaturní počítač, která má za svůj cíl poskytnou PC všem. Toho chce dosáhnout díky své nízké ceně a tím zajistit lepší dostupnost vzdělávání v informačních technologií a programování. V dnešní době se na něm dají provozovat různé druhy operačních systémů a může nahradit domácí počítač, nebo dokonce se z něj může vytvořit server či multimediální zařízení k TV. Za zrodem tohoto zařízení stojí členové Raspberry Pi Foundation. (Upton a další, 2016) (Hůna, 2016)

Obrázek 8 − Raspberry PI

Zdroj: vlastní

Myšlenka tohoto zařízení vyšla z nutkání zlepšit vzdělávací systém v Británii a připravit žáky lépe na technické vysoké školy a jejich následné uplatnění na trhu práce. Tvůrce (Eben Upton) během svého působení na Cambridgeské univerzitě postupně zaznamenával pokles úrovně dovedností u nových studentů informatiky. Uchazeči v dřívějších letech znali několik programovacích jazyků, ale postupem času se úroveň znalostí programování u uchazečů propadala. V roce 2005 byl propad znalostí programování takový, že byly rádi za znalost HTML. Autor byl touto situací ve školství inspirován a chtěl toto zařízení rozdávat uchazečům, kteří by s ním mohli předvést něco nového a něco se přiučit.

Prvotní verze zařízení nepřipomínaly zařízení, které známe dnes. Byly to pájené obvody na stole a měly znatelně menší výkon. Po odchodu do komerční sféry byla situace s uchazeči o práci obdobná jako u vysokoškoláků. Při autora setkání s několika kolegy byla těmito lidmi založena již dnes známá organizace jménem Raspberry Pi Foundation. Název vytvořila dozorčí rada a je tvořena slovem „malina“, ovoce je v názvu IT společností hojně používané a Pi je dán ze slova Python. Na začátku projektu se předpokládalo, že Python bude jediný používaný programovací jazyk zařízení.

Dalším postupem organizace bylo vytvoření takového zařízení, které by děti lákalo používat. Tedy poskytnout nejrůznější funkce, které by alespoň některé žáky přivedlo dále k programování. K zařízení byla přidána možnost použití periferií a k prototypu byl navíc přidán modul fotoaparátu, který demonstroval možnost připojení periferií. V roce 2011 publikoval Rorry Celan-Jones na svém blogu video, které ukazuje Raspberry Pi za cenu 25 dolarů. Video se velmi rychle rozšířilo a zařízení bylo známé po celém světě. Základním cílem návrhu bylo, aby zařízení umělo co nejvíce věcí za co nejmenší cenu a nepotřebovalo k sobě další drahé věci. Tedy aby pro svoji funkčnost využilo věcí, které jsou v běžné domácnosti. Postačila televize, SD karta a nabíječka od telefonu.

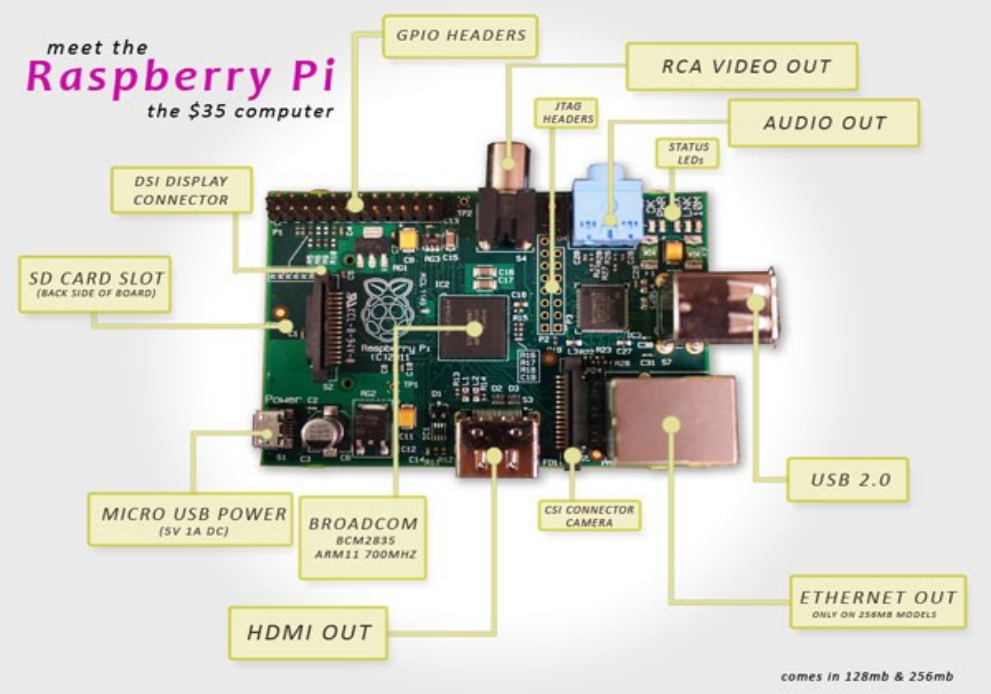
V roce 2012 se již dolaďovali poslední detaily. Zařízení zprvu běželo jen na Linuxu a bylo nutno připravit a otestovat software. Posléze se mohla vydat první verze zařízení, která měla sloužit především ke vzdělávací účelům. Do dnešní doby prošlo toto zařízení spoustou vylepšení a má několik verzí a mnoho možností využití. (Upton a další, 2016)

4.2 Seznámení s Raspberry Pi

Zařízení má velikost přibližně kreditní karty. Tato velikost je důsledkem důmyslného návrhu a technologického pokroku. I přes svoje malé rozměry poskytuje dobrý výpočetní výkon, o který se stará čip přibližně uprostřed desky. Čip je založen na ARM architektuře a dodává ho společnost Broadcom. Dále nesmí chybě samozřejmě paměť RAM, která se nachází v blízkosti čipu. Pro zobrazení je na desce umístěn port HDMI a pro starší zobrazovací zařízení i kompozitní port.

Dále je umístěn 3,5 mm Jack pro přenos zvuku. Tento port je také využíván při použití starších zařízeních bez HDMI, protože kompozitní port umí přenášet pouze video na rozdíl od HDMI. O možnost použití trvalého úložného prostoru se stará slot pro SD kartu. Celá deska je napájena z jednoho micro USB a po jejím připojení se zařízení ihned začne spouštět. Pro indikaci napájení slouží dioda na straně zařízení. Pro připojení kamery je zde umístěna sběrnice CSI,

celým názvem Cammera Serial Interface. Dále je zde umístěn konektor GPIO (*general-purpose input-output*), tento konektor je používán pro rozšíření Raspberry Pi o nové moduly. Nakonec je zde umístěno několik USB portů a jeden RJ45 pro možnost připojení zařízení k síti. V průběhu inovování přicházejí v nových revizích další možnosti a vylepšení, které umožňují použít integrovanou Wi-Fi 802.11n s Bluetooth 4.1. (Upton a další, 2016)



Obrázek 9 − Pohled na desku

Zdroj: (Donovan, 2013)

GPIO konektor obsahuje sériovou linku, I2C, SPI a 8 GPIO linek. K pinům GPIO se dá například připojit stripboard s dalšími součástkami a vytvořit si svůj vlastní jednoduchý odvod ovládaný pomocí Raspberry Pi. Pro ovládání pinů GPIO se nejčastěji používá jazyk Python. U prvních revizí se používal port GPIO s 26 piny, ale dnes v novějších revizích je používáno 40 pinů.

Dalším využitím portu GPIO je možnost pomocí něj rozšířit Raspberry Pi o další desku. Pro připojení rozšiřující desky se použije port jako celek, a nejen jeho jednotlivé piny. První kategorií rozšiřujících desek jsou desky, které umožňují pouze snazší manipulaci s GPIO piny a jednoduché připojení konkrétních komponent určených pro rozšiřující desku. Do této kategorie se mohou řadit například: Slice of Pi a Prototyping Pi Plate. O další skupině se dá říci, že je opravdu rozšiřující a přidává funkcionality navíc. Například Gertboard od společnosti

Fen Logic umožňuje přístup k dalším funkcím čipu, které nejsou normálně dostupné. Na sobě má různé prvky navíc, jako jsou například převodníky a diody.

Optický modul je u Raspberry Pi již tradicí a autoři s ním počítali již od začátku, ale pro udržení nízké ceny nebyl zahrnut do základní desky. Ale na desce je pro něj připraveno speciální rozhraní CSI určené přímo pro tento modul. Samozřejmě se dá kamera zapojit i do rozhraní USB, ale toto řešení bude mít větší spotřebu. Většina distribucí OS pro Raspberry Pi již s tímto počítají v základu a obsahují potřebné balíčky. Tento modul byl například použit pro pořizování snímků z metrologického balónu. (Upton a další, 2016)

4.3 Verze

Základní desky Raspberry Pi se podle svého rozložení, výkonu a výstupů které poskytují, dají řadit do svou základních kategorií. Model A se vyznačuje nízkou cenou a nízkou spotřebou. Má jen jeden port USB, nemá port pro Ethernet a poskytuje menší výkon. Zatímco Model B má na desce 2 USB porty, port pro Ethernet a také poskytuje větší výkon. Model B může mít v nových revizích i 4 USB porty. Jinak tyto modely mají totožné rozložení desky.

Oba tyto modely byly později modernizovány o verzi plus, tedy Model A+ a Model B+. Oba zachovávají vlastnosti uvedené výše, ale jejich desky byla přestavěny. Přišly o port kompozitního videa, který se nyní dá v případě potřeby zajistit na 3,5 mm Jacku, který se přesunul na druhou stranu desky a umožnil rozšíření GPIO z 26 pinů na 40. Dále byl ještě slot pro SD kartu nahrazen slotem pro micro SD kartu.

Nejnovější modely Raspberry Pi – 2 a 3 vycházejí z modelu B+. Poskytují 4 USB porty a stejné rozložení základní desky. Oba modely postupně dosáhly znatelně vyšších výkonů než jejich předchůdci. Hlavním rozdílem mezi Pi 2 a 3 je přidání možnosti bezdrátové komunikace pomocí Wi-Fi a Bluetooth, u 3 verze to znamenalo přesunutí kontrolních diod na druhou stranu a na jejich původní pozici byly přidány antény pro Wi-Fi a Bluetooth.

Novodobým nástupcem Modelu A se stal Raspberry Pi Zero, který dosahuje opravdu malých rozměrů a spotřeby. Také byl navýšen jeho výkon.

Poslední samostatnou kategorií jsou Raspberry Pi Compute Module. Připojuje se do speciální desky IO Board. Samostatný modul neposkytuje žádné vstupy a výstupy, pouze výpočetní výkon. Je velký přibližně jako DDR2 SO-DIMM a pomocí něj se i připojuje na desku. Je současně vyroben ve třech verzích a jeho poslední nejvýkonnější verze Compute Module 3 pokutuje obdobný výkon jako Raspberry Pi 3. (Upton a další, 2016) (Hůna, 2016) (Ježek, 2017)

Tabulka 1 − Porovnaní verzí Raspberry Pi

	Model A	Model A+	Model B	Model B+	PI 2	PI 3	Zero
Chip	BCM2835	BCM2835	BCM2835	BCM2836	BCM2836	BCM2837	BCM2835
Procesor	ARM 32-bit, 1 jádro, 700 MHz	ARM 32-bit, 1 jádro, 700 MHz	ARM 32-bit, 1 jádro, 700 MHz	ARM 32-bit, 4 jádra, 900 MHz	ARM 32-bit, 4 jádra, 900 MHz	ARM 64-bit, 4 jádra, 1200 MHz	ARM 32-bit, 1 jádro, 1000 MHz
RAM	256 MB	256 MB	512 MB	512 MB	1 GB	1 GB	512 MB
Typ úložiště	SD karta	MicroSD	SD karta	MicroSD	MicroSD	MicroSD	MicroSD
GPIO	26	40	26	40	40	40	40
USB	1x USB 2.0	1x USB 2.0	2x USB 2.0	4x USB 2.0	4x USB 2.0	4x USB 2.0	1 MicroUSB
Ethernet	10/100 Mbit/s	Ne	10/100 Mbit/s	10/10 je 0 Mbit/s	10/100 Mbit/s	10/100 Mbit/s	Ne
Audio	HDMI, 3.5mm Jack	HDMI, 3.5mm Jack	HDMI, 3.5mm Jack	HDMI, 3.5mm Jack	HDMI, 3.5mm Jack	HDMI, 3.5mm Jack	MiniHDMI
Video	HDMI, composite	HDMI, composite	HDMI, composite	HDMI, composite	HDMI, composite	HDMI, composite	MiniHDMI
Wifi	Ne	Ne	Ne	NE	NE	802.11n	Ne
Bluetooth	Ne	Ne	Ne	NE	NE	4.1	Ne

Zpracováno dle: (RPiShop.cz, 2017)

4.4 Použití a dostupné operační systémy

Raspberry Pi má velmi široké spektrum použití. Typ použití tohoto zařízení záleží na vhodném výběru operačního systému. Některé systémy se mohou hodit více jako pracovní stanice, některé jako multimediální centrum k TV a jiné zase jako server. Všechny prvotní distribuce vycházejí ze systému Linux. Prvním používaným systémem byl *Rasbian* neboli upravený Debian pro ARM architekturu.

Pro jednoduchou instalaci jsou vytvářeny speciální SD karty označovány slovem NOOBS. Tyto karty obsahují jednoduchý instalátor s možností vybrat si jeden z nejpoužívanějších systémů. Seznam systémů je generovaný jak ze systémů, které jsou obsažené na SD kartě, tak ze systémů dostupných online. Po potvrzení se systém automaticky nainstaluje a je již připraven k použití. Takto nainstalovaný systém se dá snadno změnit za jiný, při bootování stačí podržet klávesu shift a znovu se objeví nabídka s výběrem systému, který se má nainstalovat jako tomu bylo při prvotním zapnutí zařízení. (Hůna, 2016)

Při použití Raspberry Pi jako PC sice nedokáže dosáhnout stejného výkonu jako běžný počítač, ale na surfování po internetu, kancelářskou činnost a učení je dostačující. Pro aplikace, které vyžadují vyšší výkon, se dá také využít cloudových služeb a spustit aplikace skrze ně. Mezi nejznámější distribuce používané pro tyto účely jsou *Raspbian* a Ubuntu Mate.

V základních sadách aplikací se dá najít software, který využívá většina lidí. Ale hlavním cílem tohoto zařízení je vzdělání, proto je zde obsažena aplikace Scratch a dále možnost programování v Pythonu. Scratch je cílen pro ty nejmenší a ukazuje, jak se dá grafickou formou programovat pomocí přetahování stavebních bloků myší. Ačkoli je tento jazyk jednoduchý dají se na něm postavit zajímavé programy. Tento jazyk se dá obohatit i o využití rozšiřitelného HW, který se dá připojit k zařízení a umožní používat různé senzory či robotické sady. Jazyk Scratch má naučit programátorskému myšlení a připravit na další jazyky, jako je například Python.

Díky svému malému rozměru a tichému chodu je proto ideální volbou použít Raspberry Pi v domácnostech jako multimediální centrum (HTPC). Dokáže přehrávat hudbu, video, fotky a má mnohé další funkce podobající se Smart TV. U většiny distribucích sloužících jako HTPC je samozřejmostí doinstalování funkcionalit prostřednictvím nějakého repositáře. Zařízení pak dokáží přehrávat média ze sítě a po síti jej také sdílet. Mezi jedny z předních distribucí patří OpenELEC, Raspbmc, Osmc a LibreELEC.

I přes výkon, který nedosahuje úrovně běžných počítačů se může použít jako domácí server, kde je jeho výkon pro toto použití dostačující. Takže lze využít na nejrůznější testování a malé domácí projekty. Může se například využít jako webový, databázový a FTP server. Ale rozhodně se najde spousta dalších, jako je například DHCP, nebo sdílení souborů. Pro použití Raspberry Pi jako serveru se dají využít všechny distribuce založené na Linuxu, ale nejznámější bude Rasbian. (Upton a další, 2016) (Valášek, 2016) (Voříšek, 2012)

Dalším použitím je řízení elektroniky pomocí GPIO, takzvaná jednoduchá automatizace. Tato vlastnost se dá umocnit díky rozšiřující desce UniPi board a Raspberry Pi se může začít používat jako plnohodnotné PLC. Protože má teď velký počet vstupních a výstupních relé, tak umí měřit a ovládat vícero zařízení a poskytuje stejné připojení jako běžné PLC. Pro řízení může být použit řídící systém REX, který nabízí komplexní řešení. (raspi.cz, 2014) (REX Controls, 2017)

Od ostatních distribucí operačních systémů vyčnívá Windows 10 IoT, protože není jako jediný systém pro toto zařízení postaven na Linuxu. Jak již název napovídá, běží zde Windows 10

a systém je poskytován zdarma. Systém je primárně určen pro internet věcí, tedy pro síť mezi sebou komunikujících zařízeních, které mohou být vzdáleně řízeny. Po startu systému se zobrazí jedna určitá universální aplikace (UWP), tato aplikace se dá pro zařízení naprogramovat a koná určitou definovanou činnost a může využívat i GPIO Raspberry Pi. Výhoda UWP aplikací je, že dokáží běžet na všech zařízeních stejně. (Hůna, 2015) (Bechynský, 2015)

Distribucí pro Raspberry Pi je ale mnohem více a stále přibývají. Pro účely výuky je pro Raspberry Pi například vytvořena speciální distribuce PiNet, která slouží ve třídách pro interakci žáků s učitelem. Dále se dá najít distribuce pro provádění penetračních testů nazvaná Kali Linux. Nebo Pidora, což je upravená Fedora pro Raspberry Pi. Spoustu dalších nebylo jmenováno a jsou k nalezení na internetu. (Singh, 2017) (Upton a další, 2016)

5 NAS

5.1 Seznámení s NAS

Na začátku by se mělo rozdělit úložišť na typy: DAS, NAS a SAN. DAS má HDD přímo připojen k sobě a pro připojení využívá rozhraní SCSI. NAS pracuje na běžné síti a poskytuje data do své sítě. SAN pracuje na vlastní oddělené síti, nejčastěji Fiber Chanel. Dále SAN dosahuje vysokých rychlostí, protože po jeho síti nejde žádný jiný provoz a dokáže fungovat na velké vzdálenosti.

NAS neboli Network-attached storage je síťové úložiště. Pracuje v síti typu Ethernet. Vyznačuje se jednoduchostí uživatelského nastavení. Dá se říci, že je to specializovaný počítač založený z většiny případů na systému Linux. Mezi hlavní výhody patří nízká spotřeba, malá hlučnost, snadný přístup k datům po síti, možnost podpory technologie RAID, možnost nastavování uživatelských oprávnění a možnost spravovat historii souborů. Pro dobrou funkcionalitu se doporučuje do NASu dávat HDD, které jsou specializovány pro zařízení typu NAS.

Většina NAS má dnes vlastní AppCenter, který slouží pro správu aplikací. Tyto aplikace poskytují navíc možnosti, jako jsou: streamování multimédií, práce s dokumenty, synchronizace s cloudem, nebo dokonce svůj vlastní privátní cloud. Toto nejsou všechny dostupné aplikace a existuje jich mnohem více. Ty nejnovější dnes již mají i port HDMI a mohou zastávat i funkce multimediálního centra.

Pro správu domácích NAS serverů se většinou využívá webového rozhraní. Webové rozhraní se tváři jako operační systém v prohlížeči, poskytuje jak základní nastavení tak i cenné informace. Protože je systém z většiny případů založen na systému Linux je zde možnost přistupovat a nastavovat i skrze protokol SSH. Možnost přístupu pomocí konzole dává další možnosti, jako je například přidání vlastních skriptů.

NAS pro svoje připojení k síti využívá nejčastěji GLAN, někdy je těchto rozhraní více a mohou zajišťovat například i load-balancing. Připojení pomocí GLAN zajišťuje vysokou propustnost dat. Některé mají i tu možnost, že se připojí za pomoci USB a k PC a chovají se jako externí HDD. Dále mají NAS vlastní USB port či porty, které se dají například využít pro připojení externího HDD, nebo tiskárny.

Vlastní cloud na NAS není v dnešní době žádný problém. Přední výrobci tuto možnost již poskytují pomocí jejich předem připravených balíčků, tedy stačí jen cloud na NAS nainstalovat

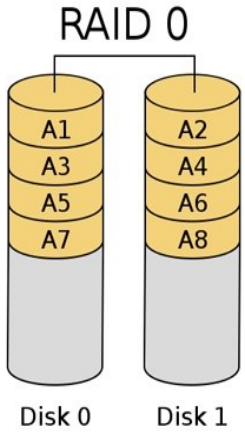
a nastavit. Takže naše data mohou být vždy synchronizována a tím pádem i zálohovány naše dokumenty z PC na náš NAS. (Čížek, 2014) (Kubeš a další, 2016) (Tuhý, 2013)

5.2 RAID

Zkratka RAID je složena z anglického označení „Redundant Array of Independent Disks“, tedy v překladu vícenásobné diskové pole nezávislých disků. Tato metoda zajišťuje bezpečné ukládání dat na více pevných disků. Základem je, že se nad disky vytvoří jeden virtuální a tím se snaží zajistit obranu proti selhání HDD, zajišťuje zabezpečení dat proti ztrátě podle typu RAIDu. Tedy data jsou při selhání jednoho či více HDD zachována na dalším HDD. Úroveň zabezpečení dat proti selhání HDD je dána číslem v názvu. Nejčastěji se používají RAID 0, RAID 1, RAID 5 a RAID 6. V případě selhání disku je HDD vyměněno a data jsou na něj z ostatních rekonstruována. (Sůva, 2012) (Tuhý, 2013) (Dočekal, 2009)

5.2.1 RAID 0

Ve skutečnosti se o RAID v pravém slova smyslu nejedná, protože nezaručuje ochranu dat proti ztrátě. Tedy pokud selže HDD, můžeme přijít o všechna data, nebo značné množství dat. Data se na diskové pole ukládají metodou zřetězení, nebo prokládání. Zřetězení ukládá na další HDD, až v případě, že je aktuální zaplněný. Jednoduše řečeno zřetězení spojuje HDD jeden za druhý. Při použití prokládání se rozkládají data na jednotlivé HDD. Výhodou použití metody zřetězení je snadné navyšování kapacity a alespoň nějaká šance na zachování dat. Metoda rozkládání nám zaručuje, rychlejší přistup k datům, ale při selhání přijdeme o všechna data. (Dočekal, 2009) (Tuhý, 2013)

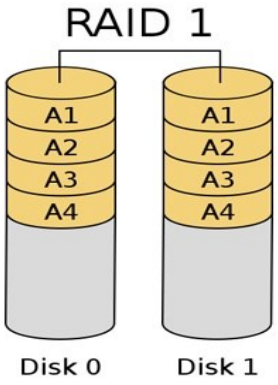


Obrázek 10 − RAID 0

Zdroj: (Binder, 2014)

5.2.2 RAID 1

Je jeden z nejvíce používaných a v principu velice jednoduchý. Zajišťuje ochranu dat proti selhání HDD za pomoci ukládání dat naráz na dva HDD, tedy data se zrcadlí. Při poruše jednoho disku z pole se vymění a data jsou obnovena z druhého. Při výpadku je možno normálně provádět operace nad polem. Nevýhodou je zejména v případě disků s rozdílnou kapacitou, kde je pak kapacita diskového pole rovna disku s menší kapacitou a ostatní je nevyužita. (Dočekal, 2009) (Tuhý, 2013)

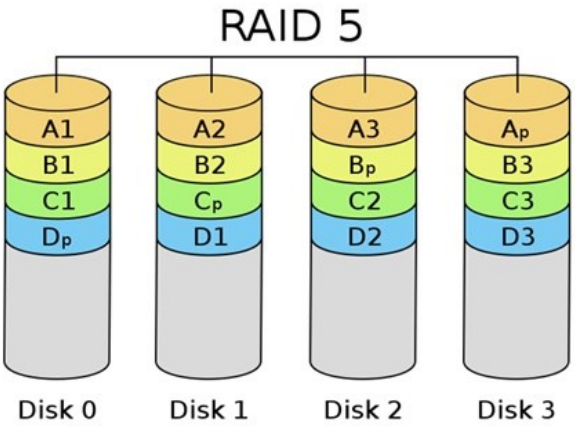


Obrázek 11 − RAID 1

Zdroj: (Binder, 2014)

5.2.3 RAID 5

U tohoto typu je nutno minimálně třech HDD. Vždy na jeden z disků se ukládá parita, která slouží k opravě dat v případě selhání jednoho z HDD a na ostatní disky jsou rozkládána data. V případě selhání jednoho disku jsou data dostupná a po výměně disku jsou na nový disk zrekonstruována data, která obsahoval disk původní, v případě selhání více disků přijdeme o všechna data. Vytváření samo opravných kódu mírně zpomaluje zápis. Kapacita pole je menší jen o velikost jednoho HDD, protože stejná kapacita je využita pro uložení parity. (Dočekal, 2009) (Tuhý, 2013)

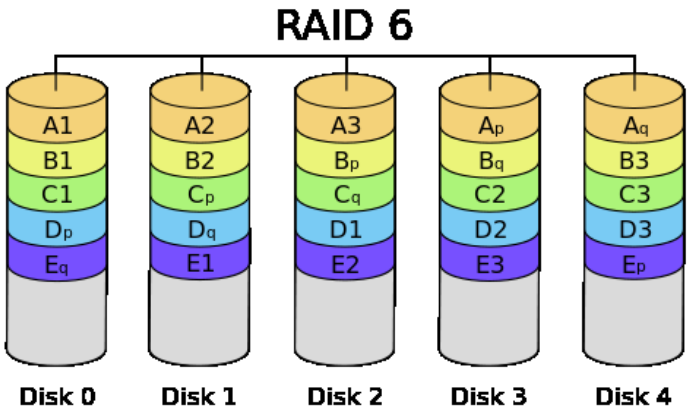


Obrázek 12 − RAID 6

Zdroj: (Binder, 2014)

5.2.4 RAID 6

Funguje podobně jako RAID 5, jen navíc přidává druhý disk s paritou, která je počítána jinak než první. Z toho vyplývá, že potřebuje další HDD navíc, tedy RAID 6 vyžaduje celkem čtyři HDD. Takže je diskové pole spolehlivé i při selhání dvou disků a nedojde tedy ke ztrátě dat. Kapacita pole je nyní snížena o velikost dvou HDD a dochází k dalšímu mírnému zvýšení režie při zápisu z důvodu výpočtu druhé parity. (Dočekal, 2009) (Sůva, 2012)



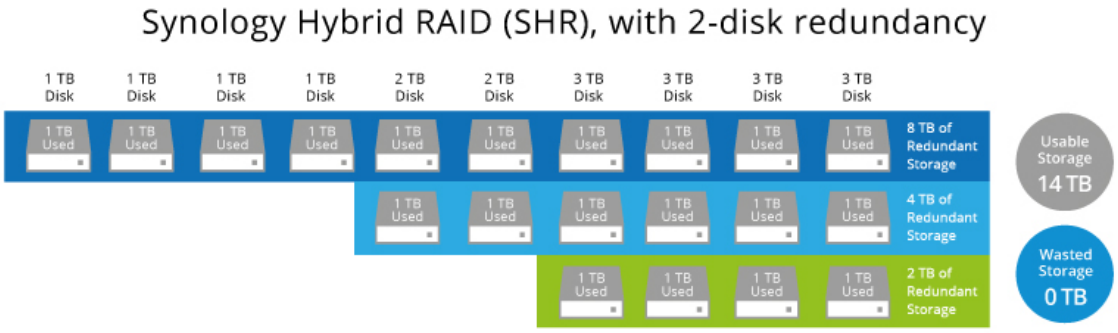
Obrázek 13 − RAID6

Zdroj: (Binder, 2014)

5.2.5 Proprietární řešení výrobců

Někteří výrobci NAS přidává svůj vlastní. Například Synology má svůj vlastní RAID jménem SHR. RAID od Synology má pomoci lidem, kteří se nevyznají v problematice RAIDů zjednodušit konfiguraci a zabezpečit jejich data před ztrátou při neplýtvání kapacitou HDD. SHR má zajistit co nejmenší nevyužitelnou kapacitu i při HDD rozdílných kapacit. SHR proto,

aby toto mohl zajistit dělí HDD na menší stejně velké oddíly, nad kterými se teprve vytváří RAID. Déle pomáhá zjednodušit navyšování kapacity pole i v případě, kdy se nebudou měnit všechny disky, tedy se jeden HDD vymění za větší. (Synology Inc., 2017)



Obrázek 14 – Ukázka SHR

Zdroj: (Synology Inc., 2017)

5.3 Protokoly

NAS pro zpřístupnění možnosti manipulace s daty ostatním zařízením v síti používá celou řadu nejrůznějších protokolů. Mezi ty nejpoužívanější patří CIFS, SMB, NFS, AFP a DLNA. (Ptáčník, 2016)

SMB (*Server Message Block*) byl vyvinut společností IBM pro systém DOS a ranných Windows. Tento protokol byl později upraven společností Microsoft a v dnešní době je používán do dnes jako CIFS (*Common Internet File Service*). Je to jeden z nejpoužívanějších protokolů, protože je používán OS od společnosti Microsoftu, ale ostatní operační systémy ho umějí již také. (Bouška, 2008)

Další možný protokol je NFS (*Network File Service*), který se používá především na systémech Linux a Unix. Následující protokol AFP (*Apple File Protocol*) je používán výhradně na systémech od společnosti Apple, ale nové verze jejich systému MAC OS X si poradí už například i s NFS. Mezi další možnosti patři použití například: FTP, FTPS, Rsync a WebDAV. WebDAV neboli Web-based Distributed Authoring And Versioning pro svoji funkčnost využívá HTTP a má velikou podporu mezi operačními systémy. (Bouška, 2008) (Tuhý, 2013) (ASUSTOR COLLEGE, 2013)

Posledním protokolem, který stojí za zmínku je DLNA, který se od ostatních protokolů liší především svým účelem. Jeho hlavním účelem je streamování multimédií po domácí síti. Je to soubor pravidel a postupů, jak zobrazit data na přijímači, například televizi. Rozšiřuje starší

protokol UpnP. Je to jednotný standard sdružení mnoha předních společností vyvíjejících elektroniku, které se nazývá Digital Living Network Aliance z toho také pochází jeho název. Automaticky navazuje spolupráci všech zařízeních certifikovaných značkou DLNA. (Čížek, 2015)

5.4 Výrobci

Mezi přední výrobce NAS se mohou řadit: Synology, QNAP, Austot, Netgear a Dell. Tyto firmy jsou vhodné jak pro domácí, tak firemní využití. Dalšími výrobci mohou být například Western Digital, D-Linl, Seagate, Zixel a mnohé další společnosti, které prodávají elektroniku či mají závody s výrobou HDD.

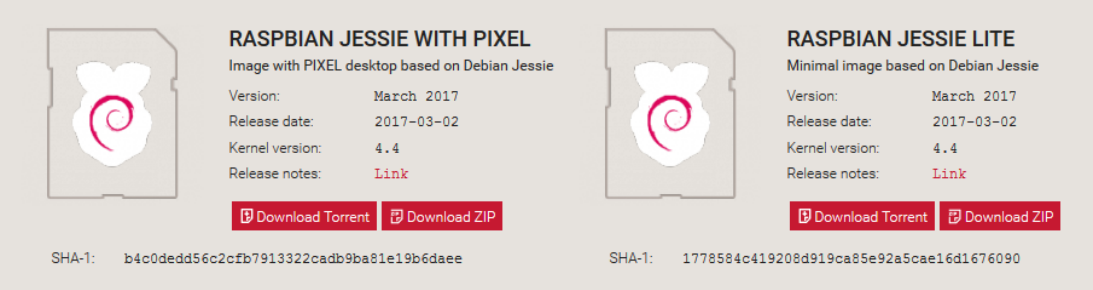
Snad každý výrobce si pro svůj NAS upravuje nějaký systém, většinou založený na Linuxu. Každý má svoje uživatelské prostředí, které je většinou dostupné pomocí webového prohlížeče. QNAP používá systém QTS. Austor zas sází na svůj Austor Data Manager. Synology používají jako OS Disk StationManager, který si i společnost vyvíjí jako ostatní výrobci. Všechny výrobce spojuje totéž, nabízí uživateli komfortní uživatelské prostředí. Například Synology má vlastní typ RAIDu nazývaný SHR, jeho funkcionalita byla popsána u podkapitoly RAID. Většina výrobců umožňuje další funkcionality doinstalovat. Jako jednu z možností doinstalování je cloud, který je pro výrobce dostupný. (Kubeš a další, 2016)

6 PRAKTICKÁ ČÁST

6.1 Příprava Raspberry Pi

Pro lepší práci s Raspberry Pi bude vytvořen image SD karty. Tato karta bude výchozí konfigurací pro všechna zařízení. Pro zařízení bude nahrán OS Raspbian a následně aktualizován. Zařízení bude nastavena IP adresa. Dále bude povolen protokol SSH a nastaven vzdálený přístup pro uživatele. Výsledkem bude image s OS, ke kterému se bude moci přistupovat vzdáleně a nebude třeba klávesnice, myši a monitoru. Postačí jen zapojený internet a napájení.

Nyní bude popsán postup vytvoření prvotního image SD karty. Jako první bude potřeba stáhnout OS Raspbian. Systém Raspbian a další systémy lze stáhnout z oficiálních stránek[6] zařízení. Při stahování OS Raspbian se vybere kompletní verze, aby se instalace na zařízení nemusela doinstalovat ze sítě jako je tomu u verze lite. Výběr verze je ukázán níže na Obrázek 15 − Stažení Raspbian.

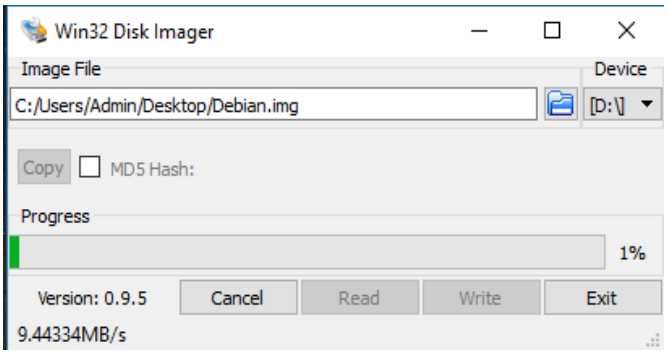


Obrázek 15 − Stažení Raspbian

Zdroj: vlastní

Po stažení je ještě potřeba image s OS pro instalaci na SD kartu rozbalit. Po rozbalení se za pomoci nástroje *Win32 Disk Imager*[7] nahrát na SD kartu. V nástroji stačí vybrat soubor image, vybrat jednotku s vybranou kartou SD a následně se klikne na tlačítko write. Operaci je nakonec ještě nutné potvrdit a po dokončení procesu je SD karta s OS Raspbian připravena.

[6] https://www.raspberrypi.org/downloads/
[7] Win32 Disk Imager je dostupný například z: https://sourceforge.net/projects/win32diskimager/



Obrázek 16 – Ukázka Win32 Disk Imager

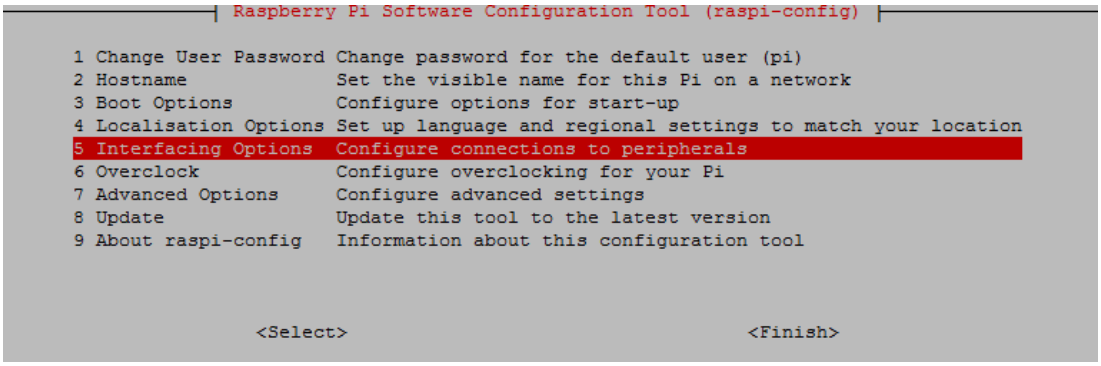
Zdroj: vlastní

Hotová SD karta se zasune do Raspbery Pi. Následně se připojí k zařízení internet, monitor, myš a klávesnice. Po zapnutí systému se spustí terminál a v něm se budou provádět následující úkony. Ze všeho nejdříve se aktualizuje seznam repositářů a následně i veškeré balíčky.

```
$ sudo apt-get update

$ sudo apt-get upgrade
```

Dále se povolí protokol SSH na zařízení za pomoci následujícího příkazu. Příkaz vyvolá nabídku, jak je tomu na Obrázek 17 – Povolení SSH na Raspberry Pi. Nejdříve se vybere volba „5 Interfacing Options“ posléze „P2 SSH“ a poté se potvrdí zapnutí. Na konec se potvrdí zapnutí a nastavení se opustí tlačítkem „<Finish>“.

```
$ sudo raspi-config
```



Obrázek 17 – Povolení SSH na Raspberry Pi

Zdroj: vlastní

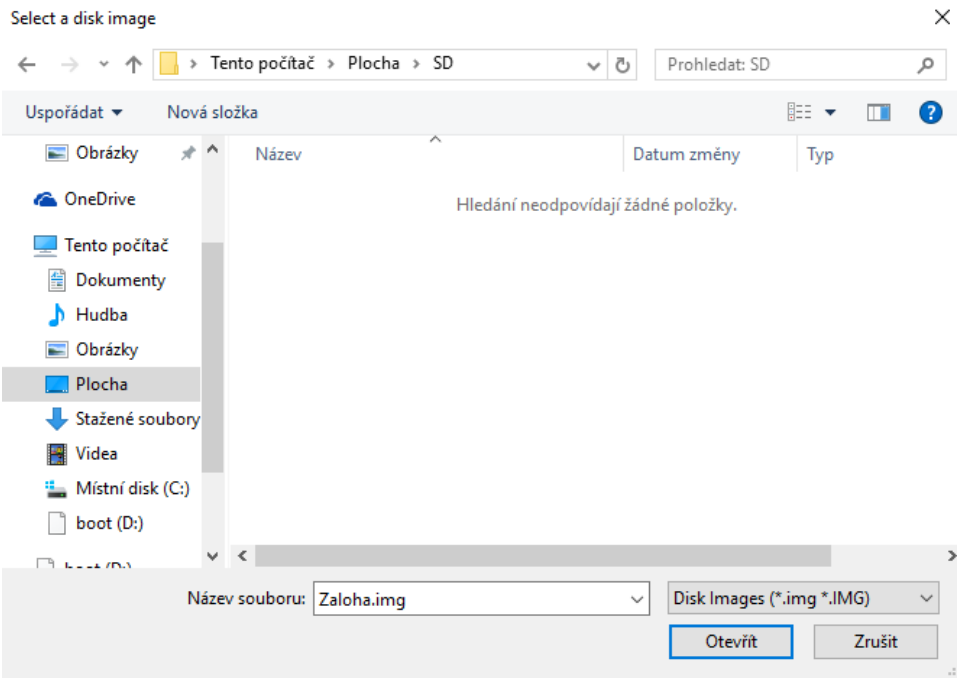
Síť nastavíme v souboru `/etc/dhcpcd.conf`, jak je uvedeno v následujícím příkladu. Nastavení předpokládá, že je takto nastavena síť, kde budou zařízení dále používána. Jako

výchozí konfigurace bude pro zařízení nastaveno: IP adresa 192.168.0.200/24, výchozí brána 192.168.0.215 a DNS 8.8.8.8.

```
$ sudo nano /etc/dhcpcd.conf

static ip_address=192.168.0.200/24

static routers=192.168.0.215

static domain_name_servers=8.8.8.8
```

Po dokončení konfigurace se zařízení může vypnout. Po vypnutí se vytvoří z SD karty image obdobným způsobem, jakým se na SD kartu kopíroval. Stačí vybrat složku a zadat název souboru jako je tomu na: Obrázek 18 − Vytvoření image SD karty. Doporučuji pojmenovat image SD karty s příponou „.img“ pro lepší opětovné použití při nahrávání na SD kartu. Image se doporučuje dát do archivu pro zmenšení jeho velikosti.

```
$ sudo halt
```



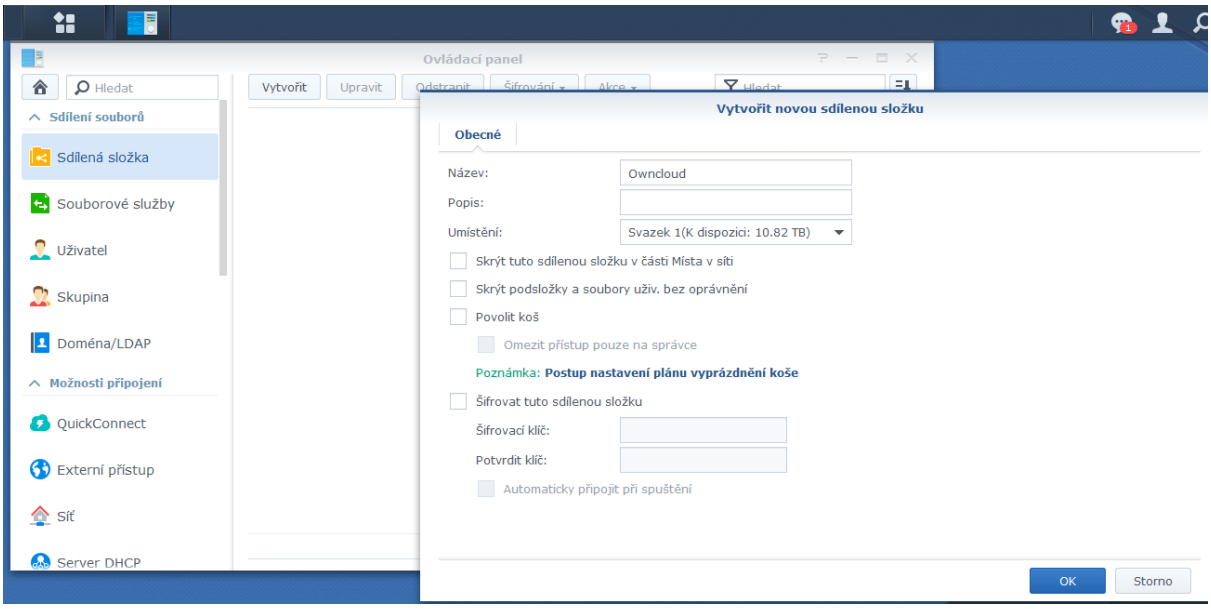
Obrázek 18 − Vytvoření image SD karty

Zdroj: vlastní

6.2 Příprava Synology NAS

Při prvotním zapnutí se musí ze stránek výrobce stáhnout *Synology Assistant*[8]. Při stažení se musí vybrat konkrétní model zařízení NAS. *Synology Assistant* pomůže s vyhledáním zařízení a jeho nastavením. Po vyhledání je na NAS umožněno připojení. Při připojení aplikace otevře webový prohlížeč s interaktivním průvodcem nastavení, kde se vytvoří první uživatel, nainstaluje systém[9] a dále umožní nastavit RAID na SHR. Po dokončení průvodce je možné se na zařízení připojit do jeho uživatelského rozhraní přístupného z webového prohlížeče na IP adrese zařízení.

Po úspěšném přihlášení k NAS se provede nastavení sdílení složky. Tato možnost je odstupná pomocí položky „Sdílená složka“ v „Ovládacích panelech“. Dále vybereme možnost „Vytvořit“ a otevře se okno s nastavením sdílené složky. Ukázka vytvoření je na: Obrázek 19 – Vytvoření sdílené složky. Takto vytvořená složka bude použita v následujících příkladech.



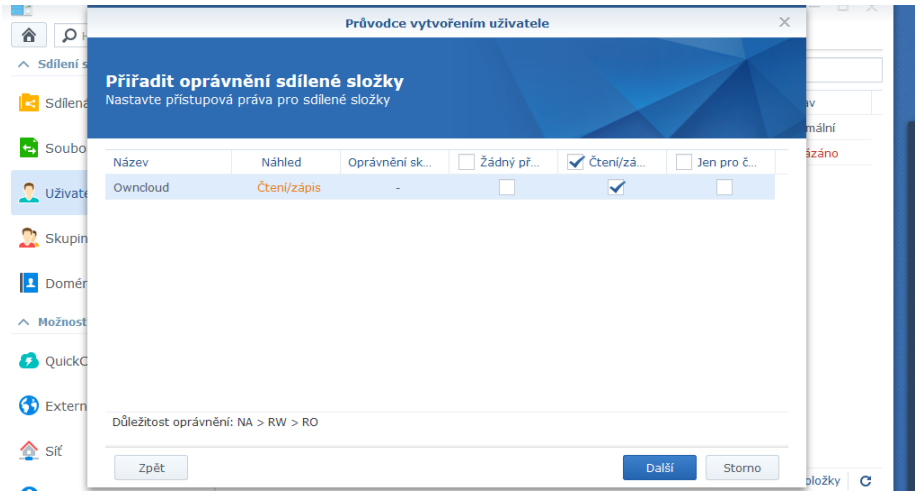
Obrázek 19 – Vytvoření sdílené složky

Zdroj: vlastní

Následně se vytvoří uživatel, který bude mít přiřazenou složku, která byla vytvořena v předešlém odstavci. Tato možnost je dostupná z ovládacích panelů v položce „Uživatel“. Při vytváření nového uživatele se spustí průvodce.

[8] https://www.synology.com/en-us/support/download
[9] Systém DSM se může nainstalovat automaticky, nebo se může manuálně stáhnout ze stránek výrobce a vybrat.

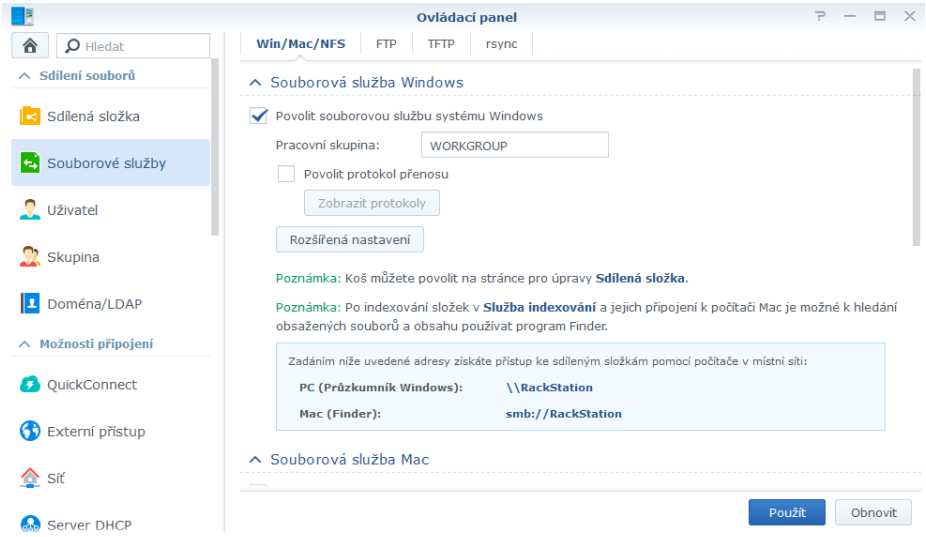
V prvním kroku se bude zadávat jméno uživatele a heslo. Dále bude následovat výběr skupiny, který se nastaví pro běžného uživatele na „users“. Následně se provede nastavení oprávnění k předem vytvořené složce jako je tomu na Obrázek 20 − Vytvoření uživatele v NAS. Další kroky průvodce se mohou přeskočit, protože se v nich nebude nic nastavovat. Mezi další kroky patři: nastavení kvót, oprávnění k aplikacím a rychlostní limity přenosu dat. Nakonec průvodce pro kontrolu zopakuje nastavení a potvrzením se nový uživatel vytvoří.



Obrázek 20 − Vytvoření uživatele v NAS

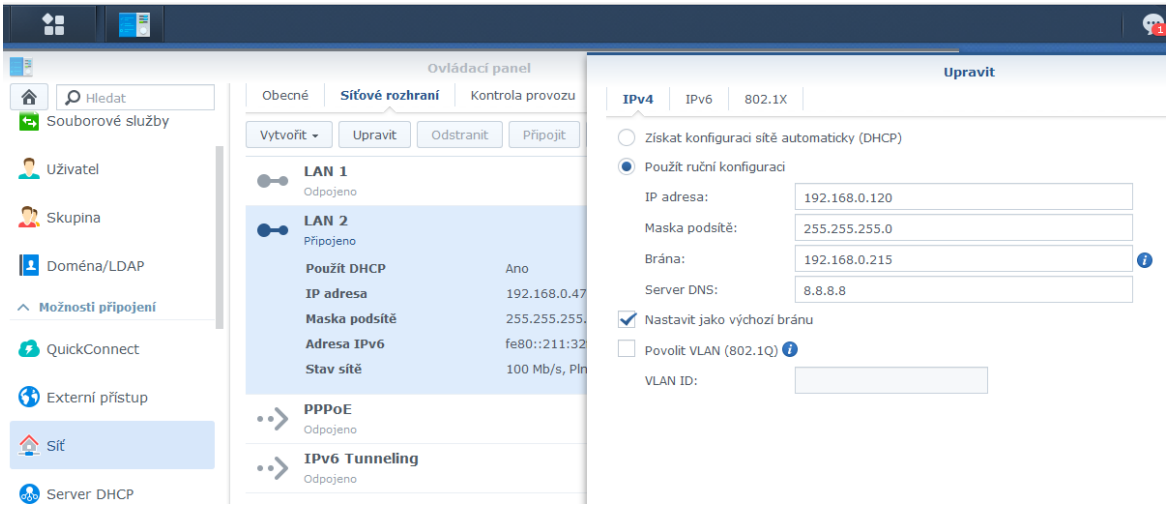
Zdroj: vlastní

V ovládacích panelech se překontroluje zapnutí souborové služby a její nastavení. Je důležité, aby byla služba zapnutá a nastavena na pracovní skupinu „WORKGROUP“. Nastavení je ukázáno v následujícím obrázku.



Zdroj: vlastní

Nakonec se pro NAS nastaví síť. V ovládacích panelech se vybere možnost „Síť“. Záložka „Obecné“ umožní změnit název, kterým se zařízení NAS bude prezentovat v síti. Změna IP adresy zařízení se provede v záložce „Síťové rozhraní“, kde se vybere rozhraní a následně mu nastaví IP adresy. Adresace je následující: IP adresa 192.168.0.120/24, výchozí brána 192.168.0.215 a DNS server 8.8.8.8. kompletní nastavení je zobrazeno na Obrázek 21 – Nastavení IP adresy NAS.



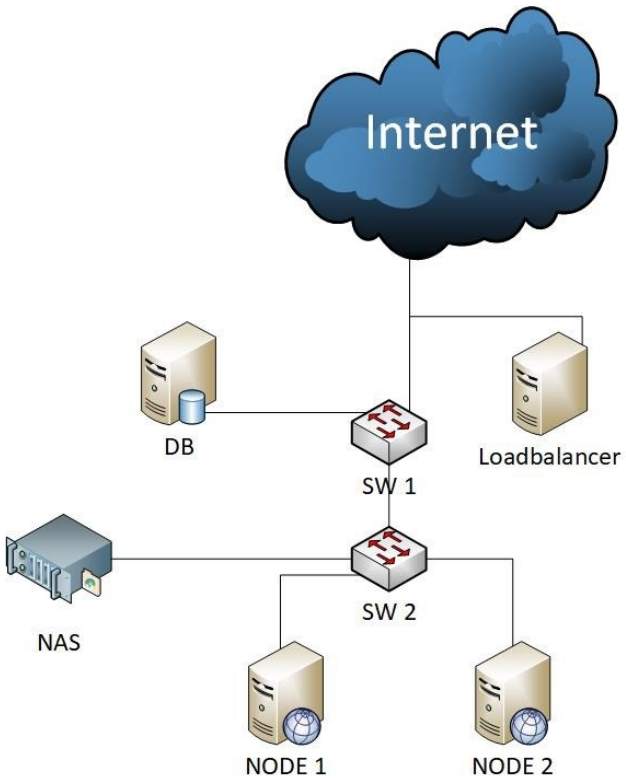
Obrázek 21 – Nastavení IP adresy NAS

Zdroj: vlastní

6.3 Cloudové uložiště

Ukázka bude demonstrovat cloud pro ukládání souborů, nazývané jako cloudové uložiště. Cloudové datové centrum bude obsahovat databázi MySQL, NAS od firmy Synology, loadbalancer a dva nody s ownCloudem. Vše s výjimkou NAS bude realizováno na zařízení Raspberry Pi.

Zařízení v datovém centru bude propojeno za pomoci switchů, které budou řešit ochranu datového centra. Zabezpečení se bude řešit za pomoci softwarově definovaných sítí, které budou řídit provoz na switchích SW 1 a SW 2. Problematika sítí SDN je řešena za pomoci jiné diplomové práce s názvem: „Využití Raspberry Pi jako OpenFlow switch“. Přístup z internetu bude povolen jen na loadbalancer a oba nody. Kompletní uspořádání sítě je znázorněno na následujícím obrázku.



Obrázek 22 − Topologie cloudového uložiště

Zdroj: vlastní

Při přístupu z internetu do cloudového uložiště bude požadavek odbaven loadbalancerem a přesměrován na jeden z nodů. Nody obsahují webovou aplikaci připojenou k zařízení DB. Data uživatelů v ownCloudu se budou ukládat do zařízení NAS. Z důvodu bezpečnosti může zařízení NAS a DB komunikovat jen s nody.

Po vysvětlení ukázkového přikladu se může přistoupit ke konfiguraci. Veškerá následující konfigurace bude vycházet ze základní přípravy zařízení Raspberry Pi a NAS provedené v předešlé podkapitole Příprava Raspberry Pi. Především u zařízení Raspberry Pi je to: konfigurace výchozí adresace a povolení přístupu za pomoci protokolu SSH.

Jako první se bude konfigurovat zařízení s databází. Po zapojení prvního zařízení se z počítače na zařízení přistoupí za pomoci protokolu SSH[10]. Následně se změní výchozí IP adresa zařízení v souboru `/etc/dhcpcd.conf` na 192.168.0.110.

```
$ sudo nano /etc/dhcpcd.conf
```

[10] Pro přístup ke vzdálenému terminálu se může volit například aplikace PuTTY, která se dá stáhnout z: http://www.putty.org/.

```
static ip_address=192.168.0.110/24
```

Dále změníme jméno zařízení na DB.

```
$ sudo nano /etc/hostname

db
```

Následně se zařízení restartuje. Po zapnutí se na zařízení opět připojí za pomoci SSH. Pro připojení se použije jeho nová IP Adresa 192.168.0.110.

```
$ sudo reboot
```

Po znovu připojení na zařízení se nainstaluje databáze MySQL. Instalace se musí potvrdit a po stažení a instalaci bude spuštěn průvodce prvotním nastavením databáze. V průvodci se zvolí heslo pro uživatele root. Heslo bylo pro demonstrační účely zvoleno: 123456.

```
$ sudo apt-get install mysql-server
```

Pro přístup z nodů do databáze se vytvoří uživatel. Pro zadání potřebných příkazů je potřeba se do databáze nedříve z terminálu přihlásit. Jméno uživatele se nastaví na cloud a heslo na 123456. Následně se uživateli ještě přiřadí oprávnění. Po vykonání příkazů v databázi se z databáze provede odhlášení zpět do terminálu.

```
$ mysql -u root -p

mysql> CREATE USER 'cloud'@'%' IDENTIFIED BY '123456';

mysql> GRANT ALL PRIVILEGES ON *.* TO 'cloud'@'%' IDENTIFIED
BY '123456' WITH GRANT OPTION;

mysql> exit
```

Jako poslední je potřeba povolit ke službě vzdálený přístup. Toto nastavení se provede v souboru `/etc/mysql/my.cnf`. V souboru se zakomentuje řádek `bind-address = 127.0.0.1`.

```
$ sudo nano /etc/mysql/my.cnf

# bind-address      = 127.0.0.1
```

Následně se už jen služba MySQL restartuje, aby se aplikovalo nové nastavení. Poté je zařízení připraveno k použití a může se přejít ke konfigurací dalších zařízení.

```
$ sudo service mysql stop

$ sudo service mysql start
```

Jako další se nakonfiguruje první node. Zase se připojí za pomoci SSH na výchozí IP adresu 192.168.0.200. Následně se nakonfiguruje nová IP adresa zařízení. Nová IP adresa bude 192.168.0.101.

```
$ sudo nano /etc/dhcpcd.conf

static ip_address=192.168.0.101/24
```

Dále se změní jméno zařízení na node1.

```
$ sudo nano /etc/hostname

node1
```

Posléze se zařízení restartuje a znovu se připojí do vzdáleného terminálu. Je potřeba se již propojit na novou IP adresu zařízení.

```
$ sudo reboot
```

Po opětovném zapnutí a přihlášení se nainstalují potřebné balíčky pro ownCloud. Mezi potřebné balíčky je zahrnut především webserver a php. Déle se přidávají balíčky pro napojení do DB a vytvoření certifikátů pro potřeby webserveru s https. Poslední kategorii jsou pomocné balíčky pro ownCloud. Jako webserver byl vybrán NGINX.

```
$ sudo apt-get install nginx openssl ssl-cert php5-cli php5-
json php5-mysql php5-common php5-cgi php-pear php-apc curl
libapr1 libtool libcurl4-openssl-dev php-xml-parser php5 php5-
dev php5-curl php5-gd php5-fpm memcached php5-memcache varnish
```

Nejdříve vytvoříme dva certifikáty pro webserver. Pro certifikáty se vytvoří speciální složka, do které se posléze příkazem vygenerují certifikáty. Při generování se spustí průvodce, který je znázorněn na Obrázek 23 – Generování certifikátů. Po vygenerování je jim potřeba ještě přiřadit patřičná oprávnění z důvodu zabezpečení.

```
$ sudo mkdir /etc/ssl/nginx

$ sudo openssl req $@ -new -x509 -days 730 -nodes -out
/etc/ssl/nginx/cert.pem -keyout /etc/ssl/nginx/cert.key
```

```
$ sudo chmod 600 /etc/ssl/nginx/cert.pem

$ sudo chmod 600 /etc/ssl/nginx/cert.key
```

```
Generating a 2048 bit RSA private key
.......................+++
.....................................................................+++
writing new private key to '/etc/ssl/nginx/cert.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:
Email Address []:
pi@raspberrypi:/ $
```

Obrázek 23 – Generování certifikátů

Zdroj: vlastní

Dalším bodem je nastavení webového serveru NGINX. Nastavení je prováděno v souboru `/etc/nginx/sites-available/default`. V souboru je nastaveno především na jakých portech a IP adrese bude naslouchat. Posléze jsou zde použity vygenerované certifikáty pro https. Dále je specifikována maximální velikost souborů, php a další nastavení. Soubor se může kompletně nahradit Příloha A – *Nastavení NGINX.*

```
$ sudo rm /etc/nginx/sites-available/default

$ sudo nano /etc/nginx/sites-available/default
```

Nastavení se dá provádět za pomoci dokumentace dostupné na stránkách ownCloud[11]. Vytvořená konfigurace se dá kontrolovat za pomoci následujícího příkazu, který zkontroluje syntaxi konfiguračního souboru.

```
$ sudo nginx -t -c /etc/nginx/nginx.conf
```

Následně se velikost souboru musí ještě nastavit i u php. Změna se provede v souboru `/etc/php5/fpm/php.ini`. V souboru se pozmění `upload_max_filesize`

[11] https://doc.owncloud.org/server/9.0/admin_manual/installation/nginx_examples.html

a `post_max_size` na 4000M. Pro snadné orientování v souboru lze využít „ctrl+w“ v editoru `nano`, které umožňuje vyhledání určité fráze v textu.

```
$ sudo nano /etc/php5/fpm/php.ini

upload_max_filesize = 4000M

post_max_size = 4000M
```

Pro funkčnost je nutné nastavit, kde bude naslouchat php. Nastavení se provádí v souboru `/etc/php5/fpm/pool.d/www.conf`. V souboru se vyhledá položka „listen“ a její hodnota se změní na 127.0.0.1:9000.

```
$ sudo nano /etc/php5/fpm/pool.d/www.conf

listen = 127.0.0.1:9000
```

Dále se na Raspberry Pi nastaví vetší velikost jednotky swap. Konfigurační soubor se nachází v `/etc/dphys-swapfile`. V souboru se změní parametr `CONF_SWAPSIZE` na 1024.

```
$ sudo nano /etc/dphys-swapfile

CONF_SWAPSIZE=1024
```

Pro dokončení všech změn je nutné zařízení restartovat. Zařízení díky tomu restartuje veškeré služby a změní velikost swap.

```
$ sudo reboot
```

Po restartu zařízení se přesune do složky, kde se nachází webový adresář. Následně se z oficiálních stránek[12] stáhne komprimovaná balíček ownCloudu. Balíček se po stažení rozbalí a změní se vlastník rozbalené složky owncloud na www-data. Stažený balíček se na konec může smazat.

```
$ cd /var/www/

$ sudo wget https://download.owncloud.org/community/owncloud-
9.1.4.tar.bz2

$ sudo tar xvf owncloud-9.1.4.tar.bz2
```

[12] https://owncloud.org/install/

```
$ sudo chown -R www-data:www-data /var/www

$ sudo rm -rf owncloud-9.1.4.tar.bz2
```

Opět je potřeba nastavit limit maximální velikosti souboru. Nastavení se provede ve složce owncloud. Kde se upraví soubor `.htaccess`, kde se změní php_value_upload_max_filesize, php_value_post_max_size a php_value_memory_limit na 4000M

```
$ cd owncloud/

$ sudo nano .htaccess

php_value_upload_max_filesize 4000M

php_value_post_max_size 4000M

php_value_memory_limit 4000M
```

Následně se ve stejné složce upraví druhý soubor `.user.ini`. Tento soubor upravuje limit velikosti souboru pro uživatele. Upraví se parametry: upload_max_filesize, post_max_size a memory_limit na 4000M.

```
$ sudo nano .user.ini

upload_max_filesize=4000M

post_max_size=4000M

memory_limit=4000M
```

Pro cloud je nutné ještě vytvořit propojení s uložištěm dat v podobě NAS. Nejdříve nainstalujeme balíček `cifs-until`, který umožní připojit síťové uložiště. Dále se v kořenovém adresáři vytvoří složka cloud. Následně se složce přiřadí vlastník a její oprávnění. Vlastníkem složky bude uživatel *www-data*. Kompletní oprávnění bude mít pro složku vlastník a skupina, ostatní nebudou mít oprávnění žádná.

```
$ sudo apt-get install cifs-utils

$ sudo mkdir cloud

$ sudo chmod -R 770 cloud/

$ sudo chown www-data cloud/
```

V adresáři uživatele se vytvoří soubor `.smbcredentials` s přístupovými údaji do NAS. Tento soubor bude sloužit pro automatické zadání přihlašovacích údajů. Přihlašovací údaje pro NAS byly nakonfigurovány v předešlé kapitole, kde se NAS připravoval.

```
$ nano ~/.smbcredentials

username=Cloud

password=123456

domain=WORKGROUP
```

Záznam o disku se připojí do konfiguračního souboru `fstab`. Zde se uvádí adresa NAS, přípojný adresář, soubor s přihlašovacím údajem a práva. Adrese je: //192.168.0.120/Owncloud a přípojný bod: /cloud.

```
$ sudo nano /etc/fstab

//192.168.0.120/Owncloud /cloud cifs
credentials=/home/pi/.smbcredentials,iocharset=utf8,gid=www-
data,uid=www-data,file_mode=0770,dir_mode=0770 0 0
```

Připojení jednotky se provede následujícím příkazem.

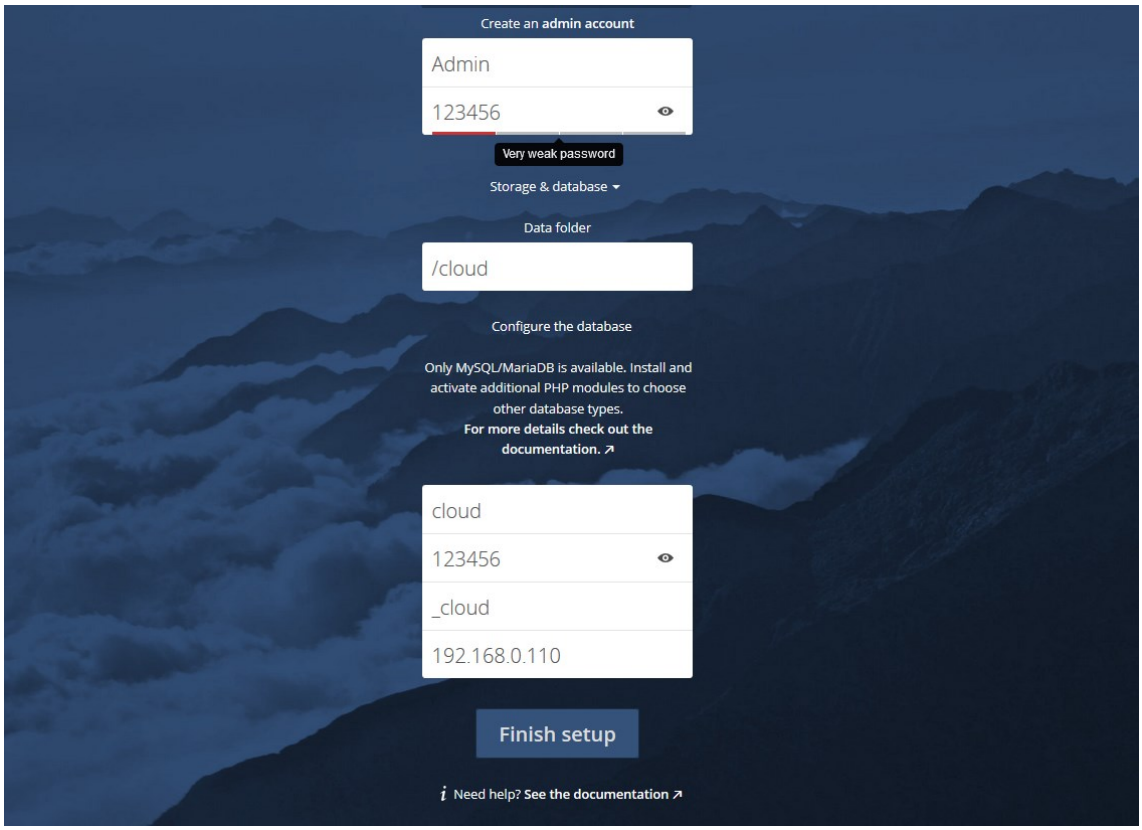
```
$ sudo mount -a
```

Následně je potřeba upravit podmínku pro kontrolu oprávnění složky s ownCloudem. Jinak by ownCloud zobrazoval chybu, že není nastaveno oprávnění složky na 0770. Soubor, který se bude upravovat, se nachází v adresáři `/var/www/owncloud/lib/private/legacy` a souboru `util.php`. V souboru se na řádku 925 upraví podmínka tak aby byla vždy „fasle“. Obsah souboru se změnou je znázorněn v následujícím kódu. Stará část kódu je zakomentována a hned pod ni je řádek který jej nahrazuje.

```
$ sudo nano /var/www/owncloud/lib/private/legacy/util.php

$perms = substr(decoct(@fileperms($dataDirectory)), -3);

     //if (substr($perms, 2, 1) != '0') {

     if (false) {

          $errors[] = array(
```

Nyní se může přistoupit k závěrečné konfiguraci ownCloudu. Zde se bude nastavovat kam má ownCloud ukládat data uživatelů, nastaví se připojení k DB a vytvoří účet správce. Pro možnost konfigurace se zadá do prohlížeče IP adresa node1, tedy 192.168.0.101. Následně se již otevře na obrazovce webová stránka, která provede prvotní konfiguraci. Webová stránka s konfigurací je zobrazena na Obrázek 24 − Prvotní nastavení ownCloud.

Jako účet správce bude zvoleno jméno Admin a heslo 12346. Data se budou ukládat do `/cloud`, který byl nastaven jako propojení s NAS v předešlém kroku. Dále se nastaví přihlášení k DB následovně: uživatel cloud, heslo 123456, tabulka _cloud a server 192.168.0.110. Posléze stačí již dokončit a bude vytvořen první účet, ze kterého se může provádět správa cloudu.



Obrázek 24 − Prvotní nastavení ownCloud

Zdroj: vlastní

Po prvotním nakonfigurování je node1 připraven k použití. Nyní se node1 může vypnout a udělá se z něj image SD karty obdobným způsobem, jako tomu bylo v kapitole: Příprava Raspberry Pi. Vytvořený image se využije pro node2. Image z node1 se tedy nahraje na SD kartu určenou pro node2. Připravený node2 se již může zapnout a začít konfigurovat.

```
$ sudo halt
```

Zapnutý node2 se musí patřičně upravit, protože teď je identickou kopií node1. Zatím se za pomoci SSH přihlásíme na IP adresu 192.168.0.101, která patří node1. Nejdříve se v souboru `/etc/dhcpcd.conf` změní IP adresa z 192.168.0.101 na 192.168.0.102.

```
$ sudo nano /etc/dhcpcd.conf

static ip_address=192.168.0.110/24
```

Dále se změní název zařízení. Změna se provede v souboru `/etc/hostname`, kde se provede změna z node1 na node2.

```
$ sudo nano /etc/hostname

node2
```

Následně se musí překonfigurovat webový server. V souboru `/etc/nginx/sites-available/`default se změní IP adresy 192.168.0.101 na 192.168.0.102. Tedy se změní parametr `server_name` pro http a https.

```
$ sudo nano /etc/nginx/sites-available/default

server_name 192.168.0.101;
```

Jako poslední je potřeba ještě změnit konfigurační soubor ownCloudu. Soubor se nachází v `/var/www/owncloud/config/config.php`. V soboru se změní IP adresa na 192.168.0.102 a ještě se přidá stejná IP adresa do pole důvěryhodných adres. Změněna je znázorněna v Obrázek 25 − Změna v konfiguraci ownCloud.

```
$ sudo nano /var/www/owncloud/config/config.php
```

```
<?php
$CONFIG = array (
  'instanceid' => 'oc01dfpryf92',
  'passwordsalt' => 'ky0IwQRvNcGV9zVMsjfp+SMPnzt9jx',
  'secret' => 'pvHkgQaFjEfci51L4ynLlctjS/TKtRcLQayGbNKBe4BvQUia',
  'trusted_domains' =>
  array (
    0 => '192.168.0.101',
    1 => '192.168.0.102',
  ),
  'datadirectory' => '/cloud',
  'overwrite.cli.url' => 'https://192.168.0.102',
  'dbtype' => 'mysql',
```

Obrázek 25 – Změna v konfiguraci ownCloud

Zdroj: vlastní

Na konec se zařízení restartuje a tím se dokončí jeho konfigurace. Jeho následná IP adresa bude 192.168.0.102. Po restartu zařízení se může opětovně zapnout node1. V některých případech se po opětovném zapnutí zařazení nemusí připojit NAS, proto je potřeba zopakovat příkaz pro jeho připojení.

```
$ sudo reboot

$ sudo mount -a
```

Jako poslední se bude konfigurovat loadbalancer. Po jeho zapojení do sítě se za pomoci ssh k němu přihlásí. Loadbalancer bude kontrolovat stav nodů a jejich vytížení. Podle těchto statistik bude směrovat na jednotlivé node.

Nejdříve se na zařízení změní výchozí IP adresa na 192.168.0.100. Konfigurace se provede stejně jako u předešlých zařízení v souboru `/etc/dhcpcd.conf`. Tato adrese bude následně sloužit jako vstupní z internetu do cloudu.

```
$ sudo nano /etc/dhcpcd.conf

static ip_address=192.168.0.100/24
```

Následně se v souboru `/etc/hostname` upraví jméno zařízení na loadbalancer.

```
$ sudo nano /etc/hostname

loadbalancer
```

Dále se nainstaluje z repositáře balíček `haproxy`.

```
$ sudo apt-get install haproxy
```

Překopíruje se výchozí soubor konfigurace haproxy `haproxy.cfg.original` do nového souboru `haproxy.cfg`. Oba konfigurační soubory se nachází v adresáři `/etc/haproxy/`. Nový konfigurační soubor se následně otevře pro následné úpravy.

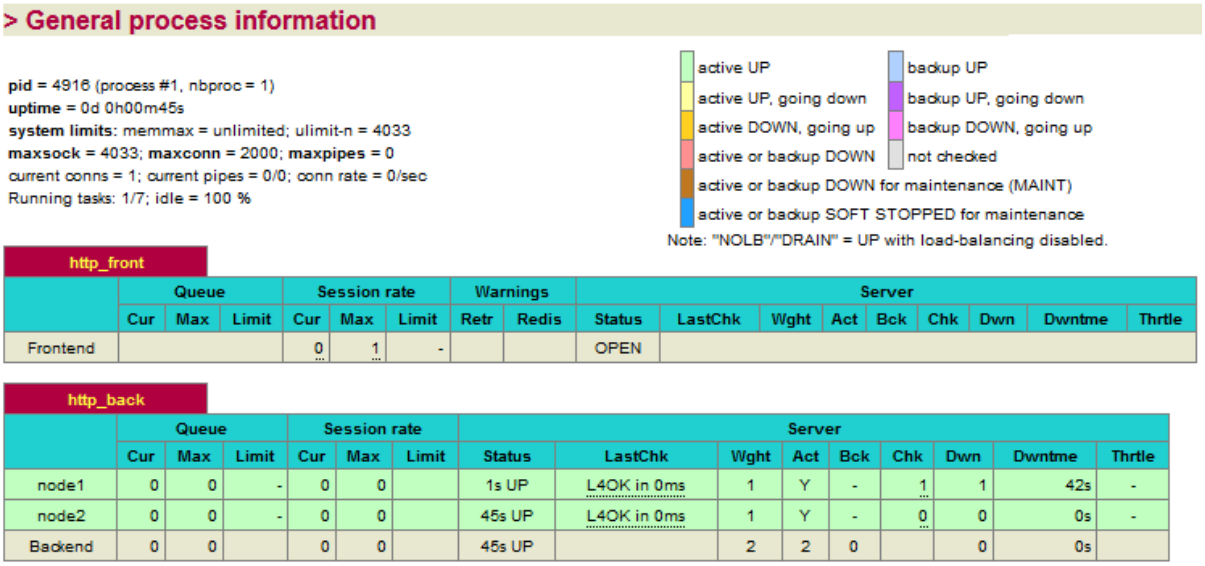
```
$ sudo cp /etc/haproxy/haproxy.cfg
/etc/haproxy/haproxy.cfg.original

sudo nano /etc/haproxy/haproxy.cfg
```

Do konfiguračního souboru se přidá následující konfigurace. Poté se soubor uzavře a uloží.

```
frontend http_front

   bind *:80

   stats uri /haproxy?stats

   default_backend http_back

backend http_back

   balance roundrobin

   server node1 192.168.0.101:80 check

   server node2 192.168.0.101:80 check
```

Následně po dokončení konfigurace stačí již službu haproxy restartovat. Posléze se na adrese http://192.168.0.100/haproxy?stats nachází statistika a stav jednotlivých nodů. Tato funkcionalita je znázorněna v následujícím obrázku.

```
$ sudo service haproxy restart
```



> General process information

pid = 4916 (process #1, nbproc = 1)
uptime = 0d 0h00m45s
system limits: memmax = unlimited; ulimit-n = 4033
maxsock = 4033; maxconn = 2000; maxpipes = 0
current conns = 1; current pipes = 0/0; conn rate = 0/sec
Running tasks: 1/7; idle = 100 %

active UP
active UP, going down
active DOWN, going up
active or backup DOWN
backup UP
backup UP, going down
backup DOWN, going up
not checked
active or backup DOWN for maintenance (MAINT)
active or backup SOFT STOPPED for maintenance
Note: "NOLB"/"DRAIN" = UP with load-balancing disabled.

http_front

	Queue			Session rate			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
Frontend				0	1	-			OPEN								

http_back

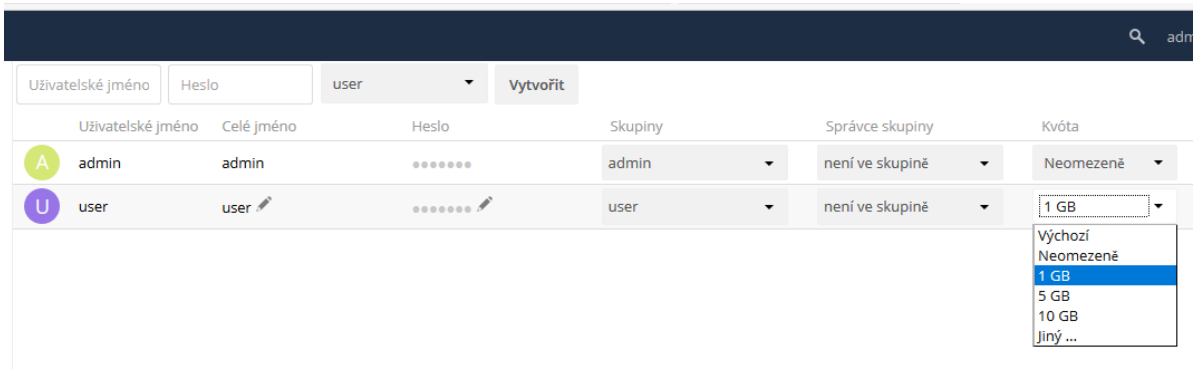
	Queue			Session rate			Server								
	Cur	Max	Limit	Cur	Max	Limit	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
node1	0	0	-	0	0		1s UP	L4OK in 0ms	1	Y	-	1	1	42s	-
node2	0	0	-	0	0		45s UP	L4OK in 0ms	1	Y	-	0	0	0s	-
Backend	0	0		0	0		45s UP		2	2	0		0	0s	

Obrázek 26 – Stav haproxy

Zdroj: vlastní

Tímto je konfigurace hotová a může se přistoupit k ukázce výsledku. Takže se přistoupí k webové stránce http://192.168.0.100 a loadbalancer přesměruje požadavek na web jednoho z nodů. Na úvodní obrazovce se přihlásí účtem admin. Za něj se dají již nahrávat soubory do cloudu, buď za pomoci této webové stránky nebo desktopového či mobilního klienta.

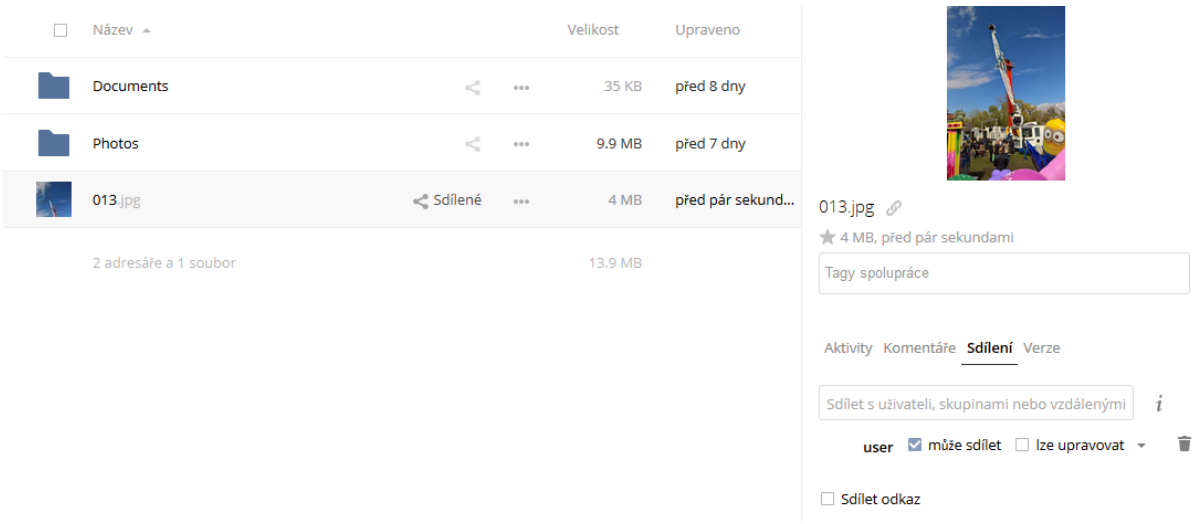
Dále se za pomoci administrátorského účtu dá nastavovat ownCloud a vytvářet další uživatelé a jejich oprávnění. Pro nového uživatele se dá nastavit členství ve skupině a stanovit jaké množství dat smí do cloudu nahrát. Vytvoření uživatele a jeho následné nastavení je zobrazeno v následujícím obrázku.



Obrázek 27 – ownCloud vytvoření uživatele

Zdroj: vlastní

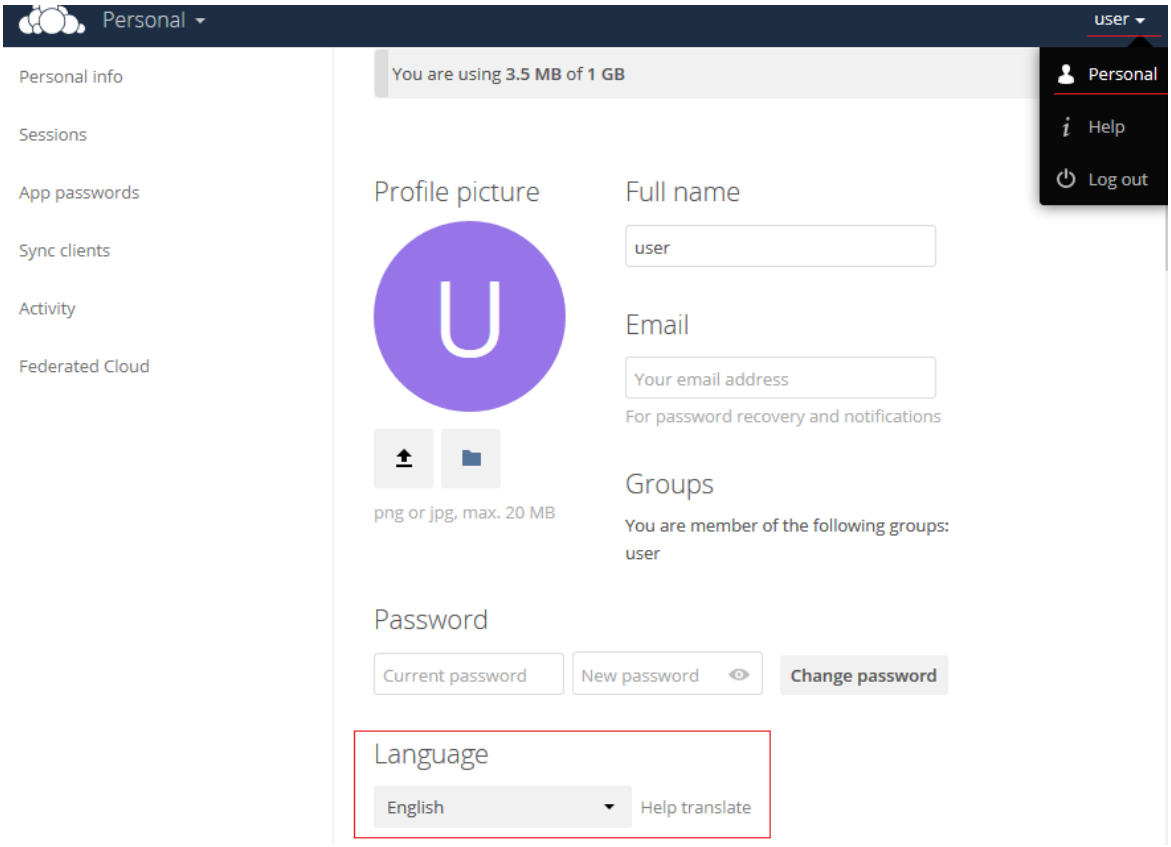
Nahrané soubory se dají jednoduše sdílet ostatním uživatelům, nebo i veřejně za pomoci odkazu na soubor. Sdíleným souborům se dají nastavovat i různá oprávnění pro uživatele, kterým je sdíleno. V následujícím obrázku je zobrazeno jedno z možných nastavení sdílení.



Obrázek 28 − Sdílení souboru

Zdroj: vlastní

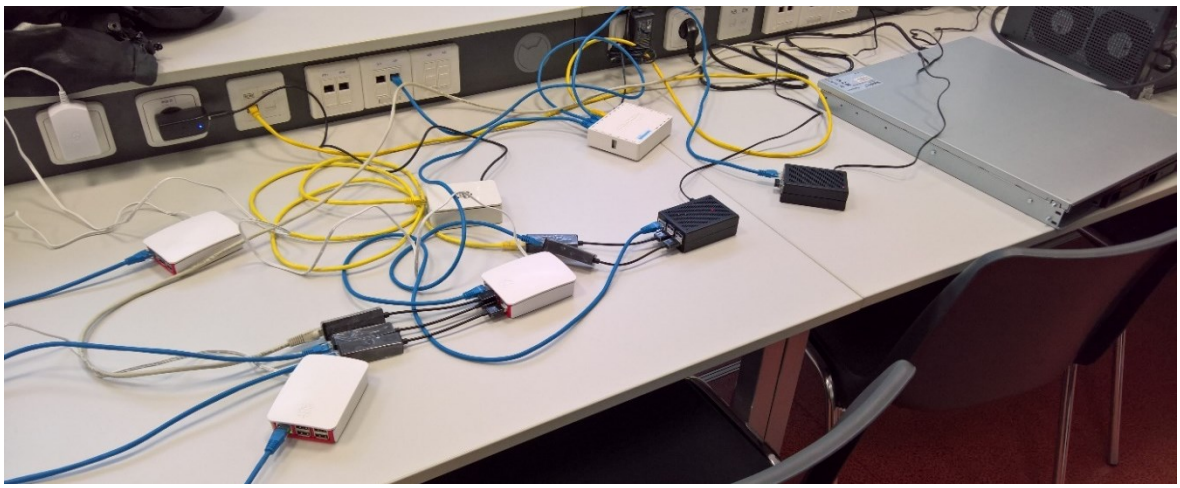
Při přístupu je možné si nastavit preferovaný jazyk uživatelského rozhraní. Nastavení se provádí v menu pod položkou „Personal“, kde se dají nastavovat všechny věci týkající se přihlášeného uživatele. Dá se například zde změnit heslo a nastavit jazyk uživatelského rozhraní.



Obrázek 29 − Nastavení jazyku ownCloudu

Zdroj: vlastní

Celé cloudové datové centrum se všemi zařízeními je zachyceno v následujícím obrázku. V pravém horním rohu obrázku je možné vidět NAS. Ostatní vyobrazené zařízení jsou Raspberry Pi s výjimkou routeru, který se přibližně nachází nahoře uprostřed obrázku. Router sloužil jako simulace internetu.



Obrázek 30 − Foto cloudového datového centra

Zdroj: vlastní

6.4 Cloudové datové centrum s kontejnery

Jako další bude představeno cloudové datové centrum, které bude využívat kontejnery pro spuštění jednotlivých služeb v cloudu. Veškeré kontejnery poběží na zařízení Raspberry Pi. Jako správce kontejnerů bude nainstalován Docker.

Nejdříve budou představeny kontejnery na jednom ze zařízení, kde se připraví pro jednoduché ovládání Dockeru webové uživatelské prostředí. Posléze se vytvoří cluster a připojí do něj dalších pět Paspberry Pi. Cluster pro Docker byl vybrán Docker Swarm. Pro Swarm se vytvoří webové zobrazení stavu jednotlivých zařízení v cluster. Dále bude představeno vytvoření ukázkových služeb v clusteru.

Postup je následující − nejdříve se připraví základní image systému, který poběží v základu na všech zařízeních. Při vytváření image s bude tentokrát vycházet z distribuce Raspbian bez GUI (Raspbian Jessie Lite). Jinak se bude postupovat obdobně, jako tomu bylo v kapitole Příprava Raspberry Pi. Provede se aktualizace, povolí se protokol SSH a nastaví se výchozí IP adresace. Jako tomu bylo při předešlém vytváření image.

```
$ sudo apt-get update

$ sudo apt-get upgrade

$ sudo raspi-config

$ sudo nano /etc/dhcpcd.conf

static ip_address=192.168.0.200/24

static routers=192.168.0.215

static domain_name_servers=8.8.8.8
```

Protože se zvolila distribuce bez GUI, může být snížená vyhrazená velikost paměti pro video. Tím, že se nastaví paměť pro video na 16 MB, se zajistí více paměti pro sytém a kontejnery.

```
$ sudo nano /boot/config.txt

gpu_mem=16
```

Následně se provede restart zařízení, aby se načetla nová konfigurace.

```
$ sudo reboot
```

Po restartování se nainstaluje Docker. Instalace se provádí za pomoci oficiálního shell skriptu dostupného z webu Dockeru. Script se spustí za pomoci příkazu `curl`[13].

```
$ curl -sSL https://get.docker.com | sh
```

Následně se po úspěšné instalaci Dockeru provede jeho povolení a nastaví se mu automatické zapnutí. Dalším příkazem se uživateli „pi“[14] povolí správa kontejnerů, toto je zajištěno přidáním uživatele do skupiny „docker“.

```
$ sudo systemctl enable docker

$ sudo systemctl start docker

$ sudo usermod -aG docker pi
```

Nyní se funkčnost Dockeru otestuje za pomoci aplikace hello-world.

```
$ sudo docker run armhf/hello-world
```

Po nainstalování se zobrazí zpráva, která signalizuje úspěch předchozích operací. Ve zprávě se v prvním řádku vypíše „Hello from Docker on armhf!“ a pokračuje úvodními informacemi k Dockeru. Zařízení se poté vypne a z jeho SD karty se vytvoří image. Tento image se nahraje na všechna ostatní zařízení. Díky tomu je na ostatních ušetřena většina prvotní konfigurace a instalace Dockeru.

```
$ sudo halt
```

Nyní se vezme jedno z připravených zařízení a zapne se. Na zařízení se změní IP adresa z výchozí adresy na 192.168.0.190. Toto zařízení bude sloužit jako první node a budou na něm předvedeny základní operace s kontejnery.

```
$ sudo nano /etc/dhcpcd.conf

static ip_address=192.168.0.190/24
```

[13] U distribuce Raspbian je balíček curl nainstalován v základu, pokud by jinde nebyl, nainstaluje se za pomoci příkazu: `sudo apt-get install curl`.
[14] Může být vybrán libovolný uživatel, u kterého se bude požadovat správa kontejnerů.

Dále se změní jméno zařízení. Změna se provádí ve dvou souborech: `/etc/hosts` a `/etc/hostname`. V obou souborech se změní „raspberrypi“ na „node1“ a soubory se uloží. Poté se zařízení restartuje.

```
$ sudo nano /etc/hosts

$ sudo nano /etc/hostname

$ sudo reboot
```

Po restartování zařízení se může již přistoupit k vytvoření prvního kontejneru. Docker má vlastní repositář image pro kontejnery a je dostupný na adrese https://hub.docker.com, ale dají se stáhnout i z externích repositářích. Vzhledem k tomu, že Docker běží na zařízeních s ARM architekturou je potřeba vybrat kompatibilní image pro touto architekturu[15].

V následujícím příkladu se vytvoří kontejner s Ubuntu. Kontejnery se startují příkazem `docker run název_image`. Za názvem image tu následuje, kam se kontejner při připojení má zapnout. Tedy při připojení na Ubuntu se zapne: `/bin/bash`. Před název image mohou následovat volitelné parametry[16]. V tomto případě je použito:

- `-i` – drží otevřený vstupní stream, i když není žádný přístup ke kontejneru,
- `-t` – otevírá pseudoterminál (důležité u kontejnerů s OS, jinak se kontejner po startu ukončí),
- `--name` – určuje jméno kontejneru,
- `--restart` – určuje, kdy se má znovu zapnout kontejner (`always` zajistí zapnutí i při startu, a nejen při chybě).

```
$ docker run -it --name ubuntu1 --restart=always armhf/ubuntu
/bin/bash
```

Při spuštění příkazu se image začne stahovat z repositáře a po stažení se nainstaluje. Po nainstalování se automaticky terminál přepne do kontejneru s Ubuntu. To, že se nachází terminál v kontejneru, se pozná tak, že vedle jména uživatele je vypsáno ID kontejneru. Kontejner se opouští za pomoci příkazu „exit“. Kontejnery se dají procházet za pomoci následujícího příkazu. Pomoci tohoto příkazu se dá zjistit například stav, jméno a ID jednotlivých kontejnerů.

```
$ docker container ls
```

[15] Image kompatibilní pro ARM se například nacházejí na: https://hub.docker.com/u/armhf/.
[16] Více parametrů dostupné na: https://docs.docker.com/engine/reference/run/.

Ke kontejneru se dá znovu připojit za pomoci následujícího příkazu. Do příkazu se zadává název kontejneru, nebo jeho ID. V tomto příkladu příkazu, je použit název kontejneru.

```
$ docker container attach ubuntu1
```

Znovu se opustí terminál kontejneru a nainstaluje se Portainer. Ten slouží ke snadné správě kontejnerů Dockeru za pomoci webového prohlížeče. Zde se v příkazu objevují další tři používané parametry. Prvním je parametr `-d` který, zajistí, že se kontejner běží na pozadí. Dalším je -p kde se mapují porty „hostitel:kontejner“, tím je kontejner dostupný na portu hostilete z vnějšku. Posledním parametrem je `-v` a ten slouží pro připojení dat z hostitele do kontejneru.

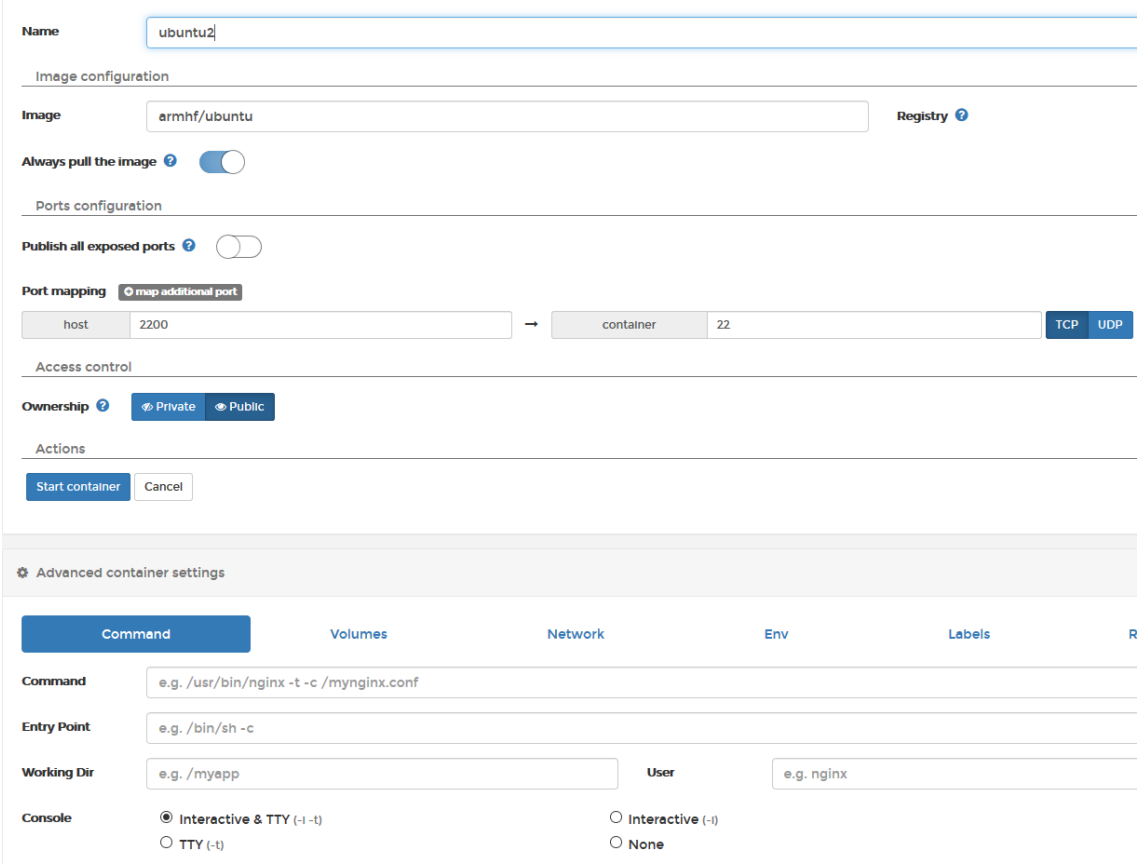
```
$ docker run --restart=always --name WebDocker -d -p 9001:9000
-v /var/run/docker.sock:/var/run/docker.sock
portainer/portainer
```

Po zapnutí kontejneru se ve webovém prohlížecí přistoupí na adresu http://192.168.0.190:9001. Na úvodní obrazovce Portaineru se nastaví heslo a potvrdí. Následuje nastavení připojení k Dockeru, jak je vidět na následující obrázku. Portaineru se nastaví, že se nachází na lokálním stroji, a potvrdí. Výchozí uživatel je: admin a heslo bylo zvoleno 123456.

Obrázek 31 – Prvotní nastavení Portaineru

Zdroj: vlastní

Nyní se již otevře úvodní obrazovka s přihlášením. Po přihlášení se již v prohlížeči načte aplikace pro správu kontejnerů. Obdobně jako tomu bylo v konzoli, tak se zde pohodlně za pomoci GUI může vytvořit kontejner s Ubuntu. V menu se vybere položka „Containers“. Na této stránce se nachází seznam kontejnerů a dají se zde vytvářet nové. Při vytvoření nového kontejneru se dají zadávat jednoduše jeho parametry. Zde bylo zvoleno: jméno ubuntu2, image armhf/ubuntu, přidáno mapování portů pro SSH z 22 na 2200 a zapnuty parametry `-it`. Po nastavení, jako je vidět v následujícím obrázku, může být kontejner vytvořen.



Obrázek 32 – Vytvoření Ubuntu v Portaineru

Zdroj: vlastní

Po vytvoření se v seznamu kontejnerů vybere nově vytvořený kontejner. Na této stránce se dá provádět nastavení kontejnerů, číst informace, nebo je zapnout a vypnout. Dále se dají číst logy a připojit k terminálu kontejneru. Teď se provede připojení do konzole, jak je znázorněno v následujícím obrázku. Pro načtení konzole se ještě musí kliknout na tlačítko „Connect“.

Obrázek 33 – Připojení ke konzoli

Zdroj: vlastní

V otevřeném terminálu se jako první příkaz provede aktualizace seznamu dostupných balíčků. Dále se nainstaluje textový editor `nano`. Posléze se nainstaluje `openssh-server`, který slouží pro povolení přístupu za pomoci SSH. Při instalaci SSH je potřeba potvrdit stažení dalších balíčků závislostí. Poté je nutné službu SSH ještě spustit a zkontrolovat její stav.

```
$ apt-get update

$ apt-get install nano

$ apt-get install openssh-server

$ service ssh start

$ service ssh status
```

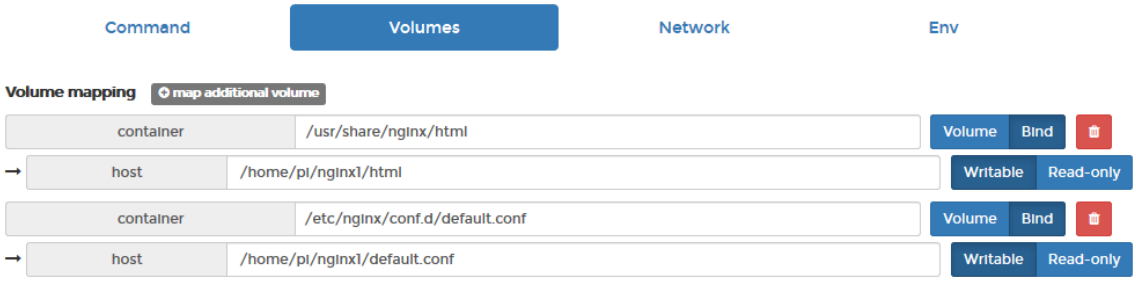
Dále se vytvoří v kontejneru nový uživatel „user“ s heslem „123456“. Posléze se nainstaluje balíček `sudo` a nový uživatel se přiřadí do skupiny sudo, aby mohl operovat s oprávněním uživatele root.

```
$ adduser user

$ apt-get install sudo

$ usermod -a -G sudo user
```

Následně se tato konfigurace může otestovat. Například za pomoci programu *PuTTY* se může klient připojit k SSH. Jako adresa se uvede 192.168.0.190 a port 2200. Takto vytvořený kontejner demonstruje poskytování operačního systému jako službu. Nyní lze kontejner opustit.

Dále se vytvoří kontejner pro web s konfiguračním souborem a složkou pro web mimo kontejner v hostiteli. Konkrétně se jedná o webserver NGINX. Opět se v menu přejde do

seznamu kontejnerů a vytvoří se nový se jménem nginx1. Jako image bude vybrán „werwolfby/armhf-alpine-nginx“. Dále se provede mapování portů na port 80. Připojení souborů z hostitele do kontejneru je vidět v následujícím obrázku.



Obrázek 34 – Připojení souborů z hostitele do kontejneru

Zdroj: vlastní

Ještě je potřena vytvořit vlastní `index.html`. Do souboru může být nakopírován obsah Přílohy B – *index.html*, nebo může být vytvořen vlastní. Konfigurační soubor může být ponechán výchozí, nebo může být dle požadavků upraven. Konfigurační soubor by vytvářel druhý příkaz. Poté může být web otestován na adrese: http://192.168.0.190.

```
$ sudo nano /home/pi/nginx1/html/index.htm

$ sudo nano /home/pi/nginx1/default.conf
```

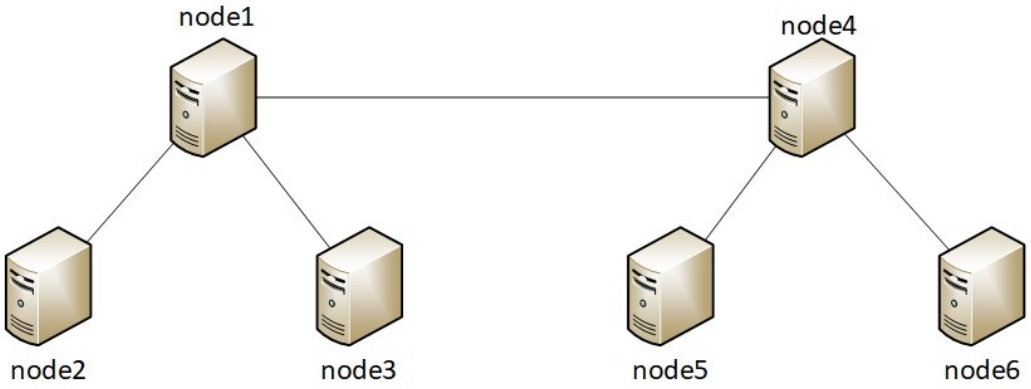
Pro porovnání se toto nastavení, které bylo nastaveno za pomocí GUI, dá provést i následujícím příkazem. Pro otestování příkazu se po vyzkoušení nastavení z GUI může tento kontejner smazat a otestovat stejnou funkčnost u příkazu.

```
$ docker run --name nginx1 -p 80:80 -
v ~/nginx1/html:/usr/share/nginx/html -
v ~/nginx1/default.conf:/etc/nginx/conf.d/default.conf -d
werwolfby/armhf-alpine-nginx
```

Kontejnery se dají mazat ze seznamu kontejnerů a přímo v detailu kontejneru na webu. Nebo se dají promazat příkazem. V následujícím příkazu se smaže kontejner „hello-world“, který byl vytvořen pro test Docker při vytváření prvotního image. K příkazu se přidává název kontejneru, nebo jeho ID.

```
$ docker container rm jovial_davinci
```

Po otestování možností kontejneru na jednom zařízení se přistoupí k vytvoření clusteru za pomoci Docker Swarm. U Swarmu se rozlišují nody na manažery a pracovníky. Manažeři jsou nody 1 a 4. Ostatní nody jsou pracovníci a vždy je spravuje jeden z manažerů. Logická topologie v následujícím obrázku znázorňuje, jak jsou jednotlivé nody se sebou provázány. Redundance managerů napomáhá při zajištění vyšší spolehlivosti.



Obrázek 35 − Logické zapojení v Docker Swarm

Zdroj: vlastní

V node1 se inicializuje cluster a stane se z něj první manažer. Inicializace se provede následujícím příkazem. Po vykonání příkazu se vypíše ihned příkaz, pomocí něhož se dá připojit další nody. Příkaz se pozná tím, že začíná `docker swarm join`. Dále příkaz obsahuje vygenerovaný token, pomocí něhož se ostatní mohou připojit, a IP adresu s portem. Tento příkaz je dobré si poznamenat.

```
$ docker swarm init
```

Po odhlášení a následném přihlášení v Portaineru se objeví další dvě položky v menu. Položka „Swarm“ znázorňuje cluster a jeho nody se stavy. Další je položka „Service“, kde se již dají spouštět a nastavovat služby, které spravují kontejnery. Aby se dal dobře sledovat cluster a stav jednotlivých nodů se službami, tak se nainstaluje visualizer pro Docker Swarm. K jeho zobrazení se dá přistoupit za pomoci webového prohlížeče na adrese http://192.168.0.190:8081. Dá se v něm dobře sledovat, jak se v následujících částech budou přidávat další nody a služby.

```
$ docker run --restart=always -it -d -p 8081:8080 -
v /var/run/docker.sock:/var/run/docker.sock
alexellis2/visualizer-arm:latest
```

Následuje nastavení druhého nodu a jeho připojení do clusteru. Jeho IP adrese se nastaví na 192.168.0.191 a jméno na „node2“ obdobným způsobem jako tomu bylo u „node1“. Poté se zařízení restartuje.

```
$ sudo nano /etc/dhcpcd.conf

static ip_address=192.168.0.191/24

$ sudo nano /etc/hosts

$ sudo nano /etc/hostname

$ sudo reboot
```

Po restartování se „node2“ připojí do clusteru. Je potřeba do příkazu zadat vlastní token, nebo použít celý příkaz, který byl předtím v node1 vypsán. Po zadání příkazu se zapojí do clusteru a tato změna je vidět ve visualizeru.

```
$ docker swarm join \

    --token VLASTNÍ_TOKEN \

    192.168.0.190:2377
```

Znovu se připojí další zařízení a postupuje se s jeho nastavením stejným způsobem jako u předchozího zařízení. Jen se změní jeho jméno na „node3“ a IP adresa na 192.168.0.192.

```
$ sudo nano /etc/dhcpcd.conf

static ip_address=192.168.0.192/24

$ sudo nano /etc/hosts

$ sudo nano /etc/hostname

$ sudo reboot

$ docker swarm join --token VLASTNÍ_TOKEN 192.168.0.190:2377
```

Znovu se přesune na terminál „node1“, kde se získá token s příkazem pro připojení druhého managera. Opět se příkaz, který je vygenerován zaznamená.

```
$ docker swarm join-token manager
```

Poté se znovu připojí další zařízení a nastaví jméno „node4“ s IP adresa 192.168.0.193. Ale pozor při připojování do clusteru, tentokrát využije nově vygenerovaného příkazu především jeho tokenu z předešlého odstavce. Po úspěšném připojení se vygeneruje příkaz pro připojení dalších pracovníků k němu. Vygenerovaný příkaz se opět zaznamená.

```
$ sudo nano /etc/dhcpcd.conf

static ip_address=192.168.0.193/24

$ sudo nano /etc/hosts

$ sudo nano /etc/hostname

$ sudo reboot

$ docker swarm join --token VLASTNÍ_TOKEN 192.168.0.190:2377

$ docker swarm join-token worker
```

Následuje zapojení dalšího zařízení a jeho následné nastavení. Nastavení se odehrává stejně jako u předešlých, jen se provede změna v připojení do clusteru a připojí se na „node4“. Tedy se změní IP adresa v příkazu pro připojení do clusteru. Jméno u zařízení je „node5“ a IP adresa 192.168.0.194.

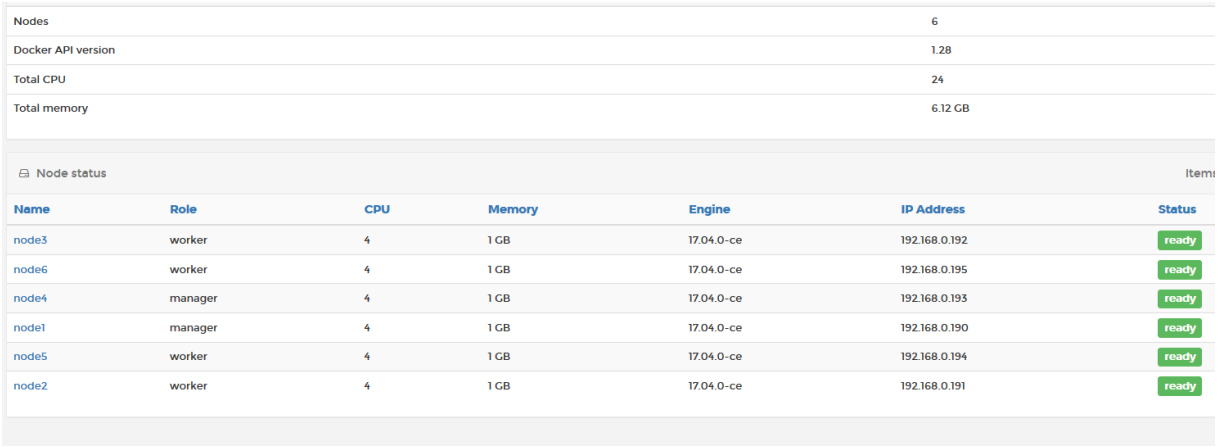
```
$ sudo nano /etc/dhcpcd.conf

static ip_address=192.168.0.194/24

$ sudo nano /etc/hosts

$ sudo nano /etc/hostname

$ sudo reboot

$ docker swarm join --token VLASTNÍ_TOKEN 192.168.0.193:2377
```

Jako další krok se již připojí poslední zařízení obdobně, jako tomu bylo v předešlém kroku. Jen se změnou jména na „node6“ a IP adresy na 192.168.0.195.

```
$ sudo nano /etc/dhcpcd.conf

static ip_address=192.168.0.195/24

$ sudo nano /etc/hosts
```

```
$ sudo nano /etc/hostname

$ sudo reboot

$ docker swarm join --token VLASTNÍ_TOKEN 192.168.0.193:2377
```

Teď jsou všechny nody v clusteru připraveny. A může se přistoupit ke spuštění ukázkových služeb. Na následujícím obrázku jsou zobrazeny všechny nody a celkový výkon clusteru.



Nodes	6
Docker API version	1.28
Total CPU	24
Total memory	6.12 GB

Node status

Name	Role	CPU	Memory	Engine	IP Address	Status
node3	worker	4	1 GB	17.04.0-ce	192.168.0.192	ready
node6	worker	4	1 GB	17.04.0-ce	192.168.0.195	ready
node4	manager	4	1 GB	17.04.0-ce	192.168.0.193	ready
node1	manager	4	1 GB	17.04.0-ce	192.168.0.190	ready
node5	worker	4	1 GB	17.04.0-ce	192.168.0.194	ready
node2	worker	4	1 GB	17.04.0-ce	192.168.0.191	ready

Obrázek 36 – Pohled na cluster

Zdroj: vlastní

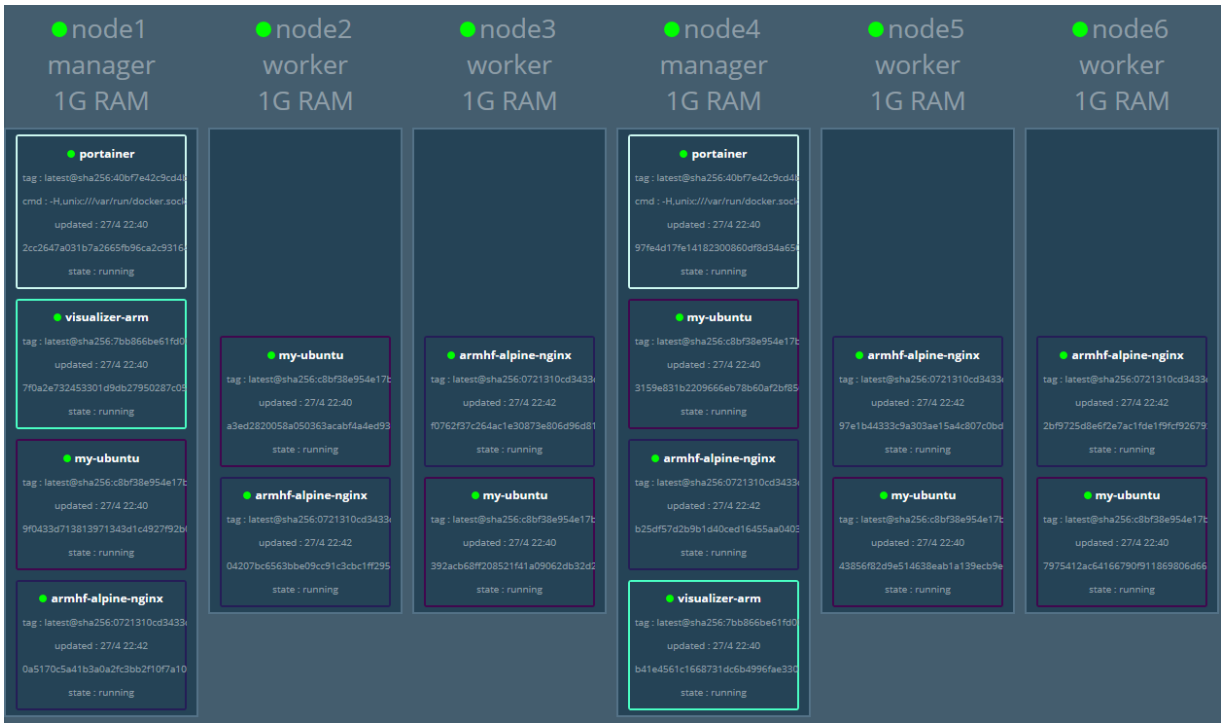
Jako první služba se nainstaluje na oba manažery Portainer. Znovu se musí vytvořit heslo a nastavit webovou aplikaci. K aplikaci se tentokrát přistoupí na adrese jednoho z managerů na portu 9000. Tedy http://192.168.0.190:9000, nebo http://192.168.0.193:9000.

```
$ docker service create --name portainer --replicas 2

--publish 9000:9000 --constraint 'node.role == manager' \

--mount
type=bind,src=/var/run/docker.sock,dst=/var/run/docker.sock \

portainer/portainer -H unix:///var/run/docker.sock
```

Další službou bude nainstalování visualizeru jako služby. Služba bude dostupná v prohlížeči na adrese obou manažerů na portu „8080“.

```
$ docker service create --name=visualizer --replicas 2 \
```

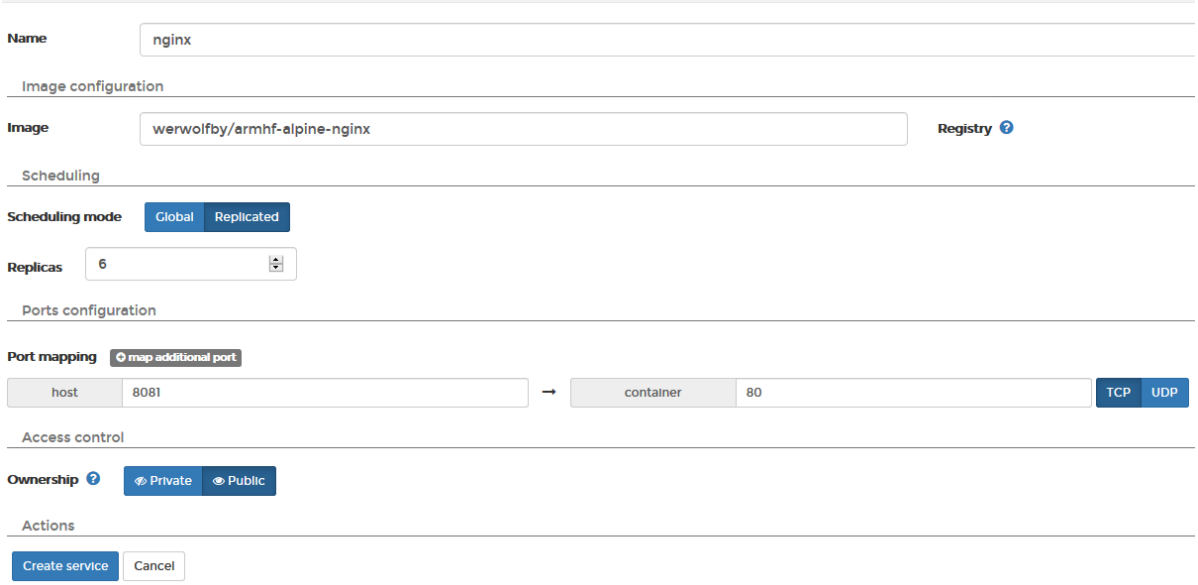
```
--publish=8080:8080/tcp --constraint=node.role==manager --
mount=type=bind,src=/var/run/docker.sock,dst=/var/run/docker.s
ock alexellis2/visualizer-arm:latest
```



Obrázek 37 – Visualizer clusteru

Zdroj: vlastní

Následně se vytvoří další služba za pomoci Portaineru. V menu se vybere položka „Services“ a v seznamu se vytvoří nová služba. Služba bude vytvořena následovně: jméno „nginx“, image „werwolfby/armhf-alpine-nginx“, počet replikací kontejneru 6 a mapování portů z „80“ na „8081“. Vytváření služby je zachyceno na následujícím obrázku. Služba se dá ve webovém prohlížeči otestovat například na adrese http://192.168.0.193:8081, ale funguje i na adresách ostatních nodů.



Obrázek 38 – Vytvoření služby NGINX

Zdroj: vlastní

Jako další služba se spustí OS s Debian, který bude replikován ve čtyřech kontejnerech. Jak se spouští další kontejnery, tak se ve visualizeru dá dobře sledovat, jak jsou kontejnery rozmístěny mezi všemi nody. Replikace kontejneru zajistí to, že když některé spadnou tak služba běží bez postihu dále. Kontejnery provázané s jednou službou spolu udržují stav.

```
$ docker service create -t --name debian --publish 2202:22 --
replicas=4 armhf/debian
```

V dalším kroku se vytvoří na zařízení „node1“ vlastní registry pro image Dockeru. Toto bude sloužit pro potřeby sdílení vlastních, či upravených image z kontejnerů mezi nody. Poté se dá publikovat a stahovat vlastní image na adrese „192.168.0.190:5000“.

```
$ docker run --restart=always -d -v /srv/registry/data:/data -
p 5000:5000 --name registry silverwind/armhf-registry
```

Dále se úplně na všech nodech musí přidat konfigurace, která povolí komunikaci s vlastními registry. Pro funkčnost bez tohoto nastavení by musel mít stroj platné certifikáty spolu s povoleným https.

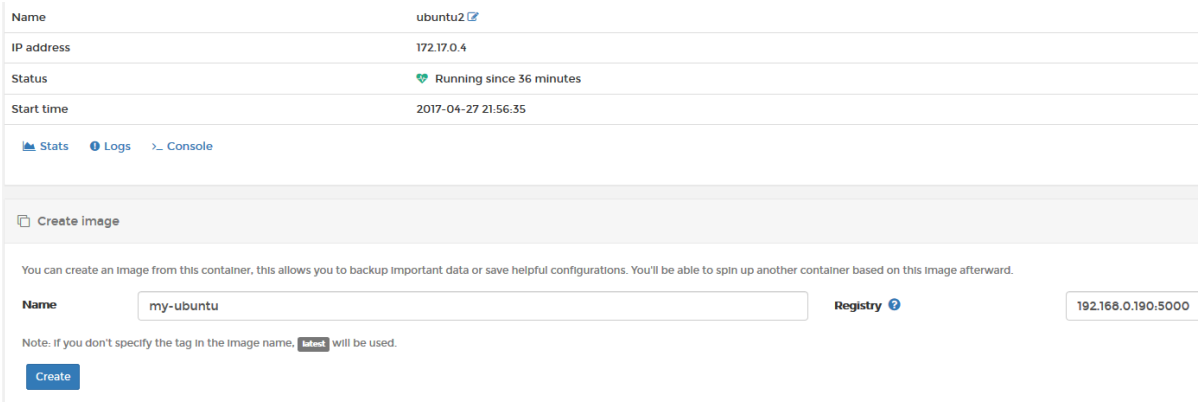
```
$ sudo nano /etc/docker/daemon.json

{

    "insecure-registries": ["192.168.0.190:5000"]
```

```
}
```

Po nastavení se služba Docker ještě musí restartovat, aby se aplikovalo toto natavení.

```
$ sudo service docker restart
```

Následně se může na „node1“ pro ukázku do repositáře uložit image s vlastním upraveným Ubuntu, do kterého se nainstalovala základní výbava balíčků. Kontejner se jmenuje „ubuntu2“ a nahraje se například jako image se jménem „my-ubuntu“. Důležité je také zadat adresu vlastního registru na adrese „192.168.0.190:5000“, jinak jsou použity výchozí registry. Celý proces vytvoření image je znázorněn na Obrázek 39 − Vytvoření vlastního image. Po vytvoření stačí už jen nahrát tlačítkem „push to registry“ a tím je image uložen již ve vlastním registru.



Obrázek 39 − Vytvoření vlastního image

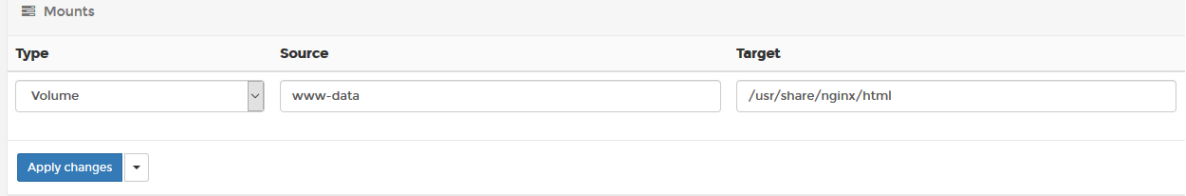
Zdroj: vlastní

Díky vlastnímu image se může vytvořit služba z vlastního image, který může být libovolně upraven dle potřeb. Jako image se při vytvoření služby zadává odkaz do vlastních registrů „192.168.0.190:5000/my-ubuntu:latest“. Image se tak stáhne do kontejnerů z vlastního registru běžícím na prvním nodu.

```
$ docker service create -t --name mojeubuntu --publish 2222:22
--replicas=6 192.168.0.190:5000/my-ubuntu:latest
```

U jednotlivých kontejnerů se dá v Portaineru sledovat využití zdrojů. Na tuto funkcionalitu se dá dostat otevřením libovolného kontejneru a následným otevřeným položky „Stats“. Na stránce se zobrazí využití zdrojů aktuálně vybraného kontejneru. Ukázka této stránky je přiložena v Příloze C – *Monitorování zdrojů*.

Mezi kontejnery se dají také mapovat sdílené jednoty, které zajistí sdílení dat mezi několika kontejnery. V menu Portaineru stačí vybrat položku „Volumes“ a v seznamu vytvořit nový. Důležité je především zadat název jednotky. Poté se již při vytváření kontejneru může vytvořená jednotka přiřadit ke kontejneru, nebo službě. Jednotka se připojí pomocí jejího názvu do libovolného umístění v kontejneru. Připojení je znázorněné na následujícím obrázku.



Obrázek 40 – Připojení vlastní jednotky ke službě

Zdroj: vlastní

Poslední ukázkou je odpojení jednoho z nodů. Při odpojení dojde clusterem k indikování výpadku a dojde ke spuštění kontejnerů, které běžely na zařízení postižené výpadkem k jejich spuštění na ostatních zařízeních clusteru. Tato situace je zachycena i ve visualizeru v Příloze D – *Ukázka výpadku jednoho z nodů v clusteru*.

Name	Role	CPU	Memory	Engine	IP Address	Status
node3	worker	4	1 GB	17.04.0-ce	192.168.0.192	ready
node6	worker	4	1 GB	17.04.0-ce	192.168.0.195	down
node4	manager	4	1 GB	17.04.0-ce	192.168.0.193	ready
node1	manager	4	1 GB	17.04.0-ce	192.168.0.190	ready
node5	worker	4	1 GB	17.04.0-ce	192.168.0.194	ready
node2	worker	4	1 GB	17.04.0-ce	192.168.0.191	ready

Obrázek 41 – Výpadek jednoho nodu z clusteru

Zdroj: vlastní

Za pomoci vytvoření clusteru Docker Swarm se dá vybudovat odolné datové centrum. Cluster dokáže udržet služby aktivní i pří výpadku několika zařízení. Stav clustery a správu clusteru řídí zařízení typu manažer. S vyšším počtem zařízení, které jsou manažery je již velice nepravděpodobné ochromení cloudového datového centra.

ZÁVĚR

Cílem této práce bylo představit princip fungování cloudového datového centra a na základě analýzy zjištěných poznatků provést implementaci cloudového datového centra. Následně podle těchto poznatků bylo za úkol implementovat vhodné řešení cloudové datového centra na zařízeních Raspberry Pi.

V kapitole Úvod jsou vysvětleny důvody pro vznik, používání cloudu a vysvětlení co je a není cloud. Na toto již plynule navazuje část, která definuje základních pět charakteristik cloudu, které musí splňovat. Jako pět charakteristik cloudu byly uvedeny: samoobslužný princip, síťový přístup, fond prostředků, elasticita zdrojů a měření prostředků. Dále bylo potřeba definovat základní termíny používané v cloudu. Poté se přistoupilo k představení modelů služeb. Byly popsány tři hlavní modely služeb: infrastruktura jako služba, platforma jako služba a software jako služba. Poté byly shrnuty ostatní modely služeb. V cloudu se řeší, jak je pracováno s daty v cloudu a kdo má mít ke cloudu přístup. Proto cloud definuje různé modely nasazení, čemuž se věnuje další část práce, která je rozdělila do skupin a podrobně popsala. Po rozdělení modelů cloudu následovalo představení jednotlivých aktérů, kteří v něm vystupují. Řešení cloudových služeb se věnuje celá řada společností a v další části práce byly představeny ty nejznámější. Protože cloud pracuje s daty uživatelů je důležité je chránit před zcizením nebo ztrátou a tomuto tématu se věnuje následující část práce. Se zabezpečením cloudu také významně souvisí legislativní předpisy v cloudu, a proto se další část práce zabývá předpisy a legislativou. Pro cloud bylo také důležité seznámit čtenáře s prostředky, které jsou ve velké míře využívány při tvorbě cloudového datového centra. Proto dále následovalo seznámení s virtualizací a kontejnery. U virtualizace a kontejnerů byly vysvětleny jejich principy a představeny prostředky pro jejich spuštění. V další části práce byly představeny zařízení Raspberry Pi a NAS, které byly potřeba pro praktickou část práce. U zařízení Raspberry Pi byl kladen důraz na jeho možnosti využití a u NAS především na RAID.

Teoretická část této práce si kladla za cíl představit fungování cloudových datových center, protože pro praktickou část této práce bylo nezbytné jednotlivým částem nejdříve alespoň částečně porozumět. Jako první se v praktické části provedla prvotní konfigurace zařízeních Raspberry Pi a NAS. Poté se již přešlo k realizaci cloudového datového centra pro sdílení souborů, jejíž vytváření je podrobně popsáno v kapitole 6.3. Další praktickou ukázkou bylo vytvoření cloudového datového centra poskytujícího služby, pomocí kontejnerů Docker. Všechna zařízení byla propojena do clusteru za pomoci Docker Swarm, což zajistilo vysokou stabilitu a dostupnost služeb.

Pokud se v cloudu budou využívat kontejnery, tak se zajistí dobrého využití zdrojů. Datové centrum, ve kterém běží služby replikovány na vícero zařízeních, má vysokou dostupnost služeb a odolnost vůči výpadku, protože při výpadku jednoho ze zařízení převezme jeho činnost některé z dalších.

POUŽITÁ LITERATURA

ANKERHOLZ, Amber. 2016. *8 Container Orchestration Tools to Know* [online]. [cit. 24. 3. 2017]. Dostupné z: https://www.linux.com/news/8-open-source-CONTAINER-ORCHESTRATION-TOOLS-KNOW

ASUSTOR COLLEGE. 2013. Seznámení s protokoly pro přenos dat. In: *Asustor* [online]. [cit. 16. 2. 2017]. Dostupné z: http://download.asustor.com/college/cs/NAS_102_Introduction_to_File_Transfer_Protocols.pdf

BANERJEE, Tobby. 2014. *Understanding the key differences between LXC and Docker* [online]. [cit. 20. 3. 2017]. Dostupné z: https://www.flockport.com/lxc-vs-docker/

BECHYNSKÝ, Štěpán. 2015. *Raspberry Pi 2 a Windows 10 IoT Core* [online]. [cit. 25. 2. 2017]. ISSN 1803-5620. Dostupné z: https://www.zdrojak.cz/clanky/raspberry-pi-2-windows-10-iot-core/

BETZ, Mark. 2016. *Five alternatives to Docker you should consider* [online]. [cit. 20. 3. 2017]. Dostupné z: http://searchcloudapplications.techtarget.com/tip/Five-development-containers-to-consider-that-arent-Docker

BEZPALEC, Pavel. 2014. *Role bezpečnosti v Cloudu* [online]. Dostupné z: https://publi.cz/books/230/06.html

BINDER, Marcel. 2014. *The Right NAS Strategy: Optimal Use of Central Storage* [online]. [cit. 15. 2. 2017]. Dostupné z: http://www.tomshardware.co.uk/nas-raid-2-way-sync-replication,review-32879-3.html

BOUŠKA, Petr. 2008. *Computer Storage - architektury, protokoly, rozhraní* [online]. [cit. 11. 2. 2017]. Dostupné z: http://www.samuraj-cz.com/clanek/computer-storage-architektury-protokoly-rozhranni/

CESNET. 2016. *Definice cloudu* [online]. [cit. 2. 3. 2017]. Dostupné z: https://wiki.metacentrum.cz/wiki/Definice_cloudu

Cloud Productivity. 2017. *What is the Cloud?* [online]. [cit. 25. 2. 2017]. Dostupné z: http://www.cloudproductivity.net/what-is-cloud/

Computer world. 2014. *Kontejnery ulehčí virtualizaci Linuxu* [online]. [cit. 24. 5. 2017]. Dostupné z: http://computerworld.cz/software/kontejnery-ulehci-virtualizaci-linuxu-51044

ČÍŽEK, Jakub. 2015. *DLNA: snadné přehrávání filmů z PC v TV* [online]. [cit. 10. 2. 2017]. ISSN 1213-8991. Dostupné z: http://avmania.e15.cz/dlna-snadne-prehravani-filmu-z-pc-v-tv

ČÍŽEK, Jakub. 2014. *Vy ještě nemáte domácí NAS?* [online]. [cit. 5. 2. 2017]. ISSN 1213-8991. Dostupné z: http://www.zive.cz/clanky/vy-jeste-nemate-domaci-nas/1-nas-ma-povedene-graficke-prostredi/sc-3-a-172446-ch-91048/default.aspx#articleStart

DOČEKAL, Michal. 2009. *Správa linuxového serveru: RAID teoreticky* [online]. [cit. 12. 2. 2017]. ISSN 1801-3996. Dostupné z: https://www.linuxexpres.cz/praxe/sprava-linuxoveho-serveru-raid-teoreticky

DOHERTY, Jim. 2016. *SDN and NFV simplified: a visual guide to understanding software defined networks and network function virtualization*. Upper Saddle River: Pearson Education, Inc. ISBN 978-0-13-430640-7.

DONOVAN, John. 2013. *Mmm—Raspberry Pi!* [online]. [cit. 20. 2. 2017]. Dostupné z: http://low-powerdesign.com/donovansbrain/2013/02/27/mmm%E2%80%94raspberry-pi/

FileCloud. 2015. *Cloud Terminology – Key Definitions* [online]. [cit. 29. 3. 2017]. Dostupné z: https://www.getfilecloud.com/cloud-terminology-glossary/

flow.ci. 2016. *Introduction to Containers: Concept, Pros and Cons, Orchestration, Docker, and Other Alternatives* [online]. [cit. 23. 3. 2017]. Dostupné z: https://medium.com/flow-ci/introduction-to-containers-concept-pros-and-cons-orchestration-docker-and-other-alternatives-9a2f1b61132c#.89m7cnf9f

FOLDYNA, Petr. 2012. *Cloud není revoluce, ale evoluce* [online]. [cit. 2. 3. 2017]. Dostupné z: https://www.systemonline.cz/virtualizace/cloud-neni-revoluce-ale-evoluce-1.htm

FORREST, Conner. 2014. *The Google Cloud Platform: 10 things you need to know* [online]. [cit. 5. 3. 2017]. Dostupné z: http://www.techrepublic.com/article/the-google-cloud-platform-10-things-you-need-to-know/

Hospodářské noviny IHNED.cz. 2015. *Cloudové služby se rozmnožily aneb co všechno je dnes v cloudu* [online]. [cit. 3. 3. 2017]. ISSN 1213-7693. Dostupné z: http://ictrevue.ihned.cz/c3-63904590-0ICT00_d-63904590-cloudove-sluzby-se-rozmnozily-aneb-co-vsechno-je-dnes-v-cloudu%20%E2%80%93%20zkontrolovat%20podle%20zdroje

HŮNA, Jaroslav. 2016. *Průvodce NOOBS* [online]. [cit. 22. 2. 2017]. Dostupné z: https://rpiblog.cz/archiv/112

HŮNA, Jaroslav. 2016. *Stručné seznámení s Raspberry Pi* [online]. [cit. 20. 2. 2017]. Dostupné z: http://rpiblog.cz/archiv/42

HŮNA, Jaroslav. 2015. *Vyšel Windows 10 pro Raspberry Pi 2* [online]. [cit. 24. 2. 2017]. Dostupné z: http://rpiblog.cz/archiv/674

iDNES.cz. 2010. *Je řešení cluster pro každého?* [online]. [cit. 2. 3. 2017]. Dostupné z: http://sdeleni.idnes.cz/je-reseni-cluster-pro-kazdeho-d2e-/tec_sdeleni.aspx?c=A100517_105452_tec_sdeleni_ahr

JEŽEK, David. 2017. *Raspberry Pi Compute Module 3 je 10× rychlejší než předchůdce* [online]. [cit. 22. 2. 2017]. ISSN 1213-2225. Dostupné z: http://diit.cz/clanek/raspberry-pi-compute-module-3-je-10x-rychlejsi-nez-predchudce

KMOCH, Ondřej. 2014. *Některé právní aspekty cloud computingu* [online]. [cit. 9. 3. 2017]. Dostupné z: http://www.cak.cz/scripts/detail.php?id=13424

KREISL, Marek a Daniel PIKAL. 2016. *Cloud a právo: pozor na umístění vašich dat* [online]. [cit. 9. 3. 2017]. ISSN 1212-8309. Dostupné z: https://www.root.cz/clanky/cloud-a-pravo-pozor-na-umisteni-vasich-dat/

KUBEŠ, Radek a Josef MIKA. 2016. *Chip Speciál - NAS a domácí síť*. Praha: Burda Praha, spol. s r. o. ISBN 1210-0684.

KUBICA, Tomáš. 2016. *DC/OS, Azure a cloud native platforma* [online]. [cit. 24. 3. 2017]. Dostupné z: http://www.cloudsvet.cz/?p=849

LACKO, Ľuboslav. 2012. *Osobní cloud pro domácí podnikání*. Brno: Computer Press. ISBN 978-80-251-3744-4.

MÁCHA, Petr. 2012. *Cloud computing – historie a budoucnost* [online]. [cit. 25. 2. 2017]. Dostupné z: https://www.ddconnect.cz/brezen-2012/o-datovych-centrech

MALÝ, Martin. 2011. *Co je a co není cloud* [online]. [cit. 26. 2. 2017]. ISSN 1213-0702. Dostupné z: http://www.lupa.cz/clanky/co-je-a-co-neni-cloud/

MIKA, Josef. 2017. Hrozba GDPR. *CHIP*. Praha: Burda Praha, spol. s r. o., roč. 2017, č. 3. ISSN 1210-0684.

National Institute of Standards and Technology. 2013. NIST Cloud Computing Standards Roadmap. In: *National Institute of Standards and Technology* [online]. [cit. 28. 2. 2017]. Dostupné z: http://ws680.nist.gov/publication/get_pdf.cfm?pub_id=913661

National Instruments Corporation. 2016. *NI Real-Time Hypervisor Architecture and Performance Details* [online]. [cit. 16. 3. 2017]. Dostupné z: http://www.ni.com/white-paper/9629/en/

NIEVES, Michael. 2014. Best practice in the cloud: an introduction. In: *AXELOS* [online]. [cit. 26. 2. 2017]. Dostupné z: http://itsmf.cz/wp-content/uploads/2014/11/Best_Practice_In_The_Cloud_An_Introduction.pdf

POMAZAL, Jiří. 2010. *Virtualizace v kostce* [online]. [cit. 16. 3. 2017]. ISSN 1802-615X. Dostupné z: https://www.systemonline.cz/clanky/virtualizace-v-kostce.htm

PTÁČNÍK, Jiří. 2016. *Kam ukládat filmy, hudbu a fotky? Na datové úložiště NAS* [online]. [cit. 10. 2. 2017]. ISSN 1802-9000. Dostupné z: http://www.digilidi.cz/datove-uloziste-nas

RASHID, Fahmida . 2016. *The dirty dozen: 12 cloud security threats* [online]. [cit. 5. 3. 2017]. Dostupné z: http://www.infoworld.com/article/3041078/security/the-dirty-dozen-12-cloud-security-threats.html

raspi.cz. 2014. *Představení nové univerzální desky UniPi board* [online]. [cit. 24. 2. 2017]. Dostupné z: http://www.raspi.cz/2014/07/predstaveni-nove-univerzalni-desky-unipi-board/

REX Controls. 2017. *Řídicí systém REX pro Raspberry Pi* [online]. Dostupné z: https://www.rexcontrols.cz/ridici-system-rex-raspberry-pi

RPiShop.cz. 2017. *RPiShop.cz* [online]. [cit. 25. 3. 2017]. Dostupné z: http://rpishop.cz/

SHELDON, Robert. 2016. *Windows Containers and Docker* [online]. [cit. 22. 3. 2017]. Dostupné z: https://www.simple-talk.com/cloud/platform-as-a-service/windows-containers-and-docker/

SINGH, Ashutosh. 2017. *11 Raspberry Pi OS for Everyday Computing – Best of* [online]. [cit. 12. 2. 2017]. Dostupné z: http://www.hongkiat.com/blog/pi-operating-systems/

SOLMECKE, Christian. 2013. *The legal aspects of cloud computing under copyright law* [online]. [cit. 9. 3. 2017]. Dostupné z: https://www.wbs-law.de/eng/it-law/the-legal-aspects-of-cloud-computing-under-copyright-law-45886/

SŮVA, Martin. 2012. *Záloha a archivace dat*. Plzeň: Západočeská univerzita.

Synology Inc.. 2017. *Co je to Synology Hybrid RAID* [online]. [cit. 14. 2. 2017]. Dostupné z: https://www.synology.com/cs-cz/knowledgebase/DSM/tutorial/Storage/What_is_Synology_Hybrid_RAID_SHR#t1

ŠPIČKA, Vladimír. 2013. *5 skutečností, co cloud doopravdy je a co není* [online]. [cit. 26. 2. 2017]. Dostupné z: http://computerworld.cz/technologie/5-skutecnosti-co-cloud-doopravdy-je-a-co-neni-50306

TUHÝ, Radan. 2013. *NAS: Práce s daty a sdílení pro pokročilé* [online]. [cit. 5. 2. 2017]. ISSN 1213-0818. Dostupné z: http://www.svethardware.cz/nas-prace-s-daty-a-sdileni-pro-pokrocile/37490

UČEŇ, Pavel. 2002. *Service Level Agreement aplikačních služeb?* [online]. [cit. 9. 3. 2017]. Dostupné z: https://www.systemonline.cz/clanky/service-level-agreement-aplikacnich-sluzeb.htm

UPTON, Eben a Gareth HALFACREE. 2016. *Raspberry Pi - Uživatelská příručka*. Brno: Computer Press. ISBN 978-80-251-4819-8.

VALÁŠEK, Michal. 2016. *Raspberry Pi mění svět: Seznamte se s nejzajímavějším počítačem dneška* [online]. [cit. 25. 2. 2017]. ISSN 1213-7693. Dostupné z: http://tech.ihned.cz/geekosfera/c1-65195330-raspberry-pi-meni-svet-seznamte-se-s-nejzajimavejsim-pocitacem-dneska

VELTE, Anthony, Rober ELSENPETER a Toby VELTE. 2011. *Cloud Computing*. Brno: Computer Press. ISBN 978-80-251-3333-0.

VISWANATHAN, Balaji. 2012. *Virtual Private Clouds and How They Compare vs Public* [online]. [cit. 29. 3. 2017]. Dostupné z: https://cloudtweaks.com/2012/06/4-things-to-know-about-virtual-private-clouds/

VOŘÍŠEK, Lukáš. 2012. *Představení a běh distribucí - OpenELEC XBMC, Raspbmc, Debian, Arch Linux, QtonPi* [online]. [cit. 23. 2. 2017]. ISSN 1213-2225. Dostupné z: http://cdr.cz/clanek/raspberry-pi-recenze/predstaveni-a-beh-distribuci

WANG, Chenxi. 2016. *Containers 101: Linux containers and Docker explained* [online]. [cit. 23. 3. 2017]. Dostupné z: http://www.infoworld.com/article/3072929/linux/containers-101-linux-containers-and-docker-explained.html

Waterford Technologies. 2016. *Cloud Computing Security Threats & Concerns* [online]. [cit. 5. 3. 2017]. Dostupné z: http://www.waterfordtechnologies.com/cloud-computing-security-threats-concerns/

PŘÍLOHY

Příloha A – *Nastavení NGINX*

```
upstream php-handler {
    server 127.0.0.1:9000;
    #server unix:/var/run/php5-fpm.sock;
}
server {
    listen 80;
    server_name 192.168.0.101;
    return 301 https://$server_name$request_uri;    # enforce
https
}
server {
    listen 443 ssl;
    server_name 192.168.0.101;
    ssl_certificate /etc/ssl/nginx/cert.pem;
    ssl_certificate_key /etc/ssl/nginx/cert.key;
    # Path to the root of your installation
    root /var/www/owncloud;
    client_max_body_size 4000M; # set max upload size
    fastcgi_buffers 64 4K;
    rewrite ^/caldav(.*)$ /remote.php/caldav$1 redirect;
    rewrite ^/carddav(.*)$ /remote.php/carddav$1 redirect;
    rewrite ^/webdav(.*)$ /remote.php/webdav$1 redirect;
    index index.php;
    error_page 403 /core/templates/403.php;
```

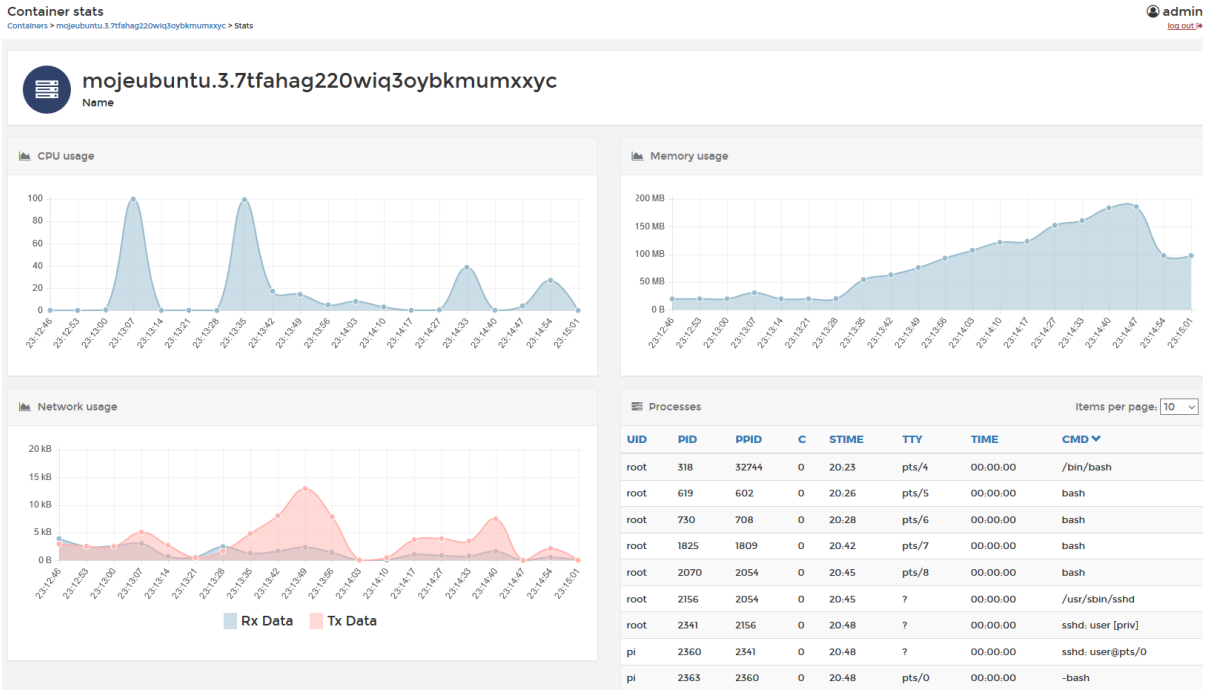
```
    error_page 404 /core/templates/404.php;
    location = /robots.txt {
        allow all;
        log_not_found off;
        access_log off;
    }
    location ~ ^/(?:\.htaccess|data|config|db_structure\.xml|README) {
        deny all;
    }
    location / {
        # The following 2 rules are only needed with webfinger
        rewrite ^/.well-known/host-meta /public.php?service=host-meta last;
        rewrite ^/.well-known/host-meta.json /public.php?service=host-meta-json last;
        rewrite ^/.well-known/carddav /remote.php/carddav/ redirect;
        rewrite ^/.well-known/caldav /remote.php/caldav/ redirect;
        rewrite ^(/core/doc/[^\/]+/)$ $1/index.html;
        try_files $uri $uri/ index.php;
    }
    location ~ \.php(?:$|/) {
        fastcgi_split_path_info ^(.+\.php)(/.+)$;
        include fastcgi_params;
```

```
        fastcgi_param                                   SCRIPT_FILENAME
$document_root$fastcgi_script_name;
        fastcgi_param PATH_INFO $fastcgi_path_info;
        fastcgi_param HTTPS on;
        fastcgi_pass php-handler;
    }
    # Optional: set long EXPIRES header on static assets
    location ~* \.(?:jpg|jpeg|gif|bmp|ico|png|css|js|swf)$ {
        expires 30d;
        # Optional: Don't log access to assets
        access_log off;
    }
}
```

Příloha B – *index.html*

```
<!DOCTYPE html>
<html lang="cs">
<head>
  <title>Docker</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
</head>
<body>
  <div class="container">
    <h1>NGINX Docker</h1>
    <p>Vítejte na webu.</p>
  </div>
</body>
</html>
```

Příloha C – *Monitorování zdrojů*



Příloha D – *Ukázka výpadku jednoho z nodů v clusteru*

Příloha E – *Obsah CD*

Na CD je přiložena diplomová práce PozdnikC_VyuzitiRaspberry_SN_2017.pdf.