Univerzita Hradec Králové
Fakulta informatiky a managementu
katedra informačních technologií

Budování infrastrukturní platformy pro IoT senzorickou IQRF full mesh síť

DIPLOMOVÁ PRÁCE

Autor: Bc. Marek Čeloud

Studijní obor: im2-p

Vedoucí práce: Ing. Jakub Pavlík, MSc.

Hradec Králové srpen, 2017

Prohlášení

Prohlašuji, že jsem diplomovou práci vypracoval samostatně a uvedl jsem všechny použité prameny a literaturu.

V Hradci Králové dne 8. srpna 2017 Marek Čeloud

Poděkování

Děkuji vedoucímu diplomové práce, Ing. Jakubu Pavlíkovi, MSc., za metodické vedení práce, odborné rady a připomínky v průběhu jejího psaní. Dále bych chtěl poděkovat za podporu své rodině, kolegům a přátelům.

Anotace

Tato diplomová práce se zaměřuje na problematiku spojenou s Internetem věcí. Tato oblast je jednou z rychle se rozšiřujících domén v celého ICT světě, která mimo jiné zasahuje i do oblastí každodenního života. S Internetem věcí jsou často spojeny i další technologie, jako je veřejný nebo privátní cloud, softwarově definované sítě apod. Všechny tyto technologie spolu utvářejí ucelenou platformu. Hlavním cílem této práce je vytvoření vlastní platformy pro Internet věcí. V závěrečné kapitole je tato platforma uvedena do praxe.

Annotation

Title: Creation of infrastructure IoT platform for sensory IQRF full-mesh network

This master thesis is focused on Internet of Things problematics. This area is one of the most dynamic domains in whole ICT world, which also involves areas of day-to-day life. There are other technologies that are tightly connected with Internet of Things like public or private cloud, software defined networking etc. All of these technologies create together comprehensive platform. Main goal of this thesis is creation of own platform for Internet of Things. The platform is brought into real world in the final chapter.

Obsah

1. Úvod

Nyní se nacházíme v době další průmyslové revoluce, která je vedená rozmachem Internetu a digitalizací. Velkou oblastí, která se v posledních letech šíří jako lavina, je Internet věcí (IoT). Internet věcí využívá technologie, které přinesla tato průmyslová revoluce, a jejich spojením vzniklo zcela nové odvětví. Jedna z prvních zmínek, která se dá alespoň vzdáleně považovat za IoT, se datuju už do roku 1984 na Tokijské univerzitě, kde se mluvilo o konceptu 'počítače všude'. Koncept, ze kterého IoT vychází, se však stává populární díky radiofrekvenční identifikaci (RFID), kde jsou věci představovány jako jedinečně identifikovatelné objekty, které mají virtuální reprezentaci. Od té doby však uplynulo mnoho let, než se dalo mluvit o Internetu věcí, a hlavním hráčem, který umožnil jeho nástup, je rozmach internetu [9]. Internet je totiž síť, která umožňuje propojit zařízení, která dříve propojit nebylo možné, a to po celém světě. IoT se stal populárním také díky šíři možnosti využití. Alespoň částečně zasahuje do všech oblastí lidského působení, jako je průmysl a výroba, zdravotnictví, doprava, služby, energetika atd. Jedním z konceptů, který poháněl IoT, je však i koncept otevřenosti - open source. Téměř každý si dnes může za minimální náklady pořídit miniaturní počítač a sadu senzorů, stáhnout volně dostupné aplikace, které jsou schopné získat analogový signál ze senzorů, přeměnit ho na digitální a vybudovat si vlastní chytrý domov. Toto je možné díky silné komunitě, která se tomuto tématu věnuje. Kromě komunity v této oblasti samozřejmě začaly participovat i velké IT firmy, které se snaží získat podíl na tomto novém trhu. Vznikají tedy celé platformy, které představují end-to-end řešení pro celý IoT systém. Tyto platformy však často končí jako proprietární řešení, kde se zákazník uzamyká jednomu dodavateli a tím může ztrácet flexibilitu. Tato práce se snaží držet konceptu otevřenosti a cílem této práce je navrhnout

vlastní IoT platformu, která je složena z komponent, které jsou open source. Většina z nich zároveň nebyla vyvinuta s účelem využití ve světě IoT. Jedná se tedy o spojení různorodých technologií dohromady tvořící ucelený systém, který by mohl mít ambice se rovnat komerčním řešením.

V této práci se nejdříve věnuji problematice Internetu věcí v obecné rovině, jaké jsou jeho komponenty a případy užití. V další kapitole se věnuji tématu IoT platforem, a to včetně popsání některých ze známých komerčních platforem, abych zjistil, jaké služby dodávají a z jakých komponent se skládají, a na základě toho mohl navrhnout alternativní IoT platformu. Tato platforma a její komponenty jsou popsány v teoretické rovině ve třetí kapitole. Důležitým bodem pro jakoukoli platformu je nasazení v reálném prostředí. To se u této platformy podařilo a byla nasazena hned na dvou konferencích. Detaily těchto implementací jsou popsány ve čtvrté kapitole.

2. IoT

V této kapitole se věnuji vymezení toho, co Internet věcí (Internet of Things - IoT) představuje, jaké jsou základní pojmy ze světa IoT, jaké existují související technologie a jaké jsou případy užití.

2.1. Vymezení IoT

Co to vlastně je Internet věcí? Zjednodušeně se dá mluvit o síti fyzických zařízení, senzorech a další elektronice, která umožňuje mezi sebou komunikovat a vyměňovat data. Internet věcí však nemá jednotnou definici a v průběhu let přišlo mnoho organizací se svou vlastní. Podle organizace CASAGRAS (Coordination and Support Action for Global RFID-related Activities and Standardization) IoT představuje globální síťovou infrastrukturu, která spojuje fyzické a virtuální objekty pomocí komunikačních možností. Tato infrastruktura obsahuje existující i nově vznikající internetový a síťový prostor. Firma SAP zase definuje IoT jako svět, kde jsou fyzické objekty bezproblémově integrovány do formy informační sítě a kde budou tyto fyzické objekty aktivně participovat v obchodních procesech. Služby budou interagovat s těmito chytrými fyzickými objekty přes internet. Budou se moci dotazovat, měnit jejich stav a informace, ale zároveň budou přitom brát ohled na bezpečnostní pravidla. IoT lze také definovat jako síť, která je složená z věcí a objektů majících identitu a virtuální osobnost. Ty pak operují v chytrém prostředí za pomocí chytrých rozhraní, jejichž prostřednictvím se pojí s uživateli a dále s nimi komunikují. Tato definice pochází od organizace EPoSS (the European Technology Platform on Smart Systems Integration).[9]

Kromě základních definic Internetu věcí můžeme definovat základní aplikační

funkce IoT systémů:

1. Informace a analýza
    - Sledování chování
    - Zvýšené povědomí o stavu
    - Rozhodovací analýza řízená senzory
2. Automatizace a kontrola
    - Optimalizace procesu
    - Optimalizace spotřeby zdrojů
    - Řízení komplexních autonomních systémů

2.1.1. Základní pojmy

V této podsekci se věnuji některým ze základních pojmů/komponent, které jsou úzce spjaté se světem IoTu. Některé další komponenty jsou vysvětleny nebo přiblíženy v následující kapitole, která je věnovaná IoT platformách. Protokoly, které jsou v Internetu věcí často používány, jsou rozebrány v další sekci.

2.1.1.1. Senzory

Senzory jsou zařízení, která detekují události a vyčítají změny v prostředí. Hrají velmi důležitou roli ve světě IoT, protože jsou jedním z hlavních zdrojů dat. Senzory však samy o sobě postrádají nějakou vyšší logiku a dá se na ně nahlížet spíše jako na konzoli, která je připojená k nějakému chytřejšímu zařízení. V dané lokalitě často utváří síť a po této síti sesbíraná data dále odesílají. Příkladem senzoru může být vodní senzor, který měří kvalitu vody spolu s množstvím znečištění. Dalším příkladem může být měřič pohybu, který zašle signál ve chvíli, kdy daný pohyb detekuje. Senzory bývají malá zařízení, nenáročná na energetickou spotřebu, často napájená z malé baterie přímo na senzoru. Některé mohou být poháněny třeba i solární energií. [2]

2.1.1.2. Aktuátory

Aktuátory představují zařízení, jež vykonávají určitou akci na základě podnětu, který byl zaslán z kontrolního bodu. Tato akce může být pohyb, rozsvícení světla, vydání zvuku, ovládání napájení apod. Jednoduchým příkladem může být otevíraní dveří pomocí dálkového ovladače. Aktuátor je zde zodpovědný za odemykání a zamykání dveří na základě instrukcí z ovladače.[2]

2.1.1.3. Označkované objekty

Radiofrekvenční identifikační (RFID) čipy umožňují jedinečnou identifikaci zařízení, které je daným čipem označeno. Čipy fungují spolu s čtečkou RFID. Ta z nich získává informace a odesílá je přes síť k dalšímu zpracování. Tato technologie je hojně využívána ve výrobě, obchodu, logistice a průmyslu. [2]

Kromě RFID do této kategorie patří NFC (Near Field Comminication). Zařízení využívající NFC spolu mohou komunikovat na krátkých radiových vlnách. Funkcionalita je zde podobná jako u RFID, ale NFC zařízení může fungovat jak čtečka, tak i čip - tedy na rozdíl od RFID může být komunikace obousměrná. Jedním z dnešních využití je čip v mobilním telefonu sloužící k platebním operacím. Stačí pouze přiložit telefon k terminálu a provést platbu díky NFC. [2]

2.1.1.4. Brány

IoT brána je zařízení, které spojuje dohromady zařízení používající různé komunikační protokoly či technologie. S bránou většinou komunikují senzory a aktuátory. Ty často slouží jako mezičlánek mezi nimi a datovým centrem.

2.1.2. Protokoly

V této podsekci se budu věnovat protokolům, kterými senzory komunikují s bránou či jiným zařízením. Těchto protokolů existuje celá řada, proto zde zmíním pouze některé. Tyto protokoly jsou alternativou k technologii IQRF, která je použita jako výchozí pro IoT platformu, jež je navrhnuta v této práci. IQRF je tedy

vysvětlen až v příslušné kapitole.

Podle [4] můžeme rozlišovat protokoly pro komunikaci na krátkou vzdálenost. Některé z těchto protokolů existují několik let, jiné byly vytvořeny nedávno.

	Protokol
Komunikace na krátkou vzdálenost	ZigBee, Bluetooth, WLAN, UWB, RFID, Bluetooth low energy, 6loWPAN
Komunikace na dlouhou vzdálenost	Optická vlákna, GSM, UMTS, LTE, Smart Grid

Tabulka 2.1.: IoT protokoly

2.1.2.1. ZigBee

Standard 802.15.4 popisuje fyzickou a linkovou vrstvu optimalizovanou pro nízkoenergetické sítě. Kromě toho ale senzory potřebují i mesh síťovou vrstvu a standardizaci pro zasílání zpráv. Za účelem standardizace této vrstvy vznikla v roce 2002 ZigBee aliance a výsledkem je ZigBee protokol.[5]

ZigBee je tedy vrstva nad 802.15.4, nad kterou formuje vlastní stoh protokolových vrstev. Fyzická a linková vrstva je, jak již bylo zmíněno, zajištěna protokolem 802.15.4. Nad těmito vrstvami je ZigBee síťová vrstva, ZigBee podpora aplikací, ZigBee objekty zařízení a ZigBee aplikační framework.[5]

ZigBee rovněž rozlišuje několik druhů rolí. První z nich je ZigBee koncové zařízení, které představuje v realitě například senzor. Koncové zařízení se musí připojit k síti za pomocí routeru. ZigBee router je druhou rolí a jedná se o zařízení, které permanentně poslouchá a chová se jako router v tradičních sítích, jakmile se připojí k ZigBee síti. Poslední rolí je ZigBee koordinátor. Koordinátor může vytvořit novou síť, nebo se připojit ke stávající. V podstatě se jedná o kontrolery v dané síti.[5]

2.1.2.2. 6LoWPAN

6LoWPAN se řadí mezi novější standardy. Tento standard umožňuje protokolu IPv6 běžet přes 802.15.4 nízkoenergetickou síť LowPAN nebo LLN.[5]

2.1.2.3. CoAP

Kromě protokolů, které se využívají pro fyzickou komunikaci v rámci IoT zařízení, je třeba zmínit protokoly, které mají na starost zpracování a odeslání dat ze senzorické sítě do nějakého centrálního místa zpracování, jako je datové centrum. Mezi nejznámější se řadí CoAP a MQTT. Jelikož je MQTT použito ve vlastní IoT platformě, je tento protokol vysvětlen až v příslušné kapitole.

Constrained Application Protocol je dalším oblíbeným protokolem ve světě IoT. Vychází ze zkušeností s HTTP protokolem a z REST architektury webového prostoru. Hlavním cílem CoAP je vytvoření generického webového protokolu pro sítě s určitým omezením, jako je třeba nízká energetická náročnost. Díky tomu, že tento protokol vychází z HTTP, je velmi snadný překlad z CoAP právě do HTTP. Interakční model je také podobný HTTP modelu client/server, který je kvůli machine-to-machine komunikaci poupraven tak, aby se zařízení mohlo chovat jako client i jako server. Čím se však CoAP liší, je využití UDP pro zprávy, které nepotřebují spolehlivé doručení. [10]

2.1.3. Čtyři pilíře IoT

Čtyři pilíře IoT tvoří M2M, RFID, WSN a SCADA. M2M (machine-to-machine) používá zařízení k zachycení nějaké skutečnosti. Toto zařízení se poté pojí přes síť k centrálnímu serveru. Zde dochází ke zpracování dat na srozumitelné informace. Druhým pilířem je RFID, které používá rádiové vlny pro přenos dat z elektronického čipu do centrálního systému přes čtečku, která je k tomuto účelu uzpůsobená. Tento čip je často připojen k objetu, o kterém má udržovat informace. Třetí pilíř se jmenuje WSN (wireless sensor network). WSN je složen z distribuovaných autonomních senzorů, které monitorují environmentální podmínky, jako je teplota, tlak, pohyb nebo znečištění. Sesbíraná data o těchto podmínkách jsou poté odeslána přes síť do centrálního místa. Posledním pilířem je SCADA (supervisory control and data acquisition). SCADA je autonomní systém založený na chytrých systémech, které spojují, monitorují a ovládají různá zařízení přes síť. Typicky se jedná o zařízení uvnitř budov nebo továren.[9]

2.2. Případy užití

V této sekci se věnuji případům užití a oblastem, ve kterých je Internet věcí používán. Mezi nejznámější patří například chytrá města, chytrý dům, chytré továrny nebo zdravotnictví.

2.2.1. Chytrá města

S rostoucím počtem obyvatel ve městech přibývá celá řada nových problémů, které je třeba řešit. Jedním z nich je třeba růst energetické spotřeby, dalším může být zhoršující se kvalita ovzduší. V neposlední řadě jsou častým jevem dopravní zácpy způsobené nadměrným počtem aut. Jedním z konkrétních problémů dnešních měst je například vysoká úroveň CO2 a nalezení řešení, jak ji snížit. Mezi příklady chytrých měst můžeme zařadit například Chicago, Rio De Janeiro, Stockholm, Hong Kong nebo Boston.

V případě Chicaga jde o tři oblasti zahrnující investice do infrastruktury, ekonomiky a komunity. Investice do infrastruktury zahrnují pokrytí města rozsáhlou optickou sítí, která umožní vysokorychlostní datové přenosy v rámci města. Zároveň také spouští projekty, jako je Health Atlas, webová stránka, která zobrazuje agregovaná data o zdravotnictví v různých lokalitách. Dále také město nabízí webový prostor pro obyvatele nebo firmy, které mají zájem spustit aplikaci, která by zlepšovala život ve městě. [8]

Případu užití chytrých měst se více věnuji v návrhu vlastní IoT platformy.

2.2.2. Chytrý dům

V případě chytrých domů se dá více méně mluvit o automatizaci vybavení domácností nebo poskytnutí pokročilejších služeb. Jednou ze součástí z oblasti chytrých domů je zabezpečení. V dnešní době existují celé řady provázaných bezpečnostních systémů, které spolu integrují alarm, IP kamery, chytré zámky apod. Kromě bezpečnosti je důležitou doménou úspora financí, která může souviset s ušetřením za energie. Zde je prostor pro chytré ovládání světel či topení. Další

oblastí, která je zajímavá pro majitele zahrad, může být chytré ovládání zavlažovacích systémů nebo chytré sekačky.

2.2.3. Chytré zdravotnictví

IoT má své využití i ve zdravotnictví například v podobě sítí WBAN (Wireless Body Area Network). Tento koncept spočívá ve vzdáleném monitorování pacientova zdravotního stavu v reálném čase. V současné době existují chytré senzory, které jsou schopné měřit třeba tepovou frekvenci, krevní tlak, hladinu glukózy nebo třeba saturaci kyslíkem. Tyto metriky jsou odeslány na IP koncové zařízení.[7]

2.2.4. Chytrý průmysl a výroba

V průmyslu a výrobě se často mluví o Průmyslu 4.0 (Industry 4.0), o Průmyslovém Internetu (Industry Internet) nebo o chytrých továrnách (smart factories). Prakticky jde o pojmenování jednoho a toho samého. Jedním z cílů chytrých továren je flexibilita. V podstatě to znamená přerod výrobních linek, které vyrábí monolitický produkt k výrobě, kde může být každý výrobek odlišný. Jednou z možných cest je vybavení produktů, nebo částí produktů RFID čipy, kde by čip nesl informaci o podobě toho daného výrobku. Další oblastí, ve které se diskutuje o IoT, jsou samotné výrobní linky a systémy, které tyto linky ovládají. Po dlouhé roky byly systémy jako EtherCAD, Modbus nebo Profibus používány pro automatizaci a ovládání průmyslových komponent. Většina těchto systémů je však proprietární a uzavřená.[8]

Velmi důležitou oblastí jsou informace, a to nejen v průmyslu a výrobě. IoT v průmyslu mění množství dat, která lze jednoduše sbírat, a možnosti, jakými lze tato data následně analyzovat. V průmyslu a výrobě však existuje další celá řada odvětví, do kterých může Internet věcí zasahovat. Můžeme mluvit například o 3D tisku, robotech nové generace nebo o přesné lokalizaci uvnitř budov. [8]

Vzhledem k tomu, jak se Internet věcí začal šířit do průmyslu a výroby, tak se kolem této oblasti začaly formovat iniciativy, které se věnují tomuto tématu. V této práci se zmíním o dvou z nich - Industry 4.0 a Industrial Internet Consortium.

2.2.4.1. Industry 4.0

Iniciativa Industry 4.0 začala jako malá skupina německých průmyslníku a výrobců. Cílem této iniciativy bylo vytvořit vizi pro novou, čtvrtou průmyslovou revoluci, která je vedena digitalizací a automatizací. V dnešní době se k iniciativě přidaly i firmy z dalších oblastí, jako jsou třeba telekomunikace - Deutsche Telekom nebo IT - IBM či SAP. Industry 4.0 se hlavně zabývá chytrými továrnami a oblastmi, které s tím souvisí, jako je třeba logistika. Šířeji se však zajímá i o oblasti chytré energetiky nebo chytrých domů.[8]

2.2.4.2. Industrial Internet Consortium

Další významnou iniciativou v oblasti chytrého průmyslu a výroby je Industrial Internet Consortium. Zakládajícími členy byly AT&T, Cisco, Inter a IBM. Kromě nich se k této iniciativě připojilo dalších 100 společností v prvním roce od jejího založení. Kromě výroby se tato iniciativa zajímá o energii, zdravotnictví, veřejný sektor nebo dopravu.[8]

3. IoT platforma

3.1. Vymezení IoT platformy

V této kapitole budou představeny některé existující IoT platformy od tradičních společností, jako jsou Intel, Microsoft, Amazon nebo Cisco. Nejprve je však potřeba vymezit, co IoT platforma představuje a z jakých komponent se skládá.

Co je IoT platforma? Zjednodušeně lze IoT platformu definovat jako IoT zařízení propojené s jinými IoT zařízeními za pomocí vhodných technologií. IoT platforma by měla představovat end-to-end řešení od senzoru až k danému koncovému uživateli.[1] Tato řešení vycházejí z referenčních modelů a standardů, které budou představeny v následujících odstavcích.

Jeden z referenčních modelů dělí komponenty do šesti vrstev. Tyto vrstvy slouží v podstatě jako předloha pro řešení buď komerční, nebo open source. Tyto vrstvy jsou:

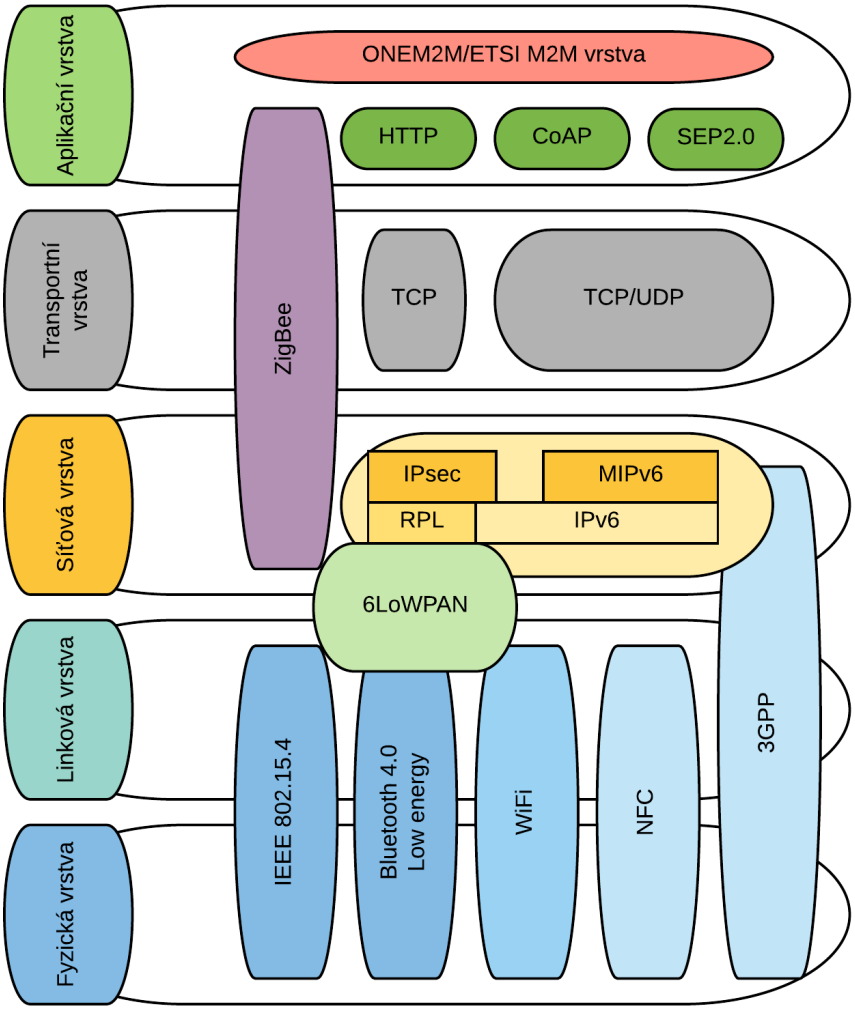
- Vrstva řešení představuje koncové aplikace pro domácnosti nebo průmysl.
- Kognitivní platforma obsahuje učící modul.
- Analytická platforma provádí šetření nad sesbíranými daty v reálném čase.
- Core platforma představuje uložiště, filtrování zpráv a komunikační middlewere.
- Komunikační vrstva v sobě zahrnuje veškeré komunikační protokoly, jako jsou MQTT, GSM, Bluetooth, IPv4, IPv6 a další.

- Fyzická zařízení jsou konkrétní IoT zařízení, jako jsou senzory, chytré přístroje a podobně. [2]

Další z referenčních modelů dělí IoT platformu na osm komponent:

1. Konektivita a normalizace zahrnuje protokoly, které umožňují propojení daných zařízení buď mezi sebou, nebo s centrálou.
2. Správa zařízení zajišťuje, že daná zařízení fungují v pořádku. Zahrnuje také způsob dodání softwarových updatů.
3. Databáze je škálovatelné uložiště, často realizované v cloudovém prostředí.
4. Správa a akce umožňuje na základě stanovených pravidel provádět činnosti, které jsou spuštěné podle daných senzorických dat.
5. Analytika poskytuje komplexní analýzu na sesbíraných datech.
6. Vizualizace umožňuje zobrazení získaných dat v člověku čitelné podobě. Často obsahuje různé grafy a 2D nebo 3D modely.
7. Nástroje poskytují vývojářům možnost vytvoření, testování a nasazení dané IoT aplikace do platformy.
8. Externí rozhraní nabízí možnost třetím stranám přistupovat k datům pomocí API.[3]

Komunikační vrstva z prvního a konektivita z druhého modelu se dá dále rozložit do dalších vrstev, které se dají napasovat na to, co již známe ze světa počítačových sítí, kde dochází k mapování daných IoT vrstev na fyzickou, linkovou, síťovou, transportní a aplikační vrstvu.



Obrázek 3.1.: IoT vrstvy[13]

Zde se fyzická vrstva skládá z velkého množství bezdrátových technologií, jako jsou Wireless Sensor Networks (WSNs), Power Line Communica-tions (PLC), Bluetooth Low Energy (BLE), Radio-Frequency IDentification (RFID), Digital Enhanced Cordless Telecommunications (DECT), Ultra-Wide Bandwidth (UWB) a Near Field Communication (NFC). Tyto technologie mají jednu charakteristiku společnou, a to nízkou energetickou náročnost. Dále se zde klade důraz na nízké pořizovací náklady, vysokou senzitivitu a vhodnou síťovou architekturu.

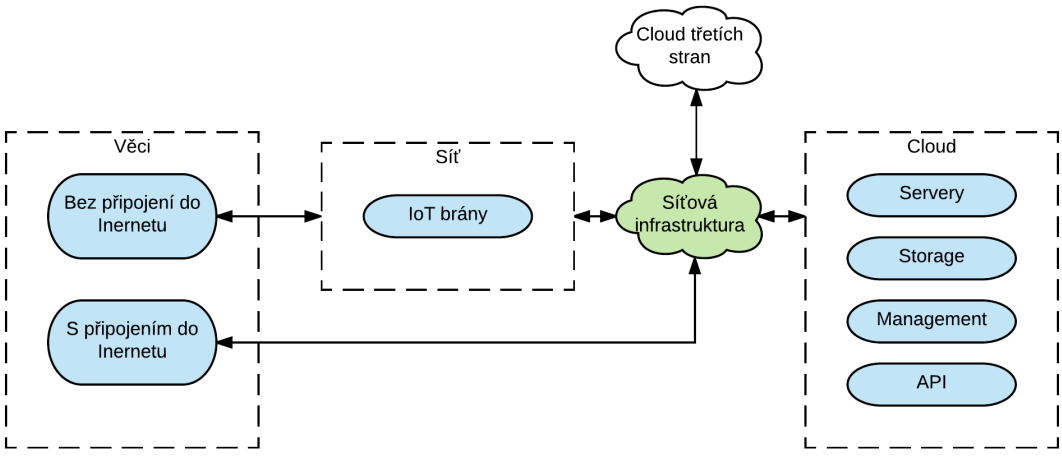
Co se týká síťové vrstvy, zde je stále největší důraz kladen na IP. Zde se však také objevují nové technologie jako 6LoWPAN nebo ZigBee, která navíc přidává i

vrstvu bezpečnosti.

Protokoly aplikační vrstvy jsou například Constrained Application Protocol (CoAP). Tento protokol se speciální webový transportní protokol, který se specializuje na sítě s nízkou energetickou náročností. Je designovaný na machine-to-machine (M2M) komunikaci a je využívaný například v chytrých domech.[10] Dalším protokolem je MQTT (Message Queuing Telemetry Transport). Tento protokol je rovněž lightweight a díky tomu ho mohou používat i ta nejmenší a nejjednodušší komunikační zařízení, a to v prostředí, kde je nízká síťová propustnost nebo vysoká odezva. [12]

3.2. Intel IoT platforma

Intel IoT platforma je end-to-end referenční model zahrnující produkty z rodiny Intel, které dohromady poskytují základ pro spolehlivou a bezpečnou komunikaci mezi IoT zařízeními a cloudovou platformou, která poskytuje analytické služby. Intel poskytuje IoT řešení pro celou řadu odvětví, jako je automobilový průmysl, obchod, strojírenství, zdravotnictví nebo třeba chytrá města.



Obrázek 3.2.: Intel IoT platforma [14]

3.2.1. Referenční architektura Intel IoT platformy

Intel system architecture specification (SAS) představuje referenční architekturu pro IoT. V současné době existují dvě verze s tím, že každá z nich je použitelná pro jiný případ užití.

- Verze 1.0 pro propojení nepropojeného - tato verze SAS řeší, jak bezpečně propojit IoT brány s legacy zařízeními (zařízení, které postrádají 'inteligenci' nebo připojení k internetu).
- Verze 2.0 pro chytré a propojení věci - tato verze popisuje jak integrovat chytré zařízení, která byla k tomuto účelu navržena. Snaží se řešit problémy integrace, bezpečnosti a kontrolu dat, která proudí mezi chytrými zařízeními a cloudem. [15]

Z výše uvedených definic je zřejmé, že Verze 1.0 slouží pro zachování kompatibility se zařízeními, která tu nyní jsou, zatímco Verze 2.0 je verze pro zařízení nová. Tato specifikace počítá s využitím technologií, jako je kontejnerizace, virtuální stroje nebo softwarově definované sítě.

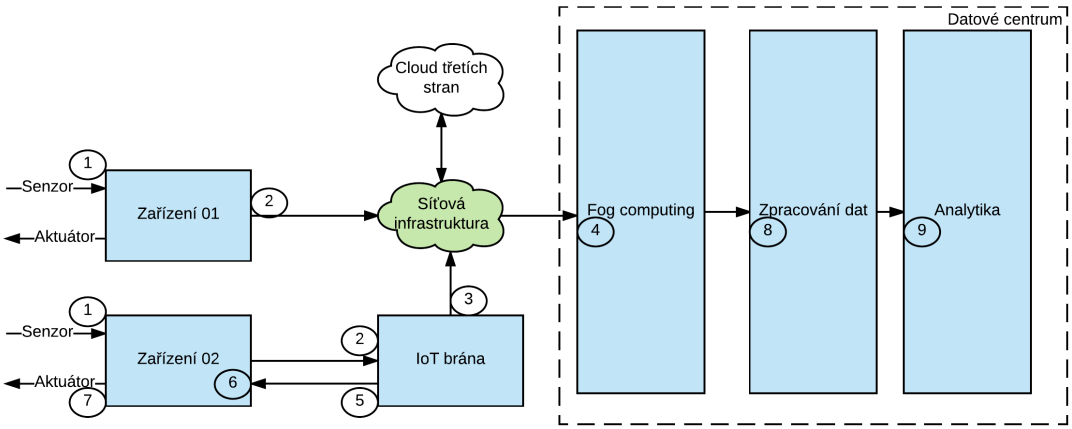
3.2.2. Tok dat

Referenční IoT architektura od Intelu představuje i jakým způsobem by mohla probíhat komunikace od senzoru až po aplikační vrstvu.

1. Senzor přijme analogový signál a přeloží ho na digitální.
2. Brána sesbírá data od senzorů. Některé senzory mohou komunikovat s datovým centrem přímo.
3. Brána připraví, očistí a agreguje data a přepošle je do datového centra.
4. Vrstva Fog Computingu přijme data. Dále komunikuje s bránou nebo se senzorem pomocí HTTPS GET nebo POST. Fog computing je decentralizovaný výpočetní výkon, který je umístěn mezi senzory a cloud z důvodu toho, že

zařízení se senzory mají malý výkon, ale výpočetní výkon datového centra se může nacházet příliš daleko. [11]

5. Brány přepošlou ovládací příkazy koncovým senzorům.
6. Senzory přeloží digitální signál na analogový.
7. Aktuátory zareagují na nový analogový signál.
8. Data jsou zpracována v datovém centru.
9. V datovém centru dochází k analytice nad nasbíranými daty a generuje report. [15]



Obrázek 3.3.: Tok dat[15]

3.3. Cisco IoT system

Cisco IoT system je IoT platforma od firmy Cisco, která cílí na řešení problémů napříč různými odvětvími, jako jsou výrobní průmysl, strojírenství, těžební průmysl, doprava nebo také veřejný sektor.[16] Cisco IoT System je složen ze šesti pilířů:

1. Síťová konektivita je zajištěna velkým množstvím zařízení pro routing, switching a bezdrátové připojení. Kromě fyzických zařízení jsou k dispozici také

virtuální varianty. Síťová konektivita zajišťuje propojení mezi datovým centrem a fog vrstvou nebo mezi datovým centrem a průmyslovou sítí.

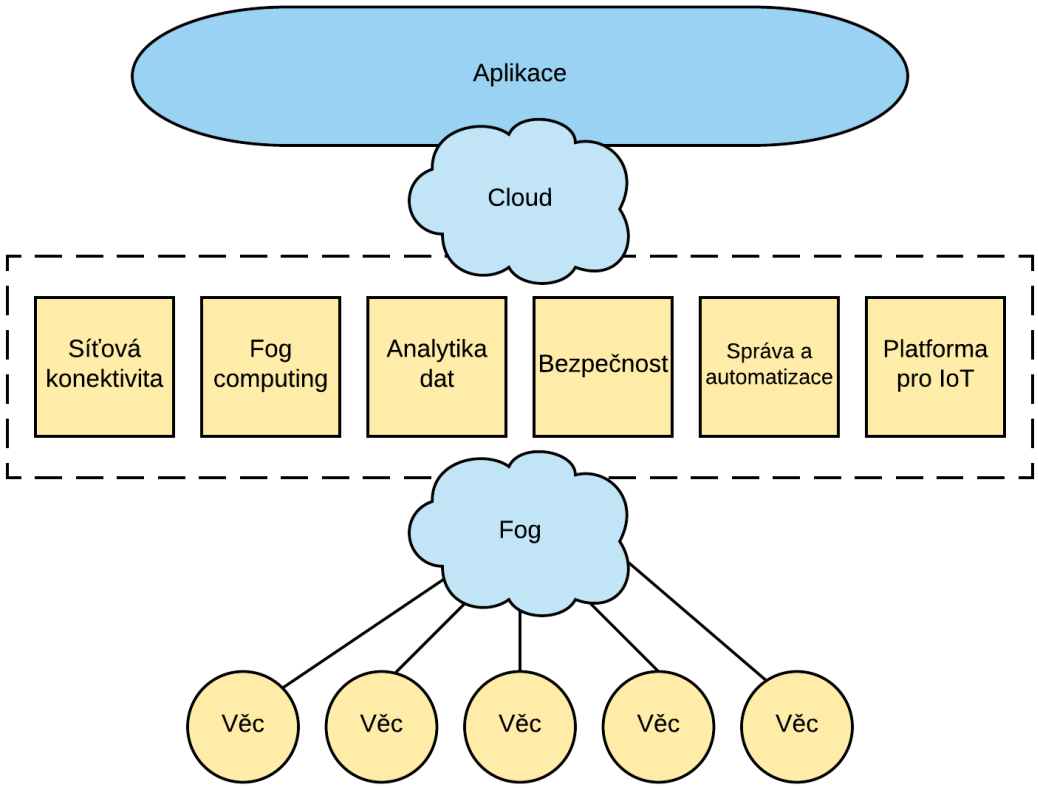
2. Fog computing, jak již bylo zmíněno u Intel IoT platfomy, přibližuje konektivitu s cloudem blíže k 'okraji', což snižuje odezvu a nároky na propustnost mezi datovým centrem a IoT zařízeními.

3. Analytika dat - zařízení z Cisco IoT system portfolia poskytují API pro napojení analytického softwaru.

4. Bezpečnost v sobě zahrnuje opatření pro bezpečnost fyzickou i kybernetickou.

5. Správa a automatizace - Cisco poskytuje rozšiřující možnosti, jak spravovat síťová zařízení. Díky tomu lze vytvářet a vynucovat pravidla automaticky a v celé infrastruktuře.

6. Platforma pro IoT aplikace nabízí celou řadu nástrojů pro podniky pro vytvoření vlastních nástrojů pro IoT platformu.[27]



Obrázek 3.4.: Cisco IoT system[27]

Pro každou z výše zmíněných vrstev nabízí Cisco celou řadu svých zařízení. V případě síťové konektivity se jedná o zařízení od vysokorychlostních switchů až po IP kamery. U fog computingu je k dispozici okolo desítky routerů s aplikačním správcem Cisco Fog Director. Bezpečnost je zajištěna velkým množstvím IP kamer v případě fyzických zařízení, která jsou propojena pomocí Video Surveillance Managera. Kyberbezpečnost zajišťuje například Defense Center nebo Cloud Security. Automatizace je realizována prostřednictvím produktů, jako jsou IoT Field Network Director nebo Fog Director. [17]

3.4. Microsoft Azure IoT platforma

V této sekci se zabývám IoT platformou od firmy Microsoft. Ta představuje end-to-end řešení, které si dává za cíl řešit například scénáře vzdáleného monitoringu,

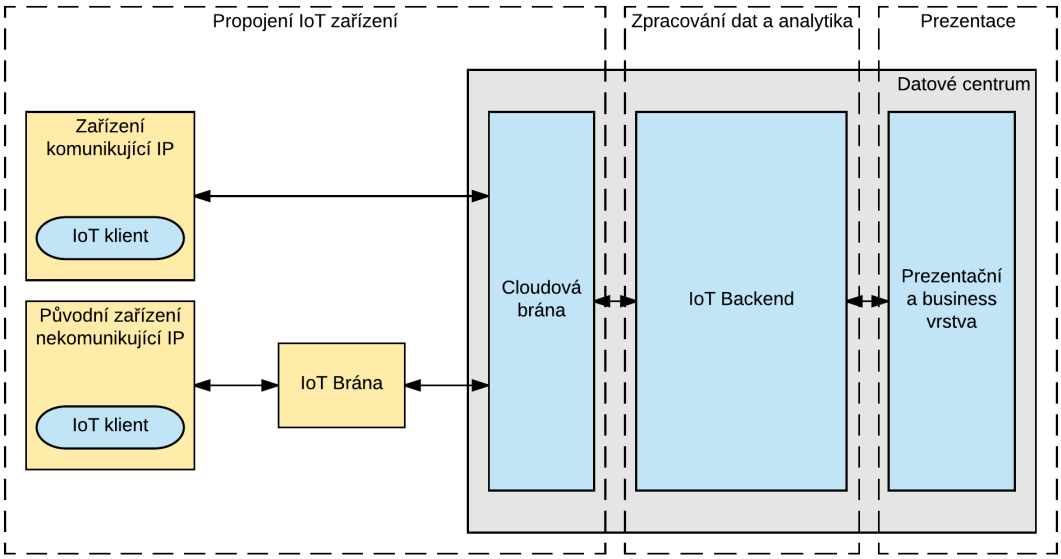
chytrých továren nebo třeba predikci výpadků.

Architektura této IoT platformy je složena ze tří hlavních komponent:

1. Propojení IoT zařízení řeší, jak propojit IoT zařízení s cloudem. Ty mohou být připojeny napřímo, nebo přes IoT bránu. Cloudová brána poté představuje komunikační mezistupeň mezi těmito zařízeními a backendem.

2. Zprocesování, analytika a management dat představuje backend pro IoT zařízení. V této části se odehrává správa zařízení, sběr dat a jejich transformace a analytika.

3. Prezentační a business vrstva je zodpovědná za integraci IoT prostředí do business procesů společnosti. [18]



Obrázek 3.5.: Architektura Mircosoft Azure IoT platformy [18]

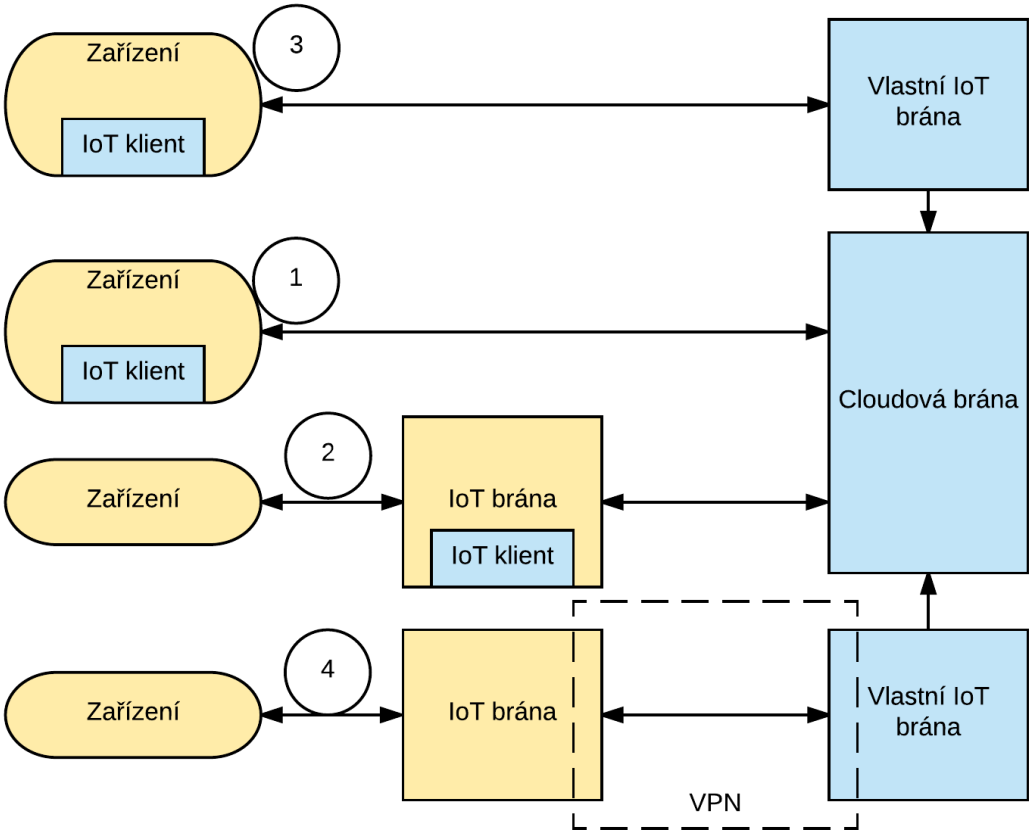
Microsoft Azure IoT platforma tvrdí, že splňuje čtyři základní vlastnosti. První z nich je heterogenita, která znamená, že jejich referenční architektura je schopná pokrýt většinu případů užití, prostředí, zařízení a standardů. Zároveň je platforma schopná pokrýt velké spektrum hardwaru a softwaru. Další vlastností je bezpečnost, která pokrývá zabezpečení napříč všemi oblastmi včetně zabezpečení

zařízení, uživatelská identita, autentizace a autorizace, bezpečnost dat apod. Důležitou vlastností je také vysoká škálovatelnost, kde Microsoft proklamuje, že platforma může škálovat od malých projektů až po miliony zařízení. S heterogenitou také souvisí flexibilita. Ta je důležitá z důvodu roztříštěnosti trhu s IoT zařízeními. [18]

Po tomto úvodu je třeba se podívat více do hloubky na jednotlivé komponenty této platformy.

3.4.1. Propojení IoT zařízení

Jak již bylo zmíněno, tato část architektury řeší připojení IoT zařízení. Zde existuje několik scénářů. Prvním z nich je přímé připojení zařízení k cloudové bráně přes internet. Druhou možností je připojení za pomocí IoT brány. To je možné za použití průmyslových protokolů, jako je CoAP, OPC, Bluetooth, ZigBee atd. Tato možnost je také vhodná v případě potřeby agregace dat na úrovni brány, než jsou poslány do cloudu. Třetím scénářem je připojení za pomocí vlastní cloudové brány. Toto je možné použít v případě, že je nutné provést nějaká specifická zpracování dat nebo transformace protokolu, než jsou data odeslána do cloudové brány. Čtvrtá a poslední možnost je kombinací druhé a třetí, tedy použití IoT brány, která je poté připojena k vlastní cloudové bráně. Tento scénář může být vyžadován v případě potřeby VPN spojení. [19]



Obrázek 3.6.: Možnosti připojení IoT zařízení [19]

Zařízení představuje širokou škálu přístrojů od malých senzorů až po komplexní tovární výrobní linky se stovkami komponent a senzorů. IoT brána je poté přístroj, který se chová jako komunikační bod, který má mimo jiné na starosti správu lokálních zařízení a lokální zprocesování dat. Ta zde mohou být agregována, filtrována a poté odeslána do cloudu. Samotná komunikace je zajištěna pomocí IoT clienta.

Cloudová brána umožňuje vzdálenou komunikaci s IoT zařízeními nebo branami, které se nacházejí v různých lokalitách. S touto branou se dá komunikovat přes internet nebo za pomocí VPN, případně privátní připojení do datových center Azuru. Cloudová brána má dva druhy - Azure IoT Hub a Azure Event Hub. Azure IoT Hub představuje hlavní roli v případě cloudové brány. Ta poskytuje podporu komunikačních protokolů AMQP 1.0, MQTT 3.1.1 a HTTP.

Vlastní IoT brána umožňuje použití jiných protokolů, než jaké podporuje Azure IoT Hub. Tato brána se nazývá Azure IoT protocol gateway a představuje open source framework pro adaptaci protokolů. Zajišťuje také mimo jiné komunikaci mezi ní a IoT Hubem. [19]

3.4.2. Zprocesování, analytika a management dat

Kromě práce s daty je tato komponenta zodpovědná za práci se samotnými zařízeními, což zahrnuje jejich identitu, vytváření a držení stavových informací.

Identita zařízení drží veškeré identifikační informace o IoT zařízení včetně certifikátů, které jsou použity pro autentizaci. Cloudová brána závisí na informacích z tohoto uložiště kvůli identifikaci, autentizaci a správě zařízení. V identitě se také vytvářejí nová zařízení. Ta pak mohou být povolena nebo zakázána. V případě zákazu má zařízení okamžitě odebraný přístup do systému, ale veškeré informace o zařízení zůstávají. Dalším uložištěm je registr zařízení. Na rozdíl od identity obsahuje metadata vztahující se k vlastnostem toho daného zařízení. Registr neobsahuje žádná senzitivní data. Důležitě je zde rovněž rozlišit metadata o zařízení od informací o stavu zařízení. Metadata jsou informace, které se mění zřídka a stavové informace se mohou měnit často. Příkladem zde může být poloha senzoru v pouličním osvětlením, které představuje metadata. Na druhou stranu poloha senzoru uvnitř motorového vozidla je stavová informace. Posledním uložištěm je stavové uložiště. To ukládá stavová data zmíněná v předešlém odstavci. [19]

Klíčovou součástí této komponenty je zpracování dat. Ta projdou nejprve cloudovou bránou do datového zpracovatele. Zde se poté rozhoduje, co se s daty stane dál. Mezi akce, které zde mohou nastat, patří například prosté uložení dat do perzistetního uložiště nebo změna stavu zařízení na základě získaných dat. Dále mohou být aktualizována metadata zařízení nebo může dojít ke spuštění alarmu. Důležitou součástí je také analytika nad proudícími daty. [19]

3.4.3. Prezentační a business vrstva

Poslední z komponent v sobě zahrnuje vrstvy, které umožňují přístup k datům koncovým uživatelům. To obvykle obsahuje webový portál, API s grafickým rozhraní nebo mobilní aplikace. Skrze vizualizaci lze zobrazit získaná data v grafech, výsledky analýz nebo informace o veškerých IoT zařízeních. Zároveň zde jsou vidět notifikace a případné alarmy. [19]

Kromě vizualizace je dostupná business vrstva, která umožňuje integraci s interními IT systémy, jako jsou CRM nebo ERP. [19]

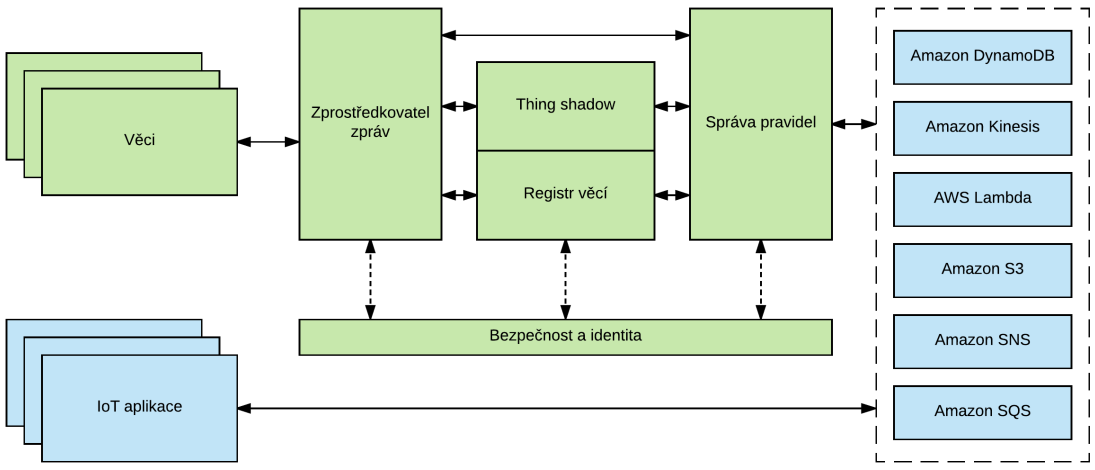
3.5. AWS IoT platforma

AWS (Amazon Web Services) IoT platforma je platformou od firmy Amazon. Tato platforma poskytuje bezpečnou oboustrannou komunikaci mezi zařízeními, jako jsou třeba senzory nebo aktuátory. AWS IoT platforma se skládá z několika různých komponent:

- Brána poskytuje možnost koncovým zařízením komunikovat s AWS cloudem.
- Zprostředkovatel zpráv představuje mechanismus zasílání a přijímaní zpráv mezi zařízeními. Je zde možné použít buď MQTT, MQTT přes WebSocket, nebo HTTP REST API.
- Správa pravidel zajišťuje zpracování zasílaných zpráv a integraci s ostatními AWS službami.
- Bezpečností a autentizační služba má na starost bezpečnost v AWS cloudu. Správa pravidel a zprostředkovatel zpráv využívají tuto službu. Tato služba uchovává citlivá dat.
- Registr věcí organizuje zdroje každé věci. Při registraci věci se přidělují atributy spolu s případným certifikátem a MQTT klientským ID.

- Thing shadow je JSON dokument, ve kterém se ukládá aktuální stavové informace o dané věci.
- Thing shadows service představuje reprezentaci věcí v AWS cloudu. Je možné upravovat informace v Thing shadow manuálně nebo nechat zařízení, aby zasílala jejich aktuální stav.[20]

Tyto komponenty spolu tvoří spolu s ostatními AWS službami ucelenou platformu. Jednoduchý příklad komunikace začíná tím, že věc odešle informaci o svém stavu do MQTT tématu. V tuto chvíli je zpráva zaslána do AWS zprostředkovatele zpráv, který je zodpovědný za odeslání dané zprávy všem odběratelům daného tématu. Tato komunikace je zabezpečena pomocí X.509 certifikátu, který může být vygenerován přímo AWS službou. Tento certifikát musí být registrovaný a aktivovaný v AWS a musí být přítomný na příslušném zařízení. Správa registrace certifikátů je realizována bezpečností a autentizační službou. Každé zařízení by mělo mít záznam v registru věcí, kde je k zařízení přiřazen daný certifikát. S přijatými zprávami je také možné provádět určité operace, jako přidat informace či je aktualizovat. Tyto operace má na starosti správa pravidel. Každá věc má také thing shadow, kde se ukládají a získávají stavové informace. Každá věc má dva záznamy - poslední stav zaslaný věcí a stav dané věci požadovaný aplikací. Aplikace může věci ovládat za pomocí změny tohoto stavu.[20]



Obrázek 3.7.: Architektura AWS IoT platformy [20]

4. Návrh IoT platformy

V předešlé kapitole jsem popsal některé existující IoT platformy od komerčních dodavatelů. Jednu věc, které mají tyto platformy většinou společnou, je to, že jejich implementací se uživatel uzavírá do jednoho ekosystému toho či onoho výrobce. V této kapitole nastíním IoT platformu, která je složená pouze z open source komponent, které jsou veřejně dostupné pro každého. Tato platforma by měla mít všechny kvalitativní znaky, které jsou potřeba pro produkční IoT platformu, jako je vysoká dostupnost, bezpečnost dat, dobrý přístup k datům a spolehlivost. V první sekci představím jeden z případů užití, jak tuto platformu využít v praxi. Dále vysvětlím jednotlivé komponenty této IoT platformy a nakonec spojím všechny tyto komponenty dohromady.

4.1. Případ užití

V teoretické části jsem na komerčních IoT platformách ukázal, že výrobci cílí na mnoho odvětví firemních činností, ale i každodenního života. V této části vymezím případ užití pro navrženou IoT platformu - chytré město. To ovšem neznamená, že tato platforma není vhodná i pro jiná odvětví a chytré město tedy slouží pouze jako reference. V následující kapitole poté aplikuji tuto platformu v menším měřítku jako důkaz schopnosti realizace na dvou chytrých konferencích v texaském Austinu a španělské Barceloně.

V rámci modernizace našeho každodenního prostředí se stále více mluví i o chytřejším městě - tedy o využití IoT pro efektivnější management v řadě oblastí městské infrastruktury a nebo pro rozvoj pro městské samosprávy. Senzorická síť ve městě může tedy sbírat data v reálném čase například v oblastech infrastruk-

tury, životního prostředí nebo veřejných služeb. Tento koncept se rovněž opírá o princip, že sbíraná data by měla být veřejně dostupná kvůli možnosti jejich využití například pro tvorbu vlastních aplikací, které mohou zlepšit životní úroveň obyvatelům těchto chytrých měst.[21]

V oblasti životního prostředí mohou senzory sbírat data o tom, jaký dopad má na dané oblasti například hluk a jaké jsou zdroje tohoto hluku, aby bylo možné tento zdroj eliminovat. Dále také jaká je zde kvalita ovzduší, jaké je světelné znečištění a další. V oblasti dopravy lze sledovat počty aut nebo třeba monitorování obsazenosti parkovišť spolu s navigováním řidiče k nejbližšímu volnému místu.

4.2. Komponenty

V následujících sekcích udělám přehled jednotlivých komponent v této IoT platformě. Každá sekce obsahuje nejdříve základní vymezení té dané komponenty, po které následuje popis toho, jak daná komponenta zapadá do platformy, a na konci této kapitoly je shrnutí toho, jak vypadá platforma jako celek. Představení komponent bude probíhat přibližně chronologicky, jak budou putovat data platformou. Nejdříve je tedy nutné začít od senzoru a o technologiích s nimi spojenými, jako je třeba IQRF nebo MQTT. Dále je zde představena kontejnerová virtualizace, která začíná už na úrovni brány senzorů. Další komponentou je cloudová platforma, jejíž virtuální instance jsou spojeny s kontejnery za pomocí jedné SDN platformy. Tento cloud hostuje databázi Graphite, ze které jsou poté vyčítány metriky, které jsou následně vizualizovány pro koncového uživatele.

4.2.1. Senzory

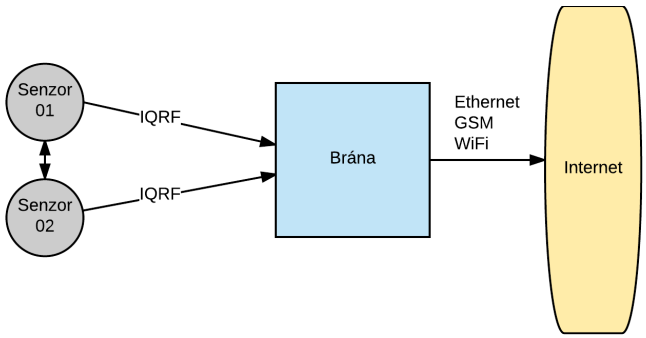
Jak již bylo zmíněno v předešlých kapitolách, senzory jsou fyzická zařízení, která mají za úkol získávat určitou fyzikální metriku ze svého prostředí.

4.2.1.1. IQRF

Senzory mohou používat různé technologie pro zasílání získaných metrik. V případě návrhu této platformy se jedná o technologii IQRF. IQRF je platforma, která pracuje při nízkých energetických nákladech, nízké rychlosti a je stavěná na zasílání malého objemu dat s dosahem až stovky metrů. Tuto technologii je možné použít s jakýmkoli jiným elektronickým zařízením a lze ji použít v průmyslu nebo třeba v případě chytrých měst. IQRF funguje na frekvencích 868 MHz, 916 MHz a 433 MHz a využívá paketově orientovanou komunikaci. [22]

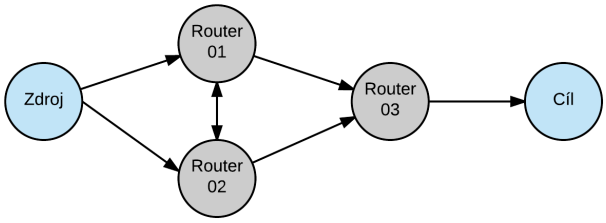
Základní komponentou IQRF jsou transceivery. Tato zařízení o velikosti SIM karty obsahují mikrokontroler s vlastním operačním systémem. Díky nim je realizována bezdrátová komunikace, která funguje oběma směry a umí tedy přijímat i odesílat. Kromě použití transceiverů v samotných senzorech, jsou použity i v IQRF branách a routerech. [23]

Brána představuje rozhraní mezi IQRF sítí a zbytkem světa. Brána, obsahující transciever, existuje v mnoha variantách - USB disk, Ethernet, GSM, WiFi.[23] V případě této platformy je použit USB disk, který je připojen k mikropočítači Raspberry Pi.



Obrázek 4.1.: IQRF brána

Další komponentou je router, který má na starosti zvýšení dosahu jednotlivých transcieverů a zároveň také přidání redundantních cest. Ve své podstatě je to opakovač, který přijme digitální signál a přepošle ho dál. [23]



Obrázek 4.2.: IQRF router

Velkou výhodou IQRF je, že kromě peer-to-peer komunikace podporuje full-mesh síť, tedy komunikaci každý s každým, čímž se odstraňuje jedinečný bod selhání.

V této IoT platformě budou využity jednak transcievery, které jsou zabudovány v senzorech, ale také routery, které budou prodlužovat signál až k branám.

4.2.1.2. MQTT

MQTT je M2M (machine-to-machine) komunikační protokol, který má široké využití ve světě IoT. Tento protokol se vyznačuje tím, že má nízké nároky na transport zpráv. Od roku 2013 probíhá proces standardizace tohoto protokolu organizací OASIS. MQTT využívá porty 1883 a 8883 pro MQTT over SSL.[24] V MQTT odesílatel odesílá zprávy do témat a příjemce se přihlašuje do těchto témat s tím, že jedno téma může mít několik příjemců. Témata představují hierarchický strom, kde jsou jednotlivé úrovně znázorněny pomocí '/'. Příkladem tématu může být 'sensory/NázevSenzoru/teplota'.[25]

4.2.1.3. Brána

Brána je v této platformě realizována pomocí malého počítače Raspberry Pi, do kterého je zapojena IQRF brána. Raspberry Pi, v současné době už třetí generace, obsahuje 64 bitový čtyřjádrový procesor s architekturou ARM. Kromě toho obsahuje 1 GB operační paměti, LAN rozhraní a 4 USB porty, do kterých lze zapojit například Wi-Fi modul.[26] Tento mikropočítač je vhodný pro tuto platformu z důvodu velmi malých rozměrů, silnému výkonu a nízké energetické náročnosti.

4.2.2. Kontejnerizace

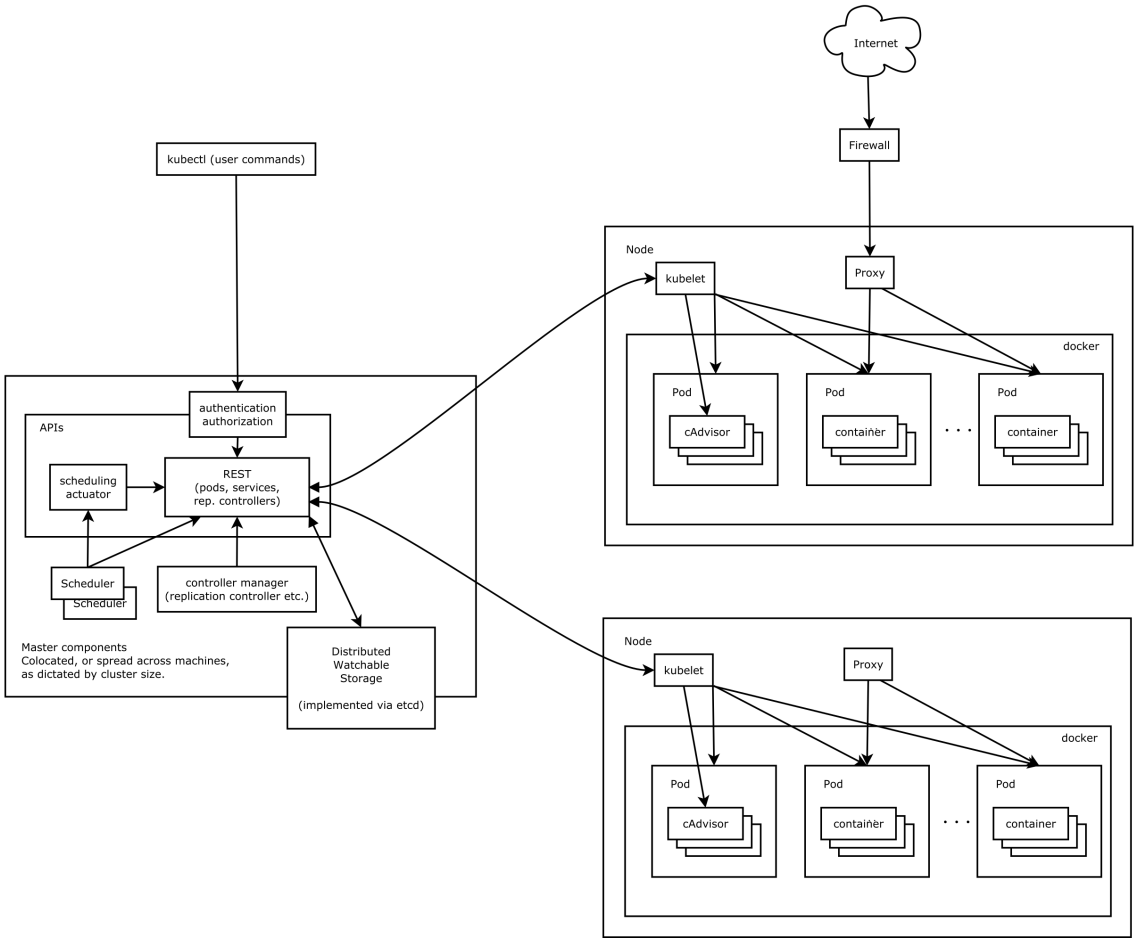
V této podsekci vysvětlím důležitost kontejnerové virtualizace pro tuto IoT platformu a také představím orchestrátor kontejnerové virtualizace, který je v ní použit. Na rozdíl od klasické virtualizace, kde jeden či více virtuálních strojů je virtualizován na fyzickém hardwaru prostřednictvím virtualizační vrstvy, kontejnerová virtualizace je realizována přímo v uživatelském prostoru na jádře operačního systému.[28] Díky tomu je kontejnerová virtualizace méně náročná na spotřebované systémové prostředky, což je u IoT zařízení jedna z klíčových vlastností.

Orchestrace kontejnerové virtualizace je zde realizována pomocí projektu Kubernetes. Kubernetes je velký open source projekt vedený firmou Google. Kubernetes se stará o to, aby dané kontejnery běžely na správných hypervizorech, kontroluje životnost kontejnerů a nahrazuje ty mrtvé, umožňuje škálovat v případě vytížení aplikace v kontejneru apod.

Než vysvětlím použití Kubernetes v této IoT platformě, je důležité nastínit základní principy Kubernetes. Kubernetes prostředí se nazývá cluster, což je množina hostitelských systému, které Kubernetes používá pro běh aplikačních kontejnerů. Cluster je složen z nodů. Nod, dříve nazývaný minion, může být fyzický nebo virtuální a jeho účelem je, aby na něm byly spouštěny pody. Na každém z těchto nodů běží několik služeb, které komunikují s kontrolní částí Kubernetes clusteru - kubelet a kube-proxy. Kubelet je služba, která má na starost samotnou správu kontejnerizace v rámci nodu. Kromě samotné správy orchestrace také pracuje s datovými svazky, reportuje stav o podech, spouští životností sondy a získává informace jako jsou Kubernetes secrets z API serveru. Kube-proxy je poté komponentou, která zajišťuje fungování Kubernetes services za pomocí vytváření iptablových pravidel.

Kubernetes master je kontrolní rolí Kubernetes clusteru. Tato role je složena ze služeb API server, scheduler a controller manager. Master komunikuje a ovládá nody. Kubernetes API server poskytuje REST API různým službám, které z něj mohou podle svých práv získávat informace nebo do něj zapisovat. Scheduler má zase na starosti plánování toho, jaké pody jsou pouštěné na jaké nody. K tomu

scheduler využívá informace z vyžadovaných zdrojů v definicích podů, politiky, značek atd. Controller manager v sobě obsahuje několik druhů managerů - kontroler replik, pod kontroler, servisní kontroler a endpoint kontroler.



Obrázek 4.3.: Architektura Kubernetes, převzato z [29]

Nejmenší jednotkou spravovaných kontejnerů je pod. Pod obsahuje jeden nebo více kontejnerů, které jsou vždy puštěny spolu na stejném nodu. To je z důvodu toho, že kontejnery spolu sdílejí stejný síťový prostor - mají stejnou IP adresu, mohou spolu komunikovat pres lokální adresu a sdílejí i portový rozsah. Kromě síťového prostoru také mohou sdílet i lokální uložiště.

Label představuje key-value pár, který umožňuje označkovat různé objekty v Kubernetes, často pody. Label je velmi důležitá komponenta, protože na základě nich se například určuje, jaký pod patří k jaké Kubernetes service. Důležité je rovněž možnost značkovat i nody. Na základě značky, kterou má daná noda, může

být rozhodováno, kde daný kontejner poběží.

Kubernetes service vystavuje přístup k podům patřícím k jedné aplikaci jiným aplikačním podům. Každá aplikace může mít jeden a více podů. V případě, že jich je více, je třeba mít mechanismus, jak k těmto podům přistupovat a jak mezi ně rozkládat zátěž. Service představuje virtuální IP, za kterým jsou schovány všechny pody, které patří k dané službě. Toto virtuální IP ve skutečnosti existuje pouze v API a přístup k tomu IP je realizován pomocí kube-proxy a vhodných iptablových pravidel. Jedna aplikace tedy nekomunikuje přímo s danými členy druhé aplikace, protože nikdy nemůže znát všechny z důvodu toho, že druhá aplikace může škálovat v reálném čase a tím zvyšovat počet svých členů. Proto komunikuje přes servisní IP, jejíž Kubernetes service automaticky přidává členy tak, jak aplikace škáluje.

Volume je uložiště pro kontejnery. V základním nastavení je uložiště kontejneru pouze v paměti a v případě úmrtí kontejneru nejsou žádná data zachována. Zde vysvětlím pouze dva nejzákladnější druhy, protože jiné nejsou v této IoT platformě použity. Prvním z nich je emptyDir. Toto uložiště vytvoří adresář na hostitelském systému a ten je namapován do cesty v kontejnerovém systému. V případě úmrtí kontejneru a jeho znovuoživení kubeletem, kontejner nastartuje s daty, která jsou uložena v emptyDir. Tato data jsou smazána v případě smazání celého podu za pomocí Kubernetes. Druhým typem uložiště je hostPath. Tento typ mapuje cestu z hostitelského systému do cesty v systému kontejneru.

DaemonSet je speciální definice podu. Tento typ zajišťuje, aby na každém nodu běžel právě jeden výskyt dané aplikace.

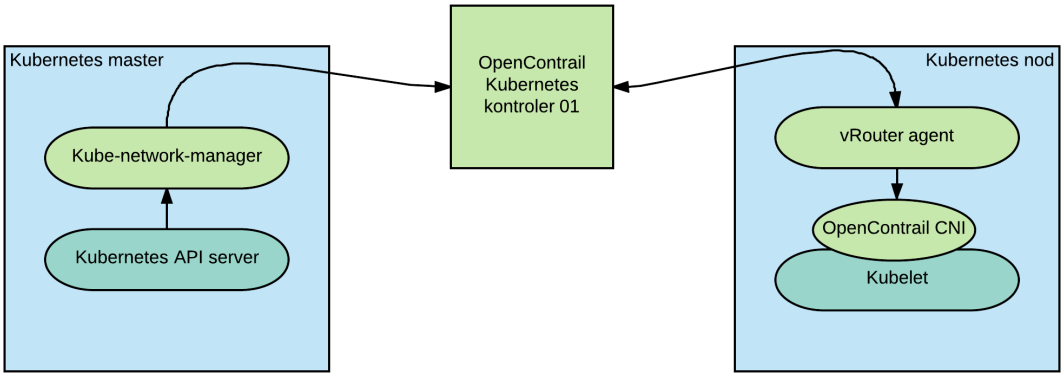
Deployment představuje pokročilou definici podu. Deployment umožňuje škálování, rolling-update a rollback podů. Rolling-update představuje bezpečný upgrade aplikace, který spočívá v postupném vypínání staré a zapínání nové revize aplikace. V případě výskytu nějakého problému je možné se vrátit ke staré revizi.

Etcd je key-value uložiště, které udržuje všechna data v Kubernetes clusteru.

Kromě výše zmíněných objektů jich v Kubernetes existuje celá další řada. Tyto byly vysvětleny z toho důvodu, že jsou přímo použity v této IoT platformě.

4.2.2.1. OpenContrail a Kubernetes

OpenContrail je v této platformě použit jako síťový plugin pro Kubernetes. Funkcionalita OpenContrail je podrobně vysvětlena v sekci Softwarově definovaná síť. Integrace spočívá ve dvou komponentách. První z nich je kube-network-manager. Tato komponenta běží spolu s Kubernetes API serverem a přímo s ním interaguje. Kube-network-manager dále komunikuje s OpenContrail API. V případě vytvoření podu vykoná Kubernetes API příslušnou akci, na základě které kube-network-manager odešle zprávu do OpenContrail API o vytvoření sítě/portu pro daný pod. Druhou komponentou je OpenContrail CNI plugin. Ten je zodpovědný za připojení síťového rozhraní, které vytvořil vRouter agent, do kontejneru.



Obrázek 4.4.: Kubernetes a OpenContrail

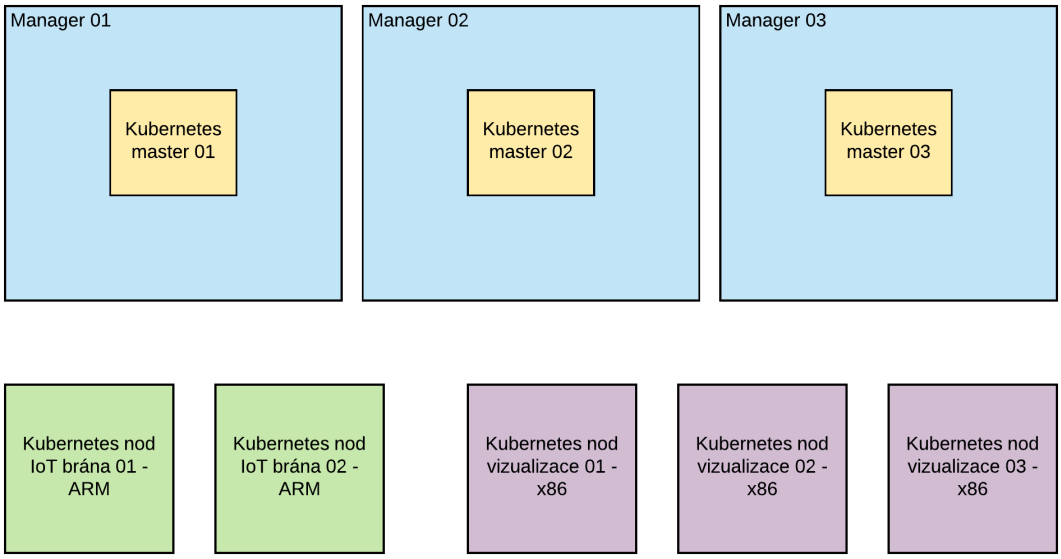
4.2.2.2. Kubernetes v IoT platformě

Kubernetes zaujímá své místo v platformě především kvůli orchestraci kontejnerů na IQRF branách. Tyto kontejnery je třeba spravovat z centrálního místa. Důležitá je také vlastnost sond, které zajišťují, že je kontejner plně funkční a v případě problémů může být resetován. Samozřejmostí je také automatické oživení kontejneru v případě jeho smrti. Druhé využití Kubernetes je pro vizualizační Django aplikaci.

Tato platforma využívá labely pro pouštění kontejnerů na daných hardwarových platformách. IoT brány jsou označkovány jako arch=arm a nody pro vizua-

lizační kontejnery jsou arch=x86. Tyto značky jsou poté použity v definici jednotlivých aplikací a jsou rozeznány schedulerem při vytváření podů.

Pro produkční platformu je velmi důležitá vysoká dostupnost. Proto Kubernetes master běží ve třech replikách. Spolu s masterem běží také etcd. Způsob balancování je popsán v sekci Cloudová platforma a tento koncept je použit i u Kubernetes. Samotné Kubernetes poté pro komponenty scheduler a controller-manager drží stav active/backup skrze etcd. IoT brány jsou nasazeny v páru také z důvodu vysoké dostupnosti. Nody pro vizualizaci je potřeba mít minimálně dva. Další lze přidávat podle potřeby škálovat.



Obrázek 4.5.: Architektura Kubernetes pro IoT platformu

4.2.3. Cloudová platforma

Pro IoT platformu je nutné mít místo, kde budou všechna data uložena, zpracována, nabídnuta k využití a analyzována. K tomuto účelu je vhodné využít například cloudovou platformu. V této sekci představím cloudovou platformu, která je použita pro tuto IoT platformu.

Pro výpočetní výkon a uložení získaných metrik slouží v této IoT platformě OpenStack. OpenStack se dá definovat jako framework pro správu, definování a

utilizaci cloudových zdrojů. V podstatě se jedná open source software pro budování privátních nebo veřejných cloudů. OpenStack se vyznačuje svojí modulární architekturou složenou z velkého množství samostatných komponent, které spolu komunikují pouze přes REST API.[30] Výchozí chování těchto komponent navíc není nutné využívat a je možné použít vlastní backend s tím, že ta daná služba slouží pouze jako API pro ostatní služby.

Jednotlivé moduly OpenStacku zajišťují různé infrastrukturní či jiné služby. Mezi nejznámější patří Nova, Keystone, Glance, Neutron, Heat, Horizon, Ceilometer a Cinder.

- Keystone je OpenStacková služba zajišťující autentizaci a autorizaci. Díky této službě je možné vytvářet uživatele s různými právy a rolemi, projekty, ve kterých uživatelé pracují s virtuálními zdroji, nebo domény, které shlukují několik projektů. Keystone je jedním z prvků zajišťující multi-tenantost, tedy oddělení zdrojů. Toto je jedním ze základních prvků cloudových prostředí, kde je nutné, aby akce jednoho subjektu neovlivnily jiný subjekt.

- Nova zajišťuje orchestraci konkrétní virtualizační platformy, mezi které se řadí například Xen, KVM nebo VMware. Dále také poskytuje plánování, kam vytvořit danou virtuální instanci podle utilizace dané infrastruktury a požadavků dané instance. Kromě toho pak také poskytuje například novnc konzoli pro přístup do instance nebo metadata, která slouží jako informace o instanci pro instanci. [31]

- Glance umožňuje ukládaní, získávání a využívání imagů. Image přestavuje zdroj pro virtuální instance, tedy disk, ze kterého je spuštěna konkrétní virtuální instance.

- Neutron má na starosti síťové služby v rámci plaftormy OpenStack. V rámci Neutronu mluvíme o softwarově definovaných sítích, které budou blíže přestaveny v další sekci. Neutron přináší další část multi-tenantnosti tím, že odděluje provoz různých virtuálních sítí. Na tento modul existuje velké množství pluginů třetích stran, které využívají Neutron pouze jako API. Kromě

pluginu OpenContrail, který je použit v této IoT platformě, existují třeba Nuage od Nokie, Cisco APIC, VMware NSX, Midonet, Calico či Plumgrid.

- Heat je orchestrační služba. Slouží k popsání celé virtuální infrastruktury do šablony, na základě které se potom celá infrastruktura spouští. Definují se zde počty a druhy instancí, rozsahy a počty virtuálních sítí, virtuální routery, blokové nebo objektové uložiště a další. Díky Heatu je možné automatizovat celý životní cyklus virtuální infrastruktury.

- Horizon představuje webové rozhraní pro jak koncového uživatele, tak administrátora. Akce povolené v Horizonu se odvíjejí od rolí, které má daný uživatel v Keystone. Přes toto webové rozhraní je možné vytvářet, spravovat či mazat veškeré virtuální zdroje, které daný cloud nabízí.

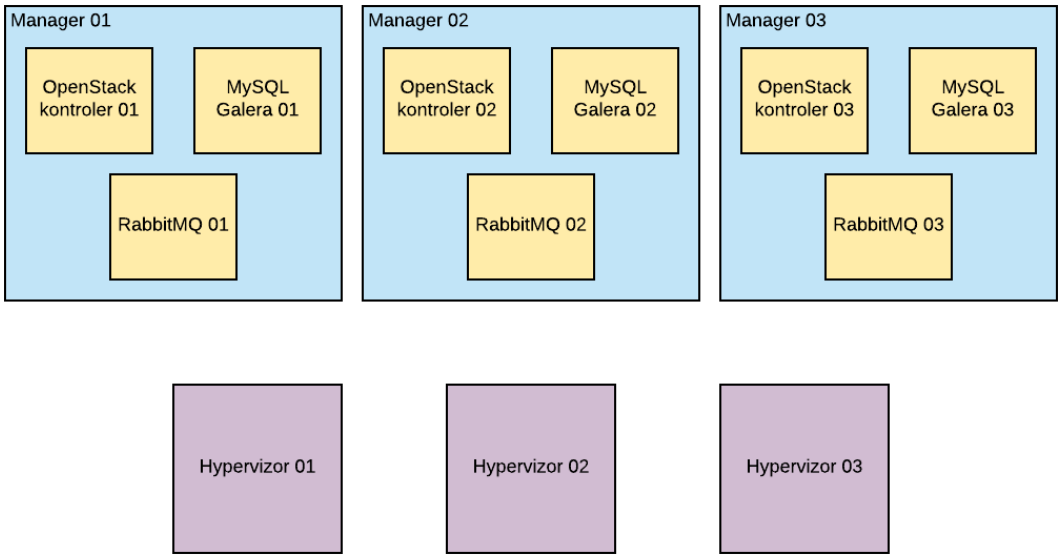
- Ceilometer je služba, která slouží ke sběru metrických dat z celé OpenStack platformy. Tyto metriky se dají použít například pro automatické škálování virtuálních instancí nebo pro vypočítání kolik zdrojů spotřebovává daný projekt. Ceilometer má několik různých druhů databází, jako jsou Mongo DB, Hbase, nebo SQL server, do kterých lze metrická data ukládat.[31]

- Cinder poskytuje blokové uložiště z nějakého uložiště třetích stran, jako jsou disková pole od tradičních výrobců. Příkladem může být třeba diskové pole od EMC, IBM Storwize, HP 3PAR nebo SolidFire.

Kromě komponent OpenStacku jsou však nutné i další podpůrné služby, bez kterých by OpenStack nefungoval. Mezi tyto služby patří například cache, message bus nebo databáze. Pro zasílání zpráv mezi komponentami v rámci jednoho modulu slouží message bus, v případě cloudu pro tuto IoT platformu je to RabbitMQ. Databáze pak slouží pro ukládání veškerých dat ze všech modulů. Databáze je realizována pomocí MySQL Galera clusteru.

Pro IoT platformu je potřeba mít alespoň 3 fyzické servery (manager) pro kontrolní a podpůrné služby OpenStacku a dále 3 fyzické servery (hypervizory) pro virtuální instance. Veškeré komponenty jsou použity ve škále tří virtuálních strojů,

aby byla zajištěna vysoká dostupnost v případě výpadku jednoho ze strojů. Přechod služeb v případě výpadku má služba Keepalived, což je Linuxová implementace protokolu VRRP. Jeden z virtuálních strojů má tedy kromě své vlastní IP adresy také IP adresu virtuální, která se zmigruje v případě výpadku. Kromě toho na této adrese poslouchá démon HAProxy, který má na starosti rozložení zátěže mezi všechny virtuální stroje. Z důvodů lepšího oddělení zdrojů jsou odděleny OpenStackové služby od databáze a od message busu.



Obrázek 4.6.: Architektura OpenStacku pro IoT platformu

4.2.4. Softwarově definovaná síť

K propojení různých komponent je zapotřebí počítačové sítě. Problém tradiční počítačové sítě je v tom, že je příliš rigidní, špatně automatizovaná a zcela v režii síťového správce. K překonání těchto problémů má sloužit Softwarově definovaná síť (SDN). V této sekci je nastíněno, co to vlastně znamená. Dále je představeno SDN řešení, které je použito v této IoT platformě.

Softwarově definovaná síť představuje nový přístup v síťovém světě, který by se dal definovat jako framework, ve kterém jsou sítě vytvářené a konfigurované dynamicky s jistou úrovní abstrakce. Dochází zde k rozdělení na control plane a

data plane. Control plane představuje centralizovaný nebo distribuovaný kontroler, který ovládá samotný průchod síťového provozu. Ten je označován jako data plane.[32]

OpenContrail je jedním z nejvyspělejších SDN řešení. Ačkoli je hlavně vyvíjené společností Juniper, tak se jedná o čistě open source řešení, které je nejvíce rozšířeno jako síťový plugin Neutronu pro cloudovou platformu OpenStack. Kromě toho však má i integraci s kontejnerovým řešením Kubernetes nebo Mesos. OpenContrail je složen z několika rolí, které by se daly rozdělit do kategorií data plane (posílání síťového provozu virtuální sítí), control plane (ovládání) a analytika.

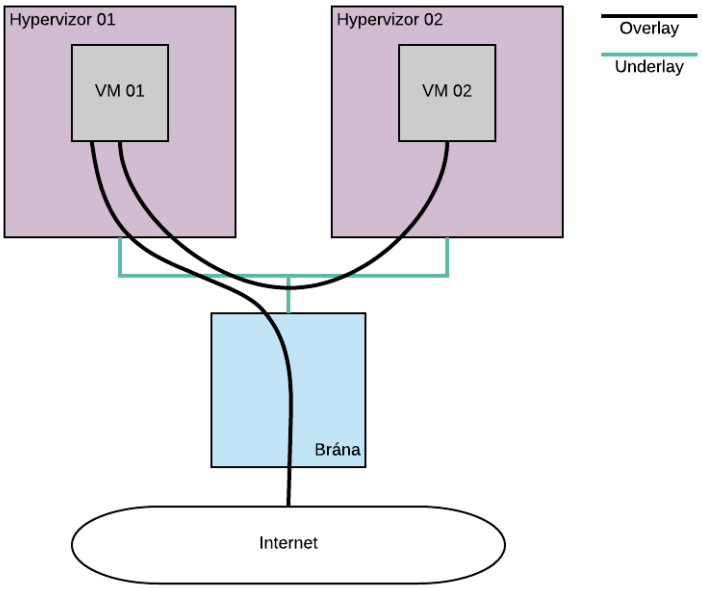
4.2.4.1. Data plane

Data plane zajišťují dva druhy rolí - compute (hypervisor) a brána.

- Compute je server, který zajišťuje virtualizaci, a tedy hostuje virtuální instance. Spojení virtuálních instancí mezi sebou a se zbytkem světa zajišťuje vRouter. vRouter je základní komponenta, která je zodpovědná za posílání síťového provozu v OpenContrailu.
- Brána představuje externí subjekt, který zajišťuje přechod mezi virtuální a fyzickou sítí. Typickým příkladem může být instance, která potřebuje komunikovat směrem do internetu.[33]

V OpenContrailu je data plane realizován pomocí overlay sítě. Overlay síť představuje sít, která je tunelovaná přes fyzickou síť (underlay). Tato technologie předpokládá zapouzdření celé síťové hlavičky i s daty virtuální instance do vnější síťové hlavičky fyzického serveru (computu). Díky tomu se provoz mezi dvěma instancemi tváří v tradiční síťové infrastruktuře jako komunikace mezi dvěma fyzickými servery. Virtuální sítě potom mohou být jakékoli, rozsahy se mohou překrývat a instance v různých sítích jsou kompletně izolované. Toto chování zajišťují speciální síťové protokoly - VXLAN, MPLS over GRE nebo MPLS over UDP. Všechny tři tyto protokoly předpokládají tunelování vnitří sítě skrze vnější síť za použití jedinečného identifikátoru, jenž slouží k rozpoznání, která vnitřní síť je

která. VXLAN (Virtual eXtensible Local Area Network) k tomuto účelu využívá VNI - VXLAN Network Identifier je 24-bitové ID, které umožňuje až 16 milionů VXLANových segmentů koexistovat v rámci jedné administrativní domény.[35] Na druhou stranu MPLS over GRE nebo MPLS over UDP tunelují MPLS návěští za pomocí UDP nebo GRE. Každý z těchto identifikátorů jednoznačně určuje, do jaké virtuální sítě daný provoz patří, a to i když se rozsahy virtuálních sítí překrývají.



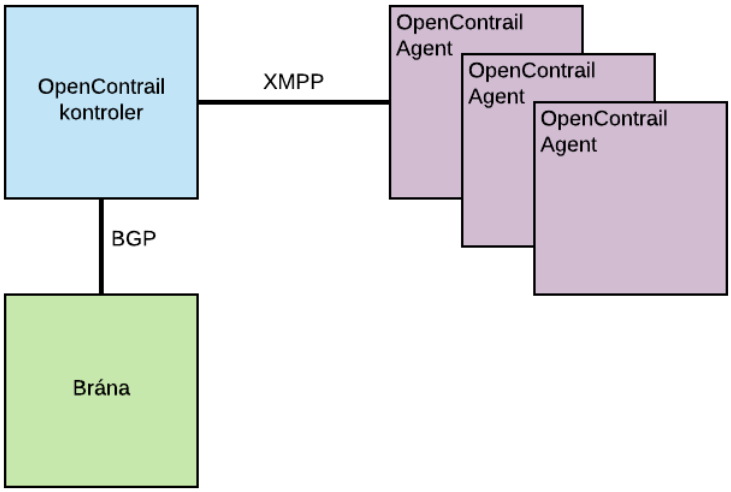
Obrázek 4.7.: Znázornění data plane provozu v OpenContrail SDN

4.2.4.2. Control plane

Veškeré řízení SDN je vykonáváno různými komponentami, které lze souhrnně nazývat Control plane. Mezi tyto komponenty patří role Kontrolní, Konfigurační, Agentová, Databázová a mimo jiné také Brána. OpenContrail má centralizovaný kontroler, který komunikuje se svými agenty ve smyslu, že agent se učí směrovací pravidla, na základě kterých poté může vykonávat samotné směrování provozu (Data plane). Tento kontroler obsahuje tři ze zmíněných rolí - Kontrolní, Konfigurační a Databázovou.

Konfigurační role má na starost držení veškeré konfigurace, přijímání dotazů

a příkazů spolu s předáváním informací Kontrolní roli. Kontrolní role je potom role, která zajišťuje předání informací komponentám Control planu, které přímo orchestrují Data plane. Tato role tedy komunikuje s Agenty a s Bránou. Agent komunikuje s Kontrolní rolí za pomocí protokolu XMPP a podle pravidel získaných z této komunikace poté programuje směrovací pravidla přímo do vRouteru. Brána představuje tradiční síťový router (buď fyzický, nebo virtuální), který komunikuje pomocí tradičních protokolů, jako je BGP, s Kontrolní rolí. Při této komunikaci se využívá MP-BGP (Multiprotocol Extensions for BGP). Toto rozšíření pro BGP umožňuje fungování technologií, jako jsou L3VPN nebo EVPN[36], které jsou využívány právě i v OpenContrail. Z předešlých vět tedy vychází, že OpenContrail je postaven na standardních síťových technologiích, které jsou známy od poskytovatelů služeb (Service providerů). Pro zjednodušení vysvětlím princip fungování na L3VPN. Podstata směrování v OpenContrailu je založená na oddělených routovacích instancích - VRF (Virtual Routing and Forwarding). Každá virtuální sít, která je vytvořena OpenContrailem přestavuje jednu VRF instanci. Každá virtuální instance, která je vytvořena, znamená záznam hostitelské cesty v této směrovací tabulce. Instance v různých virtuálních sítích spolu nemůžou komunikovat jednoduše proto, že směrovací tabulka nezná údaje o přístupu do jiné virtuální sítě. V případě, že nějaká síť potřebuje komunikovat ven ze své směrovací tabulky, potřebuje mechanismus vzájemné propagace s Bránou. Brána má předem vytvořené VRF instance a každá tato instance má přiřazen speciální indentifikátor - route target (RT), který určuje, jaký prefix přísluší jaké routovací instanci. Tento jedinečný identifikátor náležící VRF v Bráně musí být stejný jako identifikátor přiřazený virtuální síti v OpenContrail. V případě vytvoření virtuální instance Kontrol role odešle BGP UPDATE zprávu se směrovací informací o dané instanci Bráně, která přečte tuto UPDATE zprávu, na základě RT identifikuje příslušnou routovací instanci, kterou naučí příslušnou směrovací informaci. Opačným směrem Brána zasílá informace o vnějším světě, tedy například výchozí cestu, na základě které se virtuální instance dostanou ven z cloudového prostředí.

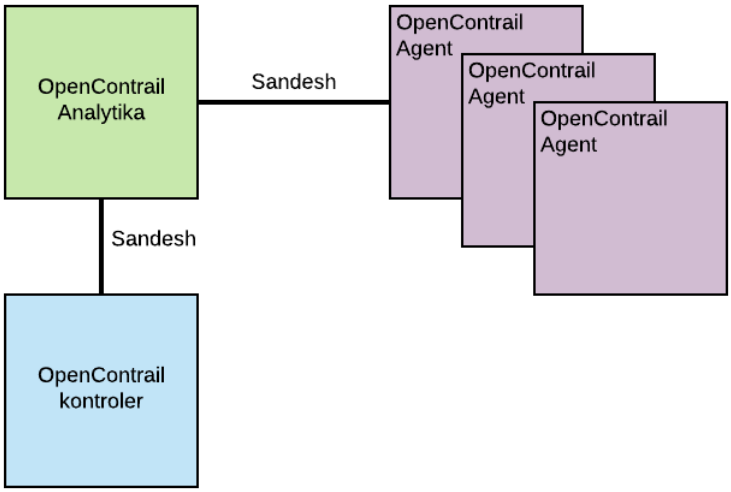


Obrázek 4.8.: Znázornění control plane provozu v OpenContrail SDN

Veškeré konfigurační informace poté ukládá Konfigurační role do databáze. V případě OpenContrail je použita NoSQL databáze Cassandra.

4.2.4.3. Analytika

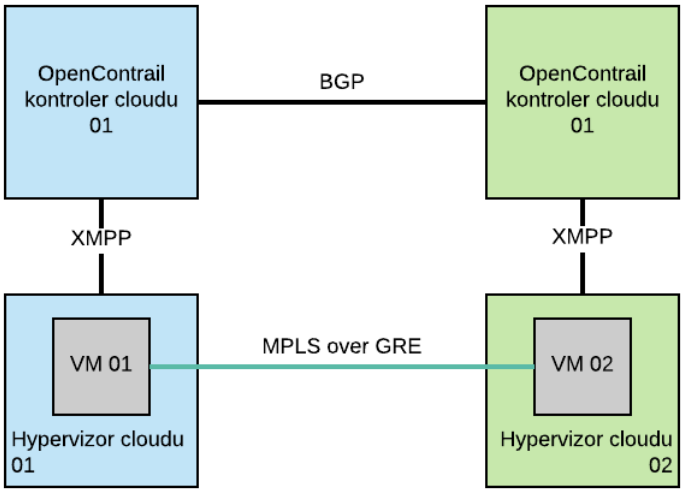
OpenContrail poskytuje kromě Control plane a Data plane také Analytickou část. Analytika sbírá status o všech OpenContrail komponentách a hlásí případné odchylky od normálu, či spustí alarm, když nějaká komponenta nefunguje tak, jak má. Kromě toho je možné sbírat údaje o veškerém síťovém provozu virtuálních instancí. Analytika je tedy napojená jednak na OpenContrail kontroler, tak i na Agentové role. Toto spojení je založené na protokolu Sandesh, který vychází z XML a je designován na zvládání velkého množství dat.



Obrázek 4.9.: Znázornění analytického provozu v OpenContrail SDN

4.2.4.4. Federace

V předchozí části jsem představil OpenContrail jako SDN řešení pro jedno cloudové prostředí. V této části popíšu, jak se dají propojit dvě a více prostředí s tímto SDN řešení. V sekci Control plane je popsáno, že komunikace mezi kontrolerem a Bránou je realizována za pomocí protokolu BGP. Na podobném principu funguje i spojení více OpenContrailů. Může se tedy jednat o dva OpenStack cloudy nebo o OpenStack cloud a Kubernetes platforma nebo dvě Kubernetes platformy a podobně. V případě propojení je třeba pouze navázat BGP spojení mezi OpenContrail Kontrol rolemi jednotlivých platforem. Výměna routovacích informací poté funguje stejně jako mezi Bránou a Kontrol rolí OpenContrailu. Výsledkem tohoto spojení je, že mohou instance dvou cloudů, nebo instance a kontejner komunikovat napřímo skrze overlay tunel a tím získat jednotnou SDN platformu pro virtuální stroje a kontejnery.



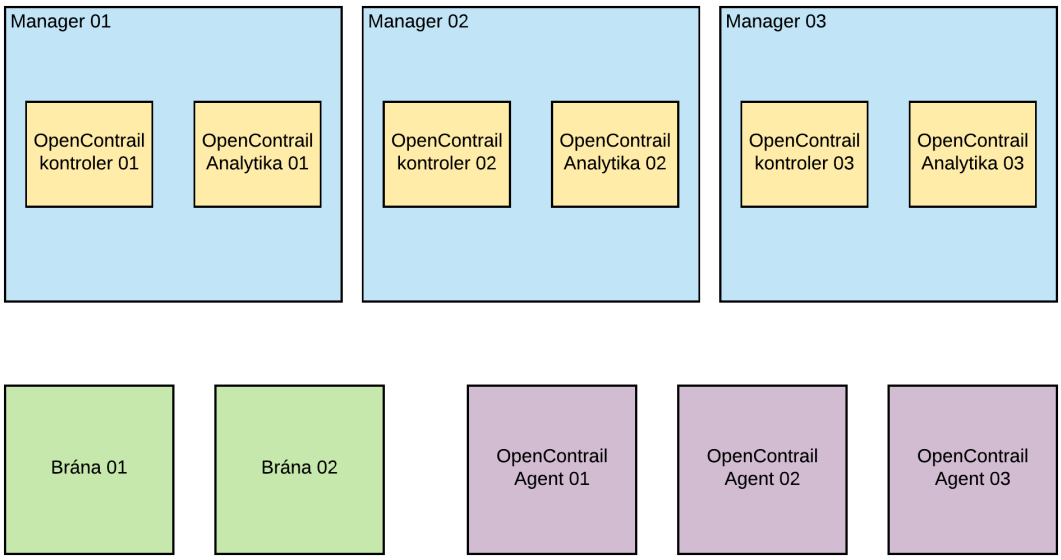
Obrázek 4.10.: Znázornění federace v OpenContrail SDN

4.2.4.5. OpenContrail v IoT platformě

OpenContrail hraje v této IoT platformě klíčovou roli. Toto SDN řešení je jednak plugin pro cloudovou platformu OpenStack, ale také jako plugin pro kontejnerovou platformu Kubernetes. Tyto dvě platformy, OpenStack a Kubernetes, budou vzájemně propojeny pomocí federace, která je popsána v předešlé sekci.

OpenContrail jako každá další komponenta v této platformě je nasazen ve vysoké dostupnosti. Z toho důvodu je každá komponenta v škále tří. Kromě toho je také potřeba oddělit jednotlivé role od sebe alespoň na úrovni virtuálních strojů, a tedy OpenContrail kontroler bude v jiném virtuálních stroji než OpenContrail Analytika. Hlavní motivací pro toto oddělení je fakt, že Analytická role není nezbytnou součástí pro funkčnost klíčové funkce SDN - posílání síťového provozu, a navíc je náchylná k přetížení z důvodu velkého množství sbíraných informací. Je tedy komponentou, která může potencionálně ohrozit celou platformu. Další úrovní oddělení je použití separovaných databázových clusterů. Princip vysoké dostupnosti je zde zajištěn stejným způsobem jako u OpenStack cluster, tedy za použití Keepalived a HAProxy. Dalším prvkem vysoké dostupnosti je použití alespoň dvou Bran buď v active/active, nebo active/backup módu. OpenContrail

Agent běží na každém OpenStack hypervisoru i Kubernetes minionu.



Obrázek 4.11.: OpenContrail SDN v IoT platformě

4.2.5. Databáze

Pro ukládání metrik z IoT senzorů se hodí databáze pro ukládání časových řad (time series database). V případě této platformy je použita databáze Graphite. Tato databáze je složena celkem ze tří komponent:

- Carbon - démon, který naslouchá datům časových řad. Tento démon cachuje tyto metriky v paměti a poté je zapisuje na disk ve formátu Whisper databáze.
- Whisper - specifikace databázového rozložení, která obsahuje i třeba metadata nebo programovací knihovnu pro Carbon a Graphite webapp.
- Graphite webapp - Djangová webová aplikace, která rendruje grafy za pomocí technologie Cairo.[37]

4.2.6. Vizualizace

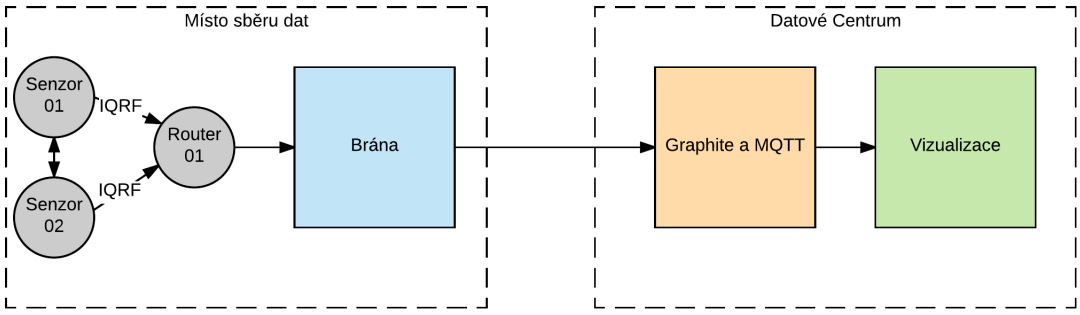
Uživatelské rozhraní, které slouží jako vizualizace získaných dat, je založeno na technologii Django. Django je pythoní framework pro tvorbu webových aplikací. Django se vyznačuje především snadnou implementací, bezpečností, univerzálním použitím a vysokou škálovatelností. Tento framework je mimo jiné open source. [38]

4.3. Platforma

V této sekci se věnuji spojení všech komponent, které jsou zmíněny v předešlých sekcích, do ucelené IoT platformy. Dále je zde nastíněna komunikace od senzoru až po vizualizaci pro koncového uživatele.

Tato platforma je složená z komponent, které se nacházejí přímo v terénu, a z komponent, které jsou v rámci datového centra. Přímo v místě sběru dat se nachází senzory, IQRF routery a IQRF brány. Z předešlých zmíněných technologií je zde použita i kontejnerová platforma Kubernetes a SDN řešení OpenContrail. V datovém centru se potom nachází cloudová platforma OpenStack, druhá část Kubernetes clusteru a také SDN platforma OpenContrail. Z aplikačního hlediska zde běží databáze časových řad Graphite a vizualizace sesbíraných dat.

Na následujícím obrázku je znázorněna logická topologie toku dat v rámci platformy. Senzor, který získá data ze svého prostředí, je odešle pomocí IQRF až k bráně. Ta odešle data prostřednictvím protokolu MQTT do MQTT kolektoru a zapíše je do Graphitu. Pak už jenom stačí vyčíst data z databáze a vizualizovat je pomocí vhodných prostředků.



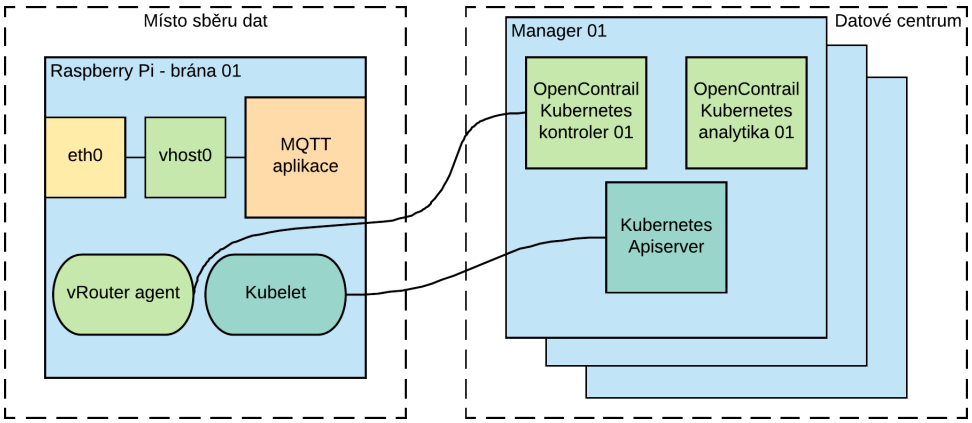
Obrázek 4.12.: Logická topologie IoT platformy

Logická topologie umožňuje zjednodušený pohled na celou platformu. Nyní je třeba představit ji i z fyzického hlediska se všemi technologiemi, které jsou zde přítomny.

Rasbperry Pi představuje bránu pro IQRF zařízení v dané lokalitě. Jeho napojení do zbytku platformy představovalo několik výzev, a to hlavně z důvodu jiné hardwarové architektury - ARM, než je běžně používána v datových centrech. Nejdříve bylo nutné zvolit vhodný operační systém. K tomuto účelu posloužila minimální instalace Debianu. První větší výzvou bylo připojení Raspberry ke Kubernetes clusteru. K tomu bylo nutné zajistit kód samotné kontejnerizace a také Kubernetes miniona. Pro tyto účely byly použity balíky Hypriotu [39]. Další a větší výzvou bylo zajistit kompatibilitu služeb OpenContrail. Ten totiž kromě služby - agenta, který je spuštěný v uživatelském prostoru Linuxu, potřebuje modul pro linuxové jádro. Komplexní systém vytváření balíků pro OpenContrail ovšem neumožňuje kompilaci pouze vRouteru a bylo tedy nutné vytvořit celou distribuci pro ARM Debian. Aby se toto podařilo, bylo nutné udělat jisté zásahy přímo do kódu vRouterového kernelového modulu. Posledním úkolem bylo vytvořit aplikační kontejner, který bude přijímat IQRF zprávy a bude je odesílat do MQTT kolektoru. Vzhledem k tomu, že kontejnerizace přejímá linuxové jádro, kontejner je závislý na hardwarové platformě, na které běží, a tedy nelze použít image například z x86 platformy. Uvnitř kontejneru se nachází Java aplikace. Raspberry má, jak jsem již dříve zmínil, připojený USB modul, který slouží jako IQRF brána. Tento USB modul je připojený z hostitelského operačního systému do tohoto kon-

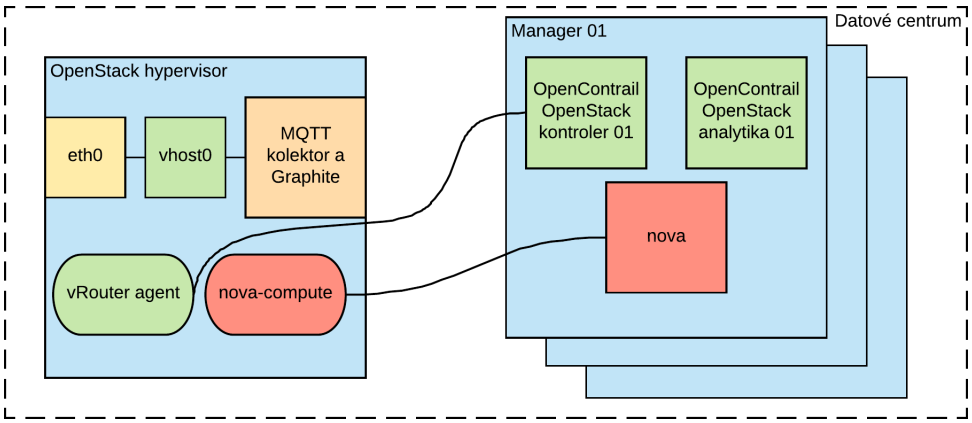
tejneru a zmíněná Java aplikace k němu přistupuje.

Na následujícím obrázku lze vidět konečný výsledek. Rasbperry Pi, na kterém OpenContrail agent komunikuje do datového centra s OpenContrail kontrolerem pro Kubernetes a také služba Kubelet, která komunikuje s Kubernetes API serverem. Dále je zde znázorněna MQTT aplikace, která běží jako dockerový kontejner, který je orchestrovaný pomocí Kubernetes.



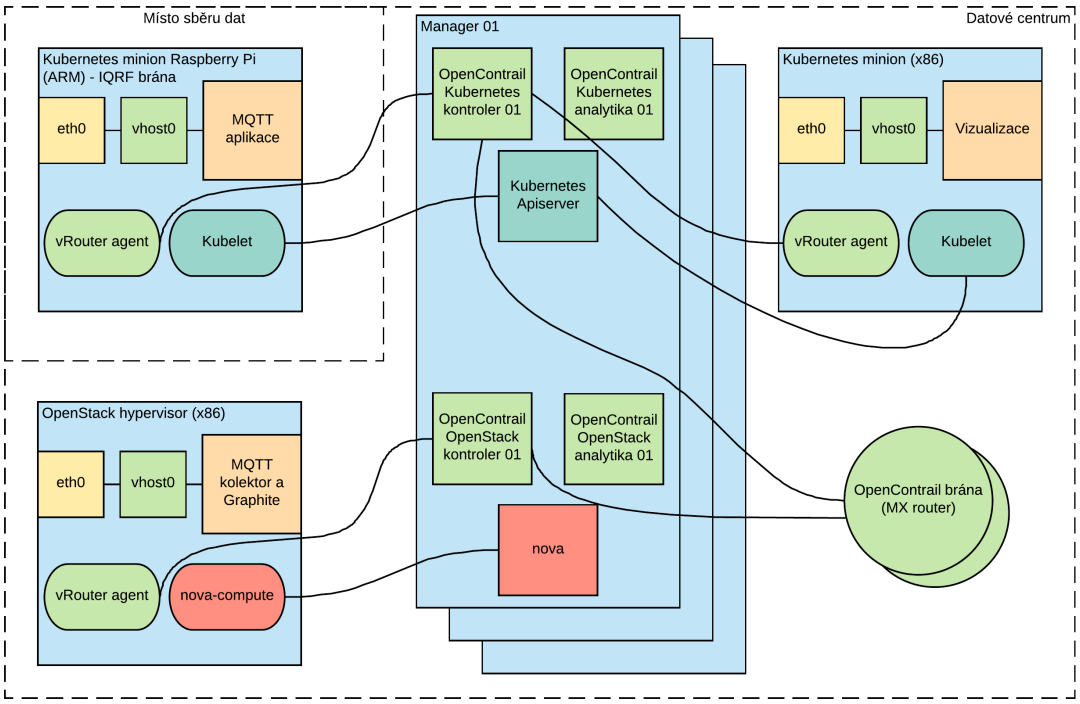
Obrázek 4.13.: Raspberry Pi s OpenContrail a Kubernetes

V předchozích odstavcích jsem zmínil komunikaci MQTT aplikace s MQTT kolektorem. MQTT kolektor běží spolu s databází Graphite na cloudové platformě OpenStack. Tato část platformy je realizována čistě v datovém centru. OpenStack hypervizor je klasický x86 server. Zde je použito Ubuntu 14.04 jako operační systém. Pro tuto platformu nebyly podobné problémy jako u Raspberry a bylo možné použít standardní balíky jak pro OpenContrail, tak pro OpenStack služby. Agent zde také komunikuje s OpenContrail kontrolerem, který je ale vyhrazený pro OpenStack.



Obrázek 4.14.: Openstack a OpenContrail

Na předešlých obrázcích byly představeny komponenty odděleně s tím, že pro lepší znázornění jsem odebral některé části. Nyní je třeba ukázat, jak vypadají jako celek. Ke Kubernetes clusteru je mimo jiné připojen další server, který slouží jako Kubernetes minion. Jedná se o běžný x86 server v datovém centru. Na tomto serveru běží kontejner s vizualizační aplikací. Mimo to jsou zde cloudové brány pro OpenContrail, které jsou realizovány například routery Juniper MX.



Obrázek 4.15.: Topologie IoT platformy jako celek

4.3.1. Vlastnosti IoT platformy

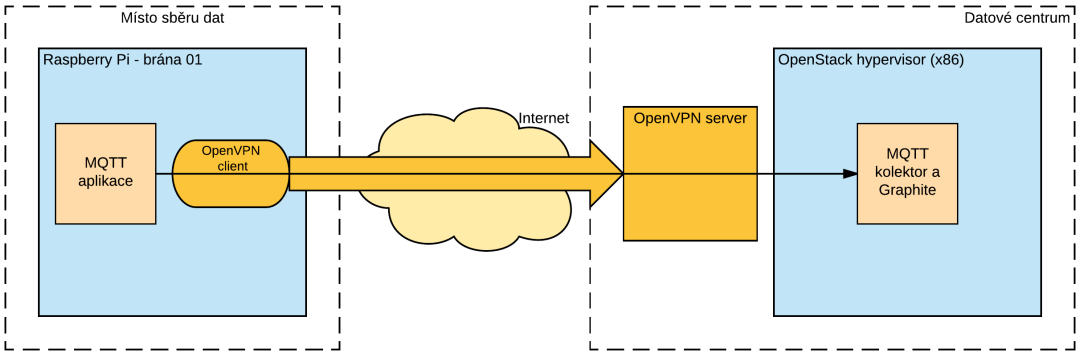
V této sekci se věnuji některým z vlastnostem, které tato IoT platforma splňuje a které jsou klíčové pro jakoukoli produkční platformu. Některé z těchto vlastností a jejich realizace již byly zmíněny v předcházejících sekcích. Nejdůležitější vlastnosti jsou:

1. Bezpečnost
2. Vysoká dostupnost
3. Škálovatelnost

4.3.1.1. Bezpečnost

Jak je zmíněno v sekci o IoT platformě Cisco IoT system, bezpečnost v sobě zahrnuje bezpečnost fyzickou a kybernetickou. V případě fyzické bezpečnosti je hlavní se zabývat bezpečností prvků v terénu, protože části platformy, které se nachází

v datových centrech, těží z jejich bezpečnostních opatření. Je tedy třeba umisťovat senzory na hůře dostupná místa, jako jsou vršky lamp. Rovněž je důležité fyzicky zabezpečit IQRF brány, které přímo komunikují s datovým centrem. Brány ale i přesto implementují několik kybernetických bezpečnostních opatření. Prvním z nich je zašifrovaný souborový systém. V případě, že by se zařízení někdo zmocnil, nebude schopen číst data, která jsou uložena na paměťovém zařízení. Dalším opatřením je zašifrovaná komunikace pro SDN a kontejnerizaci. OpenContrail umožňuje zašifrovanou XMPP komunikaci mezi agentem a kontrolerem. Stejně tak Kubernetes implementuje šifrování mezi Kubeletem a Kubernetes API serverem. Díky těmto prvkům je možné zabezpečit control plane komunikaci. Je však nutné zajistit i bezpečnost pro data plane komunikaci. Z tohoto důvodu je možné použití VPN řešení, v případě této platformy OpenVPN. OpenVPN je komunitní software, který umožňuje tunelovat komunikaci přes veřejnou síť za použití šifrování a autentizace poskytované OpenSSL knihovnou.[40]



Obrázek 4.16.: zapojení OpenVPN do IoT platformy

Kromě bezpečnosti přináší OpenVPN další důležitou výhodu. Touto výhodou je možnost napojení brány z prostředí, které nemá veřejnou IP adresu a tím pádem by s ní datové centrum nemohlo komunikovat. Díky OpenVPN její client zahájí komunikaci se serverem v datovém centru a po založení tunelu mohou služby bez problémů komunikovat.

4.3.1.2. Vysoká dostupnost

Vysoká dostupnost, nebo také redundance, je velmi důležitou vlastností každého řešení. Tato vlastnost zajišťuje funkčnost celého řešení v případě výpadku jedné komponenty. V reálu to znamená, že se každá komponenta řešení nasazuje alespoň ve dvojici, v případě například databází alespoň ve trojici.

Díky IQRF full-mesh topologii mají senzory vždy několik tras směrem k bráně. Brána realizována pomocí Raspberry Pi je poté v každé lokalitě alespoň v páru v módu active/backup (jedna brána je aktivní a druhá je pouze záloha). Aplikace uvnitř kontejnerů běžící na této bráně komunikuje s Graphite databází, které má celkem tři výskyty. Vizualizační aplikace běžící také v kontejneru je nasazena vždy alespoň ve dvojici. Vysoká dostupnost OpenStacku za pomocí HAProxy a Keepalived již byla zmíněna v předešlých sekcích. U OpenContrail a Kubernetes kontrolerů je realizace stejná. Posledním prvkem je brána pro OpenContrail - fyzický router, který je vždy také v páru.

4.3.1.3. Škálovatelnost

Další velmi důležitou vlastností je škálovatelnost. Ta určuje, zdali je platforma vhodná pro malé projekty, nebo je možné ji použít i pro velké projekty. Tato platforma je designována tak, aby byla každá komponenta škálovatelná především horizontálně, a tedy přidáváním dodatečných prvků do systémů v případě nedostatku zdrojů. Prvním příkladem je IoT brána. Zde není problém přidávat další Raspberry Pi. Backend v podobě databáze a MQTT kolektoru běží ve virtuálních instancích v cloudovém prostředí, kde se dají přidávat další stroje za běhu například za pomocí automatického škálování. Nativní podporu pro škálování má i Kubernetes.

5. Nasazení IoT platformy

V předešlé kapitole jsem představil IoT platformu založenou na otevřených technologiích. Po nastínění možného případu užití a vysvětlení jednotlivých komponent a funkčnosti je třeba představit platformu v praxi. Z hlediska případu užití jsem hlavně psal o aplikaci platformy v chytrých městech. Zde by byla realizace mnohem více složitější, a tak je třeba ji přenést do menšího prostoru. Jako vhodné místo realizace se ukázaly konferenční akce zaměřené na otevřené technologie. Cílem bylo dokázat, že lze pomocí spojení otevřených technologií vytvořit stabilní IoT platformu, která není závislá na konkrétním výrobci.

5.1. OpenStack Summit Austin - měření teploty, vlhkosti a CO2

První realizací byla konference OpenStack Summit v texaském Austinu v roce 2016. Tato realizace byla velmi důležitá a dá se považovat za zatěžkávací zkoušku. Akce se totiž konala v budově Austin Convention Center, která má celkem čtyři patra, kde vzdálenost mezi jedním koncem budovy v prvním patře a druhým koncem budovy ve čtvrtém patře tvoří několik set metrů a silné betonové stropy. V prvním patře se nacházel market se stánky jednotlivých firem, které představovaly svoje technologie spolu s halou pro Keynote prezentace. Tato hala měla možnost pojmout několik tisíc lidí. V dalších patrech se nacházely místnosti pro prezentace. Celkem bylo k dispozici 20 senzorů a 20 IQRF routerů, které byly rozmístěny napříč třemi patry tohoto konferenčního centra.

Tato implementace si dávala za cíl ukázat stabilitu IoT platformy a jejich využí-

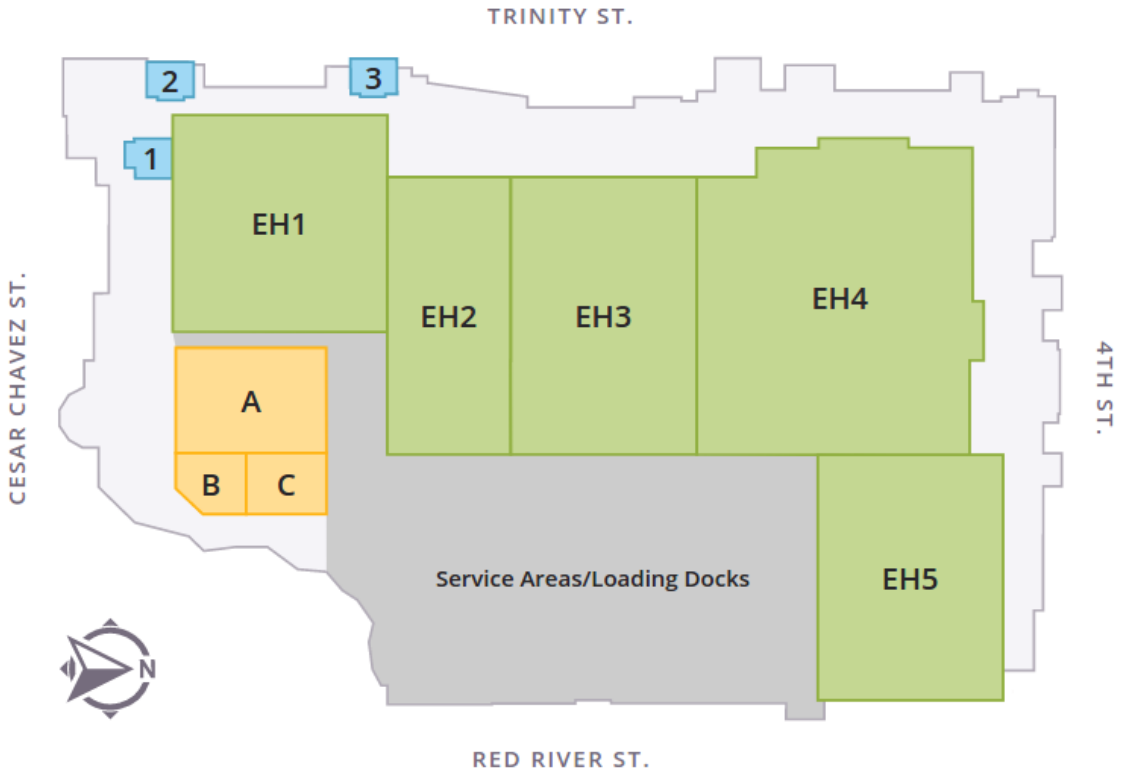
vaných technologií. Což se dá považovat za velmi obtížný úkol z důvodu použití velkého množství různých technologií, pouze omezenému testování v labovém prostředí a obtížností 'terénu', ve kterém se pohybují tisíce lidí. Co ale tato platforma vlastně poskytne za údaje? Každý ze senzorů měří tři environmentální parametry - teplota, vlhkost a CO2. Díky tomu bylo možné centrálně vizualizovat hodnoty těchto proměnných v čase, a to v celé budově konferenčního centra.

5.1.1. Senzory

Senzory pro měření teploty, vlhkosti a CO2 byly dodány firmou Protronix. Konkrétně se jedná o produkt New Line II. Tento senzor měří CO2 na základě principu závislosti útlumu infračerveného záření na koncentraci CO2 ve vzduchu. Relativní vlhkost je měřena za pomoci technologie založené na principu kapacitního polymerního snímače. New Line II je kompaktní, s rozměry 90x80x31 mm. Průměrná spotřeba energie, potřebná pro funkčnost senzoru, je velmi nízká a pohybuje se okolo 0,5 W. Senzor dokáže měřit CO2 v rozmezí od 0-2000 ppm a vlhkost od 0 do 100 % relativní vlhkosti s tím, že se odchylka od měření v 20-80% vlhkosti pohybuje v +- 3 %. Pracovní teplota je od 0 do 50 C. Senzory jsou také osazeny třemi diodami pro rychlé vizuální zjištění stavu CO2 v místnosti.[45]

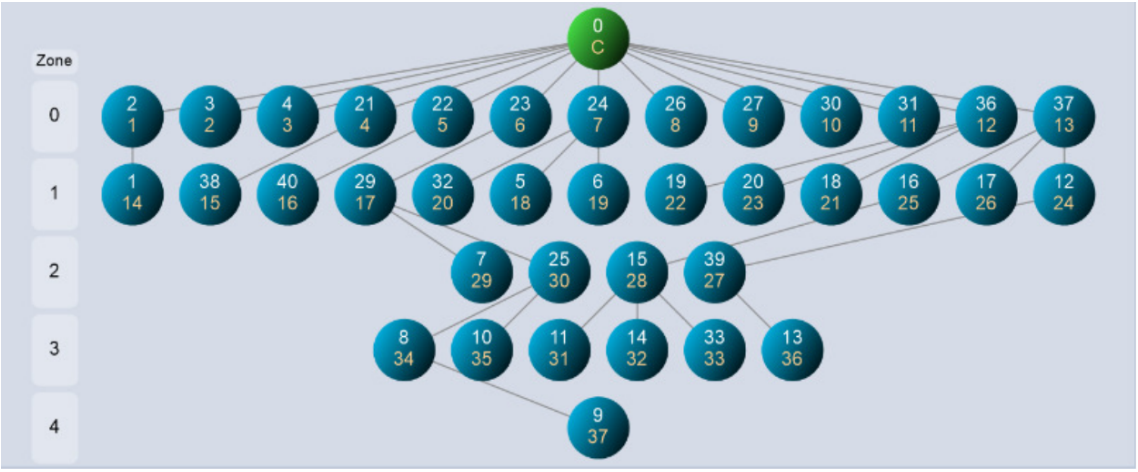
5.1.2. Implementace

Nejdříve bylo nutné naplánovat, kde dané senzory budou zapojeny. K tomu posloužily plány budovy, jejichž první patro je znázorněno na následujícím obrázku. Další patra jsou k nahlédnutí v příloze. EH4 na plánku sloužila jako místnost pro keynote prezentace. EH1-EH3 fungovaly jako market spolu se zázemím pro produkci. V tomto zázemí byla jediná možnost připojení IoT bran a bylo tedy nutné podle toho rozvrhnout rozmístění IoT routerů, aby se signál ze všech senzorů dostal až k bráně.



Obrázek 5.1.: První patro Austin Convention Center[41]

Po fyzickém zapojení senzorů a routerů přišla na řadu samotná brána. Nejdříve bylo nutné provést objevení senzorů a pospojování full-mesh sítě.



Obrázek 5.2.: Znázornění IQRF full-mesh sítě[44]

V této fázi už je IQRF síť spuštěna a senzory jsou schopné začít odesílat data.

Před tím je ale potřeba připojit bránu ke kontrolerům v datovém centru, konkrétně je řeč o připojení vRouteru k OpenContrail kontroleru a Kubeletu ke Kubernetes masteru. V tomto prostředí jsou však dostupné pouze omezené síťové prostředky, a tak nejsou k dispozici veřejné adresy. Je tedy třeba nejprve spustit OpenVPN klienta na bráně a zajistit bezpečný tunel do datového centra. Toto datové centrum se nachází v České republice v Písku. Data ze senzorů budou tedy posílána z USA přes oceán do Evropy, kde poté budou uložena a vizualizována.

Důležité je také puštění samotné aplikace, která vyčítá data z IQRF modulu a za pomocí MQTT protokolu je odešle do datového centra. Samotné vytvoření kontejneru je poměrně jednoduché ve chvíli, kdy je aplikace hotová. Jako předloha slouží kontejner obsahující javu a je vytvořen pro architekturu ARM. Poté už se jenom připraví pracovní adresář a nastaví se příkaz, který se má automaticky vykonat při spuštění kontejneru. Zdroj pro tuto aplikaci je v [42].

```
FROM dordoka/rpi-java8
RUN mkdir /var/app
ADD source /var/app/
WORKDIR /var/app
ENTRYPOINT ["sh", "run-iqrfgw.sh"]
```

Ukázka kódu 5.1: Dockerfile pro vytvoření IQRF aplikace

Samotné vytvoření kontejneru je otázkou několika dockerových příkazů. Docker build vytvoří kontejner. Docker tag pojmenuje kontejner tcpcloud/mqtt-iqrf-client:1.0, kde tcpcloud je projekt na docker hubu (veřejné uložiště, kam lze nahrávat kontejnery), mqtt-iqrf-client je název kontejneru a 1.0 je název revize. Pomocí docker push je tento kontejner nahrán na hub a je tedy dostupný z Internetu.

V předešlé kapitole v sekci věnované kontejnerizaci jsem zmínil, že se jedná o daemonSet. Tento objekt lze vytvořit v Kubernetes bez toho, aby byl Kubelet připojen k API. Objekt, který existuje v API bude spuštěn ve chvíli, kdy se brána připojí. Následující blok kódu představuje Kubernetes definici pro MQTT aplikaci. Po označkování Raspberry Pi jako noda s 'arch: arm', scheduler zde pustí pod, protože se shoduje s nodeSelectorem. Dále stojí za zmínku hostPath svazek,

který mapuje USB IQRF bránu z hostitelského systému přímo do kontejneru, aby k ní mohla přistupovat aplikace.

```
apiVersion: extensions/v1beta1
kind: DaemonSet
metadata:
  name: mqtt-iqrf-client
spec:
  template:
    metadata:
      labels:
        name: mqtt-iqrf-client
    spec:
      nodeSelector:
        arch: arm
      containers:
      - image: tcpcloud/mqtt-iqrf-client:1.0
        imagePullPolicy: Always
        name: mqtt-iqrf-client
        volumeMounts:
        - mountPath: /iqrf
          name: iqrf
      volumes:
      - name: iqrf
        hostPath:
        path: /dev/ttyACM0
```

Ukázka kódu 5.2: daemonSet manifest pro MQTT klienta

Samotná konfigurace MQTT klienta je jednoduchá a v základu stačí pouze nastavit proměnnou broker, která představuje adresu MQTT kolektoru a jeho port. Adresa 172.16.10.131 je privátní a patří instanci, která běží v OpenStack cloudu v datovém centru Písek. Komunikace po privátních adresách je možná proto, že je mezi kontejnerem a instancí přímý MPLS over UDP tunel, který je realizovaný

pomocí OpenContrail federace mezi Kubernetes a OpenStackem.

```
{
    "protocol": "tcp://",
    "broker": "172.16.10.131",
    "port": 1883,
    "clientid": "std",
    "gwid": "s827um26i73t",
    "cleansession": true,
    "quitemode": false,
    "ssl": false,
    "certfile": "",
    "username": "",
    "password": "",
    "roottopic": ""
}
```

Ukázka kódu 5.3: Konfigurace MQTT klienta

V této fázi již nic nebrání aplikaci, aby připojila senzory a začala přijímat data přes IQRF. Nejdříve je vidět objevování jednotlivých senzorů. Poté se aplikace připojí do MQTT kolektoru. V tuto chvíli začne aplikace získávat první data a odesílá je do kolektoru. V předešlé kapitole jsem nastínil fungování MQTT protokolu, konkrétně odesílání zpráv do témat. Zde je vidět, že MQTT klient odesílá tři zprávy do jednoho tématu, který náleží prvnímu senzoru. Jsou tři během jednoho okamžiku, protože každá ze zpráv představuje jednu zasílanou metriku.

```
Starting initialization of Simply ...
Creating network 1 ...
Number of bonded nodes: 14
Bonded nodes: [1, 2, 3, 4, 6, 7, 20, 21, 22, 23, 24, 25, 26, 27]
Run discovery ...
Number of discovered nodes: 2
Creating node 2:
```

```
Peripherals: [1, 2, 3, 4, 5, 6, 7, 9, 10, 12, 13]
Node created
...
Network 1 successfully created.
Initialization of Simply complete.
Connecting to tcp://172.16.10.131:1883 with client ID std
Connected
Getting data from device: 1
Getting data from device: 2
Preparing MQTT message for node: 1
Preparing MQTT message for node: 2
Sending parsed data for node: 1
Publishing at: 2016-04-24 06:46:38.04 to topic
   "02bc74dca060/sensors/protronix/1" qos 2
Publishing at: 2016-04-24 06:46:39.352 to topic
   "02bc74dca060/sensors/protronix/1" qos 2
Publishing at: 2016-04-24 06:46:39.405 to topic
   "02bc74dca060/sensors/protronix/1" qos 2
```

Ukázka kódu 5.4: Ukázka odeslání dat do MQTT kolektoru

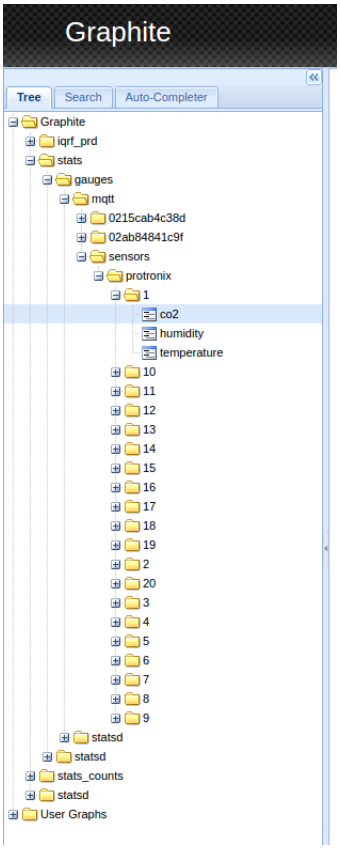
Odesílané metriky mají tvar, který je znázorněn v následujícím bloku. Teplota v jedné místnosti byla 22.9 stupňů, vlhkost 34.15% relativní hmotnosti a 568 PPM (částic na jeden milion) úrovně CO2.

```
b827ebe76977/sensors/protronix/10
   {"e":[{"n":"temperature","u":"Cel","v":22.9}],"bn":"urn:dev:mid:81009364"}
b827ebe76977/sensors/protronix/10
   {"e":[{"n":"humidity","u":"%RH","v":34.1}],"bn":"urn:dev:mid:81009364"}
b827ebe76977/sensors/protronix/10
   {"e":[{"n":"co2","u":"PPM","v":568}],"bn":"urn:dev:mid:81009364"}
b827ebe76977/sensors/protronix/11
   {"e":[{"n":"temperature","u":"Cel","v":22.3}],"bn":"urn:dev:mid:8100936A"}
b827ebe76977/sensors/protronix/11
```

```
    {"e":[{"n":"humidity","u":"%RH","v":36}],"bn":"urn:dev:mid:8100936A"}
b827ebe76977/sensors/protronix/11
    {"e":[{"n":"co2","u":"PPM","v":557}],"bn":"urn:dev:mid:8100936A"}
```

Ukázka kódu 5.5: Lokální čtení metrik

Po několika minutách jsou sesbírána první data ze všech senzorů a nyní jsou všechna témata vidět i v databázi Graphite:



Obrázek 5.3.: Ukázka dat v databázi Graphite

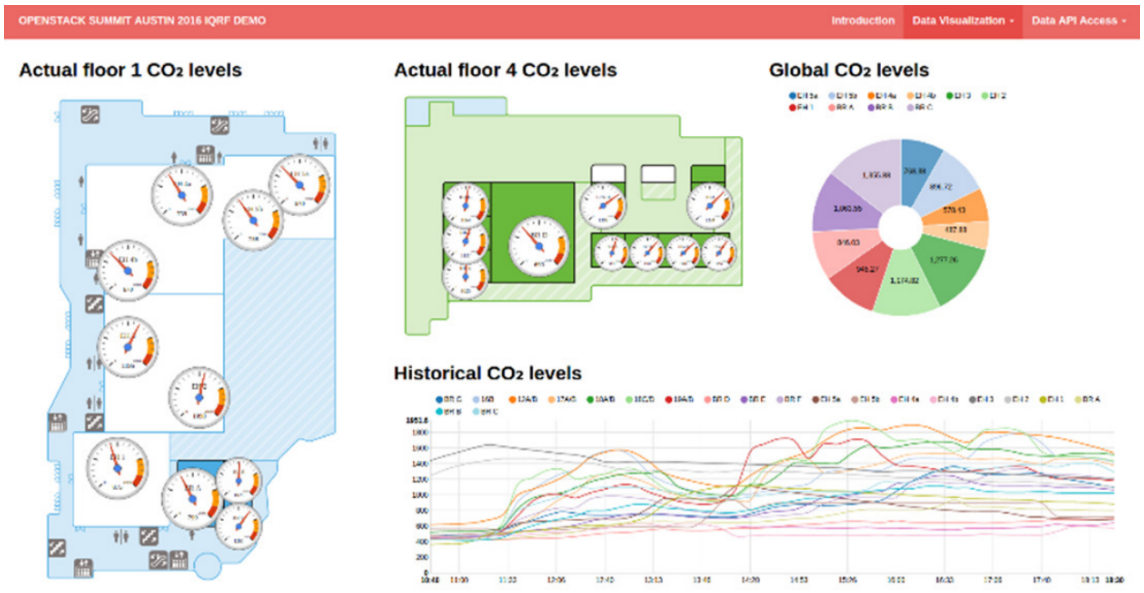
V této fázi je platforma v plně funkčním stavu a už stačí pouze vyčítat data z Graphite API a vizualizovat.

5.1.3. Výsledek

Tato IoT platforma byla v provozu necelý týden. Během té doby se podařilo získat velké množství zajímavých dat a určitě by se s nimi dalo dále pracovat a

analyzovat je. To však není cílem ani předmětem této práce, a proto se tím nebudu hlouběji zaobírat. Ale i přesto zmíním některé zajímavé skutečnosti. Jednou z nich bylo enormní zvýšení všech měřených hodnot v místnosti EH4 v době, kdy začaly dopolední keynote přednášky, na které dorazilo mezi čtyřmi až pěti tisíci lidí. Ostatní senzory si během této doby držely přibližnou hladinu jako z večera. Po skončení keynote přednášek se osazenstvo přesunulo na jednotlivé přednášky do vyšších pater. Hodnoty zde opět výrazně klesly v době prvního otevření marketu a také během oběda. Další zajímavá skutečnost se týkala dosažené výšky hladiny některých metrik. Konkrétně hladina, které dosahoval CO2 v některých místnostech. Podle [43] je zdravé množství CO2 v místnostech v rozmezí 300-1000 ppm. Při hodnotách nad 1000 dochází k poruchám soustředění a hodnoty nad 5000 dokonce představují zdravotní riziko. Tak vysoké hodnoty naštěstí naměřeny nebyly, ale v některých místnostech hodnota CO2 dosahovala až 1500 ppm.

Během celého trvání summitu byla veškerá data dostupná všem účastníkům. Na následujícím obrázku je náhled na vizualizaci. Kromě grafů bylo možné přistoupit k naměřeným datům přímo přes api.



Obrázek 5.4.: Vizualizace získaných dat[44]

Důležité je, že platforma byla provozuschopná po většinu času. Je však třeba zmínit, že případné výpadky nebyly způsobeny chybou platformy, ale nestabili-

tou internetového připojení a tím i propojení do datového centra v Písku.

5.2. OpenStack Summit Barcelona - měření počtu lidí

Po úspěšném summitu v Austinu se OpenStack foundation (organizace zaštiťující OpenStack projekt a pořadatel OpenStack summitů) rozhodla, že chtějí platformu využít i na příští konferenci, která se konala v Barceloně v říjnu 2016 a účastnilo se jí přes 5200 lidí. Úkolem však bylo použít platformu s úplně jiným druhem senzorů, a tak se zde neměřila teplota, vlhkost a CO2, ale počty lidí. Cílem bylo zjistit, jak se mění počty lidí na různých prezentacích v čase a získat tedy informace, které přednášky měly plno, ze kterých přednášek lidé brzy odešli a u kterých zůstali až do konce. Realizace byla možná dvěma způsoby. Prvním z nich bylo umístění čipu do visaček každého z účastníků. Druhý způsob byl umístění čidel pohybu ke vchodům do prezentačních místností. Z etických důvodů se OpenStack foundation rozhodla pro druhé řešení.

Konference se konala v konferenčním centru Centre de Convencions Internacional a celkem zde bylo rozmístěno 12 bran do různých prezentačních místností. Situovány byly přímo u vchodů na stojanech, které zapůjčila produkce. Kromě těchto skutečností byla platforma totožná s tou, která byla použita v Austinu.

5.2.1. Senzory

Senzory pro tuto konferenci byly vyrobeny na zakázku firmou Protronix. Jejich účelem je, jak již bylo zmíněno, počítání průchodu osob. Počítadlo se skládá ze dvou částí, které jsou umístěny proti sobě. Při počítání počtu prošlých osob rozlišuje směr, kterým dané osoby chodí, ale zároveň předpokládá pohyb osob pouze v jedné řadě za sebou. Části počítadla mohou být ve vzdálenosti od 50 do 200 cm. Senzor dokáže pracovat i ve styku s vodou, má pracovní teplotu od -20 do +60 a díky tomu lze použít ve vnitřních i venkovních prostorách. Rozměry tohoto senzoru jsou 80x160x55 mm. Počítadlo je osazeno led diodou, která slouží

k signalizaci toho, zdali jsou obě části umístěné správně proti sobě a je navázané spojení.

5.2.2. Výsledek

I když i v tomto případě vykazovala platforma podobně vysokou dostupnost, tak získaná data byla ve výsledku velmi zkreslená a neodpovídala reálnému stavu. Senzory fungují na principu +1 v případě příchodu člověka a -1 v případě odchodu člověka. Tím pádem by po odchodu všech lidí měl být součet těchto hodnot nulový, a to se bohužel nestávalo. Nelze z toho ale pravděpodobně vinit platformu ani senzory. Problémem zde byla šířka vstupu do jednotlivých místností, kde mohli pohodlně vstupovat i tři lidi najednou a tím znehodnotit získávaná data. Dalším problémem zde bylo i dodané řešení od produkce, tedy sloupky, na které se senzory instalovaly. Senzory jsou schopné správně fungovat, pokud na sebe oba konce mají přímou viditelnost. Zde však byla instalace náchylná na lidský faktor, kdy kolemjdoucí účastník mohl omylem posunout konstrukci a tím rozpojit spojení mezi senzory.

6. Závěr

V této práci jsem se věnoval jedné z velmi rychle rostoucí oblasti informačních a komunikačních technologií, Internetu věcí. V této oblasti panuje značná roztříštěnost technologií a způsobů implementace těchto technologií jednotlivými výrobci. Tato práce byla zaměřena na konkrétní oblast IoT a tou je platforma pro Internet věcí. Platforma představuje end-to-end řešení a není tedy možné se zde zabývat pouze jednou oblastí IoT, přestože by se jenom na každou z komponent IoT platformy dala napsat samostatná práce. Přesto však bylo nutné vymezit základní technologie světa IoT a právě tomu byla věnována druhá kapitola.

Cílem této práce bylo vytvoření vlastní IoT platformy za použití nového komunikačního protokolu IQRF. Tato platforma by měla splňovat určité kvalitativní znaky a měla by mít ambice na reálné produkční nasazení. Z toho důvodu bylo nutné provést rešerši již existujících platforem od renomovaných IT společností, jako je Microsoft, Amazon, Cisco nebo Intel. Tato rešerše se nachází ve třetí kapitole. Každá z těchto platforem byla svým malým či velkým přispěním inspirací pro platformu, která je navržena v této práci.

Po vymezení Internetu věcí a zmapování platforem pro něj bylo možné přistoupit k designu vlastní platformy. Ta se skládá z několika různých komponent. Dalo by se říci, že se mnohdy jedná o komponenty, kde nikdo předtím nepředpokládal jejich využití v Internetu věcí a hlavně nepředpokládal kooperaci těchto komponent za účelem komplexní jednotné platformy. Díky tomu by se tato práce dala považovat spíše za systémovou než doménově specifickou. Samozřejmostí jsou zde senzory, které získávají data ze svého prostředí. Brány realizované pomocí IQRF přijímače zapojeného do malého počítače Raspberry Pi. Tento počítač je zároveň součástí platformy pro ovládání kontejnerové virtualizace Kubernetes a softwa-

rově definované sítě orchestrované řešením OpenContrail. Data jsou za pomoci této sítě odesílána do datového centra, kde běží cloudová platforma OpenStack hostující virtuální servery, které s daty mohou dále pracovat. Tato data jsou poté dostupné skrze webovou aplikaci, která je opět realizována kontejnerovou virtualiazací. Dochází zde tedy ke spojení tří velkých open source projektů - OpenStack, Kubernetes a OpenContrail.

Kromě samotného navržení platformy, na kterém lze ukázat možné teoretické či labové použití, bylo nutné tuto platformu přenést do reálného prostředí, aby bylo možné potvrdit v páté kapitole zmíněnou ambici na produkční nasazení. Tato možnost se naskytla hned dvakrát, a to na OpenStack summitu v Austinu a Barceloně. Oba tyto testy dopadly pro platformu úspěšně. Průběh této implemetace byl popsán v páté kapitole.

Samotný návrh platformy byl zajímavý, ale realizace pro mě představovala úplně nový zážitek. Všechny tři použité projekty jsem znal již z dřívější doby, ale o světě Internetu věcí jsem nevěděl téměř nic. Díky této práci jsem tedy ze znalostního hlediska získal vhled do problematiky Internetu věcí, ale hlavním přínosem pro mě byla právě ta realizace. Příprava projektu, jehož výsledek uvidí v reálném čase několik tisíc lidí, byla opravdová výzva. Samozřejmostí bylo také poznání mnoha nových a zajímavých lidí. Veškeré vynaložené úsilí tak stálo za to.

Literatura

[1] KERAMIDAS, Georgios, Nikolaos VOROS a Michael HÜBNER. Components and Services for IoT Platforms: Paving the Way for IoT Standards. Springer International Publishing, 2017. DOI: 10.1007/978-3-319-42304-3. ISSN 978-3-319-42304-3.

[2] BALANI, Naveen, HATHI, Rajeev, ed. Enterprise IoT: A Definitive Handbook. 4. CreateSpace Independent Publishing Platform, 2016. ISBN 978-1535505642.

[3] IoT Platforms White Paper. [online]. [cit. 2017-05-02]. Dostupné z: https://iot-analytics.com/product/iot-platforms-white-paper/

[4] UCKELMANN, Dieter, Mark HARRISON a Florian MICHAHELLES, ed. Architecting the Internet of Things. Springer, 2011. ISBN 9783642191565.

[5] HERSENT, Olivier, David BOSWARTHICK a Omar ELLOUMI. The Internet of Things: Key Applications and Protocols. 2. Wiley, 2012. ISBN 9781119994350.

[6] Shelby, Z., et al. The Constrained Application Protocol (CoAP). [online]. IETF Tools. [cit. 2017-05-04]. Dostupné z: tools.ietf.org/html/rfc7252.

[7] GARDAŠEVIĆ, Gordana, Mladen VELETIĆ, Nebojša MALETIĆ, Dragan VASILJEVIĆ, Igor RADUSINOVIĆ, Slavica TOMOVIĆ a Milutin RADONJIĆ. The IoT Architectural Framework, Design Issues and Application Domains. Wireless Personal Communications. 2016, 1, 127–148. DOI: https://doi.org/10.1007/s11277-016-3842-3. ISSN 1572-834X.

[8] DIRK, Slama. Enterprise IoT. O'Reilly, 2016. ISBN 978-1-4398-9299-2.

[9] HONBO, Zhou. The Internet of Things in the Cloud a Middleware Perspective. CRC Press, 2013. ISBN 978-1-4398-9299-2.

[10] Shelby, Z., ARM, C. Bormann, and Universitaet Bremen TZI. The Constrained Application Protocol (CoAP). [online]. IETF Tools. [cit. 2017-05-04]. Dostupné z: https://tools.ietf.org/html/rfc7252.

[11] Rouse, Margaret. What Is Fog Computing (fog Networking, Fogging)? - Definition from WhatIs.com. [online]. [cit. 2017-05-07]. Dostupné z: http://internetofthingsagenda.techtarget.com/definition/fog-computing-fogging.

[12] Ibm. (n.d.): n. pag. Building Smarter Planet Solutions with MQTT and IBM WebSphere MQ Telemetry. [online]. [cit. 2017-05-07]. Dostupné z: http://www.redbooks.ibm.com/redbooks/pdfs/sg248054.pdf.

[13] OVIDIU, Vermesan a Peter FRIESS. Internet of Things: Converging Technologies for Smart Environments and Integrated Ecosystems. River Publishers, 2013. ISBN 978-8792982735.

[14] IoT Security and Scalability on Intel® IoT Platform. [online]. [cit. 2017-05-09]. Dostupné z: https://www.intel.com/content/www/us/en/internet-of-things/iot-platform.html.

[15] The Intel IoT Platform. [online]. [cit. 2017-05-09]. Dostupné z: https://www.intel.com/content/dam/www/public/us/en/documents/white-papers/iot-platform-reference-architecture-paper.pdf.

[16] Cisco *The Cisco IoT System*. In: RightScale [online]. 2015 [cit. 2017-05-13]. Dostupné z: `https://www.cisco.com/c/dam/en/us/products/collateral/wireless/industrial-wireless-3700-series/at-a-glance-c45-734164.pdf`

[17] Cisco *Internet of Things (IoT) - Cisco IoT Product Portfolio*. In: RightScale [online]. 2017 [cit. 2017-06-13]. Dostupné z: `http://www.cisco.com/c/en/us/solutions/internet-of-things/iot-products.html`

[18] Azure IoT suite [online]. [cit. 2017-05-10]. Dostupné z: https://opbuildstorageprod.blob.core.windows.net/output-pdf-files/en-us/Azure.azure-documents/live/iot-suite.pdf

[19] Microsoft Azure IoT Reference Architecture. [online]. [cit. 2017-05-10]. Dostupné z: http://download.microsoft.com/download/A/4/D/A4DAD253-BC21-41D3-B9D9-87D2AE6F0719/Microsoft_Azure_IoT_Reference_Architecture.pdf

[20] AWS IoT Documentation. [online]. [cit. 2017-05-10]. Dostupné z: https://aws.amazon.com/documentation/iot/

[21] PAVLÍK, Jakub, KOMÁREK, Aleš, SOBĚSLAV, Vladimír, Open IoT Platform Design for Urban Environments, Konference ADIBUM 2016, Open IoT Platform Design for Urban Environments,Sobeslav, Komarek, Pavlik - žurnál se SJR

[22] About IQRF. [online]. [cit. 2017-05-14]. Dostupné z: http://www.iqrf.org/iqrfabout

[23] IQRF products. [online]. [cit. 2017-05-14]. Dostupné z: http://www.iqrf.org/products

[24] mqtt. [online]. [cit. 2017-05-15]. Dostupné z: http://mqtt.org/

[25] mqtt — MQ Telemetry Transport. [online]. [cit. 2017-05-15]. Dostupné z: http://mosquitto.org/man/mqtt-7.html

[26] RASPBERRY PI 3 MODEL B. [online]. [cit. 2017-05-15]. Dostupné z: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/

[27] Cisco *The Cisco IoT System*. [online]. 2015 [cit. 2017-06-13]. Dostupné z: `https://www.cisco.com/c/dam/en/us/products/collateral/se/internet-of-things/brochure-c02-734481.pdf`

[28] TURNBULL, James. The Docker Book: Containerization is the new virtualization. James Turnbull, 2014. ISBN 9780988820203.

[29] Kubernetes documentation. [online]. [cit. 2017-05-20]. Dostupné z: https://kubernetes.io/docs

[30] BUMGARDNER, V. K. Cody. OpenStack in Action. Manning Publications, 2016. ISBN 9781617292163.

[31] JACKSON, Kevin, Cody BUNCH a Egle SIGLER. OpenStack Cloud Computing Cookbook. 3. Packt Publishing, 2015. ISBN 978-1782174783.

[32] MARSCHKE, Doug, Jeff DOYLE a Pete MOYER. Software Defined Networking (SDN): Anatomy of OpenFlow Volume I. Lulu Publishing Services, 2015. ISBN 978-1483427232.

[33] MONGE, Antonio Sanchez a Krzysztof Grzegorz SZARKOWICZ. MPLS in the SDN Era: Interoperable Scenarios to Make Networks Scale to New Services. O'Reilly Media, 2015. ISBN 978-1491905456.

[34] RIJSMAN, Bruno a Ankur SINGLA. Day One: Understanding OpenContrail Architecture [online]. Juniper Networks Books, 2013 [cit. 2017-06-25]. Dostupné z: https://www.amazon.com/Day-One-Understanding-OpenContrail-Architecture-ebook/dp/B00GTXGP7O

[35] Mahalingam M., Storvisor, Dutt D., Cumulus Networks, Duda K., Arista, Argwal K., Broadcom, Kreeger L., Cisco, VMware, Intel, Red Hat. Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks. [online]. IETF Tools. [cit. 2017-06-24]. Dostupné z: https://tools.ietf.org/html/rfc7348

[36] Bates T., Cisco Systems, Chandra R., Sonoa Systems, Katz D., Rekhter Y., Juniper Networks. Multiprotocol Extensions for BGP-4. [online]. IETF Tools. [cit. 2017-07-04]. Dostupné z: https://tools.ietf.org/html/rfc4760

[37] DIXON, Jason. Monitoring with Graphite: Tracking Dynamic Host and Application Metrics at Scale [online]. O'Reilly Media, 2017 [cit. 2017-07-04]. ISBN 978-1491916438.

[38] Django overview. Djangoproject [online]. [cit. 2017-06-04]. Dostupné z: https://www.djangoproject.com/start/overview/

[39] Hypriot. [online]. [cit. 2017-06-10]. Dostupné z: https://blog.hypriot.com/

[40] What is OpenVPN. [online]. [cit. 2017-06-11]. Dostupné z: https://openvpn.net/index.php/open-source/333-what-is-openvpn.html

[41] Floor Plans. [online]. [cit. 2017-07-11]. Dostupné z: https://www.austinconventioncenter.com/facility/plans/

[42] Iqrfsdk-incubator. [online]. [cit. 2017-07-11]. Dostupné z: https://github.com/MICRORISC/iqrfsdk-incubator

[43] Ashrae. [online]. [cit. 2017-06-20]. Dostupné z: www.ashrae.org/File%20Library/docLib/Public/20100608_62_1_2007_q_final.pdf

[44] PAVLÍK, Jakub, KOMÁREK, Aleš, SOBĚSLAV, Vladimír, Cloud Computing Platform for Internet of Things , Konference ADIBUM 2016, Cloud Computing Platform for Internet of Things, Sobeslav, Komarek,Pavlik - žurnál se SJR

[45] NLII-CO2. [online]. [cit. 2017-07-12]. Dostupné z: http://eshop.cidla.cz/data/_docs/Katalog_list_NLII_CO2_RH_2.pdf

Seznam obrázků

Seznam tabulek

Seznam ukázek kódu

Přílohy

A. Senzory

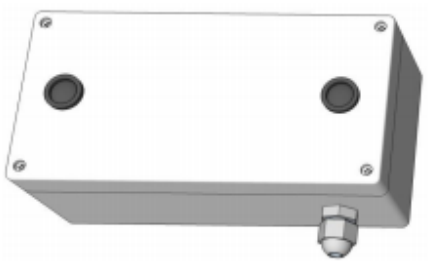
Obrázek A.1.: Senzor na měření C02, vlhkosti a teploty

PROTRONIX
SENSE & EASY

Uživatelský návod

IL-PC1 | Počítadlo průchodu osob

Světelná závora IL-PC1 slouží pro počítání průchodu osob. Skládá se ze dvou částí - primární a sekundární - umístěných v průchodu proti sobě. Naměřené hodnoty jsou k dispozici přes drátové sériové komunikační rozhraní nebo bezdrátově, komunikací přes integrovaný radiový modul



- Počítá počet průchodů s rozlišením směru
- Předpokládá se pohyb osob v jedné řadě za sebou
- Bezdrátové čtení aktuální hodnoty, předávání dat do cloudu [1)]
- Sériová drátová komunikace Modbus [2)]
- Rychlá odezva
- Vhodné do interiéru i exteriéru
- Dlouhodobá životnost

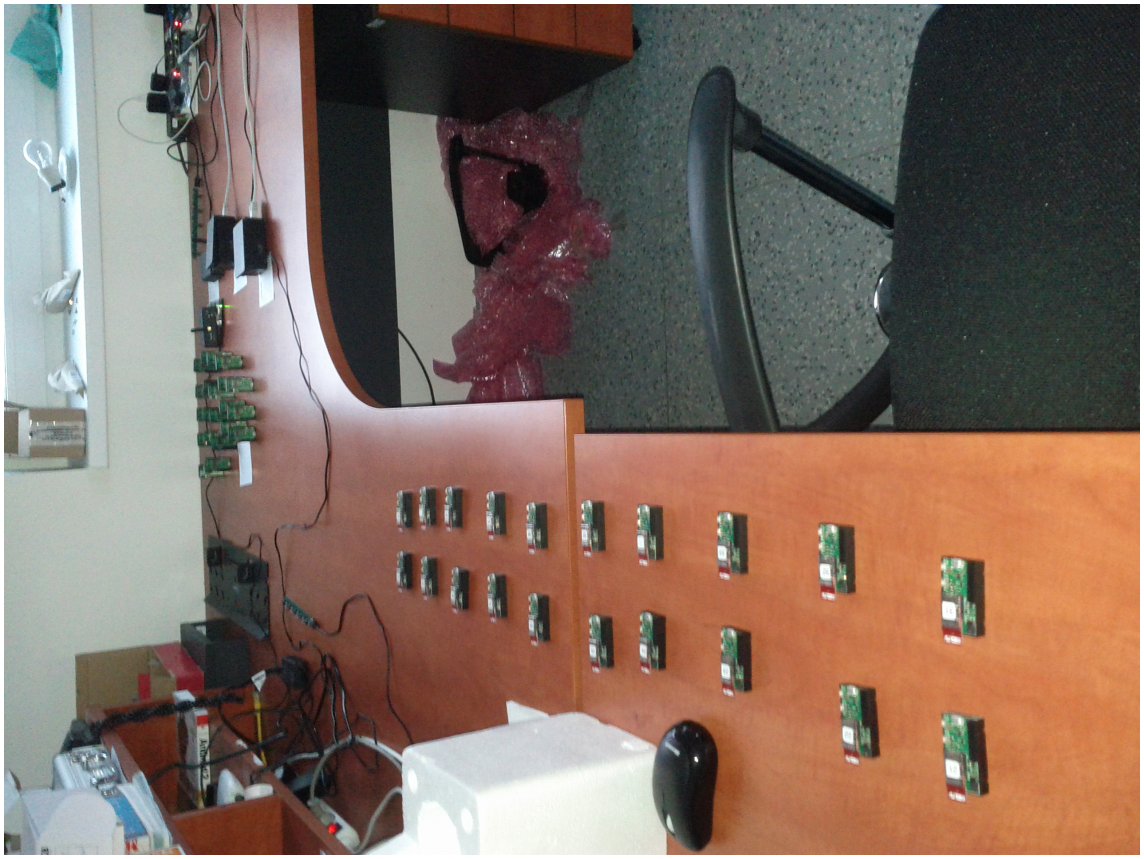
1) variantně integrovaný radiový modul a odpovídající gateway
2) variantně integrovaný komunikační modul RS485

Technická data

Parametr	Hodnota	Jednotka
Rozsah napájecího napětí	14 V – 30 V DC	
Krytí	IP65	
Odstup senzorů	50 – 200	cm
Max. intenzita okolního osvětlení	100 000	lux
Pracovní vlhkost nekondenzující	0 – 95 %	RH
Pracovní teplota	-20 až +60	°C
Skladovací teplota	-25 až +70	°C
Očekávaná životnost	min. 10	let
Rozměry	80 x 160 x 55	mm
Max. délka propoj kabelu	8	m

Obrázek A.2.: Senzor na měření počtu průchodů lidí

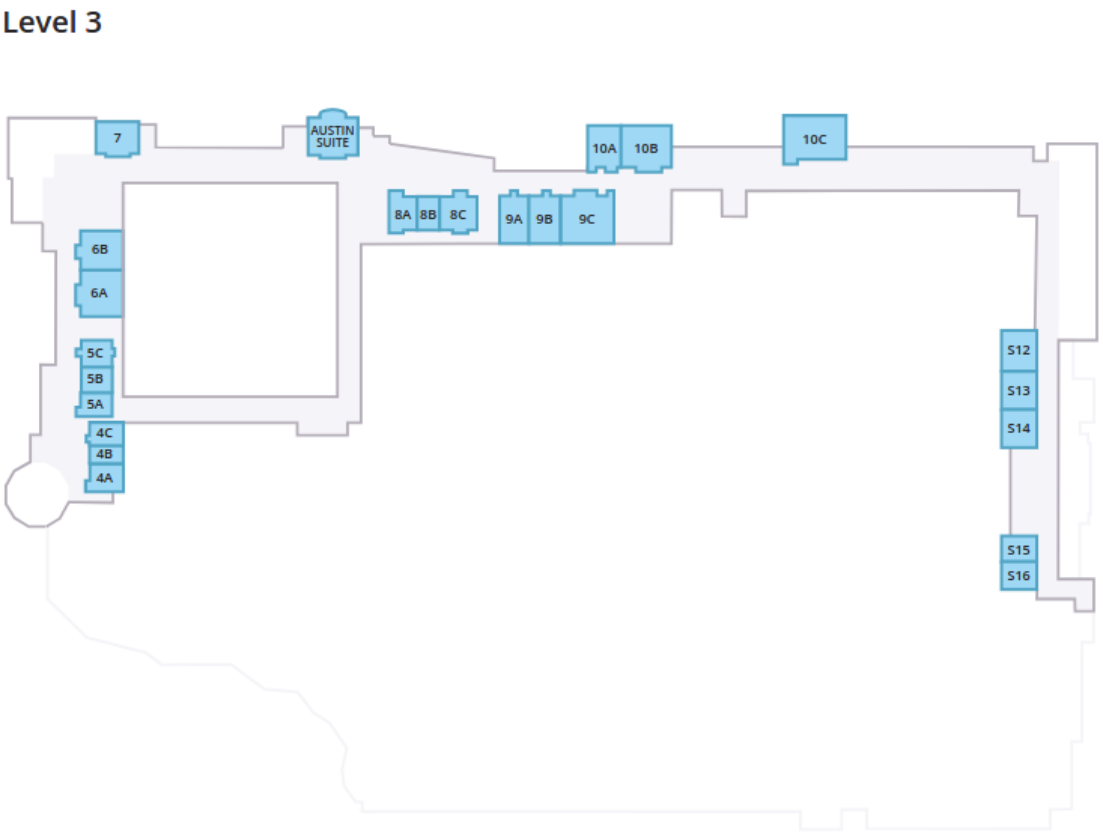
B. Příprava na OpenStack summit v Austinu



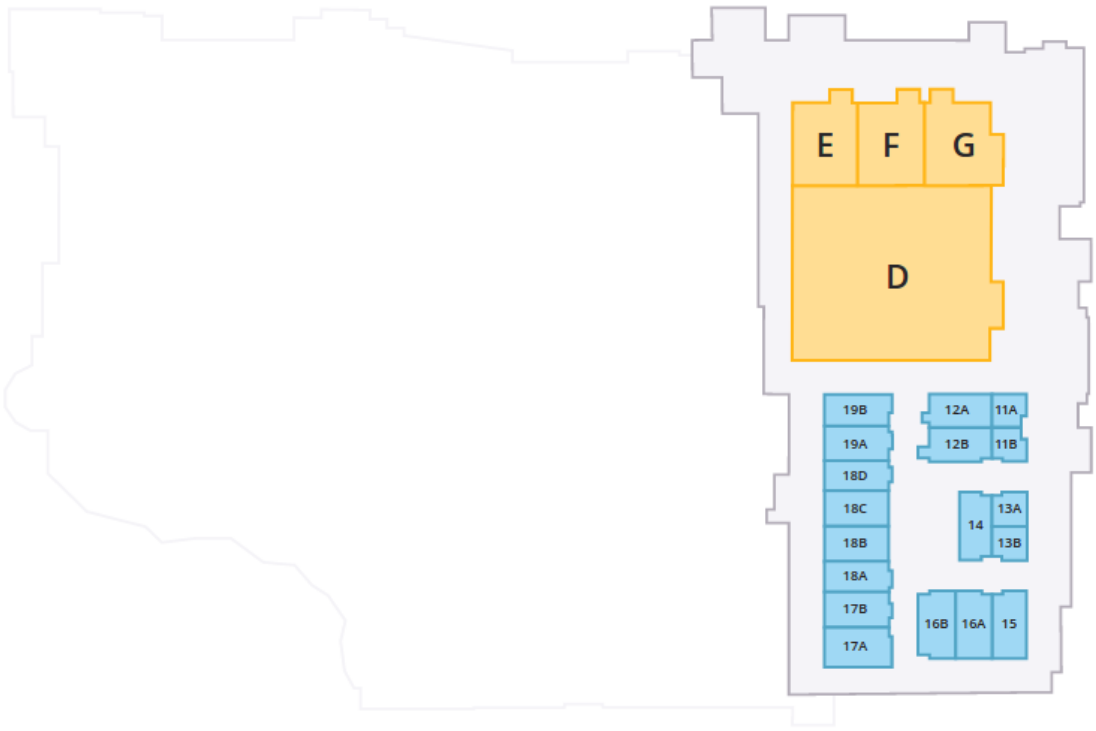
Obrázek B.1.: IQRF routery pro OpenStack summit v Austinu

Obrázek B.2.: Senzory na měření C02, vlhkosti a teploty pro OpenStack summit v Austinu

C. Austin Convention Center



Obrázek C.1.: Třetí patro Austin Convention Center[41]



Obrázek C.2.: Čtvrté patro Austin Convention Center[41]

Univerzita Hradec Králové
Fakulta informatiky a managementu
Akademický rok: 2016/2017

Studijní program: Systémové inženýrství a informatika
Forma: Prezenční
Obor/komb.: Informační management (im2-p)

Podklad pro zadání DIPLOMOVÉ práce studenta

PŘEDKLÁDÁ:	ADRESA	OSOBNÍ ČÍSLO
Bc. Čeloud Marek	Jindrova 160/20 20, Hradec Králové - Plácky	I1500726

TÉMA ČESKY:

Budování infrastrukturní platformy pro IoT senzorickou IQRF full mesh síť

TÉMA ANGLICKY:

VEDOUCÍ PRÁCE:

Ing. Jakub Pavlík - KIT

ZÁSADY PRO VYPRACOVÁNÍ:

Cílem projektu je vytvořit senzorickou platformu, která má za úkol sběr a předpracování metrik z městského osvětlení a enviromentalních hodnot. To může zahrnuje zhruba 300 IoT gateway. Jejich připojení do centrálního bodu je složité z důvodu různých přenosových infrastruktur jako LTE, GSM, WiFi. Cílem tohoto výběrového projektu je navrhnout softwarově definovanou síť, která umožní nezávislí přenos metrik do centrálního datového centra ke zpracování.

SEZNAM DOPORUČENÉ LITERATURY:

Enterprise IoT: Strategies and Best Practices for Connected Products and Services 1st Edition

Kubernetes: Up and Running: Dive into the Future of Infrastructure 1st Edition

IoT Disruptions: The Internet of Things - Innovations & Jobs

Podpis studenta: ..... **Datum:** 9.10.2016

Podpis vedoucího práce: ..... **Datum:** 10.10.2016