FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání bakalářské práce

Název:	ETCS - Tenký klient pro cloud
Student:	Vojtěch Novák
Vedoucí:	Ing. Jiří Chludil
Studijní program:	Informatika
Obor / specializace:	Počítačové inženýrství
Katedra:	Katedra číslicového návrhu
Platnost zadání:	do konce letního semestru 2024/2025

Pokyny pro vypracování

V dopravním sále na Fakultě strojní ČVUT se budují dvě stanice pro strojvedoucího určené pro ETCS simulátor. Simulátor ETCS jako takový sestává z několika komponent, které spolu navzájem komunikují. Cílem práce je prozkoumat možnosti cloud computing v rozsahu ovládání simulátoru a kde by mohl pomoci a následně do ovládacího pultu statického simulátoru nasadit tenkého klienta. Vizualizace scény simulace bude řešena v jiné práci.

1. Analyzujte možnosti užití cloudu k ovládání simulátoru.
2. Analyzujte hardwarové varianty tenkého klienta.
3. Analyzujte stav projektu ETCS a určete možnosti jeho plné integrace do cloudu.
4. Analyzujte odolnost navrženého systému proti běžným typům útokům (po síťové i hardwarové stránce)
5. Implementujte potřebný změny v simulátoru pro zapojení tenkého klienta a integraci do cloudu.
6. Výsledné zařízení podrobte testům: Rychlost, responzivnost, stabilita.
7. Analyzujte další možnosti využití této technologie v jiných projektech.

Pozn: Pro testování bude použit privátní cloud, jehož chování půjde programově i hardwarově přizpůsobit tomuto projektu.

Elektronicky schválil/a prof. Ing. Hana Kubátová, CSc. dne 14. září 2023 v Praze.

Bakalářská práce

ETCS - TENKÝ KLIENT PRO CLOUD

Vojtěch Novák

Fakulta informačních technologií
Katedra číslicového návrhu
Vedoucí: Ing. Jiří Chludil
11. ledna 2024

Obsah

Seznam obrázků

Tímto děkuji vedoucímu práce Ing. Jiří Chludilovi za veškerou pomoc a vedení v mém snažení. Dále děkuji týmu ETCS za FIT ČVUT pod vedením Ing. Jana Matouška za úžasnou práci na projektu, bez které by celý projekt byl neproveditelný a doc. Ing. Marinu Lesovi, Ph.D. za poskytnuté konzultace a přístup k simulátoru.
Poděkování patří též rodině a přátelům za podporu a trpělivost.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací. Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů, zejména skutečnost, že České vysoké učení technické v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 citovaného zákona.

V Praze dne 11. ledna 2024

Abstrakt

Tato baklářská práce otevírá otázku alternativního řešení ovládání simulátoru ETCS připraveného Fakultou dopravní ČVUT v Praze. Jedná se o simulátor jednotného Evropského zabezpečovacího systému pro vlakovou dopravu. Dosavadním řešením pro ovládání tohoto simulátoru je použití PC, jehož potenciál není plně využit. Práce nejprve analyzuje a rozebírá technologii cloudů se zaměřením na architekturu bigcloudu, poskytnutou pro vývoj. Dále uvažuje nutnosti klienta pro obsluhu ovládání ETCS simulátoru. V návrhu pokládá teoretický základ implementaci a instalaci jak klienta, tak nového hardwaru do simulátoru. V poslední části implementujeme řešení vyvíjené na Fakultě informačních technologií ČVUT v Praze do prostředí cloudu a připravujeme jeho integraci do projektu. Následně připravený software a potřebné nástroje instalujeme na klienta Raspberry Pi 4 B a testuje 2 různé možnosti řešení ovládání simulátoru. Pokládáme tak základ pro nahrazení aktuálně používaného řešení s přípravou pro další vývoj.

Klíčová slova tenký klient, komunikační protokol, cloud, zabezpečení komunikace, ETCS, Železnice 4.0, Raspberry Pi, Linux, VNC, MQTT

Abstract

This bachelor thesis poses the question for alternative solution for controlling the ETCS simulator prepared by Faculty of Transportation Sciences at CTU in Prague. The ETCS simulator is for simulating single European singalling and speed control system. Existing solution for controlling mentioned simulator uses PC which potential is not fully utilized. Firstly the thesis analyses and breaks down the technology behind cloud with a focus on architecture of bigcloud which was provided for development. Then the thesis ponders the necessities of a client to be able to satisfy the needs for controling the ETCS simulator. In the design we provide the theoretical basis for implementation of software for the client and instalation of the client into the simulator. In the last part we implement the provided solutions currently under development at Faculty of Information Technology at CTU in Prague into cloud and we prepare cloud's integration into the project. Lastly we install the prepared software and necessary tools onto Raspberry Pi 4 B – chosen client – and we test 2 possible solutions for controlling the simulator. So we pose the basis for replacement of current solution prepared for future development.

Keywords thin client, communication protocol, cloud, communication security, ETCS, Railway 4.0, Raspberry Pi, Linux, VNC, MQTT

Seznam zkratek

API	APlication Interface
DMI	Driver Machine Interface
DRBD	Distributed Replicated Block Device
EC2	Elastic Compute Cloud
ETCS	European Train Control System
EVC	European Vital Computer
JRU	Juridical Recording Unit
KVM	Kernel-based Virtual Machine
LPC	Lektorské PC
MQTT	MosQuiTTo
NIST	National Institute of Standards and Technology
RBC	Radio Block Centar
RDP	Remote Desktop Protocol
RPi	Raspberry Pi
SPI	Serial Peripheral Interface
UART	Universal Asynchronous Reciever / Transmitter
VM	Virtual Machine
VNC	Virtual Network Client
WS	WebSocket
ZFS	Zettabyte File System

Kapitola 1
Úvod

1.1 Cíl práce

Cílem této práce je prozkoumat možnost rozšíření fyzického simulátoru ETCS a jeho softwarové možnosti o cloud a s ním spojenou možnost přesunutí zásadních výpočetních operací na výkonný server. Snížení hardwarových nároků na fyzický simulátor otevírá cestu pro další rozšiřování simulátoru a podporuje myšlenku Železnice 4.0[1].

První část práce se zabývá definicí pojmů, rozborem aktuálního stavu projektu ETCS simulátoru a uvedením do problematiky cloud-computingu společně s porovnáním zvolených možností a podrobným rozborem požadavků na server nutný k provozu více simulátorů.

Druhou částí práce je implementace řešící prototyp samotného klienta, který nahrazuje výpočetní hardware aktuálně používaný simulátorem. Tato část pracuje s klientem, kterého bylo nutné objednat s velkým předstihem, a bude zde primárně řešena otázka přístupu ke cloud-computingu a jeho dopadu na ovládání simulátoru a imerzi uživatele. Výsledný prototyp uvažuje nutnosti zabezpečení a předkládá návrhy jeho řešení. V aktuálním stavu upřednostňuje další vývoj před dosažením doporučené bezpečnosti.

[1] Koncept v čele s doc. Martinem Lesem[1].

Kapitola 2
Rozbor zadání

Před vlastním přístupem k analýze stavu projektu a použitých technologií zde rozeberu výše dané cíle práce. Budou tak položeny lepší základní kameny pro nadcházející analýzu se snahou přiblížit čtenáři mé vnímání a přístup k těmto cílům.

2.1 Analýza možností užití cloudu k ovládání simulátoru

Celá vize této práce stojí na nevhodném aktuálním hardwarovém řešení simulátoru ETCS na FD ČVUT a snaze o představení možností, jak stiuaci zlepšit. V souvislosti s mým zapojením do projektu ČVUT FIT vývoje softwaru simulátoru ETCS jsem byl osloven Ing. Jiřím Chludilem, který projevil zájem o otevření hardwarové otázky kolem projektu.

Vzhledem k softwarové jednoduchosti jednotlivých komponent projektu byl jako varianta navržen přístup se zapojením tenkého klienta a cloudu. Vytvořil by se tak prostor pro zjednodušení použitého hardwaru přímo v simulátoru a otevřela by se lepší možnost pro budoucí rozšiřitelnost. Původní návrh přístupu přes čistě tenkého klienta se dále rozšířil o možnost vnitřního rozdělení projektu, který se od začátku skládá ze snadno oddělitelných komponent, na část běžící na simulátoru (UI) a část výpočetně složitější – běžící na cloudu.

Analýza možností cloudu tedy obnáší variantu připojení se ke cloudu pomocí softwaru pro sdílení obrazu a promítání celého projektu na simulátoru a variantu, kdy část projektu poběží na jednodušším zařízení (ne plnohodnotném výkonném PC) a bude s cloudem komunikovat pomocí vestavěných komunikačních kanálů.

2.2 Analýza hardwarové varianty tenkého klienta

Jak bylo zmíněno, aktuální hardwarové řešení je neoptimální. Je tedy třeba zvážit možnost nahrazení aktuálně nasazeného PC jednodušším hardwarem, který bude sloužit jako protistrana připravenému cloudu. Pro správné zvolení zařízení je nutné projít, jaké požadavky na něj mohou být uvaleny a jakým způsobem by se s nimi měl zvolený stroj vypořádat.

Nejedná se pouze o požadavky na výkon, potřebný ke spuštění komponent nebo aplikace pro vzdálený přístup na cloud, ale i o požadavky na připojení periferií. Ke splnění tohoto cíle je tedy nutné navštívit zmíněný fyzický simulátor a zjistit jeho přesné nároky a specifikace, jejichž naplnění bude od nově představeného řešení očekáváno.

2.3 Analýza stavu projektu ETCS a určení možností jeho plné integrace do cloudu

Stavem projektu v tomto případě není myšlen celkový stav, ale primárně softwarová činnost týmů na FIT ČVUT. Na fakultě se jedná o aktivně se vyvíjející projekt se zapojením studentů prakticky v plném rozsahu. Je tedy brán ohled jak na funkčnost aktuálního stavu, tak na postup v jednotlivých oblastech včetně tvoření dokumentace i očekávaný budoucí vývoj.

S plnou integrací do cloudu je také spojena analýza možností cloudu použitého během práce. Testování i analýza pracuje s konkrétní architekturou a implementací cloudu a nelze tedy výsledky práce aplikovat na libovolné řešení.

2.4 Analýza odolnosti navrženého systému proti běžným typům útoků

Poslední částí analýzy jako takové je zamyšlení se nad bezpečností výsledného řešení a případným dalším postupem pro možnost produkčního využití v rozsahu, na který projekt cílí. V celé práci je tak uvažována snížená bezpečnost vzhledem k aktuální nedokončenosti projektu a zájmu na dalším vývoji i budoucí nutné zvýšení bezpečnosti při dokončení projektu nebo alespoň dosažení stabilního stavu.

Výsledné řešení se tak snaží dosáhnout maximální bezpečnosti bez vytvoření zásadních dopadů na možnosti dalšího vývoje projektu.

2.5 Implementace potřebných změn v simulátoru pro zapojení tenkého klienta a integraci do cloudu

Návrh i implementace změn zatím pracují v čistě prototypické rovině kvůli rozpracovanosti projektu. Výsledkem tak bude prezentovatelné řešení, které bude možné nejprve předvést na Fakultě dopravní a následně se domluvit na dalším postupu, který povede ke skutečné instalaci zařízení do simulátoru a s tím spojené převedení jeho fungování do cloudové implementace.

Vzhledem k aktuálnímu stavu projektu na obou stranách tak výsledek může přinést premiéru spojení softwaru FIT s hardwarem FD a první skutečnou jízdu v simulátoru.

2.6 Podrobení výsledného zařízení testům

Cílem testů je zodpovědět otázku připravenosti technologie cloudu, nebo alespoň poskytnuté architektury, na plnou integraci do simulace v takovémto rozsahu.

Zároveň projekt nabízí i možnost řešení bez plnohodntné integrace cloudu a bude tedy tato dvě řešení porovnávat proti sobě.

2.7 Analýza další možnosti využití této technologie v jiných projektech

Jak je uvedeno výše, v několika ohledech přináší tento projekt premiéru s potenciálem otevření dalších možností v implementaci cloudu do vyvíjených projektů. Tato práce tedy zhodnotí jeho připravenost a doporučí jeho další využití.

Kapitola 3
Analýza

3.1 ETCS

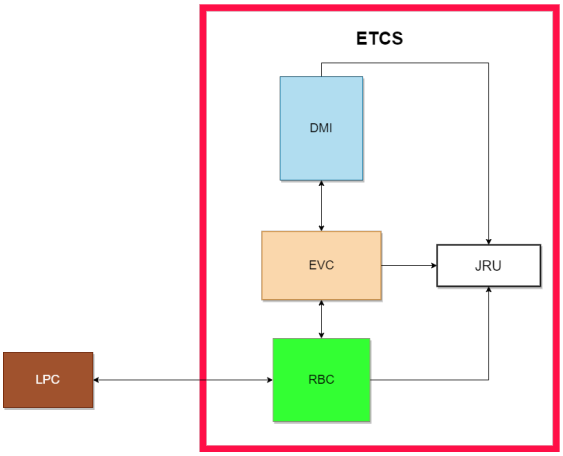
3.1.1 Představení systému

ETCS (European Train Control System) je evropský způsob zabezpečení vlakové dopravy. Jedná se o komplexní soustavu, ve které spolupracují prvky kabiny strojvedoucího s prvky implementovanými v kolejišti a centrálním "mozkem" mapujícím aktuální dění na dráze. Využitím moderních technologií dojde ke zvýšení míry autonomie jednotlivých vozidel. Cílem tohoto systému je zvýšení bezpečnosti na kolejích skrze eliminaci nebo alespoň snížení možnosti lidské chyby - primárně v důsledku únavy. Zařízení by mělo být schopno vyhodnotit kritickou situaci před katastrofou a v případě chybného či žádného zásahu od řidiče převzít kontrolu a předejít tak nehodě. V neposlední řadě pak systém hlídá, že stroj neopustí vyznačený a jemu určený úsek. Hladina autonomie a "modernosti" systému se dělí na čtyři úrovně. Základní úrovně naprosto spoléhají na člověka a prvek autonomie neexistuje nebo je naprosto zanedbatelný. Ve vyšších úrovních ETCS komunikuje svou polohu s centrálním systémem a stroj sám kontroluje oblast, ve které má povolen pohyb s možností upozornění či plného převzetí kontroly nad vlakem v případě odchýlení se způsobem, který by mohl vést k nehodě.[2]

3.1.2 Historie ETCS v rámci ČVUT

ČVUT má s ETCS již několikaletou historii. Prvotní myšlenka tohoto projektu vyšla z Fakulty dopravní, která projevila zájem o zpracování dosud v Čechách jen okrajově implementovaného dopravního systému za účelem výuky tamních studentů pro případnou kariéru v oboru. Avšak vzhledem k nedostupnosti simulátorů podporujících ETCS a nulové implementaci v provozu nebylo kde studenty se systémem seznámit a výuka tak mohla probíhat pouze teoreticky. Fakulta tedy přišla s návrhem na vytvoření vlastního simulátoru. V rámci prostor Fakulty dopravní na Albertově byla postavena zařízení simulující jak ovládací pult, tak dynamickou, pohyblivou kabinu strojvedoucího. Souběžně na Fakultě informatiky začal projekt zadaný týmům studentů, který směřuje k vytvoření softwaru pro postavené pulty a kabiny. V průběhu tří let se kolem simulátoru napsalo množství bakalářských, magisterských a doktorských prací postupně budujících pro ČVUT vlastní ETCS simulátor.

Zde zároveň vstupuje dříve zmíněný projekt železnice 4.0, jehož autorem je doc. Ing. Martin Leso, Ph.D. a který zaštiťuje ETCS projekt v rámci ČVUT. Železnice 4.0 si dává za cíl vytyčit cestu modernizace železničních tratí na našem území[1]. Tato práce má podpořit myšlenky projektu zlepšením dostupnosti simulátorů a prozkoumáním možnosti zapojení centrálního řešení



Obrázek 3.1 Zjednodušený diagram vnitřního rozdělení ETCS na komponenty[3]

v podobě cloudu na místo jednotlivých, soběstačných simulátorů a skrze to umožnit dostupný způsob zaškolování nových strojvedoucích.

3.1.3 Části ETCS

Jak je výše zmíněno, ETCS se skládá z několika částí. Těmi jsou:

EVC

EVC (European Vital Computer) je centrální logickou jednotkou každého stroje pracujícího s ETCS. Jeho smyslem je zpracovávání údajů o vlaku jako rychlost, stav na kolejišti, okolní podmínky apod. Zároveň musí tyto informace předávat strojvedoucímu. Jako centrální logická jednotka je EVC také zodpovědný za veškeré výpočty a jejich správnost. Je kritické, aby správně vyhodnocoval nutnost zásahu a v případě potřeby byl schopen převzít kontrolu nad vlakem a předešel havárii. Naopak by neměl nastat případ, kdy EVC převezme kontrolu nad strojem, aniž by před nutností zákroku předem informoval řidiče a dal mu dostatek prostoru pro reakci. Nejedná se o autopilota, pouze o systém zvýšení bezpečnosti na trati a snížení lidské chyby vlivem nepozornosti nebo únavy.[4]

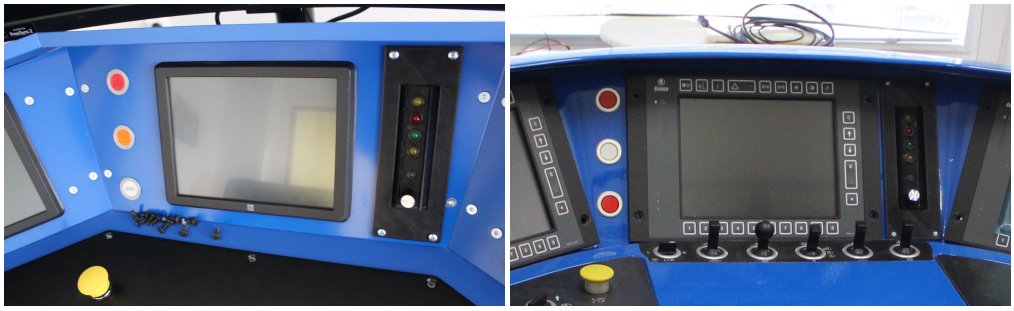
JRU

JRU (Juridical Recording Unit) je datová schránka zajišťující uchovávání informací o průběhu jízdy. Modul je stavěn tak, aby zachoval neporušené informace i v extrémních podmínkách havárie - prakticky se jedná o černou skříňku. Jeho hlavním cílem tedy je neporušeně obstát v libovolné nastalé situaci, aby bylo možné v případě havárie přezkoumat data v něm uložená při vyšetřování daného neštěstí.

DMI

DMI (Driver-Machine Interface) je hlavním komunikačním kanálem mezi zabezpečovacím systémem a strojvedoucím. Jedná se o malý displej, který v závislosti na variantě vlaku je dotykový (obr. 3.2a) nebo je orámován tlačítky (obr. 3.2b). Celý tento prvek je vsazen do řídícího pultu opatřeného pákami a tlačítky, na něž reaguje přímo EVC (DMI o jejich změnách není informován, jen prostřednictvím následků zpracovaných EVC). Displej jako takový slouží pouze pro předávání zpráv o nutnosti zásahu a informací o aktuálním a blížícím se dění v kolejišti spolu

s daty týkajícími se vlaku samotného jako rychlost, váha, počet vagónů apod. Pro snadnou orientaci je jeho rozložení jednoznačně specifikováno a napříč modely vozidel se liší co nejméně je možné. Zároveň se při jeho implementaci přihlíží ke konkrétní soupravě, do které je nainstalován, a konkrétní části, jako například nejvyšší rychlost tachometru, se mění v závislosti na kapacitách ovládaného modelu vlaku. Software ovládající displej vyhodnocuje jen minimum logiky, naopak jeho cílem je co nejrychlejší reakce na vstup strojvedoucího.[5]



(a) DMI displej dotykový [foto autora]

(b) DMI displej s tlačítky [foto autora]

Obrázek 3.2 Varianty DMI

Obrázek 3.3 Spuštěné DMI v projektu ETCS FIT ČVUT [foto autora]

Zde je také vhodné zmínit, že na DMI displeji nedochází k plynulým a rychlým změnám. Pohybuje se na něm ukazatel rychlosti na tachometru a vykreslují se brzdné křivky, které kvůli vysoké hmotnosti vlaku mají relativně vysokou reakční dobu. S tímto vědomím lze uvažovat nižší obnovovací frekvenci než u běžně používaných zařízení. Obě tyto úlevy na výkonu pomáhají při výběru řešení a hardwaru, který údaje vykresluje.

- **Realizace** Pro práci je komponenta DMI kritická. Jedná se o jediný prvek komunikující s uživatelem a díky vzájemné oddělitelnosti jednotlivých komponent se otevírá více než jedna cesta řešení připojení cloudu do systému. První možností, která přichází v úvahu, je přesunutí všech komponent na cloudový server a streamování obrazu zpět do simulátoru. Druhá varianta pak uvažuje oddělení softwarové komponenty DMI od zbytku programu a mezi simulátorem a cloudem přenášení pouze komunikace mezi komponentami.

  Fyzicky DMI displej disponuje rozlišením 640x480 pixelů a ovládá se dotykem (obr. 3.2a) nebo tlačítky umístěnými po jeho obvodu (obr. 3.2b). Oba fyzické simulátory podporují možnost výměny pultu a záleží tedy na zvoleném pultu, zda bude displej ovládán dotykem či nikoliv.[5]

Obrázek 3.4 Eurobalíza v kolejišti [foto: Armando Mendez-Villalon]

- **Oddělitelnost** Stejně jako ostatní komponenty i DMI komunikuje pomocí MQTT protkolu s ostatními softwarovými komponentami a lektorským PC, který pouze informuje o své funkčnosti skrze heartbeat. Skrze tuto implementaci je jednoduše možné přesunout komponentu na fyzicky oddělené zařízení. Bude toho využito při řešení, které nasadí komponentu na tenkého klienta[1]. Výhodou takového řešení je možnost okamžitě reagovat na uživatelský vstup přicházející přes displej (ať pomocí tlačítek nebo skrze dotykovou implementaci).

Balízy

Součástí systému ETCS, ač se nenachází uvnitř vlaku, jsou balízy. V pravidelných intervalech jsou v kolejišti rozmístěny balízy - rádiové vysílače nebo transpondéry - které slouží k určení pozice vlaku a vyznačení jeho trasy. V rámci jízdy slouží jako záchytné body, podle nichž EVC kontroluje, zdali se vlak stále vyskytuje v rámci schválené trasy. Souprava při přejetí balízy vyšle signál, na který balíza odpovídá krátkou sérií paketů v závislosti na používané úrovni ETCS. Na nižších úrovních mají balízy větší zodpovědnost, protože vysílají více dat a do jisté míry kontrolují stav dráhy; ve vyšších úrovních odesílají pouhý paket s kódem, podle kterého EVC zjišťuje polohu. Souprava pomocí nich kontroluje aktuální polohu a situaci v kolejišti, případně kontroluje průjezdy kritickými sekcemi, kde by v případě výskytu více souprav souběžně mohlo dojít k deadlocku či nehodě.[6]

RBC

Posledním klíčovým prvkem pro celkovou autonomii ETCS je RBC (Radio Block Centre). Tento prvek slouží pro komunikaci se vzdáleným pracovištěm a zajišťuje tak online získávání informací o aktuálním dění na trati. Je třeba zmínit, že tato komponenta je součástí systému až od druhé úrovně ETCS. RBC zaštiťuje klíčový prvek bezpečnosti ETCS skrze radiovou komunikaci, pomocí které se synchronizuje stav kolejiště napříč všemi soupravami, a díky tomu se předchází situacím nastalým v důsledku nečekané chyby. Další klíčovou službou RBC je zamezení interlockingu - tedy situaci, kdy se o stejnou část trasy a její využití hlásí více souprav. Zpravidla tak RBC znemožňuje přehození výhybky, přes kterou právě projíždí vlak, a kontroluje, kdy daná souprava kritickou sekci opustí, aby nebránila ostatním v plynulém provozu. Komunikace RBC je zprostředkována modulem GSM-R, tedy modulem, který i v soupravách nevybavených ETCS zabezpečuje komunikaci s dispečinkem.[7]

3.1.4 Simulátory

V době psaní této práce se ETCS v České republice aktivně používá jen velmi omezeně, ale rozšíření se připravuje skrze úpravu existující infrastruktury a nasazování balíz. Spolu s přípravou

[1] Pojem tenký klient bude podrobně vysvětlen později.

infrastruktury se řeší příprava řidičů na nový systém. Za tímto účelem se kupují a staví simulátory s hardwarem a softwarem pro imitaci skutečného vlaku používajícího ETCS v příslušné verzi – verze softwaru neodpovídá reálným úrovním ETCS, namísto toho přidává simulovatelné situace a snaží se vytvořit prostředí odpovídající reálnému nasazení ETCS do provozu. V souvislosti s těmito skutečnostmi se v prostorách Fakulty dopravní ČVUT začal takový simulátor stavět se softwarovou podporou poskytnutou Fakultou informačních technologií.

Obrázek 3.5 Statický simulátor v dopravním sále [foto autora]

V dnešní době stojí na půdě Fakulty dopravní dva simulátory včetně tří ovládacích pultů pro strojvedoucího. První ze simulátorů je prostý pult s nainstalovanými DMI displeji a širokoúhlým monitorem pro grafické zpracování světa před kabinou. Druhým je dynamická kabina posazená na třech pohyblivých ramenech, s jejichž pomocí simuluje naklánění vlaku při jízdě po kolejích. I tento simulátor je vybaven pultem - tentokrát výměnným - s nainstalovanými DMI displeji a v čele kabiny je zakomponována velkorozměrná obrazovka pro vizualizaci světa mimo kabinu. O pohyb a koordinaci dynamické kabiny se stará sestavený hardware, ale oba simulátory řídí výkonný a nákladný stolní počítač pracující v prostředí OS Windows. V obou případech se jedná o high-end hardware a cílem práce je simulátorům v tomto ohledu odlehčit a pokusit se vytvořit cloud-based alternativu, která by umožňovala postavení simulátoru bez nutnosti nákupu nákladných, výkonných počítačů.

Simulátor je ve fyzické podobě realizován Fakultou dopravní ČVUT a stojí v jejím Dopravním sále. V této době jsou v Dopravním sále fyzicky přítomny simulátory dva:

- statický3.5 - ovládací pult s monitory připravenými pro vizualizaci, u kterého bude zacvičovaný řidič sedět na židli;
- dynamický3.6 - kompletně postavená replika kabiny ukotvená na pohyblivých ramenou s velkorozměrným monitorem v místě čelního skla a vyměnitelným ovládacím pultem.

V aktuálním stavu jsou oba simulátory poháněny velmi výkonnými PC včetně grafické karty[2] schopné obsluhovat čtyři displeje s rozlišením 4k najednou. Vzhledem k faktu, že 3 z těchto displejů jsou typu DMI a mají tedy velmi nízké rozlišení (640x480) a nutná obnovovací frekvence

[2]Podle pořízených fotek se nejspíše jedná o GeForce RTX EVGA (předpokládaná řada 30 kvůli možnosti PX1).

Obrázek 3.6 Pohled na vstup kabiny do dynamického simulátoru [foto autora]

je také mnohem nižší, než u vizualizačního monitoru, není využit potenciál zvolené grafické karty. Grafická karta je tedy jedním z míst, kde je možné značně ušetřit jejím nahrazením slabší hardwarem a silnou grafickou kartu nasadit takovou, aby vyhovovala potřebám vizualizace - tedy bez nutnosti řešení dalších tří displejů. Obdobně je tomu v případě procesoru Intel Core 11. generace, který je využíván prakticky pouze vizualizací a bylo by tedy dobré zvolit takový, který vyhoví jejím potřebám.

Realizace

Softwarová část simulátoru je vyvíjena Fakultou informatiky ČVUT v rozsahu předmětu Softwarový týmový projekt 1 a 2 (SP1/2) týmy studentů. Před letním semestrem AR 2022/23 jednotlivé komponenty vyvíjely oddělené týmy studentů. Od tohoto se ustoupilo a dále jsou komponenty ETCS vyvíjeny společně a oddělené jsou jen moduly LPC a Vizualizace (rozdělená interně do dvou týmů). Aktuální verze míří pouze na statický simulátor s možností dalšího rozšíření pomocí hlubšího rozpracování fyzikálních modelů a grafické responzivity kvůli náklonům kabiny.

Lektorské PC

„Smyslem modulu Lektorské PC je především usnadnění uživateli (nebo případnému vývojáři) spuštění simulátoru, konfigurace modulů a monitorování průběhu jízdy. Sloužit by tedy měl jak vývojářům modulů, tak budoucímu lektorovi bez znalosti implementačních detailů a síťového protokolu, pro snadné a plynulé spuštění. Každý uživatel může mít na systém jiné požadavky, které bude potřeba v návrhu uživatelského rozhraní adresovat. Běžný uživatel (lektor) potřebuje simulátor samostatně spustit, zvolit konfigurace, scénáře a dále monitorovat průběh jízdy. Případný vývojář potřebuje přístup k logům, stavu jednotlivých modulů (například při debugování výpadku spojení) a nástroje pro úpravu konfigurací modulů.“[8]

V rámci provozování simulátorů vznikl i modul LPC sloužící jako ovládací prvek pro instruktora (lektora). Lektorské pracoviště není součástí simulátoru jako takového, nýbrž bude spuštěno na vlastním stroji a s ETCS simulátorem bude komunikovat vzdáleně pomocí protokolu MQTT.

Obrázek 3.7 Pult pro ovládání metra v kabině dynamického simulátoru [foto autora]

3.2 Cloud Computing

Cloudový server a jeho řešení, které je zde popisováno nereflektuje obecně fungování všech cloudů. Jedná se o snahu přiblížit čtenáři některé obecné principy jinak velmi široké a různorodé problematiky. V konkrétních příkladech pak popisuje řešení cloudu společností Casablanca INT, a. s., která nabídla servery na testování projektu a jejichž architektura je tedy pro práci stěžejní.

3.2.1 Představení problematiky

První zmínka o cloudu jakožto počítačové technologii se objevila už v roce 1961, tedy hluboko před vznikem internetu jako takového. Tehdy se jednalo o obecnou myšlenku veřejně dostupného počítače přirovnávaného k telefonní ústředně. S prvními výskyty internetu a jeho služeb se začala konkrétní představa cloud computingu teprve formovat. Mezi první představitele této myšlenky se řadí jak vyhledávače jako Google či Yahoo, tak e-mailové služby jako Hotmail, v pozdějších letech se pak přidaly služby pro zveřejňování obsahu jako MySpace a Youtube. Všechny tyto internetové aplikace spadají pod myšlenku cloudu, tedy sdílení počítačových řešení s veřejností. Navzdory těmto cloudovým předskokanům se termín "cloud computing" objevil až v roce 2006, kdy společnost Amazon nasadila první službu umožňující pronájem výpočetní síly EC2 - Elastic Compute Cloud. Samotná definice následovala krátce potom kdy ji NIST v roce 2009 vydala a následně 2011 přepsala do dnešní podoby.[9]

"Cloud computing je model umožňující všudypřítomný, příhodný, internetový přístup na vyžádání ke sdílenému zdroji nastavitelných výpočetních zdrojů (např. sítě, servery, aplikace, a služby), který může být rychle zajištěn a uvolněn s minimálním manažerským úsilím nebo interakcí s poskytovatelem služby. Tento cloudový model sestává z pěti základních vlastností, tří modelů služeb a čtyř modelů nasazení." (překlad autora) [10]

Základními vlastnostmi jsou: samoobslužnost, široký síťový přístup, sdružování zdrojů, rychlá elasticita, vyměřená služba. Cloud je architektura založená na principech škálovatelnosti, dostupnosti a schopnosti snadno a rychle vytvářet zálohy. Před tím, než rozebereme jednotlivé prvky funkčnosti cloudu, musíme definovat pojmy s nimi související.[10]

3.2.2 Motivace

Vzhledem ke skutečnosti, že IT je jedno z nejrychleji se rozvíjejících odvětví, je pro každého uživatele palčivou otázkou, zda jím používané zařízení ještě stačí držet krok s nároky, které na něj jsou kladeny. Při hledání odpovědi se zpravidla volí jedna ze tří jednoduchých strategií:

- adaptovat - tedy zajišťovat si dostatečně schopné zařízení v očekávání zvýšených nároků;
- dorovnávat - shánět dostatečné zařízení ve chvíli, kdy aktuálně používané požadavkům nestačí;
- držet krok - přidávat výpočetní kapacitu po kouskách vzhledem k postupně rostoucím požadavkům.

Dvě z těchto tří však vyžadují schopnost plánovat nebo předpovídat zvyšování nároků, což může být náročné a pro řadového uživatele dokonce nemožné, třetí naopak řeší problém až ve chvíli, kdy nastane, a uživatel tak omezuje vlastní schopnost fungovat v plném rozsahu. V neposlední řadě se v případě práce v IT může pracovní zařízení stát spotřebním zbožím a představovat značnou finanční zátěž. Oba tyto problémy řeší cloud velmi elegantně.

3.2.3 Virtuální stroj

Před vlastním rozebráním cloudu ve smyslu toho, co se skrývá na straně serveru, je třeba představit klíčový prvek v podobě virtuálního stroje. Ve své nejhrubší podstatě lze virtuální stroj vnímat jako pískoviště na pláži[3]: uživateli je zpřístupněna omezená část celku a v ní poskytnuta určitá volnost s tím, že je prakticky kdykoliv možné daný prostor upravit (např. zvětšit/zmenšit) dle uživatelova přání. Toto napodobování jiných systémů se odborně nazývá emulace[11].

V praxi se jedná o metodu poskytnutí části výpočetní kapacity zařízení a její uzpůsobení do systému tak, aby se poskytnutý celek dokázal vnějšímu světu představit jako plnohodnotný systém. Budiž jako příklad uvažován hostitelský stroj s nainstalovaným operačním systémem Windows. Virtuální stroj v tomto případě může být na systému spuštěn emulující Linuxový OS, například Ubuntu. Virtuální zařízení bude mít nárok na výpočetní sílu procesoru, část celkové paměti a kapacity disku. Nad těmito poskytnutými zdroji bude uživateli dána moc v rozsahu, který určí hostitel. Ve skutečnosti virtuálním strojem není myšleno fyzické zařízení, nýbrž proces běžící na tomto hostiteli, který simulovanému stroji poskytuje rezervované zdroje.[12]

Srovnání virtuálního stroje a dockeru

Později v práci se bude mluvit i o virtualizaci pomocí dockeru, který je hojně využíván projektem ETCS, a je tedy vhodné zde představit rozdíly mezi ním a virtuálním strojem používaným na cloudu. Virtuální stroj slouží k emulaci celého systému se záměrem suplování veškerých aktivit, které je možné provádět na hostitelském systému v izolovaném prostředí. Toho může být využito například při práci s potenciálně škodlivými daty nebo testování v rámci emulovaného systému.[13]

Docker poskytuje také izolaci spouštěného procesu, ale vytvořený obraz není plnohodnotný systém. Tento obraz vytvořený dockerem slouží zpravidla pro spouštění a testování specifických operací nebo procesů na místo vytváření celého prostředí, ve kterém by si dané procesy spouštěl uživatel sám. Výměnou za tuto specializaci je také mnohem tenčí a jednoduší na údržbu.[13]

3.2.4 Hypervizor

Zásadním pojmem pro pochopení cloudu je hypervizor. Jedná se o software, firmware nebo hardware zajišťující správu virtuálního stroje (nebo strojů) na straně serveru. Vytváří tak pro

[3] Jedná se analogii autora, na základě jeho vstupní představy. Nemá tedy spojitost s pojmem *sandbox*

klienta – hosta – hostitelské prostředí, ve kterém mu poskytuje nutné prostředky pro přístup k poskytovaným datům a aplikacím. Pro poskytování těchto služeb je hypervizor připojen do sítě (např. internetu), kde má být dostupný.

Ač je možné hypervizor provozovat i na domácím počítači, v komerční sféře se pohybujeme spíše na profesionálních zařízeních určených právě pro správu virtuálních prostředí. V takových případech se často jedná spíše o hyperskupinu[4] - skupinu hypervizorů - než o jediný hypervizor. Jednotlivá fyzická zařízení takové hyperskupiny se dále specializují na výpočetní a datovou část. Tato skupina se ven projevuje jako jediný systém, což umožňuje poskytovat prezentované zařízení s vysokými specifikacemi a maximalizovat jeho dostupnost díky možnosti přesouvání dat mezi fyzickými zařízeními v případě výpadků nebo údržby[14].

Výpočetní kapacita

Výpočetní kapacita jednotlivých hypervizorů odpovídá několikanásobku standardních domácích zařízení. Například mluvíme-li o paměti, může se jednat až o stovky GB RAM, které jsou členěny a rezervovány pro jednotlivé uživatele. Dále pak díky právě zmíněné virtualizaci je možné pro všechny klienty společně rezervovat více než 100 % kapacity jednotlivých hardwarových prvků. K tomu, aby bylo možné nabízet uživatelům více, než čeho je hypervizor schopen, se využívá předpokladu, že standardně uživatel nevyužívá veškerou poskytnutou kapacitu. Tato možnost je navíc podpořena algoritmy, které se snaží předvídat případné výkyvy ve využívání poskytnutých prostředků a zamezení selhání celého systému následnou migrací.

Datová kapacita

V případě datových hypervizorů se často dostáváme znovu do situace několika fyzických zařízení ve společné síti s přizpůsobeným filesystémem pro správu síťových disků a simulování výsledku jako jednolitého úložného prostoru. Casablancou INT používaným specializovaným softwarem pro tyto případy je DRBD (Distributed Replicated Block Device). Jedná se o otevřeně dostupný řadič jádra pro operační systém Linux, který zajišťuje jistou formu softwarové síťové synchronizace dat mezi několika zařízeními v reálném čase[15]. Pro zálohování je použit ZFS (Zettabyte File System), který dělí datovou část cloudu (nebo jiného systému s rozdělenou datovou částí) do nodů. Takto rozdělenou datovou část pak umí snadno zálohovat pomocí technologie snapshotů a skrze ni nahrazovat vypadlé nody v rámci vteřin. Spolu se správou aktuálně používaných dat zajišťuje "snaphotování" také pravidelnou zálohu systému[16]. Zásadní výhodou použití jiného fyzického zařízení je možnost geografické zálohy, díky komunikaci přes internet je tedy možné, aby uživatelova data byla na jiném místě, než na které se sám připojuje jako ke svému virtuálnímu PC.

Virtualizace a bezpečnost

Ke správě virtuálního zařízení pro připojeného uživatele se opět používá specializovaný software. V tomto případě patří mezi oblíbené nástroje například KVM (Kernel-based Virtual Machine) - volně dostupný open source software postavený nad linuxovými operačními systémy se zabudovaným VNC (Virtual Network Client) pro zprostředkování komunikace uživatele s virtuálním strojem. Mezi jednotlivými virtuálními zařízeními, provozovanými na jednom hypervizoru, dochází pro zajištění bezpečnosti k virtualizaci síťových karet a sítí. To znamená, že ač je zařízení, ke kterému se uživatel připojil, fyzicky součástí hypervizoru, komunikuje s ním prostřednictvím síťové komunikace tak, jakoby se jednalo o dvě různá, fyzicky oddělená, zařízení.[12]

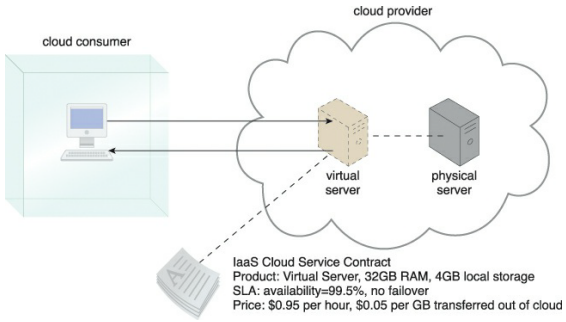
[4] Termín používaný společností Casablanca INT pro označení počítačového clusteru[14] skládajícího se z hypervizorů.

3.2.5 Modely služby

Pro práci na cloudu je třeba se seznámit se 3 vrstvami (typy), které definují, co cloud klientovi poskytuje. Jednotlivé vrstvy určují v jakém rozsahu bude klient mít možnost s cloudem pracovat a co všechno mu server poskytne. Vrstvy v tomto slova smyslu je třeba nevnímat jako vzájemně exkluzivní, naopak je v rámci jednoho cloudového řešení často propojeno více vrstev - layerů. V konkrétním případě použití jako bude tento se jedná o Software as a Service (streaming DMI na tenkého klienta) nebo Infrastructure as a Service (poskytnutní výpočentí kapacity pro obsloužení komponent mimo DMI). Model Platform as a Service tedy bude jen pro úplnost zmíněn, ale hlubší rozbor by zde nedával smysl.

IaaS - Infrastructure as a Service

Nejprimitivnější vrstvou je infrastruktura. Klientovi volícímu tento rozsah budou poskytnuty pouze prostředky. Klient tedy dostává přístup k disku, výpočetní síle nebo dokonce virtualizovaným routerům nebo switchům. Je zde důležité si uvědomit, že klient nedostává přístup k hardwaru samotnému, nýbrž pouze k virtualizovaným prostředkům simulujícím žádaný hardware. Zároveň se ale jedná o nejvolnější formu, na které si uživatel spravuje "syrové" prostředí a má tak největší povolenou míru kontroly nad poskytovanými zdroji. To s sebou samozřejmě přináší i zodpovědnost nad veškerým nastavením a využitím prostředků. Velmi jednoduchým příkladem užití IaaS je Dropbox, který každému uživateli poskytuje omezený prostor pro ukládání svých dat a přístup k nim odkudkoli z internetu s možností si úložný prostor na požádání rozšířit za příslušný poplatek.[17]



Obrázek 3.8 Příklad IaaS modelu[9] (str. 79)

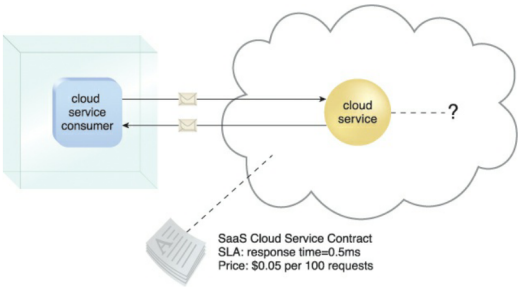
PaaS - Platform as a Service

Druhá vrstva poskytuje uživateli prostředky k vývoji aplikací. Jistým typem PaaS je libovolný software pro simulaci operačního systému na vlastním počítači jako například WSL2, který v rámci OS Windows simuluje linuxový operační systém.[18]

SaaS - Software as a Service

Poslední vrstvou cloudu je SaaS, který poskytuje konkrétní aplikaci. Na rozdíl od předchozích vrstev, tato má smysl téměř bez výjimek pouze na internetu. Na něm se zpravidla vyskytuje jako webová aplikace, je tedy dostupný z většiny zařízení a není vázaný na platformu. Kromě webových aplikací se SaaS vyskytuje v podobě dedikovaných tenkých klientů, kteří se mohou na první pohled tvářit jako běžná aplikace a standardní uživatel tak může být zmaten, když mu například Adobe Acrobat přestane správně fungovat ve chvíli, kdy nemá přístup k internetu. Zásadní výhodou tohoto modelu oproti předchozím zmíněným je prakticky nulová potřeba údržby

ze strany klienta: aplikace, kterou spouští, je plně pod kontrolou správce a ten zajišťuje instalaci potřebných updatů a opravu chyb, stejně jako dohlíží na rozložení zátěže. Na druhou stranu může být obtížné používanou aplikaci modifikovat nad rámec možností poskytnutých správcem.[19]



Obrázek 3.9 Příklad SaaS modelu[9] (str. 81)

3.2.6 Principy cloudu

Jak je výše zmíněno, hlavními vlastnostmi cloudu jsou škálovatelnost, dostupnost a snadná zálohovatelnost. Tyto stojí na architektuře cloudu jako takového a pro jejich pochopení bylo třeba si tuto architekturu přiblížit, aby na ní bylo možné ukázat, jakým způsobem se těchto principů dosahuje.

3.2.6.1 Škálovatelnost

Podstatnou věcí nutnou pro fungování cloudu jsou prostředky poskytnuté klientovi. Výše je stručně popsán hypervizor a jeho fungování, na základě tohoto konceptu je možné klientovi nabídnout základní balíček služeb s možností dalšího rozšíření. Díky virtualizaci stroje, na který klient přistupuje, a faktu, že tento stroj nijak nereflektuje možnosti a kapacity hardwaru, na kterém je simulován, je možné v rámci jednoho hypervizoru "nafukovat" pole působnosti jednoho klienta nad hranice zpočátku nabízené, s minimálním zásahem do fungování stroje jako takového a práce dalších klientů. Zároveň je skrze virtualizaci snadno možné zmenšit poskytnutý rozsah v případě, že klient už původní rozsah nepotřebuje nebo si jej aktuálně nemůže dovolit. Další výhodou virtualizace je možné skrze specializovaný hardware měnit fyzické parametry hypervizoru nebo díky síťovým diskům držícím data všech klientů přesunout jednoho či více klientů na jiný hypervizor a odlehčit tak původnímu hypervizoru.

Škálování jako takové se vnitřně dělí na dva typy: horizontální a vertikální. Každý z nich přistupuje ke změnám požadavků trochu jinak. Běžnějším z nich je horizontální škálování, které změnu požadavků klienta řeší přidáním nebo ubíráním výpočetních prostředků stejného typu. Zásadní výhodou je možnost nasazení těchto změn v reálném čase bez limitací poskytnutým hardwarem. Vertikální škálování místo toho nahrazuje výpočetní zdroj jiným s vyšší nebo nižší kapacitou. Ačkoliv je toto běžně možné v reálném čase, mohou nastat situace, kdy je nutné systém restartovat k aplikování provedených změn.[9]

3.2.6.2 Dostupnost a spolehlivost služby

Dostupnost ve spojitosti s cloudem je zásadní ve dvou významech. Jedním z nich je samozřejmá dostupnost pro velké množství klientů všude po světě. Tuto stránku snadno zajišťuje internet a používání webových aplikací, u kterých téměř nezáleží na zařízení či operačním systému, ze kterého uživatel přistupuje. I v případě nutnosti instalace klienta na vlastní zařízení je široká škála

možností, ze kterých lze vybírat. Nicméně důležitější význam dostupnosti ve spojitosti s cloudem lze vnímat jako neustálý, nepřerušovaný provoz. Od cloudu se v principu očekává, že umožní klientovi pracovat, kdykoli jej napadne, a jeho práci nebude podstatnějším způsobem narušovat ani případná potřeba změnit parametry virtuálního stroje, na kterém se pohybuje. K tomu je architektura cloudu uzpůsobena skrze používání hypervizorů, ve kterých je možné parametry virtuálního prostředí měnit za běhu. V případě nevyhnutelné nutnosti upgradu konkrétního hypervizoru jsou klienti přesouváni na jiný hypervizor. Díky síťovým diskům, ke kterým je možné snadno přistoupit z libovolného hypervizoru, se i tato operace provádí s maximálně dočasným výpadkem či omezením služeb. Nakonec je architektura datových hypervizorů rozdělena na jednotlivé nody a pomocí snapshotů je možná i obnova vypadlého nodu prakticky v reálném čase.[9][20]

3.2.6.3 Zálohování

Zálohování hraje kritickou roli ve správném fungování cloudu. Navíc je nutné, aby proces zálohování byl co možná nejrychlejší. Základním kamenem pro cloudové řešení zálohování Casablancy INT je technologie snapshotů umožňující velmi rychlé ukládání aktuálního stavu disku. Jedním z hlavních principů snapshotů je read-only přístup k datům označených pro snapshot. Samotný snapshot však data nekopíruje, pouze drží odkazy na příslušná místa v paměti. Nezabírá tak téměř žádné místo navíc. Vyvstává otázka, jakým způsobem tedy zálohuje kritická data. Zde vstupuje druhý, neméně důležitý mechanismus - copy-on-write - díky němu se snapshot postará o vytvoření kopie v případě, že by mělo dojít k jakékoliv změně zálohovaných dat. Je tedy dobré si uvědomit, že snapshot jako takový není, a nemá být, zálohou, nýbrž zálohovacím systémem či mechanismem. Díky kombinaci copy-on-write a zrcadlení původních dat je tedy možné pomocí snapshotů přistoupit k původním datům za velmi krátký čas a to bez ohledu na způsob, jakým byla data změněna a v jakém rozsahu.[20]

3.2.7 Server

Fyzicky

Zpravidla se o cloudu mluví ve spojitosti s internetovou službou, v takových případech je pro server dobré zvolit lokalitu tak, aby měl snadný, rychlý a stabilní přístup k internetu. Samozřejmě je možné provozovat cloud kdekoliv, nicméně servery umístěné na páteřních linkách internetu budou mít logicky zásadní výhodu proti těm nacházejícím se na okrajových místech sítě s nestabilním či pomalý připojením. Dále lze o cloudu mluvit i v prostředí LAN nebo WLAN, kde k internetu připojen nemusí být vůbec nebo se nejedná o kritickou součást jeho provozu, protože většina jeho klientů se bude nacházet v blízké vzdálenosti ideálně připojená přímo. V těchto případech je mnohem zásadnější architektura sítě uvnitř objektu, ve kterém se server nachází. Zde se tak otevírá otázka konkrétní implementace a je nutná domluva s uživatelem, která varianta je pro danou realizaci výhodnější.

Zabezpečení

Zásadním faktorem, který je nutný zohlednit, je zabezpečení. Za prvé je vhodné si určit, o jak citlivá data se jedná, a vzhledem k tomu patřičně upravit nároky na zabezpečení. Vzhledem k řešení cloudu pomocí hypervizorů se jeden uživatel pohybuje na řádově několikanásobně dražším hardwaru a infrastruktuře, než jaká je jeho osobní investice. Lze očekávat, že zabezpečení ze strany uživatele bude neadekvátní používané infrastruktuře. Hypervizor navíc obsluhuje několik klientů, kde každý svěřuje správci cloudové služby potenciálně citlivé údaje, za které cloud zodpovídá. Z těchto důvodů je zabezpečení primárně v rukou serveru. Je proto nutné zajistit hypervizory tak, aby narušení zabezpečení ze strany libovolného virtuálního zařízení neposkytovalo zadní vrátka

do zbytku infrastruktury a neohrozilo tak integritu dat nepostižených klientů nebo hypervizoru samotného. Uživatel od serveru očekává jak nenarušenost dat, tak nepřerušení přístupu, ke kterému by vlivem špatně ošetřeného útoku mohlo dojít. Samozřejmě některé útoky server vyřadí z provozu, ale mělo by k takovým situacím docházet výjimečně a s co nejkratšími dobami výpadku.

Výkon

Klíčovou otázkou při správě cloudového serveru je potřebný výkon. Odpověď se samozřejmě odvíjí od požadavků aplikací, které na něm budou spouštěny, a je tak většinou nemožné vytvořit přesný odhad. Avšak v případě, že se jedná o cloud s jasnou myšlenkou, která nedává uživatelům velkou volnost, lze vytvořit odhad daleko snáze a přesněji. Dobrým příkladem je právě SaaS - platforma, která klientovi poskytuje pouze konkrétní aplikaci s předem očekávatelným rozsahem a požadovanými prostředky. Je zřejmé, že i na Google Drive je možné vytvořit nadstandardně velikou a složitou tabulku, bude se však jednat o krajní případy, které nebude složité ošetřit, ani se nebude předpokládat stejně dobrá práce a rychlá odezva nad takovým dokumentem. Ještě jednodušší problém řešíme v případě, že se jedná o aplikaci s jednoznačně danými hranicemi, například hra nebo simulace. Tehdy lze redukovat otázku výkonnosti na odhad počtu současně aktivních uživatelů.

Druhou nutností, kterou je třeba zajistit ze strany bezpečnosti je připojení. Pro uživatele existuje několik možností a je jen na něm, kterou zvolí. Často používané bývají konkrétní aplikace, u kterých ani není poznat, že se jedná o klienty. V dnešní době, kdy je připojení k internetu samozřejmostí, se z každodenně používaných aplikací (jako například Office365) stává frontend cloudových služeb[21], u jiných (jako Google Drive) je tato funkcionalita zřejmá. V těchto případech, kdy uživatel přistupuje ke cloudu skrze jejich vlastní aplikaci, je většina bezpečnosti zaručována ze strany vydavatele aplikace a z uživatelovy strany je třeba jen nastavit dostatečné heslo k počítači (Google účtu), ze kterého k aplikaci přistupuje. Existuje ale také možnost použití přímého připojení skrze použití tunelu nebo virtualizačního nástroje. Tehdy je bezpečnost zaručena šifrováním zvoleného tunelu, případně třetí stranou vydávající používanou aplikaci.

3.2.8 Klient

Kromě hypervizorů, serverů, vrstev a dalších problematik spojených primárně se serverovou stranou cloudu je třeba také myslet na klienta, který se na server připojuje, a jakým způsobem tak činí. Tato práce se bude primárně zabývat tenčími klienty, nicméně je dobré si uvědomit i možnost do cloudu přistupovat pomocí tzv. tlustého klienta. V tomto případě se „tloušťkou“ vyjadřuje rozsah schopností klienta fungovat mimo prostředí cloudu.

Tlustý

Typicky se jedná o stolní počítač nebo notebook, který disponuje plnohodnotným operačním systémem, pamětí, grafickou kartou, diskem apod. a je tedy schopen plnohodnotné práce bez nutnosti připojení se k serveru.[22]

Tenký

Zpravidla za tenkého klienta označujeme zařízení, které postrádá některé hardwarové prvky nutné pro samostatnou funkci nebo je výkonově příliš slabé na to, aby se dalo považovat za plnohodnotný počítač. Mezi běžně používané příklady se řadí Raspberry Pi, které je schopné obsluhovat běžně přpojované periferie jako myš, klávesnici a monitor, ale například místo disku používá SD kartu, na které očekává nahranou obdobu OS. Ve výsledku tak díky cloudu může uživatel užívat

zařízení jako standardní počítač bez nutnosti pořizovat si výkonný hardware. Je zřejmé, že nelze dosáhnout absolutní náhrady, ale pro kancelářské použití jsou možnosti poskytované cloudem více než dostačující.[23]

- Zero

  Vzhledem k zaměření práce je dobré zmínit i existenci tzv. „Zero Client“, který bývá většinou vnímán pouze jako speciální případ tenkého. Zero klient je ještě „tenčí“ než výše zmiňovaný a není schopen prakticky žádné samostatné funkcionality. Práce se zero klientem je redukována na streaming vstupů a výstupů mezi serverem a uživatelem. Výhodou takového zařízení je samozřejmě další úleva z ceny za hardware.[23]

V případech použití tohoto jednoduššího hardwaru se do otázky bezpečnosti přidává i funkčnost samotného klienta. Na rozdíl od tlustého klienta, jehož zabezpečení je zaručováno používaným OS, tenký klient si luxus rozsáhlého a bezpečného OS nemůže kvůli slabšímu výkonu dovolit. Tyto jednodušší stroje disponují zpravidla pouze softwarem, který poskytuje strana serveru. Vzhledem ke své velikosti je tak nejjednodušší tenké klienty chránit fyzicky omezením přístupu.

3.2.8.1 Sdílení obrazu

Posledním dílem nutným pro zprovoznění tenkého klienta je zvolení aplikace zaštiťující komunikaci mezi klientem a serverem. Variantou, která se díky rozšířené používanosti snadno řadí mezi favority, je VNC - tedy Virtual Network Computing. Jedná se o systém zaštiťující sdílení obrazu používaný ke vzdálenému ovládání jiného zařízení. Zásadní výhodou tohoto řešení je jak množství variant napříč velkým množstvím operačních systémů, tak široká rozšířenost a aktivní používání; fakt, že je přímo vestavěn do Raspberry OS, je také výhodou.

Mezi alternativy se řadí například RDP - Remote Desktop Protocol - od firmy Microsoft. Stejně jako VNC umožňuje uživateli ovládání vzdáleného zařízení prostřednictvím klienta. Na rozdíl od VNC neposkytuje tak rozsáhlou podporu operačních systémů jako výše zmíněný. Zásadním rozdílem je přístup k přenášení dat. Zatímco VNC běží na principu přenášení jednotlivých pixelů, RDP přichází s vestavěným grafickým rozložením a umožňuje tak lépe komprimovat přenášená data. Rychlost přenosu je lehce ovlivněna šifrováním, které RDP na rozdíl od VNC poskytuje, ale výsledné zpoždění je stále menší než přenos celého obrazu u VNC. V otázce zabezpečení se tedy jedná o lepší možnost, která přichází s vestavěným šifrováním komunikace, které VNC nenabízí.[24][25]

3.3 Komunikace

Před rozborem hardwarového klienta jako takového je třeba představit dva způsoby komunikace, které budou v provozu využívány. Prvním z nich je websocket, který používá jedno z možných řešení streamování z cloudu přes VNC. Druhým je komunikační protokol MQTT hojně využívaný v celém ETCS projektu, který bude využit v každém případě, přinejmenším pro přenos vstupů ovládacího pultu ke komponentám.

3.3.1 Websocket

Standardní webové aplikace pracují na principu request-response, tedy na jeden dotaz právě jedna odpověď, navíc synchronně. Websocket umožňuje oboustrannou interaktivní komunikaci mezi serverem a prohlížečem[26]. Asi nejběžnějším místem použití, kde se s websockety setká naprostá většina lidí, jsou chatovací aplikace (např. Slack nebo WhatsApp Web[27]). Díky těmto vlastnostem umožňuje s jedním načtením webové stránky několik odpovědí. Bez tohoto nástroje by bylo nutné stránku opakovaně obnovovat v očekávání nového příspěvku, emailu či zprávy. Zároveň s tím umožňuje asynchronní komunikaci. Uživatel tedy nemusí čekat na zprávu od

serveru o úspěšném odeslání zprávy a místo toho může odeslat několik zpráv před tím, než je mu úspěšné odeslání první zprávy vůbec potvrzeno. Server na druhé straně může předávat prohlížeči data, aniž by přišel požadavek (request – používaný ve standardním HTTP). Mezi prohlížečem a serverem tak vzniká trvalé spojení, které nemusí být obnovováno pro předání nových dat.[27]

Díky těmto vlastnostem je možné využít webovou aplikaci založenou na websocketu ke streamování (využívají toho např. YouTube a Netflix[27]) obrazu namísto VNC klienta. Jako aplikace diváka v takovém případě poslouží libovolný prohlížeč podporující websocket API[5].

3.3.2 Protokol MQTT

Zaměřený primárně na Internet of Things, MQTT je tenký komunikační protokol fungující na principu publish/subscribe. Díky nízkým nárokům jsou jeho primárním cílem mikrořadiče, ale je používán i ve velkých aplikacích (např. Messenger[Facebook][27]).

Principiálně je jeho fungování velmi jednoduché. Brokeru MQTT se přihlásí zařízení, která chtějí dostávat zprávy. Během přihlášení také nahlásí téma, které je zajímá (subscribe). Každá odeslaná zpráva musí uvést téma, spolu s ním je předána brokeru (publish), který ji rozešle všem přihlášeným zařízením. Poslouchající aplikace je pak profiltruje podle témat, která daného přihlášeného zajímají.[28]

V případě ETCS to tedy znamená, že je třeba mít na centrálním počítači (v případě tohoto projektu cloud) spuštěný broker. Všechny komponenty se k němu přihlašují s tématy, která se jich týkají (například DMI naslouchá komunikaci mezi EVC/DMI a LPC/DMI, ostatní komponenty jej nezajímají). Aktuální řešení tuto komunikaci používá i pro předávání vstupů ovládacího pultu komponentám ETCS, primárně tedy EVC.

3.4 Nároky na klienta

Už z kraje práce bylo kvůli nedostupnosti související s dopady pandemie nutné rozhodnout se, zdali bude použito Raspberry Pi. Zbytek práce je psán s vědomím, že praktickou část implementuji na již vybraném hardwaru a výsledky tedy nemohou ovlivnit volbu pro vytvářený prototyp. Raspberry Pi bylo zvoleno pro svou oblíbenost a snadno dostupné materiály s ohledem na to, že plánovaný software hravě zvládne. Je tedy možné, že není ideální volbou pro daný projekt, ale časové možnosti nedaly dostatečný prostor pro analýzu.

Pro tento konkrétní projekt je zásadní, aby použitý tenký klient splňoval jisté specifikace:

- Výkon

  V otázce výkonu je třeba zvážit, dvě varianty: streaming a MQTT přenos. Aktuální stav projektu ETCS umožňuje pohodlné rozdělení softwaru mezi DMI a EVC komponentou. Výsledně by tak mohla celá logika komponenty DMI být implementována na tenkém klientu a na cloud odesílat informace pomocí MQTT protokolu. V případě volby streamingu by se o veškerou logiku staral cloud a tenký klient (v tomto případě už téměř zero) by pouze přehrával získaná data na displeji a do cloudu odesílal uživatelovy vstupy.

  Vzhledem k vnitřní jednoduchosti a hardwarové nenáročnosti DMI komponenty toto rozhodnutí nemá zásadní vliv na požadovaný výkon zvoleného zařízení. Jediným místem, které by bylo ovlivněno, by byla paměť a disk, které v případě streamingu drží pouze buffery pro plynulý přenos, zatímco v případě MQTT přenosu drží celou implementaci DMI a s ní spojené nároky na místo.

- Obraz

  Bez ohledu na finální interní rozdělení ETCS softwaru bude pro tenkého klienta schopnost přenosu obrazu v nízké kvalitě s důrazem na rychlost a odezvu hlavní prioritou. DMI displej

[5] `https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API#browser_compatibility`

v cíleném simulátoru navíc vyžaduje ovládání dotykem. Vhodný tenký klient tedy nemusí zpracovávat obraz s vysokým rozlišením, zato je nutné, aby umožňoval skrze zvolené připojení reakci na dotyk a odezvu s nízkou latencí.

Od tenkého klienta v tomto případě bude třeba, aby umožňoval připojení přes HDMI, DP nebo VGA a k nim USB zároveň, protože tato připojení displeje neumožňují přenos informace o dotyku. Alternativně by tenký klient mohl nabízet možnost připojení displeje přes USB-C pro přenos obrazu a dotyku zároveň. Tyto teoretické možnosti omezuje samotný simulátor, který aktuálně používá pouze HDMI, a kterému je tedy nutné se přizpůsobit.

- Připojení

  Vzhledem ke snaze ETCS simulátoru a projektu kolem něj o integraci virtuální reality ať skrze brýle či pohyblivou kabinu, rychlá odezva je nutností pro dotváření uživatelovy imerze. Pro další komerční použití mimo hranice prototypů je potřeba také zajistit stabilitu takového připojení. Pro obě potřeby jednoznačně vychází lépe připojení pomocí ethernetu. Ačkoliv Wi-Fi modul by mohl být výhodou pro specifické případy, simulátor by se nejspíše neměl vyskytnout v situaci, kdy nebude možné připojit se k internetu kabelem. Pro přenos dat potřebných k ovládání simulátoru a příslušných instrukcí postačí i nižší rychlost připojení v rozsahu několika Mb. V případě, že by výsledné řešení oddělilo modul DMI od zbytku softwaru, bude stačit i varianta pomalejších přenosů ethernetu přes USB adaptér. V případě varianty streamingu může při vyšších rozlišeních nastat problém při používání adaptéru se staršími verzemi USB, nicméně obraz streamovaný pro standardní DMI je pouze 640x480 px. Tedy i při streamování tří nebo čtyř obrazovek s takovýmto rozlišením se nutná přenosová rychlost pohybuje do 5 Mbps a kapacity USB adaptérů takové rychlosti zvládají hravě. Závěrem tak ethernetový port u klienta je výhodou, ale není nutný.[29][30]

- Periferie

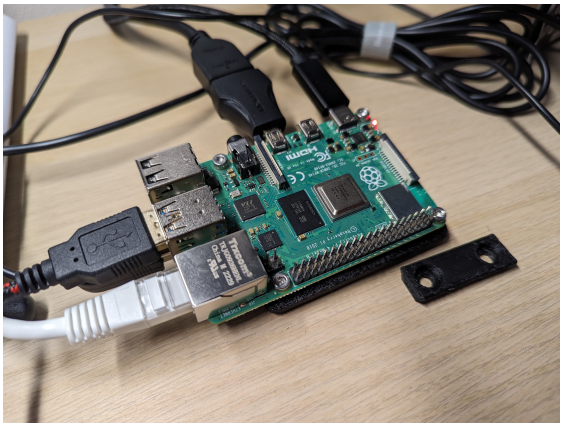
  Ovládání fyzického simulátoru se sestává z tlačítek a pák neodpovídajících standardním periferiím. Vstupy uživatele jsou interně překládány mikrořadičem před odesláním k dalšímu zpracování. Z těchto důvodů je třeba počítat s tenkým klientem umožňujícím připojení speciálního hardwaru přes USB. Aktuální stav simulátoru odesílá uživatelské vstupy přes MQTT podobně, jako je řešena komunikace v rámci komponent.

- Rozměry

  Menší velikost samozřejmě může přinášet v budoucnu jisté výhody, avšak statický i dynamický simulátor jsou v aktuálním stavu robustní zařízení a oba jsou schopny pojmout celé PC. V tomto případě na rozměrech klienta tady nezáleží a není ani znám plán na sestavení nějaké cestovní varianty simulátoru, která by mohla z menšího zařízení těžit.

3.4.1 Raspberry Pi

Zásadní výhodou je bezkonkurenční nadšení komunity kolem tohoto mikropočítače. Díky prakticky neomezené zásobě open-source projektů pro tuto platformu je jednoduché se pro vlastní implementaci inspirovat. Spolu s rozličnými projekty je bezedná i zásoba dostupných návodů umožňující snadný a plynulý start s vlastním projektem. Dalším bodem je škálovatelnost a cena. Pro tento projekt bude použito Raspberry Pi 4 B se 2 GB RAM. Podle předběžných odhadů zajišťuje dostatečný výkon pro spuštění a ovládání. V případě vyhodnocení responzivity řešení jako nedostatečné je už nyní na trhu několik výkonnějších variant, které bude možné v následujících iteracích použít. Naopak, pokud se prototyp ukáže jako zbytečně málo využívaný, je prostor i pro downgrade na slabší modely jak v řadě RPi 4, tak ve starších variantách RPi 3, které jsou i dnes hojně využívány a podporovány. Dvojsečným mečem v případě použití této platformy je čistý list, se kterým Raspberry přichází. Umožňuje tak postavit systém opravdu na



Obrázek 3.10 Raspberry připojené, připravené k obsluze [foto autora]

míru potřebám ETCS, na druhou stranu bude vyžadovat start od absolutní nuly. Alternativně je možné použít veřejně dostupného Raspberry OS na bázi Debianu.

3.4.1.1 Zabezpečení RPi

Raspberry Pi 4 B už v základní variantě přichází s několika možnostmi přístupu, které je třeba zohlednit z hlediska zabezpečení.

- **Wi-Fi**

  Obecně dobře známou formou přístupu, kterou Raspberry Pi 4 nabízí, je modul Wi-Fi (wireless fidelity), který poskytuje potenciálnímu útočníkovi nejsnažší bod přístupu, pro jehož využití není třeba se zařízením ani sdílet místnost. Vzhledem k potřebám projektu je zdaleka nejjednodušší celý tento modul odstavit mimo provoz a používat pouze připojení pomocí ethernetového kabelu.

- **Ethernet**

  Dalším přístupovým bodem, který podobně jako Wi-Fi souvisí s rozsáhlejší sítí, je ethernet. Zařízení použité v tomto projektu však neočekává žádné vnější vstupy a jediný jeho komunikační kanál tak bude veden skrze VPN tunel, který zajistí potřebné zabezpečení bez nutnosti rozsáhlých akcí v jeho softwaru.

- **USB a seriová linka**

  Přípravek disponuje několika dalšími přístupovými body v podobě USB portů a pinů pro sériovou komunikaci UART a SPI, které v případě fyzického přístupu poskytují útočníkovi možnost systém napadnout. Nejjednodušeji dosažitelným řešením možnosti tohoto přístupu je jeho fyzické oddělení do uzamykatelné schránky, ze které povedou pouze nutné periferie, a to sice způsobem, který neumožní jejich odpojení.

Jako takové je softwarově chráněno uživatelskými přístupovými údaji. V provozu však na zařízení bude spuštěno DMI a přístupové údaje tak není možné uvažovat jako formu ochrany. Zároveň v provozu bude k Raspberry Pi připojen jen displej, ethernet, napájení a ovládací panel a tedy žádná periferie umožňující zadávání vstupu a manipulaci s operačním systémem. Fyzickým oddělením zařízení se tak řeší veškerý přímý fyzický přístup a použitím ethernetu pro připojení namísto Wi-Fi je eliminován i bezdrátový přístup.

3.4.1.2 Zabezpečení komunikace

Druhou stránkou zabezpečení systému je šifrování komunikace mezi Raspberry a cloudem. Poskytnutý cloud pro připojení VNC klienta vyžaduje nejprve připojení pomocí VPN tunelu. Ten okamžitě po přihlášení uživatele vytváří bezpečnou šifrovanou komunikaci mezi oběma aktéry a zajišťuje tak nečitelnost předávaných zpráv třetí osobou.[31]

Existuje několik implementací VPN. V této práci jsou použity konkrétně dva: PPTP a OpenVPN.

- Point-to-Point Tunneling Protocol (PPTP)

  Jedná se o starý, překonaný protokol vyvinutý v 90. letech minulého století firmou Microsoft. Pomocí jednoduché konfigurace – zadáním jména a hesla – vyváří mezi klientem a serverem. Jeho zásadní nevýhodou je použití zranitelného autentikačního protokolu MSCHAP-v2, který byl prolomen.[31]

- OpenVPN

  Open-sourcová implementace VPN z roku 2001 poskytuje volný přístup pro vývojáře, kteří se mohou podílet na projektu, díky čemuž si získala silnou základnu a je aktivně rozvíjena. Pro šifrování používá AES-256, který je obecně přijímán jako standard a nabízí další zvýšení bezpečnosti použitím například 3DES.[31]

3.5 Funkční a nefunkční požadavky

Před samotnou implementací je nutné si rozvrhnout, jaké požadavky na výsledného klienta a jeho nasazení jsou. Funkční požadavky se budou týkat primárně – ale ne výhradně – spojení již existujícího simulátoru s vybraným klientem a jeho rozhraními; nefunkční se pak zaměří na nároky kladené na klienta pro další možný rozvoj. Požadavky jsou dále ohodnoceny prioritou v rozsahu: „nutné“/„optimální“/„pokud bude možné“ podle jejich důležitosti.

3.5.1 Funkční požadavky

F1: Instalace do simulátoru: Zvolený tenký klient musí být nainstalován - nasazen a připojen - do fyzického simulátoru stojícího v Dopravním sále FD ČVUT.

F1.1: Připojení DMI displeje: Jedním z řídících prvků simulátorů je DMI displej s rozměry 480x640 px s možností ovládání dotykem nebo tlačítky po jeho obvodu. Klient tedy umožní připojení takového displeje s případnou nutností dalšího vstupu pro komunikaci.
Priorita: nutné

F1.2: Připojení ovládacího pultu: Simulátory dále disponují pultem kopírujícím pult v existující vlakové kabině. Klient umožní připojení tohoto pultu.
Priorita: nutné

F1.3: Připojení k internetu: Klienta bude možné připojit k internetu tak, aby byl schopen komunikovat s cloudovým serverem, na kterém poběží většina výpočtů.
Priorita: nutné

F1.4: Připojení myši a klávesnice: K nasazenému klientu bude možné připojit myš a klávesnici pro ovládání vstupního uživatelského rozhraní.
Priorita: nutné

F1.5: Připojení dalších displejů: Simulátory disponují dalšími dvěma až třemi displeji se shodnými rozměry - 480x640 px. Klient umožní připojení i těchto dalších displejů s případnou nutnou komunikací, pokud nějaká je.
Priorita: optimální

F1.6: Připojení monitorů pro vizualizaci: Fyzický simulátor disponuje jedním 4K monitorem, dynamický pak celkem třemi, na kterých bude promítána vizualizace trati v reálném čase pro lepší imerzi. Klient by měl být schopen streamovat na tyto monitory obraz.
Priorita: pokud bude možné

F2: Uživatelské rozhraní: Rozhraní tenkého klienta nebude omezeno pouze na samotnou simulaci, ale bude poskytovat i možnosti nastavení serveru.

F2.1: Ovládání: Simulátor bude poskytovat kromě ovládání simulace jako takové také možnost ovládání pomocí myši a klávesnice.
Priorita: nutné

F2.1.1: Terminál: Kromě simulace jako takové bude možné si na simulátoru spustit terminál s jehož pomocí bude uživatel schopen simulátor nastavit.
Priorita: nutné

F2.1.2: Přepínání typu displeje: Bude možné po spuštění nastavit, zda se jedná od DMI displej ovládaný dotykem či tlačítky.
Priorita: optimální

F2.1.3: Responzivita: Uživatelský dotek či stisknutí tlačítka vyvolá reakci bez zpoždění tak, jak by reagoval počítač ve skutečné kabině.
Priorita: optimální

F2.2: Nastavení IP adresy: Uživatel bude moci nastavit přímo na simulátoru IP adresu cloudového serveru se simulací.
Priorita: nutné

F2.3: Registrace uživatele: Uživatel si bude moci na simulátoru vytvořit účet ke vstupu do simulací.
Priorita: nutné

F2.4: Přihlašování: Uživatel se bude moci přihlásit k vytvořenému účtu pomocí předem zadaných přihlašovacích údajů.
Priorita: nutné

F2.5: Rozlišení: Uživatel si bude moci na simulátoru nastavit rozlišení jednotlivých displejů a monitoru podle jejich kapacit.
Priorita: pokud bude možné

F3: Testování: Ze simulátoru bude možné spustit testy kontrolující integritu a funkčnost simulátoru a jeho jednotlivých částí včetně simulace samotné.

F3.1: Demo: Na simulátoru bude možné spustit demo simulace ve které se otestují prvky nutné pro běh skutečné simulace. Výsledky tohoto testu budou následně poskytnuty uživateli k nahlédnutí a zhodnocení.
Priorita: nutné

F3.2: Připojení: Klient bude schopen otestovat připojení k serveru a vypsat hodnoty jako rychlost, ping a latence.
Priorita: optimální

F3.3: Framerate: Uživatel bude mít možnost si zapnout zobrazování počtu snímků za vteřinu na hlavním (DMI) displeji.
Priorita: pokud bude možné

F3.4: Benchmark: Na klientovi bude možné spustit benchmark - tedy neinteraktivní demo - které otestuje kapacity používaného zařízení a připojení a v průběhu testovací simulace bude naměřené hodnoty vykreslovat na displeji.
Priorita: optimální

3.5.2 Nefunkční požadavky

N1: Připojení: Klient bude schopen se připojit k síti, na které bude očekávaný cloudový server se zbytkem komponent tak, aby bylo možné spuštění simulace.

N1.1: Rychlost: Simulace, její ovládání a vizualizace bude probíhat v reálném čase tak, aby bylo zpoždění reakce na uživatelský vstup dostatečně malé a nedošlo k narušení uživatelské imerze.
Priorita: optimální

N1.2: Stabilita: Simulace a její ovládání nebude negativně ovlivněno připojením například formou lagů, ani nebude v průběhu používání vypadávat.
Priorita: optimální

N1.3: Opětovné připojování: Klient se automaticky pokusí opakovaně navázat vypadlé připojení k serveru a až v případě opakovaného selhání nahlásí chybu.
Priorita: optimální

N2: Bezpečnost: Klient bude dostatečným způsobem chránit jak údaje uživatelů, tak citlivé informace související se simulací.

N2.1: Uživatelské údaje: Přihlašovací údaje a údaje s daným účtem spojené budou ukládány tak, aby byla zajištěna jejich dostatečná ochrana a integrita vzhledem k jejich důležitosti.
Priorita: optimální

N2.2: Komunikace: Komunikace mezi simulátorem a cloudem bude šifrována dostatečně vzhledem k citlivosti údajů předávaných mezi serverem a klientem.
Priorita: optimální

N3: Aktualizace: Klient umožní správci aktualizovat software bez nutností vyjmutí a opětovného nasazení do simulátoru.

N3.1: Stahování: Na klienta bude možné stáhnout aktualizaci prostřednictvím vestavěného terminálu.
Priorita: optimální

N3.2: Debian balíčky: Klient bude přijímat aktualizace formou Debian balíčků.
Priorita: optimální

N4: Správa DMI displeje: Existuje více variant jak tento požadavek splnit: SW komponenta/streaming, od klienta musí být schopen splnit parametry alespoň jednoho ze způsobů, není však nutné oba. Podbody budou tedy rozděleny do dvou kategorií s přidělenou prioritou v rámci kategorie a postačí uspokojit požadavky jedné z nich. Požadavky sdílené pro obě varianty jsou uvedeny první a označeny vlastní prioritou.

N4.1: Streaming: Varianta streamingu uvažuje spuštění logiky včetně DMI komponenty na straně serveru a následný přenos výstupu do simulátoru.

N4.1.1: Internetové připojení: Streamování obrazu s rozlišením 640x480 pixelů a framerate alespoň 10 snímků/s (rozlišení používané DMI displejem) potřebuje $640*480*(3*8)*10 = 70.3 Mibit/s$ rychlost stahování. Hlavním požadavkem v tomto ohledu na klienta je umožnění dostatečné rychlosti přenosu tak, aby se nestal tzv. úzkým hrdlem. Jedná se o vysoký horní odhad a dá se předpokládat použití komprese, která reálné nároky výrazně sníží.
Priorita: nutné

N4.1.2: Zpracování vstupů: Klient zaregistruje vstupy od uživatele a předá je cloudovému serveru. Reakce na tyto vstupy přijde od serveru v dostatečně krátkém čase tak, aby zpoždění jejího vykreslení nenarušilo uživatelovu imerzi.
Priorita: nutné

N4.1.3: Bezpečnost: Klient musí šifrovat uživatelské vstupy před jejich odesláním a příchozí stream naopak dešifrovat a následně zobrazit na displej.
Priorita: optimální

N4.2: SW komponenta: Druhou možností je spustit software DMI komponenty na samotném tenkém klientu a do serveru odesílat výslednou komunikaci mezi jednotlivými komponentami.

N4.2.1: Internetové připojení: Předávání informací mezi komponentami je prováděno pomocí MQTT protokolu, po kterém se posílají správy dlouhé až několik stovek bytů. Pro vykreslování změn při citlivosti minimálně 10 FPS se tedy uvažuje rychlost stahování v rámcích kibit/s. Opět je z hlediska klienta zásadní pouze to, aby se nestal pomalým článkem.
Priorita: nutné

N4.2.2: Zpracování vstupů: Klient zaregistrovaný vstup předá přímo spuštěné komponentě, nebude tak nutné s ním interagovat ani jej šifrovat pro potřeby zpracování. Následně provede nutné úkony pro jeho reakci a tu vykreslí na displej. Celý tento proces opět proběhne tak, aby uživatele nerušil v imerzi.
Priorita: nutné

N4.2.3: Bezpečnost: V tomto případě bude využit tunel nutný pro řešení použitím VNC.
Priorita: optimální

N4.3: Framerate: Displej bude obnovován rychlostí alespoň 10 snímků/s.
Priorita: optimální

N4.4: Ovládací prvky: Displej bude možné ovládat dotykem nebo postranními tlačítky dle specifikace bez nutnosti přeinstalování aplikace.
Priorita: optimální

N5: Technické požadavky: Primárně hardwarové požadavky kladené na použitého klienta.

N5.1: Periferie: Klient musí být schopen připojit DMI displej, včetně jeho ovládacích prvků, ovládací pult a ethernet. Celkem tedy musí mít 1x HDMI, 1x USB-A, 1x ethernet port.
Priorita: nutné

Dále by měl mít možnost připojení až tří dalších displejů ekvivalentních displeji DMI, jejich ovládací prvky a možnost připojení klávesnice a myši pro ovládání mimo simulaci. Další porty tedy jsou 3x HMDI a 4x USB-A.
Priorita: optimální

Kapitola 4
Návrh

Tato část se bude zabývat návrhem nasazení nového hardwaru ke statickému simulátoru a poté návrhem řešení komunikace klienta se serverem. Připojení klienta je rozděleno na tři odlišné varianty přístupu společně s posouzením výhod a nevýhod každého z nich. Veškeré testování a s ním související komplikace způsobené postupným prototypováním jsou uvedeny v implementaci. Tato kapitola se zabývá jen samotným zamyšlením se nad implementací a odůvodněním jednotlivých rozhodnutí.

4.1 Hardwarová část

Zvolený tenký klient - tedy Raspberry Pi 4 B; dále jen Raspberry - poskytuje hned do začátku ideální volbu operačního systému. Tímto nativním systémem je Raspberry Pi OS (dříve známý jako Raspbian). Jak dřívější název napovídá jedná se o unixový operační systém podporující Debian balíčky, navíc je v základu vybaven podporou pro VNC klienta i server, pomocí kterých je možné se připojit ke cloudu. Dále pak má předinstalovaný prohlížeč a poskytuje tak další možnost připojení se ke cloudu pomocí webové VNC aplikace.

Raspberry nemá běžné HDMI rozhraní, pouze jeho mikro variantu (micro HDMI). Připojení k displejům simulátoru tedy bude realizováno s pomocí redukce z HDMI na micro HDMI. Ta poskytuje možnost připojení displeje simulátoru k Raspberry aniž by vytvářela zbytečné komplikace při připojování jiných řešení (např. původního PC) související s nutností výměny kabelů. Zakoupení potřebné redukce je otázka desítek korun, a protože se jedná o pouhé převedení pinů bez přidání logické režie, neovlivní výsledné řešení.

Raspberry jako takové nedisponuje diskem, na jeho místo je používána micro SD karta. Konkrétně v případě použití Raspberry OS se doporučuje použití SD karty s kapacitou alespoň 16 GB [32]. Jelikož nároky na úložiště mimo OS se pohybují do 200 MB (zkompilovaná aplikace DMI s potřebnými assety), používání SD karty s větší kapacitou nepřinese žádnou výhodu. Pro produkční používání bude používána kvalitnější SD karta (např. Kingston SDHC 16GB Industrial), prototyp nebude vystaven tak zásadním nárokům, naopak levnější, snadno dostupnější varianta poskytne jednodušší vývoj.

Problém SD karty

Výše je zmíněn potenciál potíží při zakoupení běžné SD karty. Jedná se o obecně známý problém ve spojitosti s Raspberry. Důvodů poškozování SD karet je více, pro použití zamýšleným způsobem je však hlavním problémem nešetrné zacházení uživateli. Raspberry jako takové se zapne okamžitě při připojení do zdroje napájení. Díky nízké spotřebě tím zdrojem může pohodlně být i notebook nebo PC poskytující USB C port. Tyto dvě vlastnosti společně však

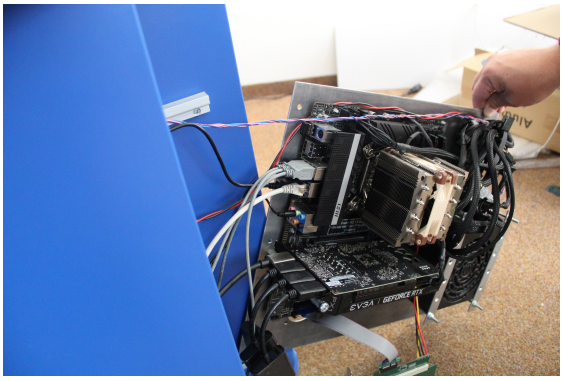
způsobují, že je Raspberry často vypínáno odpojením od zdroje namísto využití „Shutdown“ procedury. Pokud k tomuto odpojení dojde ve chvíli práce s pamětí, což je u operačního systému prakticky kdykoliv, může dojít k poškození právě používaného segmentu a v horších případech nenávratnému poškození celé karty. Tomuto však lze snadno předejít vhodným vypínacím postupem. Vedle opatrnosti při vypínání je také vhodné brát ohled na používaný zdroj a v případě používání Raspberry v místě s častými výpadky napájení před nimi zařízení ochránit.[33][34]

Tomuto problému lze v následujících iteracích předejít koupí Raspberry Pi 5, které nově disponuje PCI Express 2.0 a je tedy možné k němu připojit plnohodnotné SSD.

4.1.1 Instalace do simulátoru

Návrh řešení se bude zabývat pouze statickou variantou simulátoru. Dynamičnost simulátoru představuje potenciální problémy, jejichž řešením se tato práce nezabývá.

Aktuální stav simulátoru využívá připojení tlustého klienta v podobě plnohodnotného PC, jak je vidět na obr. 4.1.



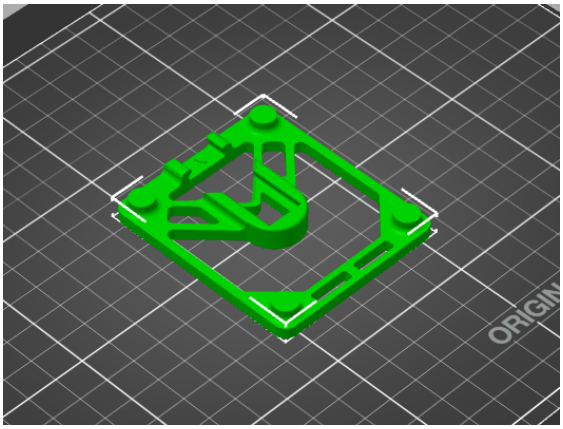
Obrázek 4.1 Připojení PC ke statickému simulátoru [foto autora]

Z obr. 4.1 je zřejmé, že se jedná o řešení improvizované a v žádném případě bezpečné. Použité zařízení nedisponuje žádnou ochranou proti fyzicky připojitelným zařízením ani snahou o omezení takového přístupu.

Navrhovaný prototyp cílí na nahrazení tohoto řešení. Připevnění zvoleného tenkého klienta pomocí jednoduchého systému (obr. 4.2) umožní snadný přístup a upravitelnost prototypu do dalších verzí a zároveň poskytne potřebné ukotvení pro bezpečnost a přehlednost kabeláže v simulátoru. Od možnosti přístupu k zařízení bude u finální verze ustoupeno a tento úchytný systém nahrazen robustnějším s dostatečným prostorem pro potřebné doplňky (např. prostor na chladící systém, redukce a huby), jejichž potřebu prototyp odhalí. V následujících iteracích bude nutné tento záchytný systém nahradit takovým, který zařízení fyzicky zabezpečí a zamezí tak interakci se zařízením nepovolaným osobám. Nepodařilo se mi najít ideálního kandidáta, ale nebude problém upravit například schránku na obrázku 4.3 a obohatit ji o možnost zámku a omezení odpojení kabelů.

4.1.2 Zapojení

Raspberry poskytuje pouze 2 HDMI porty, prototyp tedy poskytne jen možnost integrace ovládacích prvků do cloudu, bez plnohodnotné podpory všech rozhraní. V dalších iteracích mohou tyto pe-



Obrázek 4.2 Návrh modelu k uchycení Raspberry[1]

Obrázek 4.3 Model schránky pro uzavření Raspberry[2]

riferie být obsluhovány druhým Raspberry.

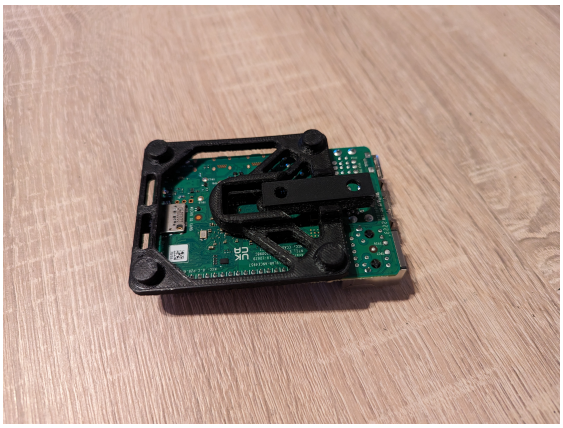
Společně s každým kabelem bude vyveden i USB kabel potřebný pro přenos ovládání displeje. Toto řešení se nijak neliší od aktuálně používaného, ale je třeba si jej uvědomit. Raspberry navíc disponuje celkem 4 porty USB-A (dva 2.0 a dva 3.0). Jeden displej tedy na Raspberry zabírá 1 HDMI a 1 USB-A port libovolné verze.

Ačkoliv Raspberry disponuje konektorem typu USB-C, podporuje pouze USB 2.0 a tedy není použitelný pro připojení displeje. Navíc zapojení nesplňuje specifikace a mohou u specifických kabelů nastat potíže s napájením[35][36]. Raspberry jako takové nastavuje napájecí požadavky na 3.0 A a 5 V, což jsou specifikace spadající až pod USB-C[37]. V případě volby napájení pomocí vizualizačního PC (nebo jiné elektroniky) bude tedy nutné zajistit dostatečně moderní USB porty (jeden či více USB-C). Alternativně je možné napájet Raspberry pomocí vlastního zdroje.

[2]Použit model `https://www.thingiverse.com/thing:3892091`, obrázek uživatele lazax `https://www.thingiverse.com/lazax`.

[2]Použit model `https://www.printables.com/cs/model/37270-raspberry-pi-wall-mount`. Snímek aplikace PrusaSlicer.

Posledním velmi významným rozhraním je ethernet. Raspberry sice disponuje možností připojení přes WiFi, ale takové připojení vystavuje projekt riziku nestability a delší odezvy. Opět se jedná o řešení, které již je v provozu a netvoří tak překážku. Potenciálně bude bezdrátové připojení k internetu možné použít během převozu a předvádění simulátoru, pokud by k tomu někdy došlo. Je tedy vhodné i s touto možností počítat a případně ji zohlednit během testování. Obecně je ale modul z důvodu bezpečnosti vypnut a jeho zapnutí není doporučeno.



Obrázek 4.4 Raspberry s nasazeným systémem pro instalaci do simulátoru [foto autora]

4.2 Softwarová část

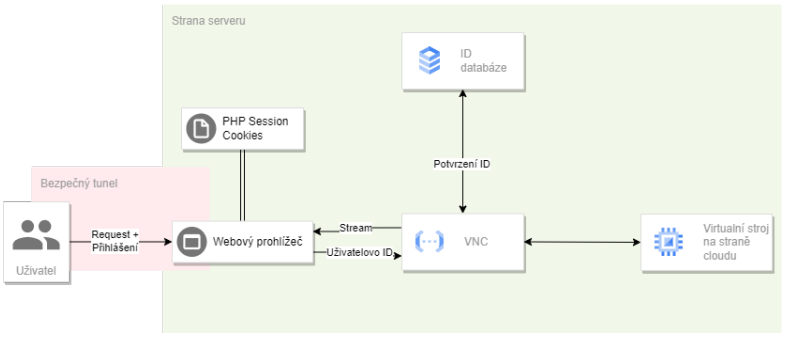
Zde se návrh dělí do několika možných variant, jejichž výsledné zhodnocení proběhne dále v práci. Všechny dále uvedené varianty počítají s podporou ze strany Casablanca INT, a. s. (dále jen jako Casablanca), která nabídla poskytnutí serverů jakožto protistrany pro klienta operujícího v simulátoru. Přesné specifikace této spolupráce budou hlouběji prozkoumány v závislosti na výsledcích práce a jejich spokojenosti na straně provozovatele simulátoru, tedy Fakulty dopravní ČVUT[3]. Softwarová část implementace se tedy opírá o rozhraní poskytnuté Casablancou a jejich cloudovými servery, stejně jako využití jejich VPN jakožto primárního zdroje zabezpečení veškeré komunikace simulátoru a serveru bez ohledu na zvolenou implementaci.

První možnost volby se naskýtá při samotném dělení softwarových komponent simulátoru. V praxi to znamená variantu, kdy se komponenta DMI nahraje přímo do Raspberry, zatímco komponenty JRU, EVC a RBC budou nahrány na straně serveru. V tomto případě veškerá logika ovládání simulátoru - ve smyslu řešení vstupu a okamžitých reakcí na něj - spadá do režie Raspberry. Cloudový server dostává od Raspberry relevantní informace skrze MQTT protokol a po vyřešení potřebné logiky na ně řádně odpovídá. Tato varianta používá Raspberry v pravém slova smyslu jako tenkého klienta.

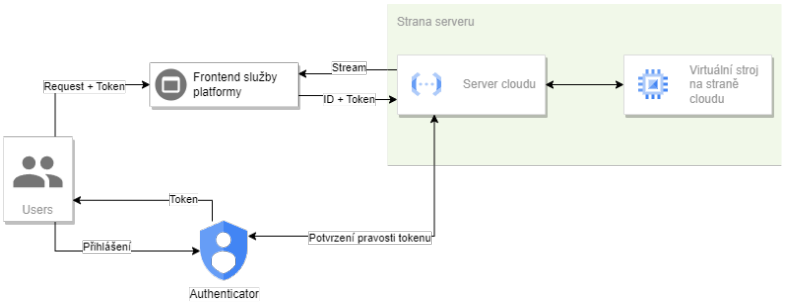
Alterantivně může Raspberry fungovat jen v režimu zero klient a spouštět VNC klienta, který bude promítat kompletní ETCS software ze strany serveru. Zero klient v tomto kontextu není přesný, protože DMI stále musí zpracovávat data kontrolního panelu a pomocí MQTT je odesílat na cloud. Vnitřně se řešení této varianty opět dělí na možnost použití instalovaného VNC klienta na Raspberry a využívání webové aplikace[4] (obr. 4.5 a 4.6 - schémata řešení používané aplikace).

V případě instalace VNC klienta do Raspberry bylo doporučeno využití jiného klienta než předinstalovaného RealVNC, kvůli nízkému počtu snímků za vteřinu a nutnosti vytváření účtu

[3] Primárním zástupcem – „objednatelem“ – tohoto projektu za FD ČVUT je doc. Ing. Martin Leso, PhD.
[4] Jedná se o aplikaci spravovanou společností Casablanca, jejíž server bude využíván jako cloud



Obrázek 4.5 Casablanca VNC solution version 1



Obrázek 4.6 Casablanca VNC solution version 2

kvůli připojení[5][38]. Pro testovací účely byl v první řadě zvolen wlvncc[6] pro jeho nativní podporu v Raspberry OS (oba postavené na waylandu). Při snaze o jeho zprovoznění se však ukázalo, že se jedná o jen málo zdokumentovaný projekt, který se navíc zdá částečně opuštěný (poslední úprava rok stará). Jako náhrada byl zvolen TigerVNC pro jeho slibovanou tenkost a na rozdíl od wlvncc dostupnou dokumentaci a zjevnou „živost“ projektu. Původní volbou nebyl z prostého důvodu a tím je wayland, respektive jeho nepoužití.

Casablanca také nabízí možnost využití webové aplikace s websocketem na místo instalace VNC klienta. Jedná se o řešení přinášející zásadní výhodu v bezpečnosti. Přesunutí komunikace VNC klienta až za hranici webového prohlížeče dává veškerá kritická data pod kontrolu serveru, na kterém webová aplikace běží, a tedy opouští nutnost vytvoření bezpečného tunelu mezi Raspberry a cloudem.

[5]Konzultace s Ing. Pavlem Podaným, systémovým inženýrem u Casablancy.
[6]`https://github.com/any1/wlvncc`

Kapitola 5

Implementace

Zde budou podrobně rozebrány všechny kroky potřebné ke zprovoznění Raspberry Pi 4 jako prostředku pro ovládání vstupů z ovládacího panelu ETCS všemi navrhovanými možnostmi a zprovoznění jim příslušné protistrany na serveru. Kapitola tak může sloužit jako návod pro zprovoznění klienta, jeho přeinstalaci a poradce při diagnostice a dalším vývoji. Zároveň se implementace nezabývá optimalizací, jejím cílem je proof of concept – prototyp ovládání simulátoru za použití cloudu – a porovnání jednotlivých řešení v jejich surovém stavu.

5.1 Instalace Raspberry

Prvním krokem po obdržení Raspberry je instalace operačního systému. Použit byl oficiální Raspberry Pi OS vydaný Raspberry Pi Foundation[1], dostupný na `https://www.raspberrypi.com/software/`. Jedná se o OS s unixovým kernelem na bázi Debianu používající protokol wayland, který je dostupný v 32 i 64 bitové variantě. Zvolen byl 64 bitový OS čistě z důvodu dosavadního vývoje na 64 bitových operačních systémech, aby se tak předešlo možným chybám způsobených touto změnou.

Samotná instalace OS probíhá pomocí instalátoru (viz odkaz výše), ve kterém uživatel zvolí místo, kam se má operační systém nainstalovat, což je v případě Raspberry Pi 4 B mikro SD karta, O vše další se již postará instalátor sám. V případě notebooku – s vestavěnou čtečkou micro SD karet – použitého k instalaci nastal problém s nečekaným odpojením SD karty před dokončením instalace, chybu se podařilo odstranit použitím externí čtečky karet[2]. Po instalaci instalátor verifikuje obsah nahrávaného zařízení, po níž je možné SD kartu odebrat a vložit do Raspberry. Při připojení Raspberry k napájení se zařízení okamžitě zapne jeho nastavení, pro tuto část je potřeba mít myš a klávesnici s USB konektorem a displej s mikro HDMI konektorem nebo jako v tomto případě redukci pro HDMI.

Průchod vstupním nastavením mimo jiné obnáší nastavení času, klávesnice (USA, CZ apod.) a uživatelského účtu, jediným pro projekt podstatným bodem je volba prohlížeče, který bude použit ve variantě s websocket klientem. Zde Raspberry nabízí možnosti Chromium a Firefox, které byly zachovány obě pro následné otestování lepší kompatibility s webovou aplikací. Nastavením výše zmíněných je instalace dokončena a otevírá se domovská obrazovka.

[1] Charitativní organizace stojící za vývojem produktů řady Raspberry Pi a souvisejícího hardware a software
[2] Použitý notebook: *Lenovo Thinkpad P14s* a čtečka: *Kingston High-Speed Media Reader FCR-HS4*

5.2 Nasazení DMI komponenty

V průběhu nasazování komponent se naskytly komplikace způsobené nedokončeností projektu a jeho zapojení do výuky na FITu. Chci zde velmi vyzdvihnout práci týmů pracujícího na ETCS, primárně skupiny studentů, která se do projektu zapojila v letním semestru 2022/23[3]. Díky jejich práci jsem na zprovozňování komponent strávil jen zlomek času, který bylo nutné investovat začátkem tohoto semestru. I tak se jedná o projekt, který se aktivně vyvíjí a nemá oficiální vydanou stabilní verzi. Zároveň se během vývoje zapojení studenti teprve učí zpracovávat dokumentaci společně s částí nástrojů nutných ke kompilaci a provozování projektů tohoto rozsahu (jmenovitě např. docker a cmake). Navzdory vynaloženému úsilí je tak dokumentace v některých místech nekompletní, decentralizovaná (a pro nezasvěcené může být zmatená) nebo (byť jen pár dnů) zastaralá a nemusí se tak podle dostupných dokumentů podařit úspěšné zprovoznění celého projektu. K mému štěstí byly členové velmi aktivní a ochotní mi v nejasnostech pomoci a problémy se mnou rychle vyřešit. Toto se týká jak samotné komponenty DMI, která vyžaduje samostatné spuštění pro její práci s grafikou, tak ostatních komponent ETCS spouštěných v dockeru společně s MQTT brokerem a emulátorem pultu simulátoru, ke kterému se přistupuje pomocí VNC.

Instalace DMI komponenty v první řadě vyžaduje kompilaci pro Raspberry. Raspberry totiž disponuje porcesorem ARM a standardně kompilované programy na zařízeních s x86 procesory nejsou na ARMu spustitelné. Ideálním řešením tedy je nastavit do git repozitáře projektu toolchain a cross-kompilaci pro ARMovksé procesory, které se nebudou spouštět při každém buildu, ale pouze v případě releasů určených pro nasazení na simulátor.

Pro první testovací účely jsem se pokusil tento proces provést na lokálním zařízení před tím, než jej nasadím na gitlab repozitář. Nejprve jsem se pokusil najít toolchain přímo určený pro Raspberry Pi, kde jsem se setkal s prvním problémem. Přes oficiální RPi fórum[4] jsem se dostal na github repozitář očividně se zabývající nástroji pro práci s Raspberry. Zde[5] jsem byl však informován, že toolchain byl odstaven a na místo něj se používá obecný toolchain pro Linux corss-kompilace na ARM procesory. Avšak ani po „úspěšné“ cross-kompilaci se aplikaci nepodařilo na Raspberry spustit. Rozhodl jsem se tedy pro účely této práce spokojit s kompilací na přípravku a toolchain pro kompilaci na gitu zřizovat až v případě úspěšných testů. Nevýhodou tohoto řešení je horší rozšiřitelnost na další zařízení (zde je vhodné zmínit, že ovládání prohlížeče na Raspberry není plynulé) a pomalá kompilace (více než 20 minut).

Pro shrnutí, aktuální prověřený postup pro nasazení a zahájení provozu DMI komponenty na Raspberry tedy je:

1. Naklonovat repozitář z gitlabu fakulty (`https://gitlab.fit.cvut.cz/etcs/dmi`[6]).
2. Aplikaci zkompilovat podle postupu uvedného v přiloženém README.
3. Spustit protistranu pro MQTT klienta a ostatní komponenty (návod a zdorojové kódy zde `https://gitlab.fit.cvut.cz/etcs/etcs_together`).
   - Alternativně je možné DMI komponentu spustit v testovacím režimu nastavením hodnoty *debug* umístěné v *config/global/app.json* na *true*. Následující kroky jsou v takovém případě zbytečné.
4. Nastavit IP adresu MQTT brokeru (zařízení protistrany).
   - V případě nutnosti změnit port z MQTT standardu 1800 na používaný.

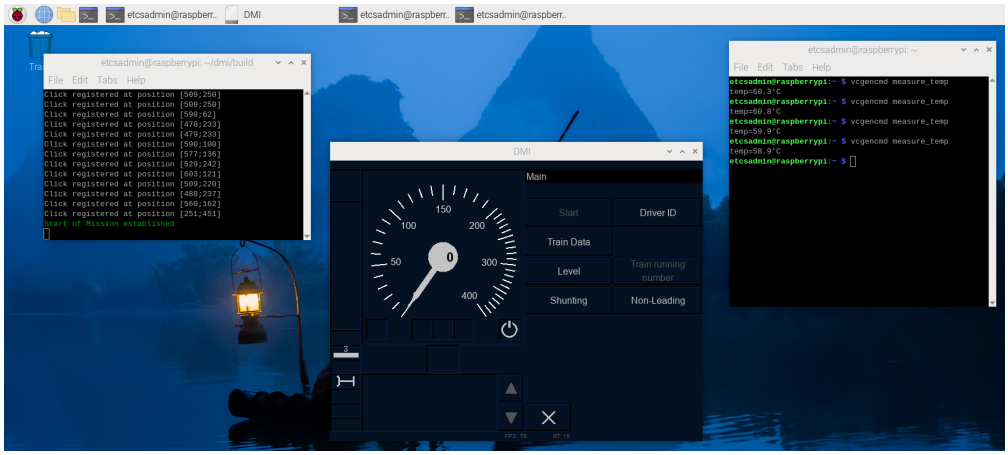
[3] Velmi pomohla i osobní účast na projektu, ačkoliv na přehlednosti a pohodlném spouštění jsem měl jen malý podíl.

[4] `https://forums.raspberrypi.com/viewtopic.php?t=232865`

[5] `https://github.com/raspberrypi/tools`

[6] V průběhu psaní se pracuje na nové iteraci, bude tedy možná nutné naklonovat aktuální repozitář umístěný zde `https://gitlab.fit.cvut.cz/etcs`

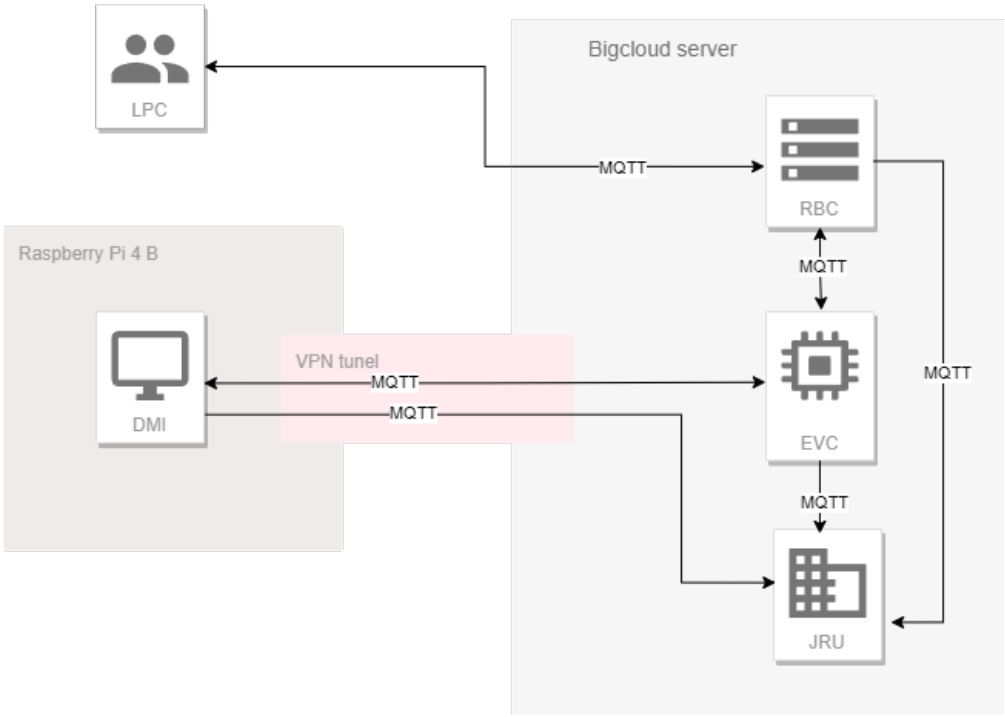
5. Spustit zkompilovanou DMI aplikaci na Raspberry.
6. Pomocí VNC se připojit k počítači, kde běží ETCS together - tím se přistoupí k emulátoru pultu.
   - Tento krok bude možné přeskočit při práci se simulátorem. Nebyla však možnost toto otestovat.
7. Odeslat zprávu o zahájení simulace - skript odesílající aktuální znění zpráv přes MQTT je přiložen v repozitáři ETCS_TOGETHER.
   - Jedná se o zprávy nutné pro zahájení simulace.
   - Všechny zprávy jsou popsány v repozitáři.
8. Pomocí ovládacího pultu nebo jeho simulátoru rozsvítit displej.
   - Pokud je vše správně nastaveno, v tento moment by se měl DMI displej rozsvítit a vyzvat uživatele k zadání vstupních údajů.



Obrázek 5.1 Spuštěné DMI na Raspberry

Ihned po spuštění nabíhá plynulá instance DMI komponenty, která při dodržení postupu vyzývá uživatele k zadání vstupních údajů (identifikátor a specifikace vlaku, používaná úroveň ETCS apod.). V rámci provozu program ukazuje rychlost vykreslování, která se v případě Raspberry pohybuje mezi 70 a 80 snímky za vteřinu a několikanásobně tak převyšuje navržené nefunkční požadavky (DMI také ukazuje dobu vykreslování posledního snímku). Vzhledem k jejímu nefinálnímu vydání se aplikace spouští přes příkazovou řádku a při přesném nedodržení postupu nemusí korektně fungovat. Při prvotních spouštěních se opakovaně stalo, že pokud se po spuštění komponent odeslaly zprávy o startu simulace před připojením se k emulovanému řídícímu pultu, komponenta EVC nezahájila komunikaci a nebylo tedy možné rozsvítit DMI displej, který na pokyn k rozsvícení kabiny očekává zprávu právě od EVC. S testovací instancí jsou spojeny i výpisy do konzole, některé velmi užitečné, jako informace o úspěšném připojení k MQTT brokeru, tak informace o každém provedeném kliknutí včetně jeho přesných souřadnic.

V aktuálním stavu je DMI naprogramováno tak, že veškeré akce s okamžitou reakcí, řeší DMI samo a není tedy možné jednoznačně určit odezvu MQTT. Během prvních experimentů bylo DMI připojeno v lokální síti pomocí Wi-Fi rozhraní k MQTT brokeru, který byl k routeru připojen pomocí ethernetového kabelu. V tomto sestavení byla reakční rychlost na veškerou interakci se simulací – tedy jak ovládání na straně pultu, tak interakce s DMI – velmi vysoká, prakticky nerozlišitelná od rychlosti při ovládání na jednom počítači. I Raspberry v aktuální konfiguraci při



Obrázek 5.2 Diagram řešení s DMI na Raspberry [diagram autora]

několikahodinovém provozu vykazovalo teploty kolem 60 °C, což ukazuje, že by výsledně mohlo být nainstalováno v simulátoru bez přídavného chlazení. Zde popisované stavy jsou patrné na snímku obrazovky Raspberry pořízené při jednom z testů (obr. 5.1).

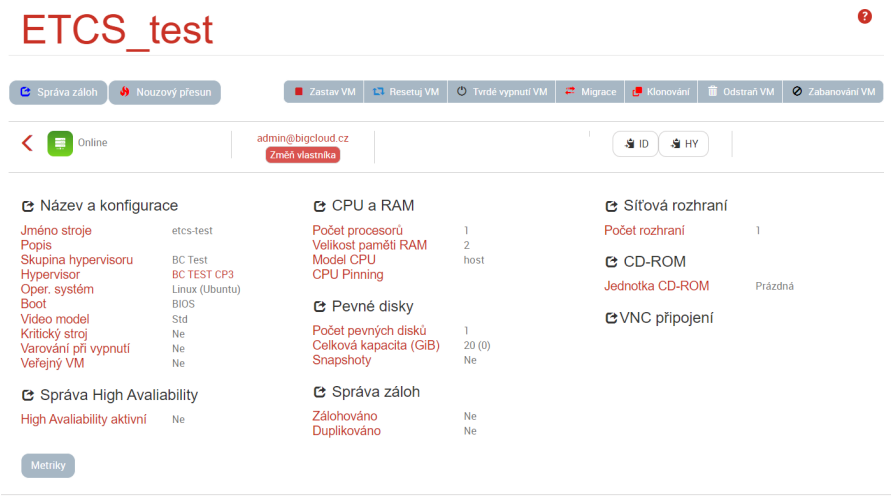
Zatím jediná pozorovaná chyba u tohoto řešení přichází se změnou jasu, kdy jas obrazovky na změnu nastavení nereaguje. Dle zpětné vazby od ETCS týmu se jedná o dosud nepozorovanou chybu jak na Windows, tak na Linux (testováno na Ubuntu[debian] - stejný základ jako Raspberry OS). Popsané pokusy byly prováděny na lokální síti s Raspberry připojeným prostřednictvím WiFi routeru, ke kterému byl ethernetovým kabelem připojen osobní počítač s Windows, na kterém běžely ostatní komponenty společně s MQTT brokerem a ovládacím pultem.

V souvislosti s instalací komponenty DMI na Raspberry bylo třeba se zamyslet i nad možností instalace ostatních komponent a nechat Raspberry provozovat celý software bez spoléhání se na cloud. Akutální stav projektu však použitím dockeru neumožňuje přímou instalaci na Raspberry a tyto snahy tak byly opuštěny.

5.3 Zprovoznění VPN

Díky funkčnímu kanálu a tedy i variantě ovládání nastává čas na vyřešení zabezpečení. Jak je zmíněno v Návrhu4 Nastavnení VPN podléhá rozhraní poskytnutému Casablancou. Ta v době psaní této práce používá OpenVPN pro vytvoření bezpečného tunelu, a pro připojení je tedy nutné si nainstalovat OpenVPN Connect[7]. Jedná se o opensourcovou[8] implementaci VPN dostupnou pro všechny všechny standardní operační systémy, včetně chytrých telefonů[9]. Je dobře dokumentovaný a práce s klientem je velmi intuitivní.

[7] Stáhnutelný např. odtud `https://openvpn.net/community-downloads/`
[8] `https://openvpn.net/source-code/`
[9] Tím je myšleno Linux, MacOS, Windows, iOS a Android.



Obrázek 5.3 Nastavení virtuálního stroje v rozhraní bigcloudu

Ačkoliv byly zástupci Casablancy velmi ochotní a pro potřeby projektu byl poskytnut účet a otevřený přístup na jejich cloudové servery, podléhala veškerá komunikace s nimi omezením způsobeným velkým vytížením a nejasném výsledku projektu. V případě úspěchu a jeho instalace do simulátorů na dopravní fakultě jsou však otevřeni spolupráci, ve které bude možné projektu přiřadit potřebnou prioritu a zdroje.

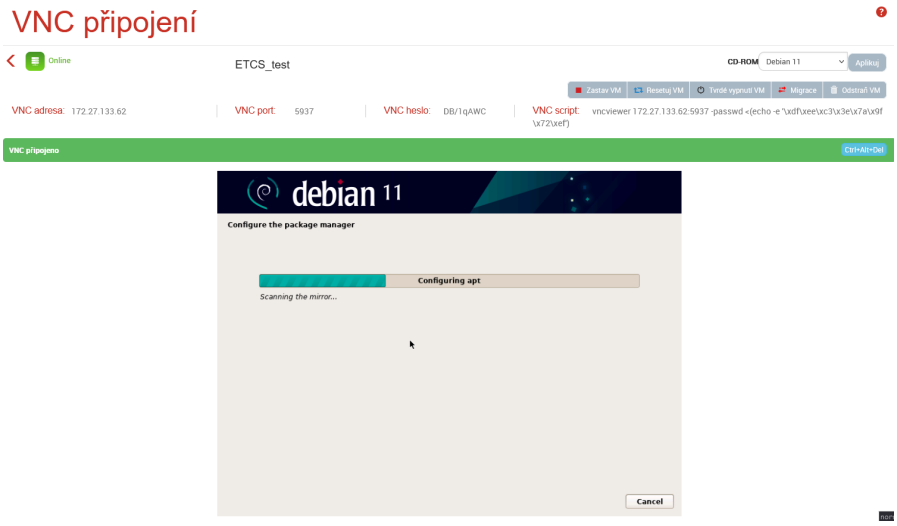
Casablanca však na serveru aktuálně neimplementuje TLS 1.2, který OpenVPN od ledna minulého roku vyžaduje[10]. Klient tedy musí být staršího data, aby bylo možné se pomocí něj k serveru připojit. Tuto informaci však OpenVPN uživateli nedá ve srozumitelné podobě, pouze nahlásí, že se připojení nezdařilo: *TLS handshake failed*. Chybu se podařilo objevit až po konzultaci, bez vnitřní znalosti používaného protokolu přesný důvod problému byl nezjistitelný.

Osobně jsem na doporučení Ing. Jiřího Chludila používal OpenVPN 2.4.6 (staženého z odkazu výše), nicméně by mělo být možné použít libovolnou verzi vydanou před rokem 2023. Po instalaci, je nutné klienta nastavit, k tomu je nutný .ovpn soubor (typ používaný pouze OpenVPN) s informacemi týkajícími se serveru, ke kterému je vytvářen tunel a definice bezpečnostních certifikátů a klíče, které uživatel poskytuje, aby byl autentizován a autorizován k přístupu na server. Na osobním počítači, odkud jsem připravoval podklady pro práci a prostředí, ke kterému se připojuje Raspberry, vše fungovalo bez sebemenších problémů a VPN se podařilo zprovoznit, na Raspberry jsem se však střetl s komplikacemi. Stažená verze OpenVPN klienta pro Debian vyžaduje dodatečné balíčky ke stažení, mezi kterými se vyskytuje i openssl, která je aktuálně ve verzi 3.0.0. Instalace klienta ovšem počítá se starší verzí a tím pádem se mi nepodařilo OpenVPN klienta nainstalovat. Pokusy o instalace postupně novějších verzí OpenVPN se střetli se stejnými problémy: Verze vydané před lednem 2023 (ke kterému skončila podpora TLS 1.0) nebylo možné nainstalovat pro jejich využívání deprekovaných funkcí openssl. Verze vydané po lednu 2023 naopak nebyly schopné navázat připojení k serverům Casablancy. Tyto problémy by měly být v budoucnu odstraněny updatem protokolu využívaného servery Casablancy.

Pro tento projekt se však podařilo domluvit se zástupci Casablancy otevření veřejného přístupového bodu pro používané Raspberry. K přímému připojení byl, po domluvě s Casablancou, použit nástroj PPTP. Jedná se o překonaný protokol používaný k vytváření tunelů mezi dvěma zařízeními[39]. Zde[11] se naskytly drobné potíže se zadáváním přihlašovacích údajů způsobené prací v příkazové řádce. Původní hlášky se špatně zadanými údaji vedli nejprve na

[10] `https://openvpn.net/security-advisory/tls-1-0-and-1-1-web-services-deprecation-notice/`

[11] Postup byl převzat z tohoto návodu `https://devtidbits.com/2013/02/19/using-a-point-to-point-tunnelling-protocol-virtual-private-network-pptp-vpn-client-on-a-raspberry-pi/`



Obrázek 5.4 Instalace virtuálního stroje na bigcloudu

stránky doporučující stažení již neexistujícího patche kvůli podpoře EAP-PEAP. Až po několika ověřeních vyplňovaných údajů, kontrole správného směrování a kontrole správnosti údajů na Windows se podařilo objevit, že heslo obsahující *$* se do nastavení nepropsalo celé a je nutné jej zapsat s použitím escape-sekvence. Po objevení tohoto pochybení se podařilo připojení k serverům úspěšně navázat. Následně byla přidána cesta pro směrování na konkrétní zařízení v rámci vnitřní sítě Casablancy a přesměrování komunikace přes VPN[12](krok bez kterého je připojení sice úspěšně navázáno, ale není možné přistupovat k vitálním prvkům cloudu. Tento poslední krok je nutné provést při každém spuštění VPN.

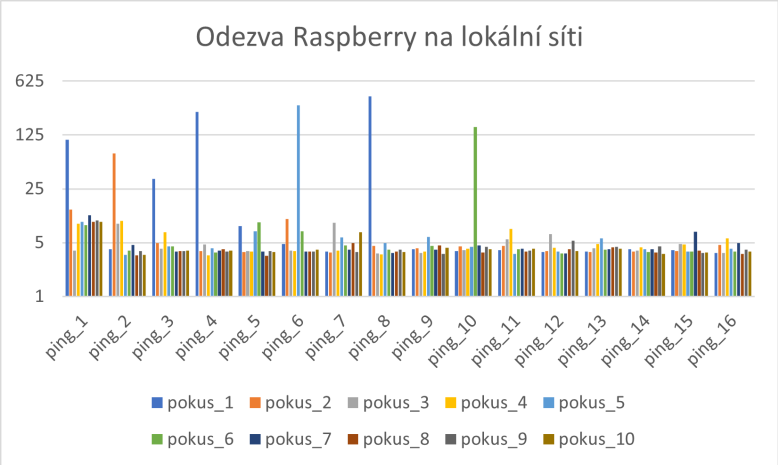
S dokončeným nastavením je možné přistoupit na *bigcloud* - cloud provozovaný Casablancou. Zde byl vytvořen nový virtuální stroj, který bude spouštět ETCS komponenty, zvolen a nainstalován byl Debian. Před dalším postupem bylo nutné zpřístupnit novému stroji internet. Jedná se o dříve popsaný problém virtuální síťové karty a jejího nastavení, ke kterému mají přístup jen interní pracovníci Casablancy, které bylo tedy nutné kontaktovat a požádat o zajištění těchto služeb. Po získání přístupu na internet se na virtuálu zprovoznila protistrana, kterou doposud simuloval osobní počítač na lokální síti[13].

5.4 Ovládání pomocí VNC

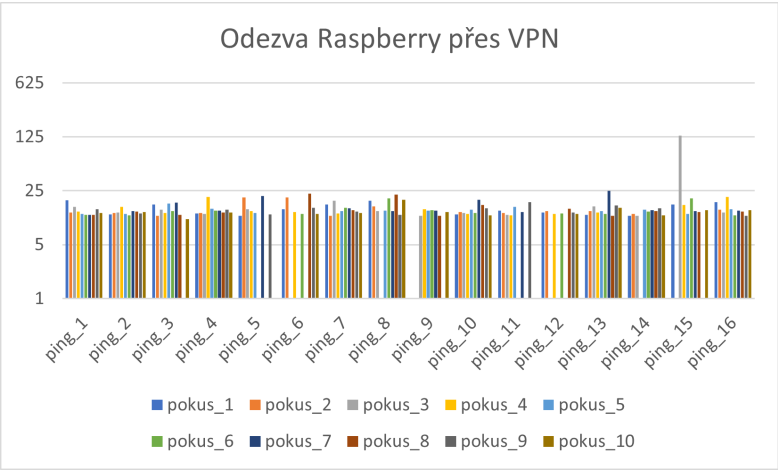
Tato část počítala s fungujícím přístupem ke cloudu, který by provozoval veškeré ETCS komponenty a dalším počítačem pro simulaci ovládacího pultu. Vzhledem k prvotním neúspěchům při zprovozňování připojení ke Casablance byl pro úvodní testy na místo jejich serverů využit osobní počítač, který slouží jak pro spuštění ETCS komponent, tak i k simulaci ovládacího pultu. Z hlediska ovládání to vytváří jistá omezení (jeden monitor pro DMI a ovládací pult zároveň) a síťový provoz neodpovídá vytyčenému cíli, ale jakožto model postačí.

Během instalace zvoleného wlvncc[14] jsem narazil na problém s knihovnou lzo a její implementací minilzo[15], kde se zdá, že se wlvncc odkazuje na verzi, která aktuálně není dostupná. Vzhledem k nedostatečné dokumentaci projektu wlvncc (a opomenutí zmínky nutnosti těchto

[12] *sudo route add -net "0.0.0.0/0"dev "ppp0"* řádek nacházející se hlouběji v odkazovaném návodu.
[13] Postup zde `https://gitlab.fit.cvut.cz/etcs/etcs_together/-/blob/master/README.md?ref_type=heads#etcs`
[14] `https://github.com/any1/wlvncc`
[15] `http://www.oberhumer.com/opensource/lzo/`



Obrázek 5.5 Graf odezvy Raspberry na lokální síti [vypracováno autorem]



Obrázek 5.6 Graf odezvy Raspberry přes VPN [vypracováno autorem]

balíčků) jsem se rozhodl použít TigerVNC. Jedná se o projekt podstatně lépe dokumentovaný, slibující tenkost a rychlost a jedinou jeho nevýhodou tak zůstává implementace pro jiné protokoly než wayland.

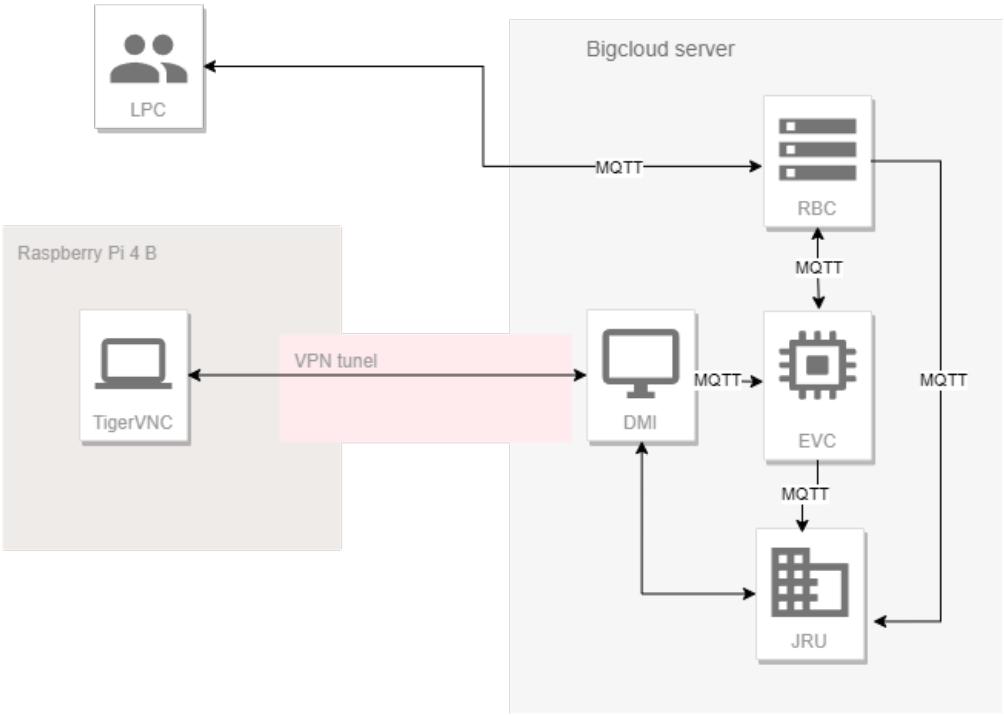
Jako protistrana byl na osobním počítači spuštěn TightVNC server, ze kterého je TigerVNC původně odštípnut. Připojení je pak velmi jednoduché: Z Raspberry je zadána adresa serveru a následně vyplněno heslo (nastavované při instalaci serveru), čímž se okamžitě otevírá okno diváka. Následně je nutné, aby server umožňoval spuštění komponent (v tomto případě i ovládacího pultu) a vše je připraveno pro simulaci.

Během testování na lokální síti se držela až na absurdní výjimky držela odezva velmi nízko a s VNC se tak pracovalo velmi pohodlně. První dva pokusy z mě nevysvětlitelných důvodů měly několik abnormalit, proto jsem testů pro lepší výslednou představu provedl více. Graf se celkem spolehlivě drží u času odezvy pod 5 ms (graf 5.5), ale je třeba poukázat na fakt, že se jedná o test prováděný na osobním počítači v době, kdy router nikdo jiný nevyužíval a pro skutečný provoz je to tedy opravdu jen velmi hrubá představa.

Po připojení přes VPN se průměrná doba odezvy zvýšila k 15-20 ms a na rozdíl od lokální sítě při přenosu nastávaly v přibližně 15 % případů výpadky paketů (graf 5.6). Výsledkem tedy celkem stabilní a rychlé připojení, které obsluhu DMI hravě zvládne.

Řešení s použitím VNC přenáší veškerou režii spojenou s udržováním aktuality spouštěné aplikace na server a odstraňuje tím i problémy spojené s kompilací pro Raspberry. S VNC je však spojeno i několik drobných problémů. První z nich je pokles počtu snímků za vteřinu. Během ovládání samotného DMI naprosto zanedbatelný, protože i s poklesem je obraz několikanásobně plynulejší než uvedený požadavek, nicméně je tento pokles citelný a může ovlivnit imerzi uživatele. S dalším drobným problémem se uživatel střetne při pokusu o nastavení jasu na DMI, které disponuje možností tlumit jas. Jas je sice úspěšně ztlumen, ale na zařízení ze kterého je vysíláno na místo DMI.

Po instalaci a zprovoznění protistrany bigcloudu se ukázalo, že akutální řešení VNC přes websocket je pro připojení se Raspberrym nepoužitelné. Oba prohlížeče – Firefox a Chromium – nepodporují práci s websocketem a tato varianta tedy musela být opuštěna.

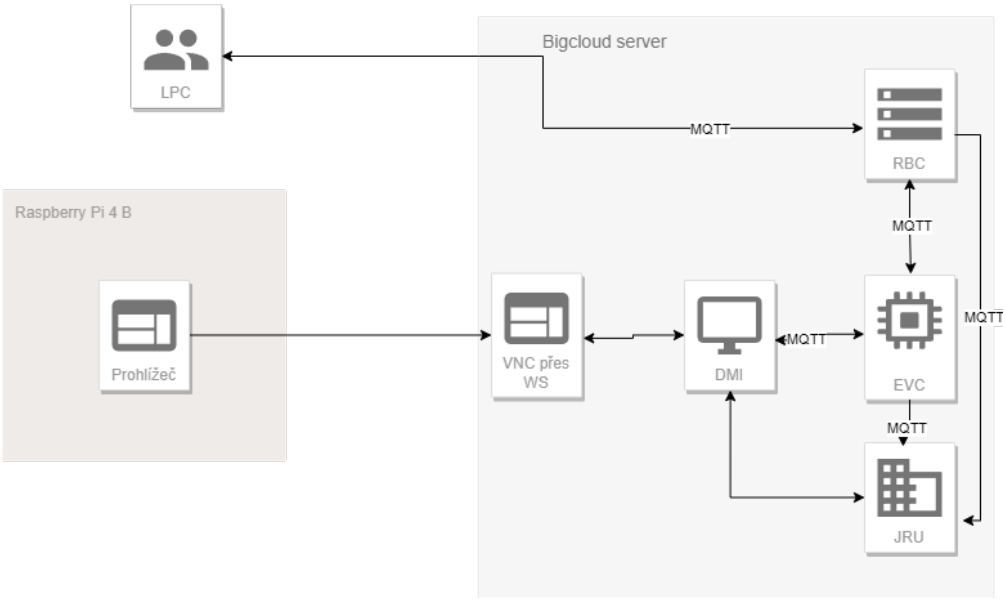


Obrázek 5.7 Diagram řešení s použitím VNC klienta

Kvůli problémům s VNC v prohlížeči byly testy s tímto řešením prováděné na osobním počítači. Jedná se však o řešení podstatně méně uživatelsky přívětivé. Sdílení obrazu řešené aktuálně v PHP je výrazně pomalejší, než nainstalovaný VNC klient a má zásadní problémy s překreslováním obrazu. Tedy i v případě, že by se podařilo toto řešení implementovat na Raspberry, zbylým dvěma by nemohlo konkurovat.

5.5 Vizualizace

Součástí ETCS je i projekt vizualizace, který má dotvořit strojvedoucímu imerzní situaci virtuální reality řízení vlaku. Tato část měla být otestována s použitým Raspberry, ale vzhledem k problémům, které Raspberry způsobuje prohlížeč, natož streamování videa na YouTube v omezeném rozlišení, byla možnost zakomponování vizualizace zavržena. Rozhodnutí bylo učiněno jak z důvodu rozpracovanosti projektu vizualizace, na kterém byly zahájeny práce teprve v letním semestru 2022/23, tak použití 4k monitoru na jehož vykreslování nemá Raspberry kapacity.



Obrázek 5.8 Diagram řešení za použití websocketu ve webové aplikaci [diagram autora]

Veškeré testy provedené na osobním počítači probíhaly za použití FullHD monitoru (1920x1080 - 1/4 rozlišení vizualizačního monitoru) s jehož vykreslováním mělo Raspberry citelnou práci. V případě spuštěné aplikace DMI (640x480 - méně, než 1/16 rozlišení vizualizačního monitoru) jen těsně překonává obnovovací frekvenci 75Hz a standard nasazený virtuální realitou je alespoň 90 snímků za vteřinu[40].

5.6 Doporučení pro budoucí vývoj

Zde budou rozebrány návrhy a postřehy spojené se zkušenostmi získané během práce na projektu s cílem navrhnout vylepšení práce a pomoci nově příchozím vývojářům s navázáním.

Práce s Raspberry

Další vývoj tohoto projektu bude jistě poznamenán omezeným přístupem k Raspberry. Představené řešení by však mělo pohodlně fungovat i na starších modelech Raspberry Pi (3), které jsou momentálně dostupné v 10. patře budovy Fakulty stavební (Thákurova 7), kde se nachází hardwarová laboratoř (v současnosti za půjčování přípravků zodpovídá Bc. Peter Guľa). Alternativně lze emulovat Raspberry OS na stolním počítači například pomocí během projektu hojně využívaného dockeru, který umožňuje nearmovským architekturám vytvářet image pro ARM. Ač žádná z těchto variant není ideální, jedná se možnosti mnohem dostupnější než Raspberry (použité v této práci), které bude nasazené na simulátoru v Dopravním sále FD ČVUT. Zároveň s tím by se většina změn v projektu neměla Raspberry nijak dotknout a bude maximálně nutné instalovat nové releasy DMI, případně pomocných nástrojů, které budou k provozu třeba. Těchto věcí by mělo být možné dosáhnout jen z příkazové řádky, stačilo by tak použití SSH tunelu a příkazové řádky (je třeba brát na vědomí, že se tím sníží bezpečnost celého systému).

V případě vývoje přímo na Raspberry (libovolném) vřele doporučuji opatření druhého páru myši a klávesnice, případně zajištění displeje/monitoru pro Raspberry (hlavně v případě práce na PC na místo notebooku). Raspberry nedisponuje periferiemi a všechny tři uvedené jsou pro práci s DMI nutné (klávesnice pro spuštění, myš k ovládání a obraz). Alternativně jsem zaregistroval

přepínač pro USB, kterým by mělo být snadno možné měnit zařízení, ke kterému je periferie připojena[16]. V začátku projektu jsem pracoval pouze s druhou myší a přepojoval klávesnici, což se velmi rychle ukázalo jako nepohodlné. Co se týče samotného ovládání Raspberry, doporučuji se omezit v co největším rozsahu na příkazovou řádku, případně spuštěné DMI. Jak jsem uváděl dříve, práce v prohlížeči na Raspberry je pomalá. Prohlížeč reaguje s velmi citelnou odezvou včetně interakce s jednotlivými prvky na stránce. Osobně jsem naprostou většinu webových stránek otevíral na notebooku a potřebné texty přepisoval raději, než abych příslušnou stránku otevřel na Raspberry a text do příkazové řádky jednoduše kopíroval.

Cloud

Veškeré nové releasy a obecně změny provedené v ETCS projektu se dotknou primárně bigcloudu, na který bude třeba vývojaři/ům vytvořit účet a zpřístupnit VPN. Přesné rysy spolupráce s Casablancou v tento moment ještě neexistují, ale měly by se rychle vyjasnit po prezentaci výsledků této práce. Na bigcloudu bude mít následně vývojář přístup k virtuálnímu stroji s debianem a nainstalovanými potřebnými nástroji (docker a ETCS Together). Bude tak možné s použitím vydáleného přístupu (přes VNC) jednoduše stahovat nové změny z gitlabu, případně specifické problémy odstraňovat přímo na daném systému.

Vzhledem ke komplikacím, které vyvstaly během připojování se k serverům Casablancy nemohu doporučit vhodný postup. S OpenVPN verzí 2.4.6 se mi na Windows pracovalo velmi dobře, ale původně stažená aktuální verze (3.4.3) se té použité nijak nepodobá. Nová verze aplikace na mě působila velmi čistě a přehledně, ale neposkytovala některé možnosti nastavení, kterými verze 2.4.6 disponuje a výsledné ovládání tak nemohu dobře posoudit. Verze programu 2.4.3 vyžaduje neintuitivní přesunutí souborů (certifikátů a klíče) do složky, o kterém uživatele informuje pouze vyhozením chyby o nenalezení zmíněných souborů. Veškerá další práce s aplikací je prakticky bezúdržbová, stačí po nastartování mezi skrytými ikonami zvolit OpenVPN GUI a v jeho otevíracím menu zvolit možnost *connect*.

Práce s PPTP, pokud bude zachována, byla o poznání méně pohodlná, ale jen z důvodů neexistence grafického rozhraní. Na druhou stranu nejhorší příkaz je spouštěn pouze při prvním navazování připojení, kdy je nutné do příkazové řádky vypsat delší text obsahující IP cílového serveru společně s přihlašovacími údaji. Existuje i možnost zadání těchto údajů v editoru místo příkazové řádky, v takovém případě je třeba spustit příkaz *sudo vim /etc/ppp/chap-secrets* (možné otevřít v jiném editoru, ale super-user je povinný), který otevře dokument s ukládanými údaji k otevíraným tunelům. Zde je možné data vyplnit: *user[TAB]tnlname[TAB]pswd[TAB]IP*[17] a obejít tak spouštění příkazu *pptpsetup*. Další zapínání VPN přístupu nevyžaduje opětovné zadávání všech informací, stačí spustit příkaz *pon tnlname* (podrobný postup je uveden výše v části zprovozňování VPN). Silně doporučuji si po použití příkazu *pptpsetup* zkontrolovat obsah souboru */etc/ppp/chap-secrrets* kvůli správnosti zadávaných údajů. Chybová hláška společně s kontrolou správnosti řádku v terminálu mě zavedly naprosto špatným směrem a jen náhodou jsem se rozhodl zkontrolovat uvedený soubor, před snahou o instalaci patchů. Jedinou zásadnější věc, kterou bych vytkl používání tohoto nástroje je nutnost při každém spuštění znovu zadávat přesměrování internetové komunikace přes vytvořenou VPN. Tomuto nepohodlí však lze snadno předejít vytvořením velmi jednoduchého skriptu, který uvedené příkazy vykoná za vývojáře.

Veškerá další práce a spolupráce s Casablancou je podmíněna jejich spokojeností s dosaženými výsledky a není tedy možné dopředu předpokládat další vývoj věcí.

[16] Nevlastním takové zařízení a nemohu jej tedy doporučit. Jedná se však o použitelnou alternativu

[17] Převzato z `https://www.domoticz.com/wiki/Installing_a_PPTP-VPN_server_on_a_Raspberry_Pi`

Nové releasy

Jak je zmíněno výše, naprostá většina úprav ETCS projektu se projeví na straně cloudu. Na něm budou komponenty EVC, JRU a RBC bez ohledu na zvolenou variantu stejně jako MQTT broker, v případě řešení pomocí VNC zde bude i komponenta DMI. Výhodou prvních zmíněných je jejich zahrnutí v ETCS Together, které tak umožňuje jejich společné updatování. DMI vyžaduje, kvůli nemožnosti zahrnutí do společného dockeru, samostatný přístup i pokud bude společně s ostatními na jednom zařízení. Vzhledem k dostupnosti a kapacitám zařízení bude snadné tyto změny řešit přímo na tamním virtuálním stroji, nicméně se i tak s sebou připojení k němu nese jistou režii.

Poněkud horší práce bude s udržováním Raspberry. Potenciálně se bude nacházet přímo na dopravním sále FD ČVUT (Horská 3, Praha 2) a tedy fyzicky hůře dostupné. Většina práce by mohla být odvedena skrze SSH tunel z příkazové řádky.

Osobně bych doporučil řešení, které je na vstupu poněkud pracnější, ale ušetří veškeré další ruční updaty. Díky Raspbian OS stojí na distribuci Debianu, podporuje i Debian balíčky, se kterými s gitlab hravě poradí. Gitlab je totiž možné nechat z buildu vytvořit přímo Debian balíček odlehčený od všech zbytečností, obsahující pouze části nutné pro update a následně tento vytvořený balíček nasadit na jmenované zařízení, v tomto případě Raspberry. Hlavní překážkou v tomto řešení je nalezení toolchainu pro Raspberry (a jeho ARM procesor), v čemž jsem neuspěl. Na druhou stranu, kompilace pro Raspberry nebylo, ani nemělo být cílem této práce a při střetnutí se s výše zmíněným problémem jsem zvolil nejjednodušší dostupnou cestu, kterou bylo spustit kompilaci přímo na používaném přípravku. Domnívám se tedy, že se nejedná o neřešitelný problém a snad ani o příliš složitý, pokud mu bude možné dát dostatečnou prioritu. Vzhledem k rychlosti, kterou se projekt vyvíjí a projeveném zájmu o jeho prezentaci, vnímám vylepšení workflow při nasazování nových releasů jako nejvyšší prioritu.

Bezpečnost

V aktuálním stavu se projekt vznáší v poněkud bezpečnostním vakuu. Je to způsobeno problémy s vytvářením VPN mezi Raspberry a Casablancou. Momentálně používaný PPTP je zřejmě nedostatečný pro seriózní použití a už při nasazování bylo zřejmé, že se jedná o dočasné řešení. Doporučil bych na základě výsledků projektu dojednat s Casablancou lepší řešení, v ideálním případě možnost vylepšení používaného TLS 1.0 na OpenVPN podporovanou TLS 1.2, a následné nainstalování OpenVPN klienta na Raspberry. Vzhledem k prototypickému stádiu celého projektu (včetně simulátorů a software) se jedná o vzdálený cíl a během vývoje je pptp dostačující. Zvyšování bezpečnosti komunuikace bych tedy věnoval jen nízkou prioritu.

Vizualizace

Vizualizace se tato práce dotýká jen okrajově a není tedy mnoho co doporučit. Po zapojení DMI a úspěšných testech, které ukazují na možnost obsluhování ovládání simulátoru pomocí DMI, je možné zaměřit výkonnost počítače v simulátoru čistě na vizualizaci a provozování nižšího množství monitorů (typicky jednoho) nebo prozkoumat možnosti řešení i vizualizace pomocí cloudu a v takovém případě prozkoumat ideální hardware pro streamování, pro které je Raspberry zřejmě nedostačující.

Kapitola 6
Závěr

Motivací pro tuto práci byl aktuální stav projektu ETCS a návrh na jeho zlepšení.

Nejprve tedy byla provedena analýza přesného stavu jak po stránce fyzické – tedy jak vypadá simulátor a hardware, kterým je ovládán – tak softwarové. Následně byla provedena analýza i technologie cloudu a architektury použité u poskytnutých serverů společností Casablanca INT, a. s. Na základě získaných informací byla navržena tři řešení jakožto alternativa k aktuálnímu, které bylo shledáno neideálním.

Práce se tak v jednom z řešení opírá o architekturu poskytnutou týmy předmětu BI-SP1, které v uplynulých letech implementovaly software ETCS na FIT ČVUT. Primárně pak o výsledky letního semestru 2022/23, který doposud zvlášť vyvíjené komponenty nechal vyvíjet jediným týmem, díky čemuž se podařilo zajistit kompatibilitu komponent.

Zbylá dvě řešení se naopak zcela odkazují na fungování poskytnutého serveru a uvažují úplnou integraci projektu do cloudu. Prvním z nich je řešení sdílející běžným způsobem obraz s virtuálním strojem emulovaným protistranou. Druhé využívá webovou aplikaci zmíněného cloudu a vyhýbá se tak nutnosti instalovat vlastního klienta pro sdílení obrazu. Toto druhé řešení se však nakonec ukázalo jako nerealizovatelné na kombinaci zvoleného klienta a aktuální verze webové aplikace.

Celá práce klade důraz na nutnost zabezpečení celého systému i jednotlivých jeho částí, ale z části zabezpečení upouští ve prospěch usnadnění dalšího vývoje jak tohoto řešení, tak projektu ETCS jako takového.

Nakonec se podařilo úspěšně vytvořit prototyp klienta pro ovládání simulátoru ETCS pomocí přípravku Raspberry Pi 4 B ve dvou modelech přístupu. Naplněny byly i cíle práce a většina hlavních požadavků vytyčených v závěru Analýzy. Výsledkem je nainstalovaný hardware připravený k představení, který uloženou činnost zvládá plynule a bez obtíží. Zároveň výsledné řešení ukazuje, že úplná integrace do použitého cloudu neposkytuje ani zdaleka tak dobré výsledky jako instalace DMI přímo na použitý hardware.

Výsledné řešení ukazuje, že úplná integrace do cloudu je možná, ale ne v případě projektů cílících na simulaci skutečného dění, jako jízda vlakem, a imerzi uživatele.

Bibliografie

1. LESO, Martin. *Koncept Železnice 4.0. - vize digitální železnice v ČR* [https://zeleznice40.cz/wp-content/uploads/2022/03/Leso_Zeleznice4_0_Final.pdf]. 2020. Navštíveno 4.9.2023.
2. DIRECTORATE-GENERAL FOR MOBILITY AND TRANSPORT. *What is ERTMS and how does it work?* [https://transport.ec.europa.eu/transport-modes/rail/ertms/what-ertms-and-how-does-it-work_en]. [B.r.]. Navštíveno 8.1.2024.
3. ROSHCHUPKINA, Daria. *ETCS - Modul pro komunikaci mezi EVC a RBC*. FIT ČVUT, 2022. bakalářská práce.
4. MALÝ, Matěj. *ETCS - EVC - Implementace módů reversing, shunting a post trip*. FIT ČVUT, 2023. bakalářská práce.
5. MĚŠŤAN, Ondřej. *ETCS DMI II - Implementace rozdílů na lokomotivách provozovaných v České republice oproti normě ETCS*. ČVUT FIT, 2023. Bakalářská práce.
6. UNISIG. *FFFIS for Eurobalise* [https://www.era.europa.eu/system/files/2023-01/sos3_index009_-_subset-036_v310.pdf]. 2015. Navštíveno 8.1.2024.
7. UNSIG. *FIS for the RBC/RBC Handover* [https://www.era.europa.eu/system/files/2023-01/sos3_index012_-_subset-039_v320.pdf]. 2015. Navštíveno 8.1.2024.
8. STERNWALD, Jiří. *ETCS - Lektorské PC - Uživatelské rozhraní*. ČVUT FIT, 2023. Bakalářská práce.
9. ERL, Thomas; MAHMOOD, Zaigham; PUTTINI, Ricardo. *Cloud Computing: Concepts, Technology & Architecture*. Prentice Hall, 2013.
10. MELL, Peter; GRANCE, Timothy. *The NIST Definition of Cloud Computing* [https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-145.pdf]. [B.r.]. navštíveno 29.8.2023.
11. HOWE, Denis. *Free On-line Dictionary Of Computing* [https://foldoc.org/emulation]. 2023. Navštíveno 13.11.2023.
12. VMWARE, INC. *What is a Virtual Machine?* [https://www.vmware.com/topics/glossary/content/virtual-machine.html]. [B.r.]. navštíveno 13.11.2023.
13. ARORA, Simran. *Docker vs. Virtual Machines: Differences You Should Know* [https://cloudacademy.com/blog/docker-vs-virtual-machines-differences-you-should-know/]. 2023. Navštíveno 2.10.2023.
14. KIRVAN, Paul. *Cluster* [https://www.techtarget.com/whatis/definition/cluster]. 2022. Navštíveno 15.10.2023.

15. LINBIT. *DBRD9 User's Guide* [https://linbit.com/drbd-user-guide/drbd-guide-9_0-en]. 2024. Navštíveno 8.1.2024.
16. SUN MICROSYSTEMS, Inc. *What is ZFS?* [https://web.archive.org/web/20100503201205/http://hub.opensolaris.org/bin/view/Community+Group+zfs/whatis]. 2009. Navštíveno 15.8.2023.
17. IBM. *What is IaaS (Infrastructure-as-a-Service)?* [https://www.ibm.com/topics/iaas]. [B.r.]. Navštíveno 9.1.2024.
18. IBM. *What is Platform-as-a-Service (PaaS)?* [https://www.ibm.com/topics/paas]. [B.r.]. Navštíveno 9.1.2024.
19. IBM. *What is SaaS (software-as-a-service)?* [https://www.ibm.com/topics/saas]. [B.r.]. Navštíveno 9.1.2024.
20. SLIWA, Carol. *ZFS* [https://www.techtarget.com/searchstorage/definition/ZFS]. 2017. Navštívena 15.8.2023.
21. MICROSOFT CORPORATION. *Microsoft 365 and Office 365 service descriptions* [https://learn.microsoft.com/en-us/office365/servicedescriptions/office-365-service-descriptions-technet-library]. 2023. Navštíveno 9.1.2024.
22. GILLIS, Alexander S. *Thick Client (Fat Client)* [https://www.techtarget.com/searchnetworking/definition/thick-client]. 2021. Navštíveno 9.1.2024.
23. GILLIS, Alexander S. *Thin Client (Lean Client)* [https://www.techtarget.com/searchnetworking/definition/thin-client]. 2021. Navštíveno 9.1.2024.
24. CLOUDFLARE, INC. *What is the Remote Desktop Protocol (RDP)?* [https://www.cloudflare.com/learning/access-management/what-is-the-remote-desktop-protocol/]. [B.r.]. Navštíveno 4.1.2024.
25. SCHHMITT, Matt. *RDP vs VNC: Which Remote Desktop Technology Should I Use in 2023?* [https://cloudzy.com/blog/rdp-vs-vnc/]. 2023. Navštíveno 5.1.2024.
26. *The WebSocket API (WebSockets)* [https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API]. 2023. Navštíveno 12.12.2023.
27. DIACONU, Alex. *What are WebSockets used for?* [https://ably.com/topic/what-are-websockets-used-for]. 2023. Navštíveno 12.12.2023.
28. HIVEMQ TEAM. *MQTT Tutorial: An Easy Guide to Getting Started with MQTT* [https://www.hivemq.com/article/how-to-get-started-with-mqtt/]. 2020. Navšítveno 20.12.2023.
29. GOOGLE LLC. *System requirements & supported devices for YouTube* [https://support.google.com/youtube/answer/78358?hl=en]. [B.r.]. Navštíveno 9.1.2024.
30. UGREEN. *The Complete Guide to a USB Ethernet Adapter* [https://blog.ugreen.com/usb-ethernet-lan-adapters/]. 2021. Navštíveno 9.1.2024.
31. DANAO, Monique; BOTTORFF, Cassie. *What Is A VPN Tunnel And How Does It Work?* [https://www.forbes.com/advisor/business/what-is-vpn-tunnel/]. 2023. Navštíveno 10.1.2024.
32. RASPBERRY PI FOUNDATION. *Getting started* [https://www.raspberrypi.com/documentation/computers/getting-started.html]. [B.r.]. navštíveno 10.11.2023.
33. VORONOVA, Arya. *Raspberry Pi And The Story Of SD Card Corruption* [https://hackaday.com/2022/03/09/raspberry-pi-and-the-story-of-sd-card-corruption/]. 2022. Navštíveno 5.12.2023.
34. HIRTENMACHER, Mark. *Understanding Raspberry Pi SD Card Wear Out: A Guide* [https://www.howto-do.it/raspberry-pi-sd-card-wear-out/#The_Lifespan_of_Raspberry_Pi_SD_Cards]. 2023. Navštíveno 7.1.2024.

35. RASPBERRY PI FOUNDATION. *Raspberry Pi 4 Model B (REDUCED)* [https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-reduced-schematics.pdf]. [B.r.]. Navštíveno 22.11.2023.

36. COOK, Jeremy. *Faulty Raspberry Pi USB-C Design: Pi 4 USB-C Problems & Fixes* [https://www.arrow.com/en/research-and-events/articles/faulty-usb-c-design-in-raspberry-pi-usb-c-cable-evolution]. [B.r.]. Navštíveno 22.11.2023.

37. USB IMPLEMENTERS FORUM, INC. *Universal Serial Bus Type-C Cable and Connector Specification* [https://www.usb.org/document-library/usb-type-cr-cable-and-connector-specification-release-23]. 2023. Navštíveno 28.11.2023.

38. POT, Justin. *VNC Connect Review* [https://www.pcmag.com/reviews/vnc-connect]. 2022. Navštíveno 27.12.2023.

39. SAVICKAITĖ, Miglė. *What is a PPTP VPN and why it°s the wrong choice* [https://surfshark.com/blog/what-is-pptp]. 2022. Navštíveno 7.12.2023.

40. CHARNOCK, Phil. *What Does the Frame Rate of a Virtual Reality Headset Indicate* [https://drawandcode.com/learning-zone/what-does-the-frame-rate-of-a-virtual-reality-headset-indicate/]. 2023. Navštíveno 19.12.2023.