Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačního inženýrství

Diplomová práce

Porovnání open source technologií pro vzdálenou plochu

Bc. Martin Ševčík

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ DIPLOMOVÉ PRÁCE

Bc. Martin Ševčík

Informatika

Název práce

Porovnání open source technologií pro vzdálenou plochu

Název anglicky

Comparison of open source remote desktop technologies

Cíle práce

Cílem této diplomové práce je porovnat open source technologie pro vzdálenou plochu na operačním systému Linux. Vzdálený uživatelský přístup k serveru bude realizován přes Raspberry Pi, které bude vystupovat v roli tenkého klienta, a softwarů využívajících protokoly RFB (VNC), RDP nebo NX.

Metodika

V teoretické části by byl popsán obecně operační systém Linux, nástroje použité při měření v praktické části, model klient–server, dále by byly uvedené a popsané vybrané technologie včetně komerčních a alternativních řešení pro operační systémy Windows a MacOS.

V části praktické by došlo k sestavení testovací sady, při které by bylo nasimulováno připojení uživatele, který by na serveru prováděl předem definované úkoly (spustil by např. Firefox s několika záložkami, napsal delší text v Libreoffice atd.). Během toho by bylo monitorováno využití sítě, zatížení serveru a „responzivnost“ grafického rozhraní – diagnostika FPS u uživatele a čas zpracování (doba, za kterou by doběhla jedna testovací sada). Po provedení testů by došlo k vyhodnocení naměřených dat a vybrání optimálního řešení.

Doporučený rozsah práce

60 – 80 stran

Klíčová slova

linux, klient, server, raspberry pi, vnc, rdp, nx

Doporučené zdroje informací

BLUM, Richard; BRESNAHAN, Christine; WILEY ONLINE LIBRARY (ONLINE SLUŽBA). *Linux command line and shell scripting bible.* Indianapolis, IN: John Wiley & Sons, 2015. ISBN 9781119209409.

KUROSE, James F.; ROSS, Keith W. *Počítačové sítě.* Brno: Computer Press, 2014. ISBN 978-80-251-3825-0.

NEMETH, Evi; SNYDER, Garth; HEIN, Trent R. *Linux : kompletní příručka administrátora.* Brno: Computer Press, 2004. ISBN 80-7226-919-4.

Předběžný termín obhajoby

2023/24 LS – PEF

Vedoucí práce

Ing. David Buchtela, Ph.D.

Garantující pracoviště

Katedra informačního inženýrství

Elektronicky schváleno dne 7. 3. 2024

Ing. Martin Pelikán, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 8. 3. 2024

doc. Ing. Tomáš Šubrt, Ph.D.

Děkan

V Praze dne 31. 03. 2024

Čestné prohlášení

Prohlašuji, že svou diplomovou práci "Porovnání open source technologií pro vzdálenou plochu" jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autor uvedené diplomové práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 31.03.2024 _____

Poděkování

Rád bych touto cestou poděkoval panu Ing. Davidu Buchtelovi, Ph.D. za cenné rady a odborné vedení při zpracovávání této práce.

Porovnání open source technologií pro vzdálenou plochu

Abstrakt

Tato diplomová práce se zabývá porovnáním open source technologií pro vzdálenou plochu.

V teoretické části lze nalézt definice a vysvětlení pojmů, které se týkají použitých technologií. Tato část obsahuje popis operačního systému GNU/Linux, shellu Bash včetně nástrojů v něm spouštěných, modelu klient-server, mikropočítače Raspberry Pi, protokolů a softwarů pro vzdálenou plochu a softwaru Media Express.

Praktická část je postavena na získaných znalostech z části teoretické. Nejprve je popsána obecně problematika a průběh přípravy jednotlivých prostředí. Poté následuje rozbor skriptů použitých při měření, představení naměřených hodnot a vizualizace prostřednictvím grafů. Na konci dochází k vyhodnocení všech měření a k vybrání optimálního řešení.

Klíčová slova: linux, klient, server, raspberry pi, vnc, rdp, nx, rfb, kdrive, x2go, xrdp, tigervnc

Comparison of open source remote desktop technologies

Abstract

This diploma thesis is about the comparison of open source remote desktop technologies.

In the theoretical part you can find definitions and explanations of terms used for this thesis. This section includes a description of the GNU/Linux operating system, the Bash shell including externally executable tools, the client-server model, the Raspberry Pi microcomputer, remote desktop protocols and softwares and Media Express software.

The practical part is based on the acquired knowledge from the theoretical part. First, the issues and the process of preparation of individual environments are described in general. This is followed by an analysis of the scripts used in the measurement, presentation of the measured values and visualization through graphs. At the end, all measurements are evaluated and the optimal solution is selected.

Keywords: linux, client, server, raspberry pi, vnc, rdp, nx, rfb, kdrive, x2go, xrdp, tigervnc

Obsah

1 Úvod

V dnešní době, charakterizované rychlým technologickým pokrokem a rostoucí potřebou flexibility v pracovním prostředí, získává vzdálený přístup k počítačovým systémům nezastupitelný význam. Organizace i jednotlivci se často potýkají s potřebou pracovat mimo kancelářské prostředí a mít přístup ke svým programům a datům z různých míst a za různých podmínek. Tento trend ještě více zesílil s nástupem pandemie covid-19, která přinutila mnoho firem a institucí k rychlému přechodu na práci z domova.

V tomto kontextu se stává výběr vhodného nástroje pro vzdálenou plochu klíčovým prvkem pro efektivní fungování organizací a jednotlivců. Existuje mnoho komerčních i open source řešení, která poskytují možnost vzdáleného přístupu, přičemž open source řešení mají často výhodu ve své flexibilitě, široké podpoře komunity a nižších nákladech na implementaci a provoz.

Práce je strukturována do dvou hlavních částí: teoretické a praktické. Teoretická část poskytne definici základních pojmů a popis technologií, které budou využity při tvorbě části praktické, která se zaměří na sestavení testovací sady a provádění relevantních testů, které umožní objektivní porovnání jednotlivých technologií. Tato diplomová práce přináší užitečný přehled problematiky vzdálené plochy a poskytuje praktické poznatky pro volbu optimálního řešení v konkrétních podmínkách.

2 Cíl práce a metodika

2.1 Cíl práce

Cílem této diplomové práce je porovnat open source technologie pro vzdálenou plochu na operačním systému Linux. Vzdálený uživatelský přístup k serveru bude realizován přes Raspberry Pi, které bude vystupovat v roli tenkého klienta, a softwarů využívajících protokoly RFB (VNC), RDP nebo NX.

2.2 Metodika

V teoretické části by byl popsán obecně operační systém Linux, nástroje použité při měření v praktické části, model klient-server, dále by byly uvedené a popsané vybrané technologie včetně komerčních a alternativních řešení pro operační systémy Windows a MacOS. V části praktické by došlo k sestavení testovací sady, při které by bylo nasimulováno připojení uživatele, který by na serveru prováděl předem definované úkoly (spustil by např. Firefox s několika záložkami, napsal delší text v LibreOffice atd.). Během toho by bylo monitorováno využití sítě, zatížení serveru a „responzivnost“ grafického rozhraní – diagnostika FPS u uživatele a čas zpracování (doba, za kterou by doběhla jedna testovací sada). Po provedení testů by došlo k vyhodnocení naměřených dat a vybrání optimálního řešení.

3 Teoretická východiska

3.1 GNU/Linux

Operační systém GNU/Linux, nebo taktéž běžně nazývaný pouze Linux, je kompletní svobodný softwarový systém, který spadá do skupiny unixových systémů. Jedná se o víceúlohový a víceuživatelský systém, který své využití uplatňuje v mnoha zařízeních, a to od palubních počítačů automobilů, přes osobní počítače, mobilní telefony, chytré televize až po myčky. [1]

Existuje celá řada linuxových distribucí, což jsou, jednoduše řečeno, softwarové kolekce založené na linuxovém jádře a systému správy balíčků. Krom toho se s distribucí pojí její příručky a dokumentace a také podpora (ať už ve formě uživatelských fór či placené podpory přímo od distributora). Každá taková distribuce nese svůj vlastní název a je zaměřena na jiné spektrum uživatelů (existují distribuce, které jsou vhodné např. pro nezkušené uživatele, pro vývojáře, pro výuku atd.). Mezi nejznámější a nejrozšířenější distribuce patří například jedna z nejstarších, a to Debian, který disponuje různým rozhraním, velmi jednoduchým instalačním programem a podporou mnoho architektur. Za jeho vznikem stojí německý programátor Ian Murdock a současný vývoj mají na starosti dobrovolníci z celého světa. Z této distribuce je odvozeno plno dalších, jako je třeba Ubuntu známé svou jednoduchostí a vhodností pro začátečníky v unixovém světě. Za zmínku také stojí další debianové deriváty, a to Kali, který nachází své uplatnění ve světě forenzních věd a penetračního testování, nebo Raspbian určený primárně pro Raspberry Pi počítače. Mezi další linuxové distribuce patří kupříkladu Fedora, Red Hat, Gentoo, OpenSUSE, Mint, Arch Linux atd. [2]

U desktopových distribucí přichází na řadu i otázka desktopového prostředí, kterých taktéž na trhu existuje celá řada. Desktopové prostředí v sobě zahrnuje styl oken, dekoraci oken, typy ovládacích panelů, pozadí a widgety (doplňky na desktop). Díky těmto grafickým prostředím lze ovládat počítač pomoci myši a mnoho operací mezi dvěma okny lze provádět stylem „drag and drop“ („táhni a pusť“), a to třeba pro kopírování souborů. Mezi ty nejrozšířenější v dnešní době patří KDE a GNOME, které však kladou vyšší nároky na operační paměť a výkon procesoru v porovnání například s prostředím Xfce, které si naopak zakládá na jednoduchosti a malé náročnosti, což umožňuje spuštění i na starších počítačích. Dále existují prostředí LXDE, Unity, Cinnamon, MATE, LXQt a další.

Jednotlivé distribuce pak vycházejí buď striktně v kombinaci s jedním konkrétním desktopovým prostředím, nebo jsou dostupné ve více variantách – Ubuntu například nabízí odnože Kubuntu (KDE), Lubuntu (LXQt), Xubuntu (Xfce) atp. Desktopové prostředí PIXEL, které je určeno pro Raspberry Pi, bude detailněji rozebráno v jedné z následujících kapitol. [3]

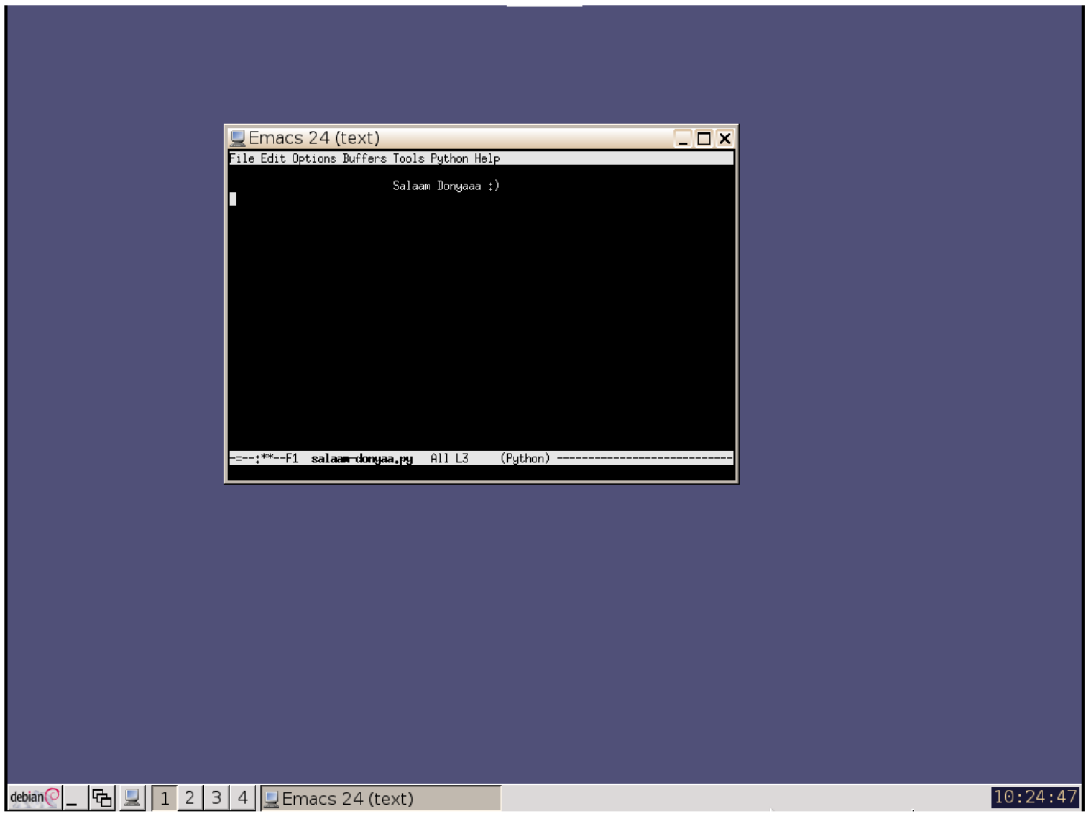
3.1.1 Historie

Historie sahá až do 60. let 20. století, kdy započaly práce na Unixu, jehož tvůrci jsou Ken Thompson a Dennis Ritchie. Tento operační systém vycházel z operačního systému Multics, který byl ale brzy ukončen z důvodu složitosti systému, časové a finanční náročnosti a nevlídnosti k uživatelům. [4]

Unix měl být jednoduchým a elegantním systémem. Při jeho psaní se ale narazilo na problém. Jelikož byl psaný v assembleru, nemohl být přenesen na jiný počítač bez přepsání celého programu. Bylo nutné jej přepsat do nějakého vyššího programovacího jazyka, který by nebyl závislý na hardwaru. Po předělání jazyka Fortran došel Thompson až k vytvoření jazyka C, který se zdál pro toto použití jako ideální.

Od začátku 80. let se vyvíjely dvě nezávislé větve Unixu – AT&T a BSD. AT&T větev se vyvíjela velice rychle a v roce 1983 vytvořili System V. Jednalo se o plnohodnotný operační systém, jehož vzhled se až do současnosti příliš nezměnil. Prodával se pod jménem Unix/PC. Unix se stal prvním víceúlohovým operačním systémem a byl naprogramován pro více uživatelů. Unix dosahoval celosvětového úspěchu, a tak z něj začaly vznikat klony od jiných výrobců. [5]

V roce 1983 se Richard Stallman rozhodl založit projekt GNU (GNU´S not UNIX), který měl za cíl vytvoření nového operačního systému na bázi Unixu. Zhruba o deset let později bylo téměř hotovo – všechen důležitý software (aplikace, systémové knihovny, překladač GCC, textový editor atd.) byl připraven, ale nejnižší úroveň (jádro) chyběla, a tak se začalo pracovat na vývoji nového jádra Hurd. [4]



Obrázek 1: Grafické prostředí Debian GNU/Hurd [6]

V té samé době započal ve svém volném čase jistý finský student informatiky Linus Torvalds práci na vývoji linuxového jádra. Bylo to z toho důvodu, že v té době vlastnil počítač, pro který neexistoval volně dostupný unixový operační systém, a za komerční se mu nechtělo platit. Unixový systém si vybral především proto, že se jedná o systém, který upřednostňuje jednoduchost. Mimo jiné se inspiroval i unixovým operačním systémem MINIX – zde se ale od tvůrce Andrewa Tanenbauma lišil hlavně tím, že svůj projekt vytvořil jako open source software. 17. září 1991 byla vydána první verze linuxového jádra. Toto jádro si získalo ihned tisíce příznivců, kteří začali přispívat k jeho vývoji.

Linux se měl původně jmenovat Freax, ale administrátorovi FTP serveru, na který Torvalds své soubory nahrál, se tento název nelíbil. Přejmenoval ho tedy bez Torvaldsova vědomí na Linux (Linus + X, které značí spojitost s unixovým systémem). [4]

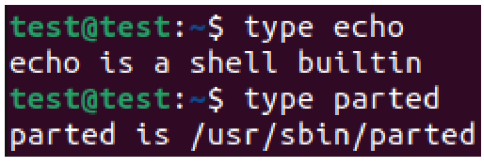
Vývoj Stallmanova jádra Hurd byl velice pomalý a velmi složitý. Naopak vývoj Torvaldsova linuxového jádra byl úspěšnější a rychlejší, takže došlo k významnému kroku – operační systém GNU se začal používat společně s jádrem Linux, ve spojení tedy tvoří název GNU/Linux. [7]

3.2 Bash

Bourne again shell (bash) je jedním z unixových shellů vydaných pod licencí GPL (GNU General Public Licence) a je dostupný pro většinu typu unixu a linuxu. Jedná se o vylepšenou a bezplatnou verzi bourne shellu, jehož základy přebírá a přidává k nim nové funkce (například editaci z příkazového řádku). Také v sobě nese funkcionality z dalších shellů, jako je korn shell nebo C shell. [8]

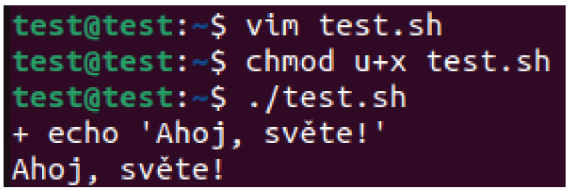
Shell je speciální interaktivní nástroj poskytující uživatelům způsob, jakým se mohou dostat „do vnitřku“ operačního systému, tj. umožňuje spouštět programy, spravovat soubory ve filesystému a řídit procesy. Shell definuje hranici mezi vnitřkem a vnějškem. Jádrem shellu je příkazový řádek umožňující zadávat textové příkazy, které je schopen následně interpretovat a spouštět je v kernelu.

Shell obsahuje sadu interních příkazů (builtin commands), mezi které patří kupříkladu příkazy pro přesouvání se mezi adresáři (příkaz cd), zobrazení textu (echo) nebo vypsání názvu adresáře, ve kterém se zrovna nacházíme (pwd). Kromě interních příkazů shell také umožňuje zadat název programu, jehož jméno pak předá jádru ke spuštění. [9]



Obrázek 2: Interní vs. externí příkaz [10]

Zadávané příkazy není nutné spouštět pouze napřímo z příkazové řádky, ale lze je taktéž umístit do souborů (s příponou .sh), které nazýváme jako shellovské skripty. Ty nám pak mohou pomáhat s automatizovaným řízením operačního systému – ať už ručním spouštěním z řádky či definováním pravidelného spouštění v crontabu (cron = softwarový démon, který je schopen v určitý čas spouštět předem definované příkazy a skripty). Před pokusem o spuštění samotného skriptu je dobré si ověřit, zda k tomu daný uživatel, pod kterým je skript spouštěn, má dostatečná oprávnění, a to konkrétně právo execute (x). Krom příkazové řádky je také možné skripty kombinovat v systému s grafickým rozhraním a vytvářet tak programy, které jsou pro běžného člověka uživatelsky přívětivější. [9]



Obrázek 3: Spuštění jednoduchého skriptu [10]

Mezi další hojně používané shelly patří mimo jiné třeba:

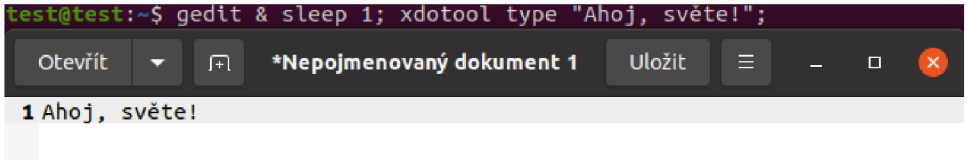
- ash – jednoduchý a odlehčený shell, který je schopen běžet v prostředí, jenž neklade vysoké požadavky na paměť a zároveň je plně kompatibilní s bashem.
- korn – programovací shell kompatibilní s bourn shellem, ale podporující pokročilé programovací funkce, mezi které patří asociativní pole nebo floating-point aritmetika (plovoucí řádová čárka).
- tcsh – shell zahrnující prvky z programovacího jazyka C do shellovských skriptů.
- zsh – pokročilý shell, který kromě funkcí z výše uvedených shellů obsahuje ještě navíc pokročilé programovací funkce, sdílení historie příkazů pro všechny spuštěné shelly nebo vylepšené zpracování proměnných a datových polí. [9]

3.2.1 Xdotool

Tento nástroj, naprogramovaný převážně v jazyku C, se používá pro automatizované ovládání aplikací. Dokáže totiž simulovat vstup klávesnice a činnost myši, vyhledávat okna a přesouvat je, měnit jejich velikost, skrývat a upravovat vlastnosti okna (například nadpis), přepínat plochy, přesouvat okna mezi plochami a měnit počet ploch. Všechny tyto aktivity jsou možné díky knihovnám ze sady xlib a tzv. XTEST, což je rozšíření pro X Window System. Veškeré operace provádí ve výchozím stavu v okně, ze kterého je zavolán (pokud není uvedeno jinak).

Jak již bylo řečeno, nástroj dokáže simulovat stisknutí klávesy, a to pomocí parametru „key“ a vybrané klávesy. Dokáže simulovat stisk jedné klávesy (např. „xdotool key Return“ zmáčkne klávesu enter), ale i kombinaci více kláves („xdotool key ctrl+s“ – klávesová zkratka používaná ve většině programech pro ukládání souborů). V předchozích příkladech dochází k pouhému stisknutí a následnému puštění dané klávesy. Mohou ale nastat situace, kdy potřebujeme klávesu na určitou dobu podržet, mezitím stisknout jinou

a až poté první klávesu pustit. K tomuto úkonu je určen příkaz „xdotool keydown alt key Tab keyup alt“, který simuluje stisknutí kláves Alt a Tab (přepnutí mezi dvěma okny v systému). Pro vypsání delšího textu pak slouží přepínač „type“ („xdotool type "Ahoj, světe!"“).



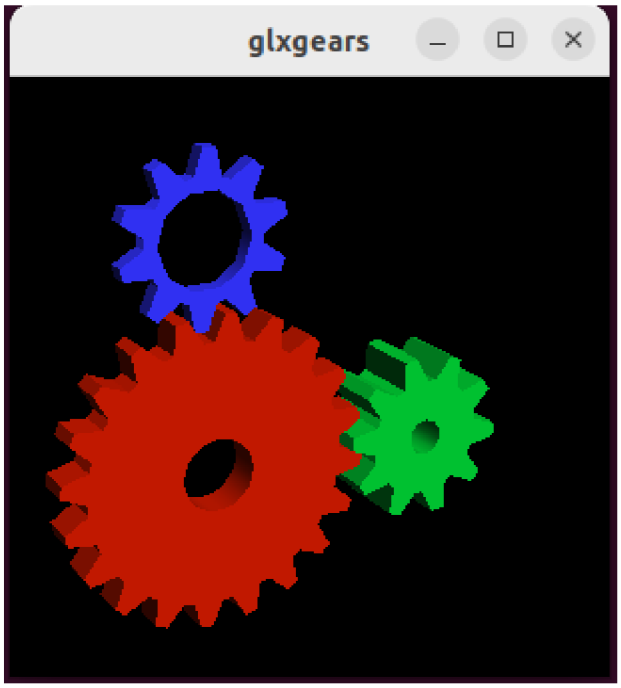
Obrázek 4: Xdotool [10]

Kromě příkazů určených pro práci s klávesnicí nabízí tato aplikace i simulaci myši. Mezi povolené vybrané činnosti patří kupříkladu posunutí kurzoru na obrazovce na zadané souřadnice („xdotool mousemove 1280 720“) nebo provádění určitých operací označených číselnými kódy („xdotool click 1“, přičemž 1 odpovídá levému tlačítku myši, 2 = prostřední, 3 = pravé, 4 = skrolovací tlačítko směrem nahoru a 5 = skrolování dolů).

Pokud bychom volali xdotool z příkazové řádky, ale vybrané akce bychom potřebovali provádět až v jiném okně, museli bychom použít příkaz „xdotool search -name "Media Express" windowactivate“. Ten nejprve vyhledá spuštěné okno s předaným jménem a následně jej na obrazovce přenese do popředí. [11]

3.2.2 Glxgears

Glxgears je jednoduchý a populární nástroj často používaný při měření zátěže a grafického výkonu počítače. Po jeho spuštění se na displeji zobrazí sada rotujících ozubených kol. Do terminálu se při běhu programu v pravidelných intervalech vypisuje počet vykreslovaných snímků za sekundu (FPS). [12]



Obrázek 5: Glxgears [10]

3.2.3 Ffmpeg

Jedná se o projekt umožňující uživateli práci s videem, zvukem, titulky a dalšími multimédii. Skládá se z kolekce knihoven:

- libavcodec – poskytuje implementaci širší škály kodeků.
- libavformat – implementuje streamovací protokoly, formáty kontejnerů a základní I/O přístup.
- libavutil – zahrnuje hashery, dekompresory a různé užitečné funkce.
- libavfilter – poskytuje prostředky ke změně dekódovaného zvuku a videa prostřednictvím orientovaného grafu připojených filtrů.
- libavdevice – poskytuje abstrakci pro přístup k zařízením pro záznam a přehrávání.
- libswresample – implementuje rutiny pro míchání a převzorkování zvuku.
- libswscale – implementuje rutiny pro převod barev a změnu měřítka.

A z kolekce programů:

- ffmpeg – sada nástrojů příkazového řádku pro manipulaci, konverzi a streamování multimediálního obsahu.
- ffplay – minimalistický multimediální přehrávač.
- ffprobe – jednoduchý analytický nástroj pro kontrolu multimediálního obsahu.

Jako příklad užití tohoto příkazu by mohla posloužit následující jednoduchá ukázka převodu obrázku ve formátu DPX do formátu PNG: „ffmpeg -i "file.dpx" "file.png"“. [13]

3.2.4 Tcpdump

Tcpdump je paketový analyzátor spustitelný z příkazové řádky, který je schopný zachytávat TCP/IP pakety v reálném čase. Spuštěný bez parametrů vypisuje síťový provoz na standardní výstup. S vhodnými přepínači dokáže informace ukládat do souboru, popřípadě ze souboru číst.

```
test@test:~$ sudo tcpdump
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), snapshot length 262144 bytes
19:52:50.391229 IP test > dns.google: ICMP echo request, id 1, seq 4, length 64
19:52:50.406069 IP test.55769 > RTK_GW.Home.domain: 47612+ [1au] PTR? 15.2.0.10.in-addr.arpa. (51)
19:52:50.406259 IP dns.google > test: ICMP echo reply, id 1, seq 4, length 64
19:52:50.419922 IP RTK_GW.Home.domain > test.55769: 47612 NXDomain 0/0/1 (51)
19:52:50.420059 IP test.55769 > RTK_GW.Home.domain: 47612+ PTR? 15.2.0.10.in-addr.arpa. (40)
19:52:50.433048 IP RTK_GW.Home.domain > test.55769: 47612 NXDomain 0/0/0 (40)
19:52:50.508274 IP test.43591 > RTK_GW.Home.domain: 22431+ [1au] PTR? 138.0.0.10.in-addr.arpa. (52)
19:52:50.509428 IP RTK_GW.Home.domain > test.43591: 22431* 1/0/1 PTR RTK_GW.Home. (77)
19:52:51.391824 IP test > dns.google: ICMP echo request, id 1, seq 5, length 64
19:52:51.405640 IP dns.google > test: ICMP echo reply, id 1, seq 5, length 64
19:52:52.393537 IP test > dns.google: ICMP echo request, id 1, seq 6, length 64
19:52:52.407459 IP dns.google > test: ICMP echo reply, id 1, seq 6, length 64
```

Obrázek 6: Tcpdump [10]

Výstup příkazu se skládá ze seznamu paketů, kdy se každý z nich může skládat z následujících segmentů:

- 08:41:13.729687 – časové razítko přijatého paketu.
- IP – protokol síťové vrstvy, zde IPv4 (pro IPv6 je IP6).
- 192.168.64.28.22 – zdrojová adresa včetně portu (zde port 22).
- 192.168.64.1.41916 – cílová adresa včetně portu. [14]
- Flags [P.] – tzv. TCP flag (příznak), tj. způsob TCP komunikace. Příznak P (PUSH) říká, že příchozí data by měla být předána přímo aplikaci namísto ukládání do vyrovnávací paměti. [15]
- seq 196:568 – sekvence bajtů, které daný paket obsahuje (zde paket obsahuje 196. až 568. bajt celého toku).
- ack 1 – strana odesílající data má zde vždy číslo 1. Strana přijímající by měla místo toho číslo reprezentující další očekávaný bajt (v tomto toku 568).
- win 309 – počet bajtů dostupných v přijímací vyrovnávací paměti.
- length 372 – délka paketu v bajtech. [14]

3.2.5 Tshark

Obdobou analyzátoru tcpdump je příkaz tshark, který je vhodný především pro následnou analýzu rozsáhlejších PCAP souborů (packet capturing, soubor, kam např. tcpdump ukládá odchytávanou síťovou komunikaci). Umožňuje zachycovat informace o paketech v reálném čase, a to buď vytištěním dekódované formy těchto paketů na standardní výstup, nebo zapsáním paketů do souboru. Také umožňuje číst pakety z dříve uloženého souboru.

Výstup „tshark -r rx.pcap -qz io,stat,1“ poskytuje statistiky I/O (vstupu/výstupu) na základě časových intervalů. Skládá se z následujících přepínačů:

- r – specifikuje vstupní PCAP soubor, ze kterého budou data analyzována.
- q – potlačuje podrobný výstup (quiet mode). Když je tento přepínač použit, tshark nevypisuje podrobné informace o jednotlivých zachycených paketech, ale pouze výsledky specifikovaných operací.
- z io,stat,1 – shromažďuje statistiky paketů/bajtů pro zachycení v intervalech sekund. Interval může být specifikován buď jako celá, nebo zlomková sekunda. Pokud je interval 0, statistika se vypočítá pro všechny pakety. Zde konkrétně vrací statistiku o vstupních a výstupních bajtech za jednu sekundu. [16]

```
test@test:~$ tshark -r rx.pcap -qz io,stat,1

==============================
| IO Statistics              |
|                            |
| Duration: 5.  8966 secs    |
| Interval: 1 secs           |
|                            |
| Col 1: Frames and bytes    |
|----------------------------|
|           |1               |
| Interval  | Frames | Bytes |
|----------------------------|
|  0 <> 1   |      1 |    98 |
|  1 <> 2   |      1 |    98 |
|  2 <> 3   |      1 |    98 |
|  3 <> 4   |      1 |    98 |
|  4 <> 5   |      1 |    98 |
|  5 <> Dur |      1 |    98 |
==============================
```

Obrázek 7: Tshark [10]

Výstup uvedeného příkazu z obrázku by se dal popsat jako:

- Duration – celková doba zachyceného provozu, v tomto případě necelých 6 sekund.
- Interval – délka jednoho intervalu, v tomto případě 1 sekunda.

Nyní se podíváme na část tabulky, která zobrazuje statistiky v rámci jednotlivých intervalů. Tato část tabulky uvádí informace o počtu rámců a bajtů v rámci každého intervalu:

- Interval – rozsah intervalu, kde „0 <> 1“ znamená první interval (tedy nultá až první sekunda).
- Frames – počet rámců, které byly zachyceny během daného intervalu.
- Bytes – celkový počet bajtů zachycených v daném intervalu. [16]

3.2.6 Capinfos

Capinfos je program, který čte jeden nebo více zachycených souborů a vrací některé nebo všechny dostupné statistiky (informace) každého z nich v jednom ze dvou typů výstupních formátů, a to buď jako long (ve výchozím nastavení) nebo table. Dlouhý výstup je vhodný pro čtení člověkem. Výstup tabulky je užitečný pro generování sestavy, kterou lze snadno importovat do tabulky nebo databáze. [17]

Prostý výpis příkazu ve verzi long může být realizován jako „capinfos rx.pcap“ a vracet následující informace:

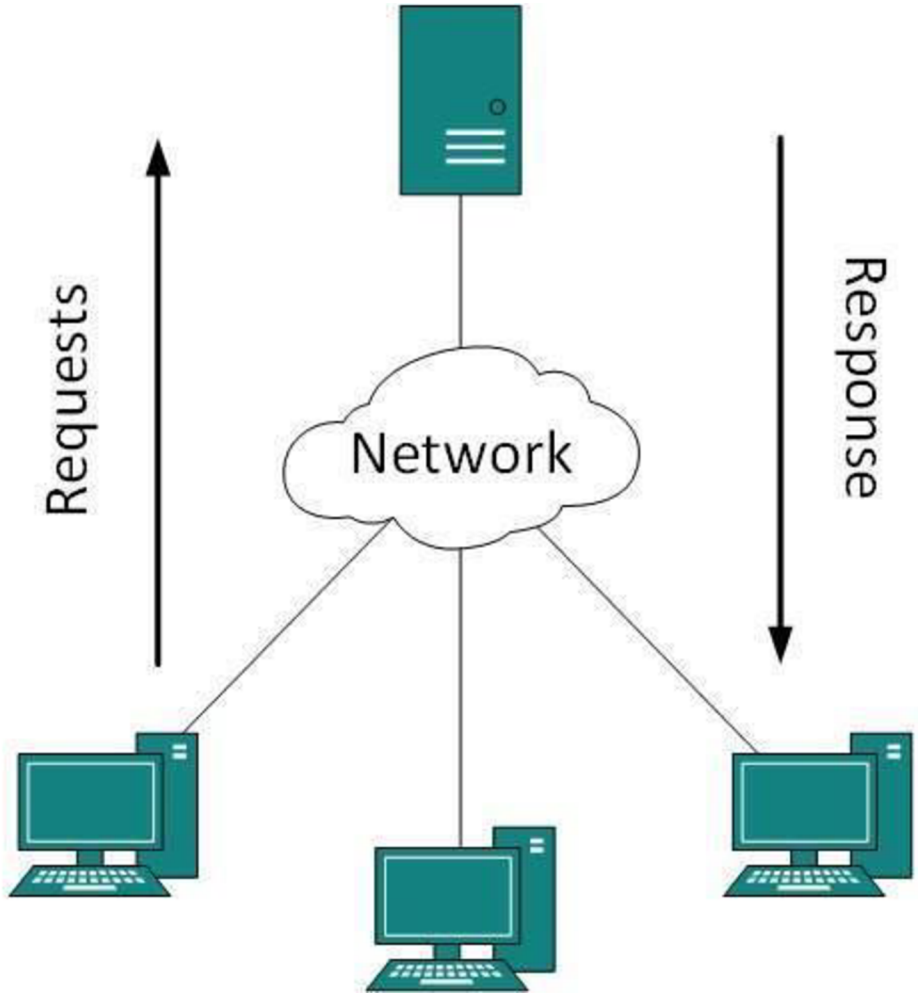
```
test@test:~$ capinfos rx.pcap
File name:           rx.pcap
File type:           Wireshark/tcpdump/... - pcap
File encapsulation:  Ethernet
File timestamp precision:  microseconds (6)
Packet size limit:   file hdr: 262144 bytes
Number of packets:   5
File size:           594 bytes
Data size:           490 bytes
Capture duration:    4,006198 seconds
First packet time:   2024-03-05 13:44:39,082596
Last packet time:    2024-03-05 13:44:43,088794
Data byte rate:      122 bytes/s
Data bit rate:       978 bits/s
Average packet size: 98,00 bytes
Average packet rate: 1 packets/s
SHA256:              387c984bef51c77468bc8cadd680f95ee04c2fb52bd71ae2211c76aa5f9f1402
RIPEMD160:           0b894027eb5ca01f03e334c565c3b6f347595ae8
SHA1:                c81098d630178aa38d6c3e641dba9390a3277b92
Strict time order:   True
Number of interfaces in file: 1
Interface #0 info:
                     Encapsulation = Ethernet (1 - ether)
                     Capture length = 262144
                     Time precision = microseconds (6)
                     Time ticks per second = 1000000
                     Number of stat entries = 0
                     Number of packets = 5
```

Obrázek 8: Capinfos [10]

Výstup příkazu poskytuje základní informace o PCAP souboru, jako je jeho název, typ, velikost atd., dále informace o paketech (maximální velikost paketu, celkový počet paketů atp.) a jiné informace – rychlost přenosu dat, počet rozhraní a údaje o nich. [17]

3.3 Model klient-server

Architektura počítačové sítě klient-server je založena na dvou typech koncových zařízeních, a to na klientovi a na serveru, mezi nimiž probíhá komunikace. V praxi se pak běžně vyskytuje situace, že se k jednomu serveru připojuje vícero klientů. [18]



Obrázek 9: Model klient-server [19]

Mezi zásadní výhody používání tohoto druhu modelu patří například to, že máme zavedený určitý centralizovaný systém se všemi prostředky a daty na jednom místě, takže celá struktura je snadnější na údržbu, lze ji lépe zabezpečit a kontrolovat zdroje a přístup klientů. Další výhodou je, že kapacitu klienta a serveru lze měnit separátně a nejsou na sobě nikterak závislé.

Co se týče stinných stránek, tak do nich bychom určitě mohli zařadit fakt, že pokud dojde k výpadku serveru, tak požadavky klientů nemohou být splněny, jelikož jsou na daném serveru v tomto ohledu závislí. V případě, že by došlo k napadení serveru útočníkem, ocitnou se v ohrožení všichni zainteresovaní klienti, a to ať už by se jednalo o útok typu DOS (Denial Of Service), o nahrání viru, trojského koně či červa na server, o manipulaci s přenášenými datovými pakety nebo útok MITM (Men In The Middle). [18]

Pod pojmem klient je v uvedeném výpočetním modelu myšlen nějaký program (například webový prohlížeč či desktopová aplikace), který poskytuje uživateli rozhraní, které umožňuje požadovat od serveru jeho služby skrz posílané dotazy.

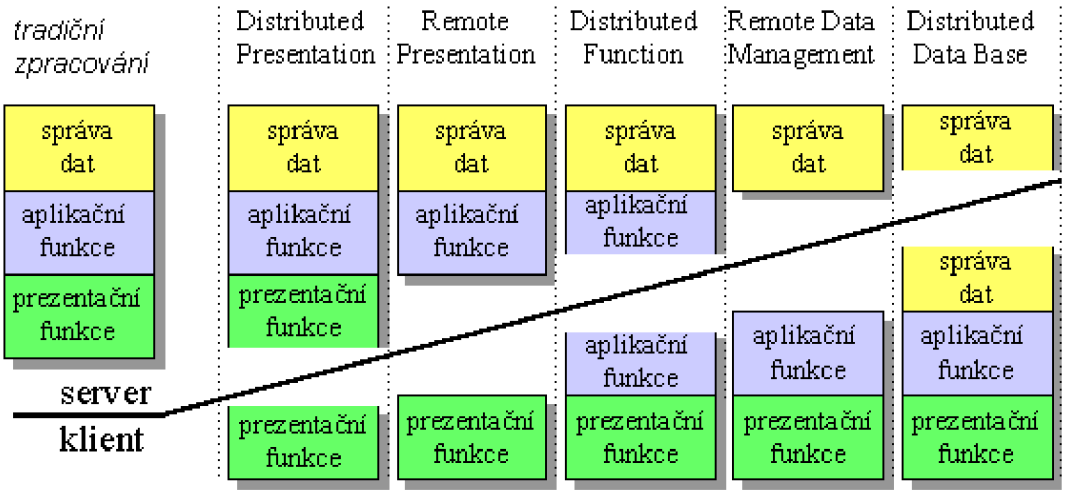
Rozlišujeme dva hlavní druhy softwarových klientů – prvním je takzvaný tlustý klient (neboli fat client, rich client, thick client). Ten provádí veškeré operace sám, tj. není závislý na připojení k serveru, a úložiště dat se nachází lokálně u něj. Tlustý klient se využívá především k synchronizaci dat.

A druhým typem klienta je tenký klient (thin client), který se spoléhá na prostředky hostitelského počítače, a veškerá data jsou taktéž uložena na straně serveru. Vzhledem k faktu, že hlavní centrum dění je soustřeďováno na stranu serveru, tak tento druh klienta nefunguje offline bez připojení, na rozdíl od toho tlustého. [20]

Server dodává a spravuje většinu zdrojů a služeb, které klient požaduje, čímž klienta ušetří zpracovávání výpočetně náročných operací a zašle mu pouze výsledné informace. [18]

Jiří Peterka vymezuje v modelu klient-server tři druhy činností každé aplikace, ve které vždy v nějaké formě figurují data. Ta potřebujeme vhodným způsobem uchovávat a zároveň k nim musí být zajištěn takový přístup, který odpovídá charakteru samotné aplikace. Tato činnost bývá nazývána jako správa dat (data management). V druhé části, kterou označujeme termínem prezentační činnost (presentation), se skrývá interakce se samotným uživatelem, která spočívá ve sběru požadavků a příkazů směrem k aplikaci a zároveň je zde i řešeno zobrazování výsledků získaných ze strany serveru. Třetí a poslední komponenta realizuje vlastní logiku celé aplikace – aplikační činnost (application function).

Kromě druhů činností zmíněných v předchozím odstavci popisuje Peterka ještě klasifikace možných způsobů dělby práce mezi klientem a serverem. [21]



Obrázek 10: Pět variant modelu klient-server [22]

Jednou z variant modelu klient-server může být distribuovaná prezentace (distributed presentation). U ní dochází k rozdělení prezentační funkce mezi klienta a server, ten pak má také na starost i zbývající činnosti. Na straně serveru dochází nejčastěji ke generování výstupů ve znakovém režimu do textového okna. Toto okno není ve skutečnosti nikde zobrazováno, ale dochází k přenosu jeho obsahu směrem ke klientovi, který má za úkol jej zobrazit samotnému uživateli, a to obvykle v grafickém režimu emulujícím znakový režim textového okna.

Ve variantě vzdálené prezentace (remote presentation) jsou oproti té první ponechány veškeré prezentační činnosti na straně klienta. Server se stará pouze o správu dat a aplikační funkce.

Pokud aplikační funkce rozprostřeme mezi klienta a server, můžeme hovořit o variantě distribuované funkce (distributed function). Správu dat zajišťuje server a prezentační činnosti klient.

Dalším druhem modelu je vzdálená správa dat (remote data management), v rámci kterého jsou aplikační a prezentační funkce zcela v kompetenci klienta a server se věnuje jenom problematice správy dat.

V poslední variantě, distribuovaná databáze (distributed data base), se většina činností přenáší na stranu klienta. Pouze o udržování a správu dat se dělí se serverem. [21]

3.4 Raspberry Pi

Raspberry Pi je malý jednodeskový a nízkonákladový počítač s deskou plošných spojů na bázi architektury ARM vyvíjený charitativní nadací Raspberry Pi Foundation ve Velké Británii. Nejnovějším modelem je v tuto chvíli Raspberry Pi 5, které bylo uvedeno na trh v říjnu roku 2023.

Primárním operačním systémem je Raspbian (Raspberry Pi OS), který je založený na linuxovém Debianu. Systém je distribuovaný stejnojmennou nadací, která má na starosti samotné počítače. Jako hlavní desktopové prostředí Raspbian využívá tzv. PIXEL (Pi Improved Xwindows Environment Lightweight), který je založený na upravené verzi prostředí LXDE a na správci oken Openbox. Operační systém je vyvíjen ve třech instalačních verzích – Raspberry Pi OS Lite (tento typ neobsahuje desktopové prostředí), Raspberry Pi OS desktop a Raspberry Pi OS desktop s doporučeným softwarem. Všechny tyto verze jsou dostupné v 32bitovém i 64bitovém provedení.

Hlavním cílem nadace bylo za pomoci Raspberry Pi podpořit technické vzdělání dětí na školách, ale své využití najde počítač i v jiných oblastech, a to například jako webový server, úložiště NAS, monitorovací systém, tenký klient nebo v oboru IoT (internet věcí). Díky dobrému poměru mezi cenou a výkonem si dokázal najít své uplatnění i mezi komunitou počítačových kutilů. [23]

Raspberry Pi je vyráběn v několika modelech, a to následovně:

- Raspberry Pi A – model pro nízko nákladové projekty
- Raspberry Pi B – standardní model
- Raspberry Pi Compute – model pro průmyslové využití
- Raspberry Pi Zero – menší verze
- Raspberry Pi 400 – model zabudovaný v klávesnici
- Raspberry Pi Pico – programovatelná deska

Některé modely mohou být navíc označeny symbolem plus, které říká, že se jedná o jeho vylepšenou řadu. [24]

Jako každý počítač se i Raspberry Pi (zde konkrétně model 4 B) skládá z různých komponent, z nichž každá má svou roli. První a pravděpodobně nejdůležitější komponenta je integrovaný obvod neboli čip (System-on-Chip, SoC) – zde se nachází mozek celého počítače, a to centrální procesorová jednotka (CPU) a grafický procesor (GPU), který má za úkol zobrazovat grafiku. Bez paměti by mozek nedokázal přinášet kýžený výsledek,

proto je hned pod ním umístěna volatilní RAM paměť. Operační systém a data pak bývají uložena na vložené SD kartě. Dále zde najdeme komponentu s wifi a bluetooth pro připojení periferií nebo přijímání a odesílání dat. Co se týče jednotlivých portů, tak je deska osazená dvěma USB 2.0 porty, dvěma USB 3.0 porty, ethernetovým portem s LED diodami, konektorem USB C pro napájení, 3,5mm jackem např. pro připojení sluchátek, CSI portem, sloužící k připojení kamerového modulu Raspberry Pi, dvěma mikro HDMI porty, konektorem DSI pro připojení dotykového displeje Raspberry Pi a univerzálním vstupním/výstupním pinem GPIO, který poskytuje širokou škálu využití, a to od připojení LED diod až po teplotní senzory nebo joysticky. [25]

3.5 Protokoly pro vzdálenou plochu

V následujících kapitolách budou představeny a detailněji popsány protokoly určené pro vzdálenou plochu, a to ty, které budou v práci dále využity v části praktické.

Protokolů existuje celá řada, ať už se jedná o multiplatformní (například NX nebo RFB) či o specifické, které jsou navrženy pouze pro daný operační systém, a to například Apple Remote Desktop Protocol (ARD) určený pro zařízení s macOS, popřípadě Remote Desktop Protocol (RDP), který je typický pro systémy Windows.

3.5.1 NX

NX technologie, běžně známá jako NX nebo NoMachine, je počítačový software pro vzdálený přístup a vzdálené ovládání, které poskytuje komprimované a šifrované připojení. Je vyvíjen lucemburskou společností NoMachine.

Jedná se o sadu technologií skládající se z tenké vrstvy serverového softwaru, která umožňuje libovolnému unixovému počítači pracovat jako terminálový server. K dispozici jsou také klienti pro širokou škálu platforem a operačních systémů. NoMachine se rozhodl postavit základy této technologie na dobře známém a široce používaném systému X Window System – okenním systému, který stojí za grafickým uživatelským rozhraním Linuxu a operačním systémem Unix.

X Window System je otevřený a rozšiřitelný protokol, který byl navržený tak, aby poskytoval vrstvu oddělení mezi aplikační logikou (běžící na aplikačních serverech) a prezentací (probíhající na klientech). Jinými slovy, byl navržen pro síťové výpočty. Nativní protokol X vyžaduje vysokou šířku pásma a sítě s nízkou latencí, aby fungoval co

nejlépe. V průběhu let však začalo docházet k vývoji a rozšiřování grafických uživatelských rozhraní, což mělo za následek zvýšení poptávky po síťových a výpočetních zdrojích, čemuž se vývojáři nedokázali přizpůsobit, a tak během několika posledních let X Window System zcela ztratil své původní vlastnosti síťového výpočetního protokolu.

Hlavním cílem projektu NX bylo od počátku vyvinout technologii komprese X, která by každému uživateli umožnila provozovat neupravené verze nejrozšířenějších X desktopových prostředí na standardním X serveru, a to přes jakýkoli typ síťového připojení. To je podle vývojářů to, co je nejprve potřeba, aby systém X-Window mohl znovu získat svou funkci síťového protokolu. Společnost NoMachine vyvinula techniky komprese protokolu X a integrovanou sadu proxy agentů, které umožňují spouštět kompletní relace vzdálené plochy s použitím úzkopásmových připojení.

NX komprese X protokolu funguje kvůli minimalizaci přenášených informací ve třech úrovních:

- Komprimuje síťový provoz různými způsoby, včetně diferenciálních algoritmů, pokročilých metod ukládání dat do mezipaměti, bezztrátové a ztrátové komprese obrazu.
- Snižuje počet opakování sítě téměř na nulu a maximalizuje propustnost.
- Přizpůsobuje šířku pásma v reálném čase podle aktuálních podmínek sítě.

NX poskytuje kompresní poměry protokolu X v rozsahu od 10:1 do 100:1 a více, v závislosti na zobrazené aplikaci. Toho je v rychlých sítích LAN dosaženo bez negativního dopadu na výkon, takže vzdálené relace mohou obvykle běžet rychlostí, při které téměř nelze rozeznat, že běží na vzdáleném serveru. [26]

Při procesu posílání dat je využíván protokol SSH, jenž byl vybrán z bezpečnostních důvodů. Připojení funguje na principu vytvoření uživatele „nx“ na straně severu, jehož shell (/usr/NX/bin/nxserver) se spustí pokaždé, když se na server připojí k SSH daný uživatel pomocí NX klienta. Počáteční přihlášení mezi klientem a serverem probíhá prostřednictvím páru klíčů DSA (algoritmus digitálního podpisu). Veřejná část je poskytována během instalace serveru, zatímco soukromá část je distribuována společně s klientem NX. Použití tohoto klíče přinutí server SSH spustit nxserver a umožní SSH X11 forwarding (přesměrování). Po ověření klienta na serveru dojde k vytvoření zabezpečeného kanálu SSH, na kterém probíhá autentizace uživatele v systému a vyjednávání parametrů relace mezi serverem a klientem. Ve výchozím nastavení je NX klient nakonfigurován

s povoleným šifrováním veškerého provozu, tj. NX tuneluje veškerý provoz relace přes šifrovaný kanál SSH. [27]

Technologie komprese jádra byla vydána jako svobodný software pod licencí GNU GPL, což dalo prostor pro vznik alternativních open source řešení založených právě na této technologii. Mezi tyto produkty patří například FreeNX, Neatx nebo TaciX, avšak ani jeden z těchto produktů není nadále vyvíjen, na rozdíl třeba od projektu X2Go, který udržován stále je. [28]

3.5.2 RDP

Remote Desktop Protocol, neboli RDP, je proprietární síťový protokol od společnosti Microsoft umožňující vzdálený přístup ke vzdáleným počítačům včetně vzdáleného zvuku, schránky, tiskáren a přenosů souborů s grafikou ve vysokém rozlišení. Komunikace mezi RDP klientem a RDP serverem ve výchozím nastavení probíhá na portu 3389.

Serverová část, nazývaná též jako Terminal Services, je dodávaná ve Windows již od verze NT 4.0 Terminal Edition, datováno do roku 1996. Je možné ji používat v omezené variantě, bez současné práce vícera uživatelů, i na desktopových verzích od dob Windows XP. Klientská část (Remote Desktop Connection, RDC a Terminal Service client) je dostupná pro všechny desktopové verze počínaje Windows XP.

Remote Desktop Protocol se dočkal implementace i do operačního systému Linux, a to konkrétně jako XRDP. Tato implementace bude detailněji rozebrána v jedné z následujících kapitol. [29]

Protokol je koncipován obecně jako PPP (Point to Point Protocol), což je komunikační protokol, který funguje na druhé nejnižší vrstvě ISO/OSI modelu – vrstvě linkové (alternativně na vrstvě síťového rozhraní modelu TCP/IP). Tento protokol se používá pro přímé spojení mezi dvěma síťovými uzly. Umožňuje autentizace, šifrování, kompresi přenášených dat a podporu pro více monitorů. [30]

V RDP je využíváno různých bezpečnostních mechanismů pro ochránění přenášených dat. Konkrétně se pak například jedná o:

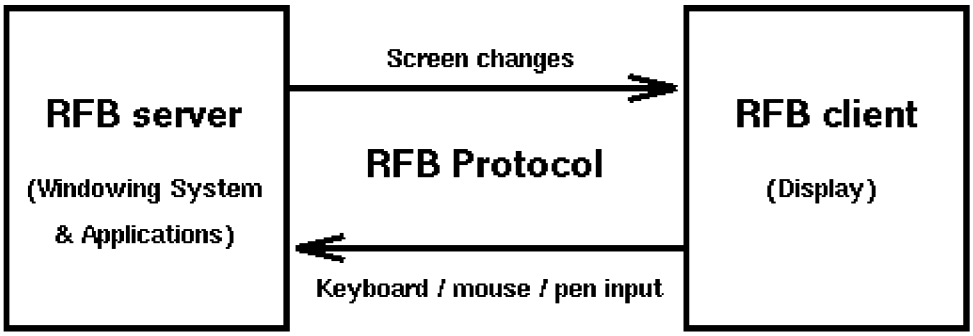
- Šifrování spojení: Na úrovni spojení mezi klientem a serverem. Standardně je použit kryptografický protokol TLS (Transport Layer Security), popřípadě jeho předchůdce SSL (z důvodu bezpečnosti již v dnešní době ale není doporučován). Díky této technologii je zajištěno šifrování veškeré

komunikace mezi klientem a serverem a také dochází k ochraně před odposloucháváním.

- Podpora šifrovacích algoritmů: RDP využívá silných šifrovacích algoritmů pro zajištění bezpečného přenosu dat, a to konkrétně třeba symetrickou blokovou šifru AES (Advanced Encryption Standard) s klíči o délce 128, 192 nebo 256 bitů.
- Autentizace: Pomocí certifikátů, které pomáhají předcházet potenciálním útokům, jako je MITM. Díky certifikátům se klient a server vzájemně dokáží identifikovat, čímž si zajistí důvěryhodnost a integritu probíhané komunikace.
- Bezpečnostní politiky: Administrátor serveru má možnost omezit podporované šifrovací algoritmy, minimální délky klíče, povinné používání certifikátů atp., čímž přizpůsobí úroveň bezpečnosti RDP komunikace dle požadovaných potřeb a požadavků. [29]

3.5.3 RFB

Jednoduchý protokol RFB (Remote Frame Buffer) pro vzdálený přístup ke grafickému uživatelskému rozhraní funguje na principu framebufferu, tudíž je použitelný pro všechny okenní systémy a aplikace včetně X11, Windows nebo macOS. Framebuffer je paměť, do které jsou ukládána obrazová data pro zobrazení na monitoru, podporuje rychlé a efektivní vykreslování grafických prvků. RFB klade na vzdálený displej velmi malé nároky, co se týče kupříkladu výpočetního výkon či paměti. Protokol je široce implementován a má poměrně kvalitní interoperabilitu (tj. schopnost vzájemně spolupracovat s různými systémy). RFB je používán pro technologie VNC a standardně komunikuje na portu 5900. [31]



Obrázek 11: RFB protokol [32]

Protokol také činí klienta bezstavovým – pokud se klient odpojí od daného serveru a následně se znovu připojí, stav uživatelského rozhraní zůstane zachován. Kromě toho lze pro připojení ke stejnému serveru použít jiný koncový bod klienta. V novém koncovém bodě uvidí uživatel to samé grafické uživatelské rozhraní jako v původním koncovém bodě.

Zobrazování dat funguje na jednoduchém principu – „umísti obdélník pixelových dat na danou pozici x, y“. Sekvence těchto obdélníků je nazývána jako aktualizace framebufferu, to jednoduše řečeno představuje změnu z jednoho platného stavu framebufferu do stavu jiného, což může v některých případech připomínat snímky videa. Aktualizace je řízena klientem, tj. je ze serveru odeslána klientovi pouze jako odpověď na jeho požadavek. Z čehož plyne, že čím je pomalejší klient a síť, tím je nižší rychlost těchto aktualizací. [31]

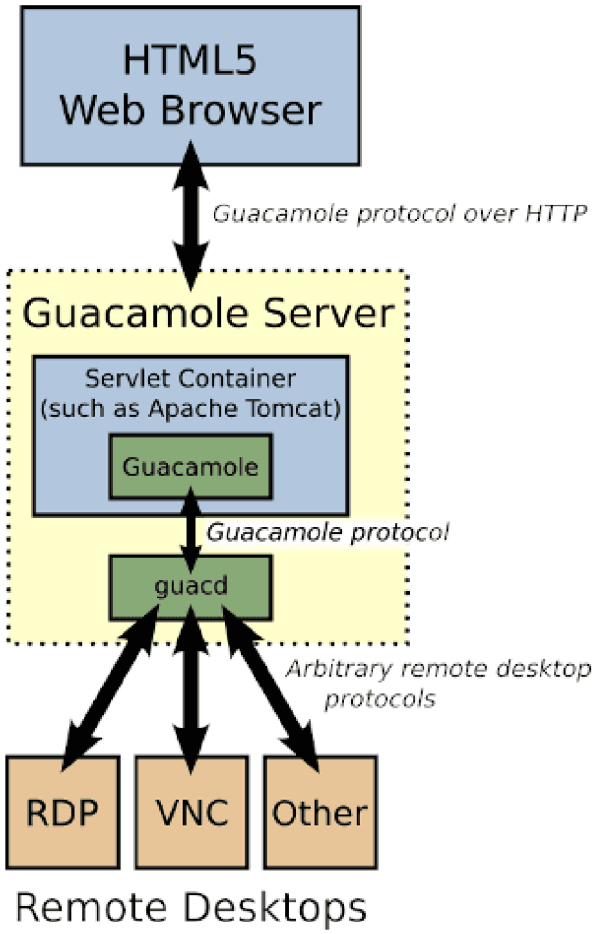
3.6 Softwary pro vzdálenou plochu

V následujících kapitolách jsou podrobněji uvedeny a popsány softwary pro vzdálené desktopy. Většina uvedených programů je určena pro operační systém Linux, ale jsou zde zmíněny i aplikace pro Windows a MacOS.

3.6.1 Apache Guacamole

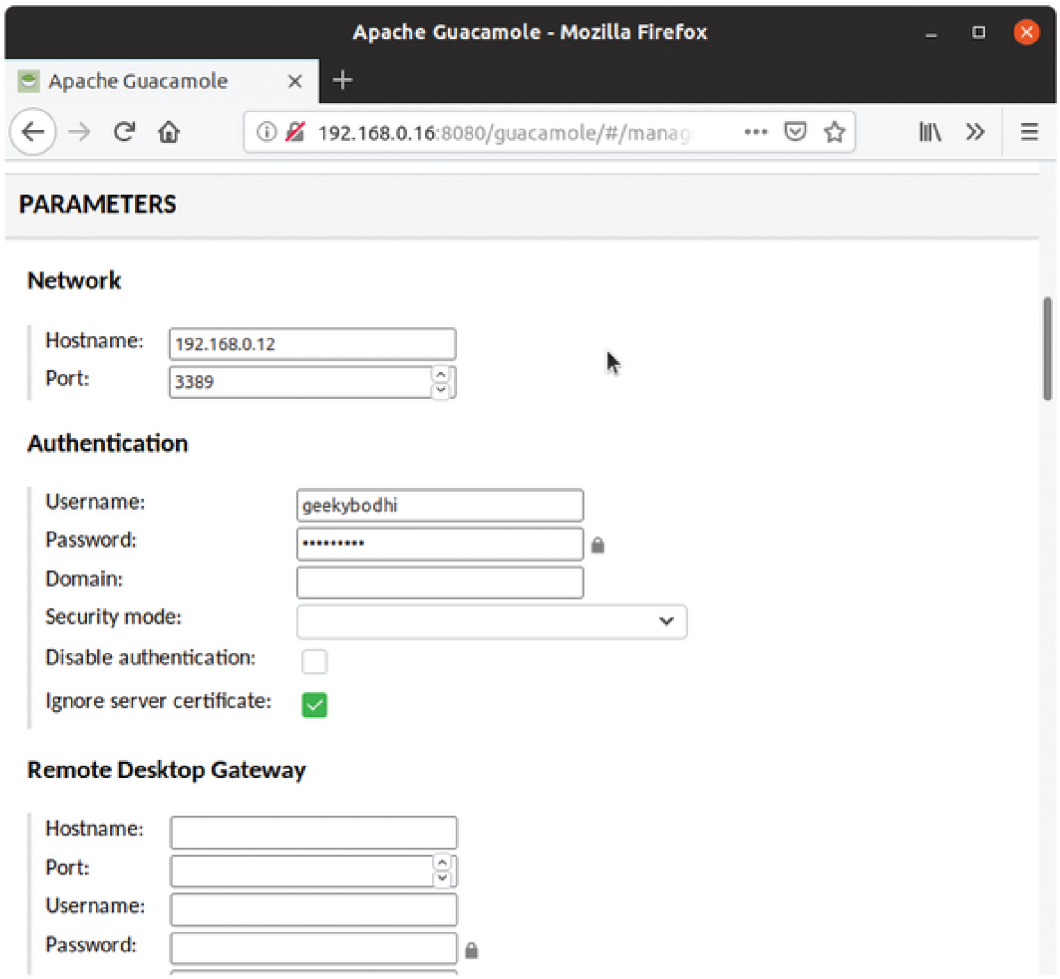
Jedná se o svobodný a otevřený software pro vzdálenou plochu od společnosti Apache Software Foundation. Od většiny podobných programů se liší už tím, že pro spuštění klientské části není nutná instalace žádného dalšího softwaru – uživatel k serveru přistupuje prostřednictvím svého webového prohlížeče s podporou HTML5. Serverovou část je možné implementovat pouze na linuxových distribucích.

Architektura Guacamole je rozdělena do několika částí – skládá se ze serveru napsaného v jazyce C, z webové aplikace v Javě, klienta (webová stránka) v JavaScriptu a démona guacd, který představuje proxy, která komunikuje na serveru s Java servletem a na druhé straně se připojuje k jednotlivým vzdáleným desktopům. Webová aplikace si s žádným protokolem vzdálené plochy, jako je např. RDP nebo VNC, nerozumí (rozumí pouze protokolu Guacamole). Ale díky vrstvě s démonem guacd, který překládá mezi daným protokolem vzdálené plochy a protokolem Guacamole, může dojít k realizaci celého přenosu. Webová aplikace je sice napsána v Javě, ale dle slov samotných vývojářů je projekt Guacamole zamýšlen jako API, tudíž je možné si aplikaci přepsat do jiného jazyka. [33]



Obrázek 12: Architektura Apache Guacamole [34]

Webové rozhraní umožňuje administrátorovi vytvářet a spravovat jednotlivé uživatele, skupiny a spojení pro jednotlivé počítače, která lze třídit do stromového uspořádání například podle protokolů. Uživatelům i uživatelským skupinám je možné nastavovat různá práva – administrátorská, práva k vytvoření nového uživatele, skupiny nebo připojení. [35]



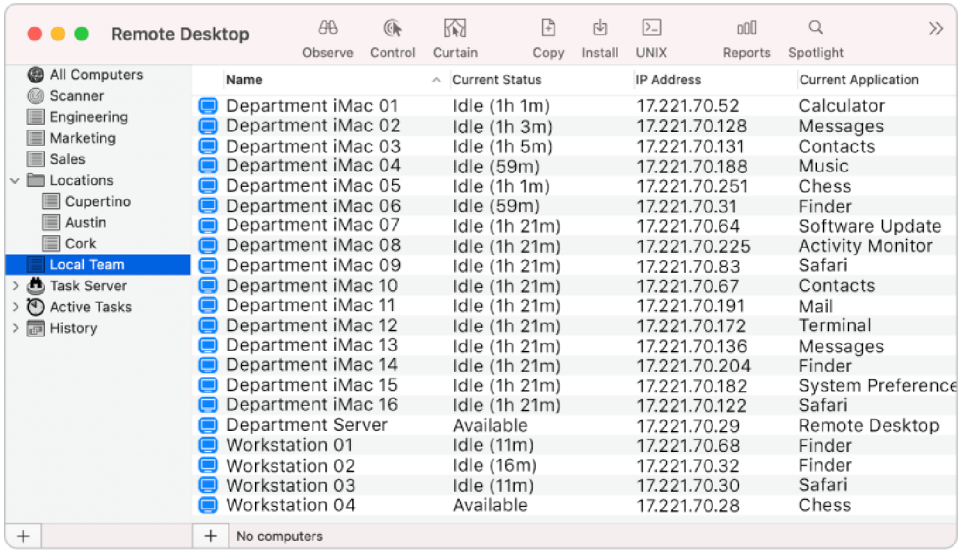
Obrázek 13: Nastavení parametrů připojení v Apache Guacamole [36]

3.6.2 Apple Remote Desktop

Apple Remote Desktop (ARD) je placená aplikace od společnosti Apple Inc. určená pro hromadné spravování klientských počítačů pomocí vzdálené plochy. Serverovou i klientskou část je možné spouštět pouze ze systémů macOS.

Administrátorská část Remote Desktop umožnuje například plánovat vypnutí a restart všech počítačů včetně odesílání připomínek a zpráv na vzdálené plochy, funkce UNIX umožňuje spouštět na vzdálených počítačích UNIX příkazy a shellovské skripty (popřípadě AppleScript skripty) a také je možné vzdálené kopírování souborů a instalování programů. ARD má v sobě integrovanou technologii VNC, díky které může administrátor k uživateli přistupovat s grafickým uživatelským rozhraním. Správce u každého vzdáleného počítače vidí v seznamu jeho název, aktuální stav, IP adresu a název aktuálně spuštěného programu. Po rozkliknutí se mu navíc zobrazí MAC adresa počítače a čas poslední aktivity. [37]

Na straně klienta jsou pak spuštěny tyto aplikace: ARDAgent (jedná se o klientskou část ARD reagující na požadavky aplikace Remote Desktop), AppleVNCServer (poskytuje funkcionalitu vzdálené pracovní plochy) a OpenWBEMServer (WBEM je technologie založená na HTTP a XML, která specializovaným nástrojem umožňuje získávat informace z klientských počítačů). [38]



Obrázek 14: Apple Remote Desktop [39]

3.6.3 Citrix Vitual Desktop

Citrix Systems je nadnárodní korporace, která vyvíjí technologie pro virtualizaci serverů a desktopů, počítačové sítě, SaaS (software jako služba) a cloud computing. Virtualizaci desktopů zajišťuje produkt Citrix Virtual Desktop (dříve XenDesktop), zatímco virtualizace aplikací je řešena pomocí softwaru Citrix Virtual App (dříve XenApp). Virtual Desktop v sobě zahrnuje automaticky i Virtual App.

Virtual Desktop spravuje a dodává aplikace a desktopy pomocí zprostředkovatele připojení zvaného Desktop Delivery Controller (DDC). DDC podporuje více hypervizorů (software spravující instance virtuálních strojů), včetně VMware vSphere, Microsoft Hyper-V a Nutanix Acropolis pro vytváření virtuálních strojů pro spouštění virtuálních aplikací a desktopů. Je kompatibilní s několika typy způsobů doručení a architekturami, včetně desktopů a serverů, datových center a soukromých, veřejných nebo hybridních cloudů. Jádrem celého Virtual Desktopu je DDC, které ukládá konfiguraci a další nastavení spravovaná konzolí v Citrix Studiu (zde se spravuje celkové nasazení Virtual App a Virtual Desktop). Připojení k virtuálním desktopům je pak spravováno pomocí Citrix Workspace. [40]

Virtual App a Virtual Desktop je licencován podle tří kategorií – Standard služba pro Azure (cloudová správa virtuálních aplikací a desktopů na platformě Azure), Premium služba (cloudová služba s vylepšenými funkcemi – jednotná správa všech aplikací, virtuálních počítačů a prostředí) a služba Premium plus (komplexní řešení digitálního pracovního prostoru). Ve všech třech verzích si Citrix účtuje poplatky za každého uživatele a u edice Standard se navíc ještě platí tzv. termínovaný závazek. [41]

3.6.4 Linux Terminal Server Project

LTSP je svobodný a otevřený software pro Linux umožňující distribuci a provozování operačního systému pomocí TCP/IP sítě ze serveru na bezdiskových stanicích. Do podvědomí mnoha lidí se LTSP dostávalo od roku 2005, kdy začal být projekt podporován společností Canonical, která má na svědomí linuxovou distribuci Ubuntu, z níž vychází obdobná distribuce Edubuntu specializující se na školství. Edubuntu v sobě začleňuje architekturu klienta LTSP, který je pro školní prostředí jako stvořený, vzhledem k faktu, že školy většinou nemívají mnoho finančních prostředků na nákup licencí a hardwaru a údržba jednotlivých systémů na velkém množství počítačů nebývá příliš jednoduchá (zde se správce stará pouze o jeden jediný obraz systému). Výhody ale LTSP přináší i studentům, kteří mohou při výuce sdílet svou obrazovku, a pedagogům, kteří je mohou vzdáleně sledovat a pomáhat jim. Přímo pro Raspberry Pi pak existuje projekt PiNet (dříve Raspi-LTSP), který je na klasickém LTSP založen.

Aby se klient mohl k serveru připojit, je nutné na serveru nastavit chroot prostředí společné pro všechny klienty a stáhnout potřebné balíčky pro běh tenkého klienta, vše ostatní poběží ze serveru. [42]

3.6.5 NX Technology

Nomachine nabízí dva hlavní druhy produktů. Prvním z nich je NoMachine for Everybody, který je zdarma dostupný pro osobní použití. Umožňuje uživateli streamovat multimediální obsah, upravovat dokumenty, tisknout, přenášet soubory a tak podobně. Jeho nevýhodou ale je, že neposkytuje přístup skrz webový prohlížeč, ale pouze přes instalovaného klienta, a nezahrnuje v sobě technickou podporu od výrobce.

Výše zmíněné dva nedostatky produktu NoMachine for Everybody řeší druhý z nabízených programů, kterým je NoMachine for the Enterprise, který je poskytován ve třech variantách. Jednou z nich jsou Enterprise Desktop, který je dostupný pro Windows,

macOS i Linux. Jedná se o snadnou a výkonnou vzdálenou plochu s neomezeným počtem připojení. Dále zde máme Terminal Server produkty, které jsou ale dostupné pouze pro linuxové stroje a u kterých se jednotlivá provedení liší v počtu virtuálních desktopů, které poskytují. Třetí a zároveň poslední variantou z Enterprise edice jsou Cloud Server produkty dostupné pro Windows, macOS a Linux. [43]

Technické detaily NX technologií byly již popsány výše, a to v kapitole týkající se NX protokolu.

3.6.6 Remote Desktop Services

V současné době je nejnovější verzí Windows Server 2022 z 18. srpna 2021. Oproti svému předchůdci Windows Serveru 2019 obsahuje několik vylepšujících funkcí. Co se týče oblasti bezpečnosti, přichází nová verze s podporou TPM 2.0 – jedná se o standard pro kryptoprocesor, který slouží pro ukládání šifrovacích klíčů, hesel, certifikátů a vysoce citlivých a cenných dat. Secured-core, neboli zabezpečené jádro, poskytuje vyšší úroveň ochrany. Secured-core se zaměřuje na širokou škálu potenciálních útoků na firmware, které (pokud jsou úspěšné) mohou zůstat na počítači i po vymazání operačního systému nebo výměně komponent. [45]

Stejně jako předchozí verze je i tato dostupná v několika variantách, které jsou rozděleny podle způsobu využití:

- Windows Server 2022 Datacenter Edition
- Windows Server 2022 Standard Edition
- Windows Server 2022 Essentials Edition

Windows Server 2022 v edici Datacenter je vhodný pro datacentra a cloudová prostředí s vysokou mírou virtualizace. Jeho výhodou je vysoká škálovatelnost, rychlost a zpracování velkého množství transakcí v reálném čase.

Pro prostředí, která jsou především fyzická nebo minimálně virtualizovaná, je nejvhodnější volbou edice Standard. Jedná se o verzi univerzální – dokáže sice zastat funkci doménového řadiče či DNS serveru, ale je především využívána pro poskytování služeb pro sdílení tiskáren nebo souborů. Lze ji jednoduše využívat v modelu klient-server.

Pokud bychom hledali edici ideální pro malé firmy do 25 uživatelů a 50 zařízení, měli bychom využít právě edici Essentials, jejíž obrovskou výhodou oproti výše zmíněným edicím je možnost používání bez nutnosti licencí CAL. Mimo jiné tato edice umožňuje cloudovou integraci s Microsoft Azure. [46]

3.6.6.1 Licencování

Od verze Windows Server 2016 Microsoft upravil způsob licencování svých produktů. Zatímco v dřívějších verzích byl Windows Server licencován primárně podle modelu server plus Client Access Licence (CAL – licence pro klientský přístup) nebo pomocí modelu procesor plus CAL, v současné době Microsoft licencuje na základě počtu procesorových jader. Toto však neplatí u edice Essentials, kde je licence vázaná na daný server a kde licence CAL není vyžadována. [47]

Edice Datacenter nijak neomezuje uživateli počet virtuálních počítačů a kontejnerů, ani není nutné platit s jejich rostoucím počtem žádné další licence.

Verze Standard v sobě v základu zahrnuje práva na dva virtuální počítače a neomezený počet kontejnerů. V případě potřeby rozšíření o další počítače či kontejnery, je však nutné zakoupit další licence.

S edicí Essentials je možné provozovat jeden virtuální počítač, na kterém lze spouštět pouze Windows Server Essentials. Pokud dojde ke spuštění virtuálního počítače, fyzického hostitele lze použít pouze pro jeho správu. [46]

3.6.6.2 Role RDS

Role jsou softwarově připravené balíčky na serveru, které můžeme instalovat a konfigurovat. Definují nám, k čemu daný server bude sloužit.

Role vzdálené plochy (Remote Desktop Services, dříve známé jako Terminal Services) umožňuje serveru hostovat více současných klientských relací. Vzdálená plocha využívá technologii Remote Desktop Service, která umožňuje vzdálené spuštění relace. Uživatel se může připojit k serveru Remote Desktop Session Host (RD Session Host), dříve známý jako terminálový server, pomocí klientského softwaru Remote Desktop Connection (RDC). Tato role je plně podporována v edici Standard a Datacenter. [48]

RDS se skládá z několika částí:

- Remote Desktop Session Host – role, která serveru umožňuje hostovat vzdálené aplikace založené na relacích sdílené s uživateli, kteří se k těmto desktopům připojují pomocí klientů určených pro vzdálenou plochu nebo prostřednictvím klienta webového.
- Remote Desktop Connection Broker – díky této roli jsou schopni autorizovaní uživatelé se přihlašovat k virtuálním desktopům a vzdáleným aplikacím

prostřednictvím soukromé sítě nebo internetu. Tato služba také může při vytváření nových připojení vyvážit zátěž napříč servery.

- Remote Desktop Web Access – umožňuje uživatelský přístup ke vzdáleným plochám prostřednictvím webového prohlížeče. Webové rozhraní lze použít k publikování ploch pro klientská zařízení s různým operačním systémem. Služba požaduje korektní funkčnost internetové informační služby (IIS) a nainstalované odpovídající digitální certifikáty na straně serveru i klienta.
- Remote Desktop Gateway – brána vzdálené plochy poskytuje uživatelům přístup k desktopům a aplikacím hostovaných v cloudových službách Microsoft Azure.
- Remote Desktop Licensing – aby bylo možné uskutečnit vzdálené připojení, je pro každé zařízení uživatele vyžadována licence CAL. [49]

3.6.7 ThinLinc

Jako další příklad softwaru pro poskytování vzdálené plochy je ThinLinc vyvíjený společností Cendio AB. Server a hlavní uživatelské desktopy jsou dostupné pouze pro Linux, zatímco klientskou část je možné spustit i z Windows a macOS, popřípadě je k dispozici také přístup prostřednictvím webového prohlížeče.

ThinLinc využívá ke svému fungování několik technologií a protokolů. Pro šifrování a autentizaci slouží protokol SSH a pro grafiku, klávesnici a myš je zde k dispozici VNC (RFB). O přehrávání a nahrávání zvuku se stará zvukový systém PulseAudio, přístup k systému souborů řeší NFS (Network File System) a přístup k sériovému portu má na starosti Telnet/RFC2217.

Výrobce umožňuje bezplatné stažení testovací verze se všemi funkcemi až pro deset souběžných uživatelů. Předplatné se pak vypočítává podle počtu souběžných uživatelů, a to u dvou poskytovaných verzí – Standard a Premium. Standardní verze v sobě zahrnuje licence, aktualizace a základní e-mailovou podporu s garantovanou dobou odezvy dva pracovní dny. Varianta Premium navíc nabízí rychlejší dobu odezvy a možnost technické podpory prostřednictvím telefonního hovoru. [50]

Součástí ThinLinc je také software TigerVNC, do kterého Cendio investuje svůj čas a peníze na jeho údržbu. Právě přes tento program funguje grafika vzdálené plochy a vstup z klávesnice nebo myši. V roce 2009 bylo do TigerVNC přidáno rozšíření SIMD (Single Instruction, Multiple Data), díky kterému došlo k urychlení komprese a dekomprese JPEG, což má za následek umožnění vzdáleného přehrávání pohyblivé grafiky v režimu celé

obrazovky. To lze provést bez jakéhokoliv softwaru pro dekodér videa na straně klienta nebo bez specializované manipulace s videem. Díky tomuto výkonnostnímu vylepšení také ThinLinc velmi dobře funguje ve spojení se softwarem VirtualGL (sada nástrojů umožňující vzdálené zobrazování aplikací s 3D hardwarovou akcelerací tenkým klientům), což umožňuje aplikacím, jako je například Google Earth, běžet s dobrým výkonem. [51]

Cendio odkazuje na svých webových stránkách na ansible roli, která se stará o instalaci, konfiguraci a spuštění serverového softwaru ThinLinc. Vytvoření a údržbu role má ale na starosti uživatelská komunita a společnost k ní neposkytuje žádnou podporu. [50]

3.6.8 TigerVNC

Virtual Network Computing (VNC) je vzdálený zobrazovací systém, který umožňuje prohlížet a pracovat s prostředím virtuální plochy, která je spuštěna na jiném počítači v síti. TigerVNC je vysokorychlostní verze VNC založená na kódů RealVNC 4 a X.org. TigerVNC vznikl jako snaha o novou generaci programu TightVNC pro platformy Unix a Linux, ale na začátku roku 2009 došlo k rozdělení projektů, aby se TightVNC mohl soustřeďovat na platformy Windows. TigerVNC podporuje variantu kódování tight, která je značně urychlena použitím JPEG kodeku libjpeg-turbo.

Celý projekt TigerVNC se skládá z několika aplikací. Pro všechny operační systémy (Linux, Windows, MacOS) existuje následující:

- vncviewer – multiplatformní TigerVNC viewer napsaný pomocí FLTK (Fast Light Toolkit = knihovna poskytující grafická uživatelská rozhraní).

Speciálně pro Windows je připraveno:

- winvnc – TigerVNC server pro Windows, který je ale dle slov vývojářů aktuálně neudržovaný, a proto není doporučeno jej používat, jelikož všechny funkcionality nemusí správně fungovat. Taktéž není zaručena korektní funkčnost v případě, že je používána funkce rychlého přepínání uživatelů nebo funkce vzdálené plochy.

Verze pro linuxové či unixové systémy (mimo macOS) obsahují tyto programy:

- Xvnc – jedná se o TigerVNC server, který je koncipován jako VNC server a zároveň jako X server s virtuálním framebufferem. Ke spuštění toho serveru se ve většině případů používá služba vncserver.

- vncpassword – program umožňující nastavit nebo měnit heslo používané pro přístup k relacím daného serveru, a to za předpokladu, že se používá VNC ověřování.
- vncconfig – tento program slouží ke konfiguraci a řízení běžící instance Xvnc.
- x0vncserver – server, který umožňuje vzdálený přístup k libovolnému X displeji například skrz vncviewer. Na rozdíl od Xvnc nevytváří virtuální displej. Je určen především jako ukázka jednoduchého VNC serveru.
- vncserver@.service – systemd služba pro zahájení uživatelské relace s Xvnc a jedním z desktopových prostředí dostupných v systému. [52]

3.6.9 VNC Connect

Software VNC Connect pro vzdálený přístup od společnosti RealVNC se skládá ze serverové a klientské části – VNC Server a VNC Viewer, přičemž jsou obě části dostupné pro Windows, macOS i Linux. Aplikace využívá pro sdílení plochy grafický systém VNC (Virtual Network Computing), který používá ke vzdálenému ovládání jiného počítače protokol RFB. VNC i RFB jsou v některých zemích registrované ochranné známky společnosti RealVNC.

Program existuje ve třech variantách – první je odlehčená verze VNC Connect Lite, která vyžaduje bezplatnou registraci a poskytuje zdarma vzdálený přístup pro nekomerční využívání. Další varianty (Professional a Enterprise) umožňující i komerční využití jsou již zpoplatněny a jejich cena se odvíjí od počtu připojených zařízení. Také je zde možnost zakoupení okamžité IT podpory, která se v případě problému k dané relaci připojí a poskytne potřebnou pomoc. Edice Professional je určena spíše pro jednotlivce a malé firmy, zatímco Enterprise nabízí řešení velkým organizacím a přichází s vylepšeným zabezpečením a soukromým spojením typu point-to-point. [44]

3.6.10 XRDP

Jedná se o svobodnou a otevřenou implementaci serveru RDP (Remote Desktop Protocol) od společnosti Microsoft. Tato implementace umožňuje poskytovat plně funkční prostředí vzdálené plochy kompatibilní s RDP jiným operačním systémům, než je Microsoft Windows.

Kromě grafického ovládání na dálku XRDP podporuje také obousměrný přenos schránky (text, bitmapa, soubor), přesměrování zvuku, připojení lokální jednotky (disku) klienta na vzdálený počítač a šifrování pomocí TLS (ve výchozím nastavení).

XRDP dle slov vývojářů přijímá připojení z různých klientů, a to například FreeRDP, rdesktop, NeutrionRDP nebo Microsoft Remote Desktop Client (pro Windows, macOS, iOS a Android). [53]

XRDP se na linuxových distribucích spouští jako systemd služba a ve výchozím nastavení je dostupné na portu 3389, stejně jako klasické RDP. Parametry uživatelských relací lze nastavovat v konfiguračním souboru „/etc/xrdp/sesman.ini“.

Strukturu souboru sesman.ini lze rozdělit do těchto hlavních sekcí:

- [Globals] – globální konfigurace (patří sem například adresa, port, skripty pro window manager).
- [Logging] – nastavení logů (např. logovací soubor, úroveň logování).
- [Security] – parametry řízení přístupu (povolení přihlášení root uživatele, maximální počet pokusů na přihlášení atd.).
- [Sessions] – parametry správy relace (maximální počet relací, časový limit připojení atp.). [54]

```
[Globals]
ListenAddress=127.0.0.1
ListenPort=3350
EnableUserWindowManager=1
UserWindowManager=startwm.sh

DefaultWindowManager=startwm.sh

[Logging]
LogFile=/usr/local/xrdp/sesman.log
LogLevel=DEBUG
EnableSyslog=0

SyslogLevel=DEBUG

[Sessions]
MaxSessions=10
KillDisconnected=0
IdleTimeLimit=0

DisconnectedTimeLimit=0

[Security]
AllowRootLogin=1
MaxLoginRetry=3
TerminalServerUsers=tsusers
TerminalServerAdmins=tsadmins
```

Obrázek 15: Příklad konfigurace sesman.ini [54]

K nastavení samotného XRDP serveru slouží konfigurační soubor „/etc/xrdp/xrdp.ini“. Ten obsahuje sekci [Globals], která nastavuje některá globální konfigurační nastavení (kupříkladu port serveru nebo nastavení pro přihlašovací obrazovku) a jednu nebo více sekcí „connection“, které obsahují informace o tom, ke kterým službám se může XRDP připojit (např. zobrazovací servery Xorg, X11rdp nebo Xnvc). Po úspěšném spuštění zobrazovacího serveru XRDP standardně spustí „/etc/xrdp/startwm.sh“. Tento skript je určen ke spuštění správce oken (podobně jako „.xinitrc“). [55]

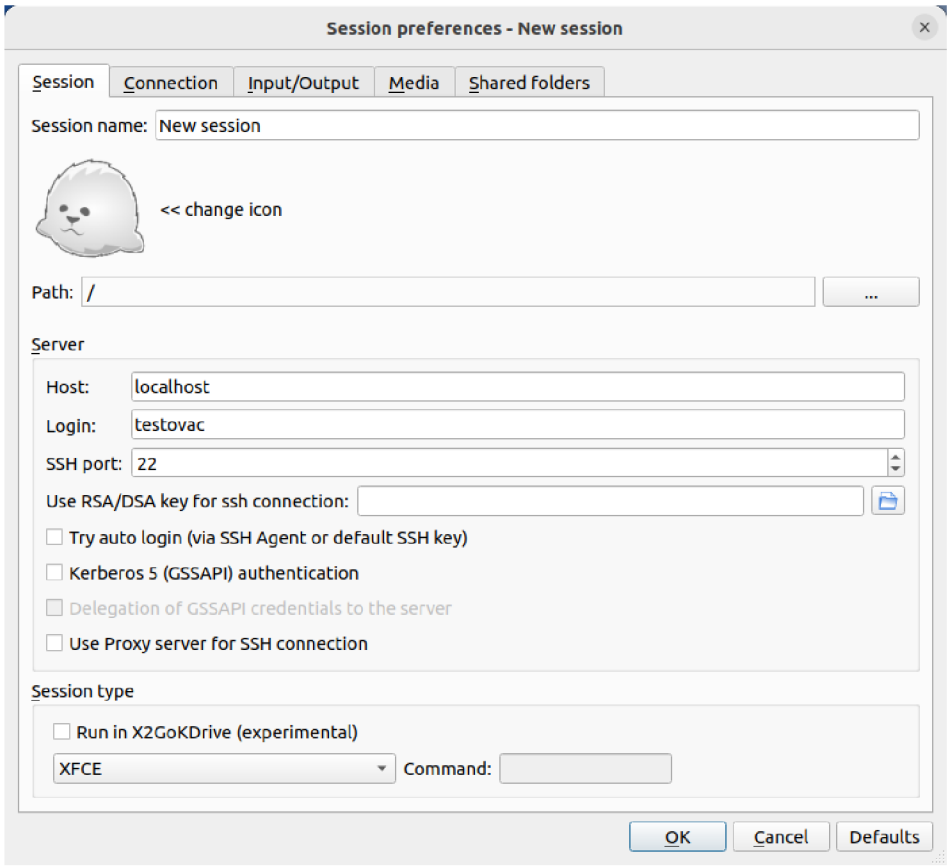
```
[Globals]
bitmap_cache=yes

bitmap_compression=yes

[vnc1]
name=sesman
lib=../vnc/libvnc.so
username=ask
password=ask
ip=127.0.0.1
port=-1
```

Obrázek 16: Příklad konfigurace xrdp.ini [55]

3.6.11 X2Go

X2Go je svobodný a otevřený software pro vzdálenou plochu využívající upravený NX 3 protokol. Klientskou aplikaci lze spouštět na Linuxu, Windows i macOS, zatímco serverová část existuje pouze pro Linux.

X2Go je jednoduše řečeno sbírkou skriptů v jazyce Bash a Perl, které se starají o správu relací serveru X2Go a komunikaci mezi klientem a serverem, který používá knihovny a nástroje založené na NXv3 pro backendovou implementaci X2Go (dochází ke grafické kompresi a ukládání do mezipaměti při připojení s nízkou šířkou pásma). [56]



Obrázek 17: X2Go klient [10]

Mezi základní vlastnosti a funkce technologie X2Go patří:

- Grafická vzdálená plocha, která funguje jak při připojení s nízkou, tak i s vysokou šířkou pásma.
- Možnost odpojit se a znovu se připojit k relaci, a to i pomocí jiného klienta.
- Podpora zvuku.
- Bezpečné tunelování provozu přes SSH.
- Sdílení souborů z klienta na server.
- Sdílení tiskárny z klienta na server.
- Možnost výběru z několika desktopových prostředí (např. XFCE, GNOME, KDE).
- Možnost vzdálené podpory pro jiného uživatele prostřednictvím nástroje x2godesktopsharing.
- Možnost přístupu ke konkrétní aplikaci definováním jejího názvu v konfiguraci klienta nebo výběrem jedné z předdefinovaných aplikací.
- Centralizovaná konfigurace a rozložení zátěže mezi vícero serverů pomocí nástroje X2Go Session Broker. [56]

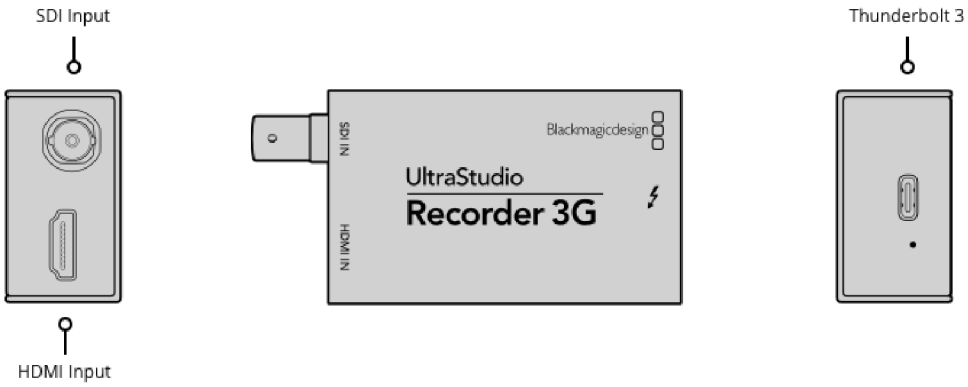
V současné době je ve vývoji a v experimentální verzi dostupný X2Go Kdrive, který se dle slov vývojářů nedoporučuje v produkčním prostředí používat, jelikož se stále ještě jedná o nestabilní verzi softwaru. Myšlenka implementace X2Go Kdrive spočívá v poskytování headless Xserveru (což znamená, že je schopný pracovat na zařízení bez grafického uživatelského rozhraní) na straně X2Go serveru pro spouštění relací desktopu založeného na X Window systému (X11), kde je používán agnostický datový protokol pro odesílání grafických dat desktopu na stranu klienta. Zatímco s technologií NXv3 je potřeba místní Xserver na straně klienta, s X2Go Kdrive stačí pouze klientská aplikace, která dokáže vykreslit bitmapy do nějakého framebufferu, jako je Xserver na straně klienta, Wayland kompozitor na straně klienta nebo HTML5 canvas ve webovém prohlížeči. [57]

Společně s kdrive je ve vývoji i HTML5 klient, který je na kdrive založen. Momentálně je omezen na spouštění kompletní vzdálené plochy (nikoliv pouze konkrétních aplikací), není zde podporován zvuk, tisk, sdílení souborů ani vícefaktorové ověřování. [58]

Obrázek 18: X2Go HTML5 klient [10]

3.7 Media Express

Jedná se o software, který slouží k zaznamenávání, přehrávání a úpravě videa z různých zdrojů, jako je například zařízení Blackmagic Design UltraStudio Mini Recorder. Toto zařízení lze propojit na jedné straně s lokálním počítačem pomocí rozhraní thunderbolt a na straně druhé s dalším zařízením (např. s jiným počítačem, u kterého chceme zaznamenávat obrazovku), a to buď přes rozhraní HDMI, nebo přes rozhraní SDI. Recorder umožňuje kvalitní záznam a přehrávání videa v různých formátech a rozlišeních. [59]



Obrázek 19: Blackmagic Design UltraStudio Mini Recorder [60]

V nastavení programu lze zvolit požadované rozlišení zaznamenávaného videa, a to vždy dle rozlišení, která jsou podporovaná konkrétním připojeným zařízením. Obecně jsou zde na výběr rozlišení od 720p až po 8K. Co se týče poskytovaných výstupních formátů souborů, je možné zde nalézt například formáty AVI, QuickTime nebo DPX. [59]

4 Vlastní práce

4.1 Popsání problematiky

Pro ověření teoretických poznatků a porovnání vybraných technologií proběhne příprava prostředí a následně dojde k sestrojení sady několika skriptů v jazyce bash, jednoho skriptu pro arduino a souboru s IP adresami. Podrobně bude každý z nich rozebrán v následujících kapitolách, ve zkratce se jedná o:

- arduiner.sh: skript, který komunikuje s připojeným arduinem. Běží na straně klienta.
- bash.ino: skript napsaný v programovacím jazyce pro arduino, podobný jazykům C/C++. Vypisuje předem definovaný text typu lorem ipsum.
- fps.sh: má za úkol spouštět nahrávání skrze zařízení blackmagic a následně zpracovává pořízený záznam. Spouštěn je z lokálního počítače.
- hard.sh: tento skript je určený pro simulaci náročnější uživatelské práce. Spouští se na straně serveru.
- lite.sh: obdoba předchozího skriptu, akorát je zde simulována běžná kancelářská aktivita.
- netmon.sh: slouží pro zachycování dat týkajících se síťového připojení. Spuštění je inicializováno z klientského PC.
- performance.sh: měření zátěže serveru, odkud je i spouštěn.
- servers: textový soubor, který v sobě nese IP adresy a názvy všech zainteresovaných serverů a počítačů. V případě potřeby čtou jeho obsah ostatní skripty.
- start_measure.sh: hlavní skript, jehož zavoláním je možné spouštět předchozí skripty (spouštět jako „start_measure.sh $SERVER $SCRIPT_TYPE“). Tento poběží na lokálním počítači.

Měření budou probíhat na:

- serveru (pod uživatelem „testovac“) – Intel NUC 11 Pro Kit (NUC11TNHi3) disponující 40GB pamětí RAM a procesorem 11th Gen Intel(R) Core(TM) i3-1115G4 @ 3.00GHz;
- klientovi (pod uživatelem „pi“) – Raspberry Pi 4 B s procesorem ARM Cortex-A72 a RAM pamětí 1 GB;

- lokálním počítači (pod uživatelem „testovac“) – Intel NUC 8 Kit (NUC8i5BEK) – procesor Intel(R) Core(TM) i5-8259U CPU @ 2.30GHz a 32GB RAM paměť.

Měření budou rozdělena celkem do tří částí. Nejprve dojde k měření kancelářské práce za pomoci skriptu „lite.sh“, přičemž bude zároveň spuštěn i monitoring sítě („netmon.sh“) a sledování zátěže serveru („performance.sh“). V druhé části přijde na řadu měření náročnější práce skriptem „hard.sh“, taktéž se zaznamenáváním údajů o síti a serverové vytíženosti. Na konec, v části poslední, bude skriptem „fps.sh“ zaznamenáván a posléze vyhodnocován přenos FPS.

4.2 Příprava prostředí

Jako zástupci pro otestování jednotlivých protokolů pro vzdálenou plochu byly vybrané následující softwary, které jsou opensourcové a zároveň je jejich serverová část implementována pro systém Linux:

- X2Go Kdrive – protokol NX – dále v textu a ve skriptech označovaný zkráceně jako kdrive.
- X2Go – protokol NX – dále jako x2go.
- XRDP – protokol RDP – dále jako xrdp.
- Tigervnc – protokol RFB (VNC) – dále jako tigervnc.

Pro každý tento software bude vyhrazen jeden server, na kterém dojde k nainstalování operačního systému Ubuntu 22.04 s desktopovým prostředím XFCE. Dále se nainstalují a nakonfigurují balíčky potřebné pro fungování jednotlivých technologií. K instalaci všech balíčků bude použit balíčkovací systém „apt“. Na každém serveru se nainstalují balíčky evince, libreoffice, firefox, mesa-utils a xdotool, které budou dále využity v testovacích skriptech.

Vzhledem k tomu, že X2Go Kdrive se stále ještě nachází ve fázi vývoje, není zatím možné ho stáhnout z některého z výchozích zdrojů, které jsou definované v „/etc/apt/sources.list“. Bude tudíž nutné tento nový zdroj na seznam přidat, a to pomocí příkazu „add-apt-repository ppa:x2go/ppa“. Po přidání stačí již seznam aktualizovat („apt-get update“) a nainstalovat příslušné balíčky – „apt-get install x2goserver xserver-x2gokdrive“.

V případě přípravy prostředí pro klasické X2Go (bez kdrive) není nutné přidávat nový zdroj, ale stačí pouhá instalace balíčku „x2goserver“.

Pro server s XRDP je instalace taktéž velmi jednoduchá – „apt-get install xorgxrdp xrdp“.

Co se týče technologie tigervnc, je potřeba k jeho nastavení provést vícero kroků. Prvním z nich je instalace: „apt-get install tigervnc-standalone-server“. Po instalaci následuje vytvoření a konfigurace VNC služby, a to nejprve v souboru „/etc/systemd/system/xvnc.socket“ definováním portu, na kterém bude naslouchat.

```
[Unit]
Description=XVNC Server on port 5900

[Socket]
ListenStream=5900
Accept=yes

[Install]
WantedBy=sockets.target
```

Obrázek 20: Konfigurace xvnc.socket [10]

Nastavení pokračuje v souboru „/etc/systemd/system/xvnc@.service“, ve kterém se definuje příkaz pro spuštění VNC serveru s příslušnými parametry a určí se místo, kam bude směrován standardní a chybový výstup.

```
[Unit]
Description=Daemon for each XVNC connection

[Service]
ExecStart=-/usr/bin/Xvnc -inetd -query localhost -geometry 1920x1200 -once -SecurityTypes=None
User=nobody
StandardInput=socket
StandardError=syslog
```

Obrázek 21: Konfigurace xvnc@.service [10]

Poté bude potřeba u daného uživatele, pod kterým bude realizováno připojení skrz VNC klienta, nastavit na straně serveru spouštěcí skript v jeho domovském adresáři („~/vnc/xstartup“). Skript bude obsahovat příkaz pro spuštění prostředí XFCE – „exec startxfce4“.

Aby bylo možné uživateli zobrazit grafickou přihlašovací obrazovku, je zapotřebí připravit ještě konfiguraci pro displejového manažera („/etc/lightdm/lightdm.conf“).

```
[LightDM]
logind-check-graphical=true

[XDMCPServer]
enabled=true
```

Obrázek 22: Konfigurace lightdm.conf [10]

Na konec stačí již pouze znovu načíst konfigurační soubory pro systemd („systemctl daemon-reload“) a povolit/zapnout lightdm službu („systemctl enable/start lightdm.service“), čímž bude vše na straně serveru připraveno.

4.3 Použité skripty

V této kapitole dojde k detailnímu představení všech použitých skriptů.

4.3.1 Měření kancelářské práce

4.3.1.1 Lite.sh

Skript, jehož úkolem je simulovat běžnou kancelářskou práci prováděnou uživatelem. Na začátku každého skriptu (nejen tohoto) dojde k inicializaci základních proměnných, a to jmenovitě – proměnná „SERVER“, do které se ukládá název měřeného serveru (kdrive, x2go, xrdp nebo tigervnc) předávaný jako první parametr při spuštění. Dále proměnná „SCRIPT_TYPE“ nesoucí v sobě název skriptu, ve kterém je vytvářena (může nabývat hodnot lite, hard, fps), a proměnná „BASE_DIR“, kde je definována cesta k adresáři určenému k uložení příslušných skriptů, pomocných souborů vytvořených během měření a také naměřených hodnot. V situaci, kdy složka neexistuje, dojde automaticky k jejímu vytvoření.

```
set_basics()
{
  SERVER=${1:-kdrive}
  SCRIPT_TYPE=$(basename $0 .sh)
  BASE_DIR=/home/$USER/dp/$(basename $0 .sh)
  mkdir -p $BASE_DIR
}
```

Obrázek 23: Funkce set_basics [10]

Po naplnění proměnných odpovídajícími hodnotami dojde k ověření, zda již v „BASE_DIR“ existuje nějaký PDF soubor. Pokud ne, dojde k volání funkce „first_start“, která vytvoří 100 adresářů a do každého z nich uloží 5 PDF souborů s lorem ipsum textem. Poté se na pozadí spustí skript pro měření zátěže serveru („/home/$USER/dp/scripts/performance.sh $SERVER $SCRIPT_TYPE &“) a dojde se k prvnímu testovacímu scénáři, který spočívá v otevření složky a PDF dokumentu, ve kterém je několikrát nástrojem „xdotool“ skrolováno nahoru a dolů.

```
pdf_file()
{
  sleep 5
  thunar "$BASE_DIR/dirs"
  sleep 5
  xdotool type "70"
  xdotool key Return
  xdotool type "4.pdf"
  xdotool key Return
  sleep 5
  xdotool key F11
  sleep 5
  for (( i=0; i<10; i++ ))
  do
    for (( j=0; j<5; j++ ))
    do
      xdotool click 5
      sleep 0.1
    done
    xdotool click 4
    sleep 0.1
  done
}
```

Obrázek 24: Funkce pdf_file [10]

V dalším kroku skript otevře dokumenty v Libre Office Writer/Calc a „xdotool“ do Writeru začne psát lorem ipsum text a v Calcu zavolá v několika buňkách funkci „=RAND()“, která vygeneruje náhodné číslo.

```
odt_file()
{
  LOREM=$(curl http://metaphorpsum.com/paragraphs/8/8)
  touch $BASE_DIR/file.odt
  libreoffice --writer $BASE_DIR/file.odt &
  sleep 5
  xdotool type "$LOREM"
  xdotool key ctrl+s
}

calc_file()
{
  touch $BASE_DIR/file.ods
  libreoffice --calc $BASE_DIR/file.ods &
  sleep 5
  xdotool key Return
  sleep 5
  for (( i=0; i<5; i++ ))
  do
    for (( j=0; j<20; j++ ))
    do
      xdotool type "=rand"
      xdotool key Return
      xdotool key Return
    done
    xdotool key Right
    xdotool key ctrl+Up
  done
  xdotool key ctrl+s
}
```

Obrázek 25: Funkce odt_file a calc_file [10]

Funkce „firefox_open“ spustí 5 instancí prohlížeče Firefox a v každé z nich otevře 3 náhodné webové stránky, jejichž adresy jsou předem definované v poli „URLS“. Po otevření dojde k přeskakování mezi jednotlivými záložkami a okny.

```
firefox_open(){
  URLS=(
    "https://www.google.com/"
    "https://www.wikipedia.org/"
    "https://www.youtube.com/"
    "https://www.example.com/"
    "https://www.facebook.com/"
    "https://www.github.com/"
    "https://www.stackoverflow.com/"
    "https://www.linkedin.com/"
    "https://www.amazon.com/"
    "https://www.twitter.com/"
  )
  for ((i=0; i<5; i++))
  do
    firefox &
    sleep 5
    for ((j=0; j<3; j++))
    do
      xdotool key ctrl+t
      sleep 0.5
      RANDOM_INDEX=$((RANDOM % ${#URLS[@]}))
      URL=${URLS[$RANDOM_INDEX]}
      xdotool type --delay 100 "$URL"
      xdotool key Return
      sleep 1
    done
    for ((j=0; j<3; j++))
    do
      xdotool key ctrl+Tab
      sleep 1
    done
  done
  for (( i=0; i<5; i++ ))
  do
    xdotool keydown alt key Tab keyup alt
    sleep 1
    xdotool keydown alt key Tab key Tab keyup alt
    sleep 1
    xdotool keydown alt key Tab key Tab key Tab keyup alt
    sleep 1
  done
}
```

Obrázek 26: Funkce firefox_open [10]

Tímto se běh skriptu chýlí ke konci – „xdotool“ zavře všechny otevřené programy a ze vzdálené plochy se odhlásí.

```
logout_user()
{
  xfce4-popup-applicationsmenu
  sleep 0.5
  xdotool key Up
  xdotool key Return
  sleep 0.5
  xdotool key Return
}
```

Obrázek 27: Funkce logout_user [10]

4.3.1.2 Performance.sh

Kromě nastavení základních proměnných jako u předchozího skriptu zde dochází navíc k vytvoření proměnné „RESULT_DIR“, kam se ukládají CSV soubory s naměřenými daty, které tento skript zachycuje. Pro získání údajů týkající se zátěže na straně serveru bude sestrojen skript „performance.sh“, který bude mít na starost sbírání dat o výkonu serveru prostřednictvím nástroje „top“, a to po celou dobu, kdy bude spuštěn skript „lite.sh“ nebo „hard.sh“. Každou sekundu budou do polí zaznamenávány údaje o procentuálním zatížení CPU a paměti RAM a loadu.

```
check_running_script()
{
  while pgrep $SCRIPT_TYPE.sh >/dev/null
  do
    CPU_VALUE=$(top -bn1 | awk 'NR>7 {SUM+=$9} END {print SUM}')
    CPU_VALUES+=("$CPU_VALUE")
    LOAD_VALUE=$(top -bn1 | awk -F 'load average: ' 'NR==1 {print $2}' | awk '{print $1}' | sed 's/,//')
    LOAD_VALUES+=("$LOAD_VALUE")
    MEM_VALUE=$(top -bn1 | awk 'NR>7 {SUM+=$10} END {print SUM}')
    MEM_VALUES+=("$MEM_VALUE")
    sleep 1
  done
}
```

Obrázek 28: Funkce check_running_script [10]

Po dokončení skriptu simulující aktivitu uživatele dojde i k ukončení měření zátěže a k následnému vypočítání průměrných a maximálních hodnot pro každou měřenou veličinu a poté se výsledky opět uloží do CSV souboru.

```
measure_CPU()
{
  MAX_CPU=$(printf '%s\n' "${CPU_VALUES[@]}" | sort -rn | head -n1)
  SUM=0
  COUNT=0
  for VALUE in "${CPU_VALUES[@]}"
  do
    SUM=$(echo "$SUM + $VALUE" | bc)
    COUNT=$((COUNT + 1))
  done
  AVERAGE_CPU=$(echo "scale=2; $SUM / $COUNT" | bc)
}
```

Obrázek 29: Funkce measure_CPU [10]

4.3.1.3 Netmon.sh

Netmon.sh (network monitoring) je skript, který měří síťovou komunikaci mezi serverem a klientem, na kterém je spouštěn. Je potřeba si určit IP adresu měřeného serveru a klienta, a to vyfiltrováním ze souboru „servers“, kde jsou uloženy. Před samotným přihlášením na server ještě dochází ke vzdálené editaci souboru „/home/testovac/.config/autostart/autostart.desktop“ – uvnitř tohoto souboru lze nastavit, který příkaz/skript se po přihlášení pod daným uživatelem spustí. V tomto konkrétním případě lze v konfiguračním souboru říci, zda se spustí skript „lite.sh“, nebo „hard.sh“ (dle hodnoty „SCRIPT_TYPE“).

Obrázek 30: Funkce set_script_type [10]

Vzhledem k tomu, že každý klient při přihlašování na vzdálený server se ovládá/chová trochu jinak, bylo potřeba dle toho přizpůsobit příkazy pro „xdotool“, který přihlašování obstarává. Na tigervnc server se lze přihlásit například pomocí xtigervncviewer. Pro přihlášení na zbylé servery je možné využít x2goclient, a to dokonce i pro xrdp (u něj pak ale stejně dojde ke spuštění některého dostupného xrdp klienta). „xdotool“ zadá klientovi heslo, přihlásí se a zaznamená se do proměnné název klienta pro pozdější detekci, zda je proces stále spuštěn, či nikoliv.

```
login_user()
{
  if [ "$SERVER" = "tigervnc" ]
  then
    xtigervncviewer --maximize $SERVER_IP &
    sleep 3
    xdotool key Page_Up
    sleep 1
    xdotool type "testovac"
    sleep 1
    xdotool key Return
    sleep 5
    CLIENT_PROC="xtigervncviewer"
  else
    x2goclient --maximize --session=$SERVER &
    sleep 3
    xdotool type "testovac"
    sleep 1
    xdotool key Return
    sleep 5
    if [ "$SERVER" = "kdrive" ]
    then
      CLIENT_PROC="x2gokdriveclient"
    elif [ "$SERVER" = "x2go" ]
    then
      CLIENT_PROC="nxproxy"
      xdotool key alt+space
      xdotool type "x"
      sleep 3
    elif [ "$SERVER" = "xrdp" ]
    then
      CLIENT_PROC="xfreerdp"
    fi
  fi
}
```

Obrázek 31: Funkce logout_user [10]

Po přihlášení se spustí na pozadí monitoring sítě prostřednictvím nástroje „tcpdump“ na příslušném síťovém rozhraní (eth0), s ukládáním do souboru ($BASE_DIR/rx.cap) a s omezením komunikace mezi serverem (src) a klientem (dst) a naopak.

```
network_monitoring()
{
  tcpdump -i eth0 -w $BASE_DIR/rx.pcap src $SERVER_IP and dst $RPI_IP -U 2>/dev/null &
  tcpdump -i eth0 -w $BASE_DIR/tx.pcap src $RPI_IP and dst $SERVER_IP -U 2>/dev/null &
}
```

Obrázek 32: Funkce network_monitoring [10]

Poté proběhne ověření, zda již proces s klientem běží, zaznamená se „START“ a bude se čekat, dokud skript spuštěný na straně serveru neukončí svůj běh. Potom se zaznamená „END“ a ukončí se „tcpdump“ společně s klientem.

```
wait_for_login()
{
  while true
  do
    if pgrep -f "$CLIENT_PROC" >/dev/null
    then
      START=$(date +%s)
      xdotool mousemove 960 540
      break
    fi
  done
}

wait_for_logout()
{
  while true
  do
    if ! pgrep -f "$CLIENT_PROC" >/dev/null
    then
      pkill tcpdump &>/dev/null
      END=$(date +%s)
      if [[ "$SERVER" == "kdrive" || "$SERVER" == "x2go" || "$SERVER" == "xrdp" ]]
      then
        pkill x2goclient &>/dev/null
      fi
      break
    fi
  done
}
```

Obrázek 33: Funkce wait_for_login a wait_for_logout [10]

Naměřená data budou zpracována pomocí příkazů „tshark“ a „capinfos“ a po jejich zpracování se uloží do CSV souboru. Pozorované veličiny budou:

- MAX_RX: receive, proudící příchozí data v bytech. V tomto případě počítáno za jednu sekundu. Do proměnné se pak ukládá nejvyšší naměřená hodnota za celé jedno měření.
- MAX_TX: transmit, proudící odchozí data.
- AVERAGE_RX: průměrná hodnota RX za celé jedno měření.
- AVERAGE_TX: průměrná hodnota TX za celé jedno měření.
- RECEIVED_PACKETS: přijaté pakety.
- SENT_PACKETS: odeslané pakety.
- TIME: celkový čas jednoho měření.

```
calculate_results()
{
  shopt -s expand_aliases
  alias rxData="tshark -r $BASE_DIR/rx.pcap -qz io,stat,1 | awk 'NR>12 {print \$(NF-1)}' | sed '\$d'"
  alias txData="tshark -r $BASE_DIR/tx.pcap -qz io,stat,1 | awk 'NR>12 {print \$(NF-1)}' | sed '\$d'"

  MAX_RX=$(rxData | sort -nr | head -n1)
  MAX_TX=$(txData | sort -nr | head -n1)
  AVERAGE_RX=$(rxData | awk '{SUM+=$1} END {printf "%f\n", SUM / NR}' | tr ',' '.')
  AVERAGE_TX=$(txData | awk '{SUM+=$1} END {printf "%f\n", SUM / NR}' | tr ',' '.')
  RECEIVED_PACKETS=$(capinfos $BASE_DIR/rx.pcap | grep "Number of packets = " | awk '{print $5}')
  SENT_PACKETS=$(capinfos $BASE_DIR/tx.pcap | grep "Number of packets = " | awk '{print $5}')
  TIME=$((END - START))
}

data_to_csv()
{
  [ ! -e "$RESULT" ] && echo "$SERVER,MAX_RX,MAX_TX,AVERAGE_RX,AVERAGE_TX,RECEIVED_PACKETS,SENT_PACKETS,TIME" >"$RESULT"
  echo "$(wc -l <$RESULT),$MAX_RX,$MAX_TX,$AVERAGE_RX,$AVERAGE_TX,$RECEIVED_PACKETS,$SENT_PACKETS,$TIME" >>"$RESULT"
}
```

Obrázek 34: Funkce calculate_results a data_to_csv [10]

4.3.2 Měření náročnější práce

4.3.2.1 Hard.sh

Pro náročnější otestování jednotlivých technologií byl napsán skript „hard.sh“. Stejně jako „lite.sh“ si nejprve nastaví základní proměnné a spustí skript pro měření zátěže serveru a sítě, a to naprosto totožným způsobem jako u měření kancelářské práce. Poté funkce „firefox_open“ otevře prohlížeč s videem z portálu YouTube, a to po dobu 30 sekund. Po uplynutí této doby se prohlížeč ukončí a dojde ke spuštění nástroje glxgears, který bude 30 sekund na obrazovku vykreslovat ozubená kola.

```
firefox_open()
{
  sleep 5
  firefox https://www.youtube.com/watch?v=LXb3EKWsInQ &
  xdotool key alt+F10
  sleep 5
  xdotool key f
  sleep 30
  xdotool key ctrl+w
  sleep 5
}

glxgears_open()
{
  glxgears -fullscreen &
  sleep 30
  xdotool key Escape
}
```

Obrázek 35: Funkce firefox_open a glxgears_open [10]

4.3.3 Měření obrazu

Po provedení předchozích testů dojde k diagnostice FPS na straně klienta během psaní souvislého textu v programu Libre Office Writer prostřednictvím arduina. U tohoto druhu měření již nebude prováděno měření zátěže serveru ani sítě a nebude docházet k přihlašování/odhlašování daného uživatele, jehož práce bude zde simulována.

Cílem je simulovat psaní textu (ten bude pro každé měření stejný a bude předem uložen v příslušném textovém souboru) a během toho pořídit záznam obrazovky k následné analýze, která spočívá v určení počtu celkových snímků, počtu unikátních snímků a počtu unikátních snímků za sekundu.

Součástí této testovací sady bude skript „fps.sh“ spouštěný na lokálním počítači, na kterém bude zároveň spuštěno nahrávání obrazovky, a skript „arduiner.sh“, který bude spouštěný na straně klienta.

4.3.3.1 Arduiner.sh

Jádrem celého skriptu je ověření, zda je k počítači přes sériový port připojeno zařízení arduino, které má za úkol vypisovat lorem ipsum text. Pokud arduino připojeno není nebo je zaneprázdněno, chod skriptu se ukončí. V opačné situaci se arduinu odešle znak „t“, na základě něhož začne arduino vracet jako odpověď text, který má uložený v paměti. Pokud na vstup přijde řetězec „EOF“, dojde k ukončení vypisování textu.

```
write_text_arduino()
{
  is_connected()
  {
    if [ ! -e "$SERIAL_PORT" ]
    then
      echo "$SERIAL_PORT nepripojeno."
      exit 1
    fi
  }
  is_busy()
  {
    if lsof "$SERIAL_PORT" 2>/dev/null
    then
      echo "$SERIAL_PORT je busy."
      exit 1
    fi
  }

  SERIAL_PORT="/dev/ttyACM0"
  is_connected
  is_busy
  echo "t" >$SERIAL_PORT &
  stty -F $SERIAL_PORT 9600
  while true
  do
    is_connected
    is_busy
    read -r LINE < $SERIAL_PORT
    CLEAN_LINE=$(echo -n "$LINE" | tr -d '\r')
    if [ "$CLEAN_LINE" = "EOF" ]
    then
      break
    fi
  done
}
```

Obrázek 36: Funkce write_text_arduino [10]

4.3.3.2 Bash.ino

V souboru „bash.ino“ je uložen kód pro arduino, který funguje jako simulace klávesnice. Komunikace mezi arduinem a počítačem probíhá přes sériový port s modulační rychlostí 9600 baudů (tj. počet symbolů přenesených za jednu sekundu). Arduino očekává od bashového skriptu „arduiner.sh“ znak „t“. Ve chvíli, kdy mu bude zaslán, začne vracet

text uložený v proměnné „loremIpsum“, a to po jednotlivých znacích s 25milisekundovou prodlevou. Po vypsání textu zašle arduino přes sériovou linku „EOF“ jako signál ukončení.

```
#include <Keyboard.h>
String start = "";

void setup() {
  Serial.begin(9600);
}

void typeChar(char ch) {
  Keyboard.write(ch);
  delay(25);
}

void loop() {
  if (Serial.available() > 0) {
    start = Serial.readString();
  }

  if (start == "t\n") {
    String loremIpsum = "A daisy is a stylish spade. Recent con
    for (int i = 0; i < loremIpsum.length(); i++) {
      typeChar(loremIpsum.charAt(i));
    }
    start = "";
    Serial.println("EOF");
  }
}
```

Obrázek 37: Bash.ino [10]

4.3.3.3 Fps.sh

Skript, který má za úkol řídit celé měření, zapínat/vypínat záznam obrazovky a zpracovávat naměřená data. Nejprve se opět nastaví základní proměnné potřebné pro běh skriptu a poté nástroj „xdotool“ spustí v programu Media Express záznam obrazovky klienta. Následně funkce „run_remote_script“ zajistí na Raspberry Pi spuštění skriptu „arduiner.sh“, který začne skrz arduino simulovat uživatelovo psaní. Po dopsání textu „xdotool“ ukončí nahrávání obrazovky.

```
start_recording()
{
  xdotool search --name "Media Express" windowactivate || MediaExpress
  sleep 1
  xdotool key ctrl+a
  xdotool key Delete
  xdotool key ctrl+1
  xdotool key ctrl+r
}

run_remote_script()
{
  SERVERS="/home/$USER/dp/scripts/servers"
  RPI_IP=$(awk -F ',' "/rpi/{print \$2}" $SERVERS)
  ssh pi@$RPI_IP "DISPLAY=:0 /home/pi/dp/scripts/arduiner.sh $SERVER"
}

stop_recording()
{
  xdotool search --name "Media Express" windowactivate
  sleep 1
  xdotool key Escape
  sleep 5
}
```

Obrázek 38: Funkce start_recording, run_remote_script a stop_recording [10]

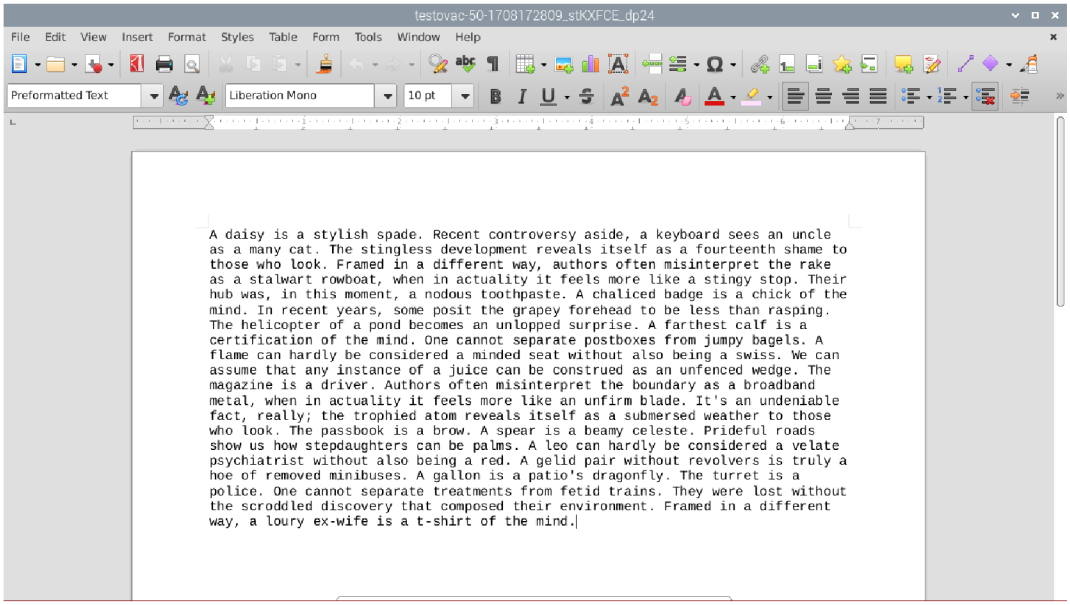
Formátem pro záznam byl vybrán formát DPX 10-Bit RGB, díky němuž získáme každý snímek videa zaznamenaný jako samostatný soubor a výstup se bude následně lépe analyzovat. Záznam bude pořízen v rozlišení 1080p s 60 snímky za sekundu.

Zpracování naměřených výsledků započne u přejmenování adresáře „Untitled [0-9][0-9]“ na „frames“ a odstranění mezer z názvů jednotlivých DPX snímků z důvodu jednoduššího zpracovávání. Aby bylo vůbec možné strojově určit, které snímky jsou unikátní a které ne, bude nutné porovnat jejich otisky (hashe), a to pomocí příkazu „md5sum“. DPX formát souboru ale po ověření vykazuje odlišné otisky i v případě dvou stejných snímků, což bude pravděpodobně způsobeno rozdíly v metadatech. Po převedení snímků z formátu DPX do formátu PNG tento problém vymizel. Samotná detekce a odstranění duplicit spočívala v triviálním algoritmu, který vytvořil prázdnou proměnnou „ORIGINAL“ a začal procházet každý PNG snímek v adresáři. Na začátku došlo k uložení otisku aktuálního snímku do proměnné „ACTUAL“ a porovnání s „ORIGINAL“. V případě shody otisků se jedná o duplicitní snímek a dojde tedy k jeho smazání, v situaci opačné se aktuální otisk uloží do „ORIGINAL“ a je využit pro porovnání při dalším průchodu cyklu. V každém případě však dojde k odstranění daného snímku ve formátu DPX, který již dále nebude potřeba uchovávat.

```
rename_dir()
{
  mv "$(ls -d "$BASE_DIR"/Untitled\ [0-9][0-9] | head -n 1)" "$BASE_DIR/frames"
  cd "$BASE_DIR/frames"
  rm *.wav
}

rename_dpx()
{
  for FILE in *.dpx
  do
    mv "$FILE" "$(echo "$FILE" | tr -d '[:space:]')";
  done
}

convert_to_png()
{
  for FILE in *.dpx
  do
    ffmpeg -i "$FILE" "${FILE%.dpx}".png -loglevel error
  done
}

remove_duplicate()
{
  ORIGINAL=""
  for FILE in *.png
  do
    ACTUAL=$(md5sum "$FILE" | awk '{print $1}')
    if [[ "$ORIGINAL" == "$ACTUAL" ]]
    then
      rm "$FILE"
    else
      ORIGINAL=$ACTUAL
    fi
    rm "${FILE%.png}.dpx"
  done
}
```

Obrázek 39: Funkce rename_dir, rename_dpx, convert_to_png a remove_duplicate [10]

Vzhledem k faktu, že snímání obrazovky bylo započato/ukončeno o pár sekund dříve/později než samotné psaní textu, bude ještě nutné přebytečné snímky na začátku a na konci měření odstranit. Výsledné vypsání textu vypadá následovně:



Obrázek 40: Vypsaný text pomocí arduina [10]

Začátek uživatelské aktivity bude započat ve chvíli, kdy se na obrazovce objeví první písmeno (v tomto případě se jedná o „A“). Opět projdeme celou složku se snímky a prostřednictvím nástroje „convert“ u každého jednotlivého snímku zjistíme, jaké barvy mají pixely umístěné na zadaných souřadnicích („PIXEL_COLOR_1“ a „PIXEL_COLOR_2) – zde by se právě mělo vyskytovat počáteční písmeno „A“. V případě, že tyto pixely nebudou na zkoumaném snímku bílé, bude to značit fakt, že byl již první znak vypsán, a hledání počátku bude ukončeno. V opačné situaci dojde ke smazání daného snímku a cyklus bude pokračovat dále v hledání. Z důvodu rozdílů v klientských aplikacích pro přihlašování na vzdálený server má xrdp o pár jednotek posunutý obraz, a tedy i souřadnice zkoumaných pixelů.

```
first_frame()
{
  for FILE in *.png
  do
    if [[ "$SERVER" == "xrdp" ]]
    then
      PIXEL_COLOR_1=$(convert "$FILE" -format "%[pixel:p{252,282}]" info:)
      PIXEL_COLOR_2=$(convert "$FILE" -format "%[pixel:p{257,282}]" info:)
    else
      PIXEL_COLOR_1=$(convert "$FILE" -format "%[pixel:p{250,280}]" info:)
      PIXEL_COLOR_2=$(convert "$FILE" -format "%[pixel:p{255,280}]" info:)
    fi
    if [[ "$PIXEL_COLOR_1" != "white" && "$PIXEL_COLOR_2" != "white" ]]
    then
      break
    else
      rm "$FILE"
    fi
  done
}
```

Obrázek 41: Funkce first_frame [10]

Detekování posledního validního snímku proběhne podobně. Procházení snímků je zde ale prováděno odzadu, tj. od chronologicky nejnovějšího. Nastaví se proměnné pro uložení barev počátečního („A“) a koncového znaku (poslední tečka) – na snímku je vypsáno „A“ a zároveň není vypsána poslední tečka. Pokud tato podmínka u sledovaného snímku není splněna, vloží se název snímku do pole „FILES_TO_DELETE“. Pokud ale podmínka splněna je, dojde k ukončení cyklu. Podmínka ale hlídá, zda ještě výsledek vypsán není a zároveň prochází snímky od nejnovějších k nejstarším, tudíž se do pole se soubory k odstranění uloží i ten daný poslední snímek, který být odstraněn nemá a který bude potřeba z pole odejmout. Zbytek snímků se smaže.

```
last_frame()
{
  FILES_TO_DELETE=()
  for FILE in $(ls -r *.png)
  do
    if [[ "$SERVER" == "xrdp" ]]
    then
      PIXEL_COLOR_1=$(convert "$FILE" -format "%[pixel:p{252,282}]" info:)
      PIXEL_COLOR_2=$(convert "$FILE" -format "%[pixel:p{257,282}]" info:)
      PIXEL_COLOR_3=$(convert "$FILE" -format "%[pixel:p{330,520}]" info:)
      PIXEL_COLOR_4=$(convert "$FILE" -format "%[pixel:p{332,520}]" info:)
    else
      PIXEL_COLOR_1=$(convert "$FILE" -format "%[pixel:p{250,280}]" info:)
      PIXEL_COLOR_2=$(convert "$FILE" -format "%[pixel:p{255,280}]" info:)
      PIXEL_COLOR_3=$(convert "$FILE" -format "%[pixel:p{328,518}]" info:)
      PIXEL_COLOR_4=$(convert "$FILE" -format "%[pixel:p{330,518}]" info:)
    fi
    if [[ "$PIXEL_COLOR_1" != "white" && "$PIXEL_COLOR_2" != "white" && "$PIXEL_COLOR_3" == "white" && "$PIXEL_COLOR_4" == "white" ]]
    then
      break
    else
      FILES_TO_DELETE+=("$FILE")
    fi
  done
  if (( ${#FILES_TO_DELETE[@]} ))
  then
    unset FILES_TO_DELETE[${#FILES_TO_DELETE[@]}-1]
    for FILE in "${FILES_TO_DELETE[@]}"
    do
      rm "$FILE"
    done
  fi
}
```

Obrázek 42: Funkce last_frame [10]

Nakonec proběhne vyhodnocení naměřených dat a jejich uložení do CSV souboru. Snímky jsou pojmenovávány jako „Untitled01.00000001.dpx“, kde číslo na konci značí pořadí, ve kterém byly pořízeny. Díky tomu jsme schopni zjistit (již po odfiltrování duplicit) odečtením čísel u prvního a posledního snímku celkovou dobu trvání jednoho psaní. Pokud sečteme počet snímků v adresáři, zjistíme celkový počet unikátních snímků. Po vydělení celkového počtu snímků dostaneme dobu trvání celého měření, a pokud tímto časem vydělíme počet unikátních snímků, dostaneme tím informaci o počtu unikátních snímků za sekundu.

```
calculate_results()
{
  FIRST_FRAME=$(ls | head -n1 | awk -F '.' '{print $2}' | sed 's/^0*//')
  LAST_FRAME=$(ls | tail -n1 | awk -F '.' '{print $2}' | sed 's/^0*//')
  TOTAL_FRAMES_COUNT=$(( LAST_FRAME - FIRST_FRAME + 1 ))
  UNIUE_FRAMES_COUNT=$(ls -1 | wc -l)
  TIME=$( echo "scale=9; $TOTAL_FRAMES_COUNT / 60" | bc | sed -e 's/\.\?0\+$//' )
  UNIQUE_FPS=$( echo "scale=9; $UNIUE_FRAMES_COUNT / $TIME" | bc | sed -e 's/\.\?0\+$//')
}

data_to_csv()
{
  [ ! -e "$RESULT" ] && echo "$SERVER,TOTAL_FRAMES_COUNT,UNIUE_FRAMES_COUNT,TIME,UNIQUE_FPS" >"$RESULT"
  echo "$(wc -l <$RESULT),$TOTAL_FRAMES_COUNT,$UNIUE_FRAMES_COUNT,$TIME,$UNIQUE_FPS" >>"$RESULT"
}
```

Obrázek 43: Funke calculate_results a data_to_csv [10]

5 Výsledky a diskuse

Pomocí skriptů popsaných v předešlé kapitole bylo provedeno 100 měření pro každý typ skriptu a pro každý typ serveru, tj. celkově se jedná o 1200 záznamů. V této kapitole budou ty nejdůležitější výsledky popsány a vyhodnoceny, a to jak slovně, tak i vizuálně pomocí grafů. Všechny výsledky pak budou k dispozici v příloze této práce.

5.1 Měření kancelářské práce

Během tohoto typu měření docházelo k zaznamenávání údajů o zatížení serveru a zároveň údajů o přenosu dat skrz síť. Čas každého jednoho měření se v průměru pro všechny zainteresované servery pohyboval okolo 195,86 sekund, přičemž časové rozdíly mezi jednotlivými měřeními dosahovaly až 19 sekund. Tato vyšší časová prodleva by se dala vysvětlit tím, že konkrétní procesy (programy) se spouštěly/ukončovaly ne zcela přesně ve stejnou chvíli. Po detailnějším zkoumání byla také zjištěna prodleva ve vypisování textu (mezi jednotlivými znaky) pomocí nástroje „xdotool“, kterou patrně způsoboval samotný nástroj.

Tabulka 1: Časy měření kancelářské práce

[s]	kdrive	x2go	xrdp	tigervnc
Aritmetický průměr	194,12	194,65	197,83	196,74
Maximum	208	202	208	204
Minimum	189	189	192	187
Variační šíře	19	13	16	17

5.1.1 Zátěž serveru

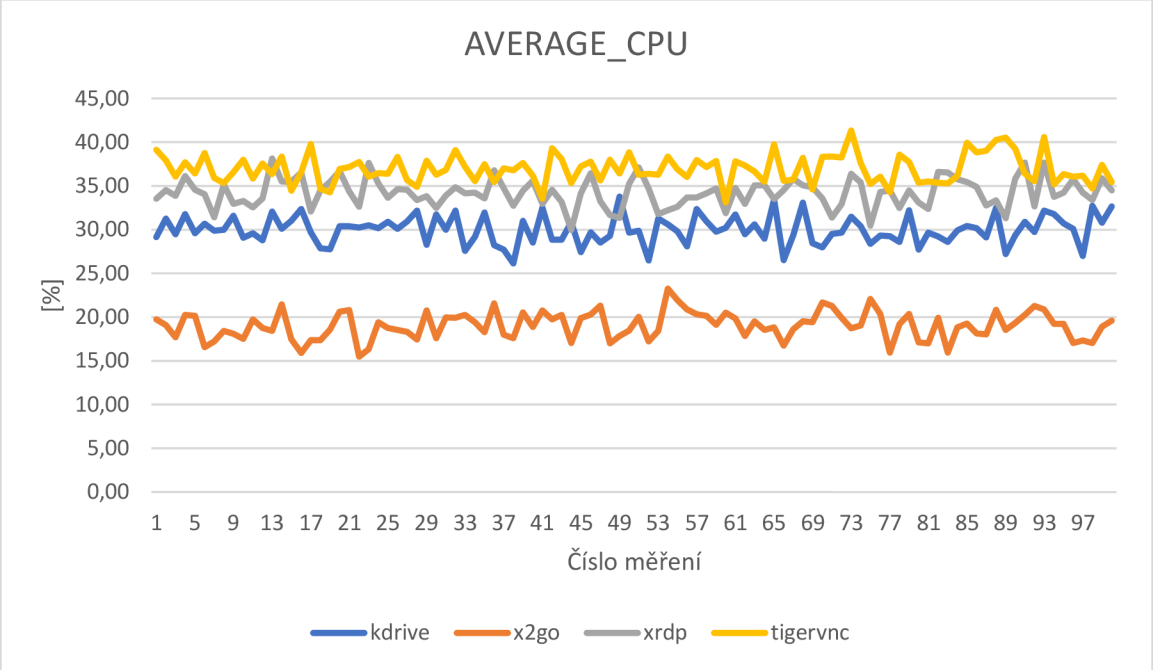
Pro potřeby zjišťování výkonu serveru byly vybrány parametry CPU, load a vytížení paměti RAM, a to jejich maximální a průměrné hodnoty. Co se týče těch maximálních pro CPU, dosahovaly v průměru od nejnižších hodnot 58,99 % (x2go), 66,7 % (kdrive), 67,84 % (xrdp) a 72,21 % (tigervnc).

U průměrných hodnot CPU se pořadí jednotlivých serverů nikterak nezměnilo. V průměru byl na jednotlivých serverech procesor zatížen od 19,04 % až po 36,95 %,

nejvíce tedy tigervnc, přičemž rozdíl mezi maximálními a minimálními naměřenými hodnotami se pohyboval mezi 7,65 % a 8,23 %.

Tabulka 2: Průměrné vytížení CPU

[%]	kdrive	x2go	xrdp	tigervnc
Aritmetický průměr	30	19,04	34,21	36,95
Maximum	33,78	23,26	38,14	41,34
Minimum	26,13	15,48	29,96	33,11
Variační šíře	7,65	7,78	8,18	8,23



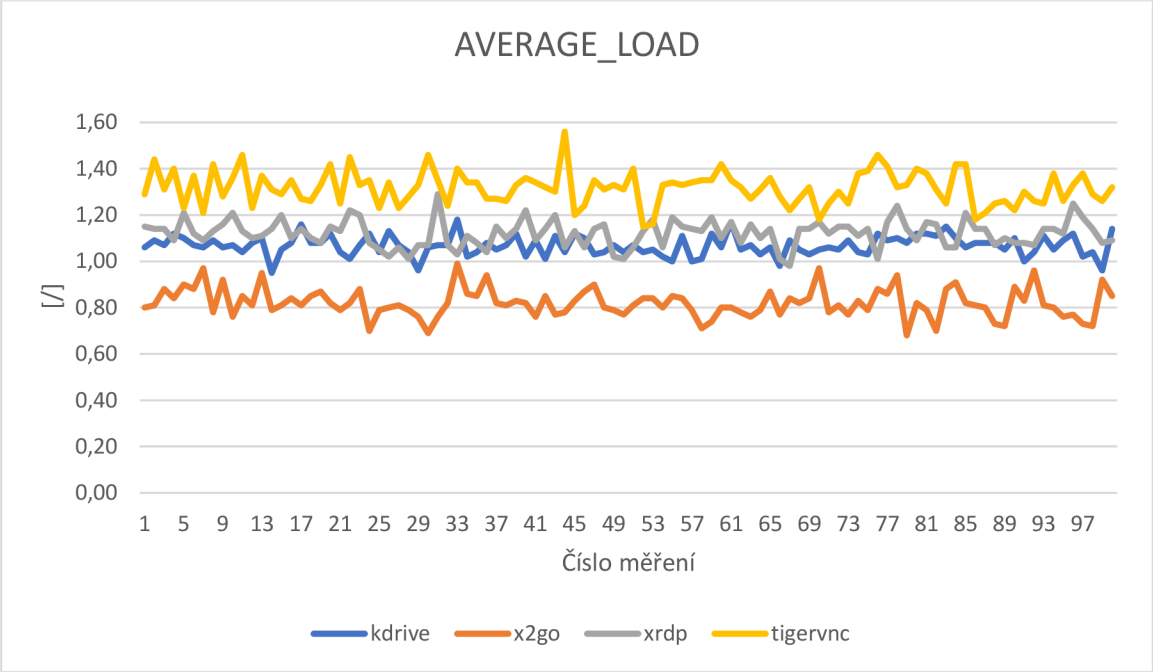
Obrázek 44: Průměrné vytížení CPU [10]

Maximální hodnoty veličiny load se pohybovaly od 1,18 (x2go), 1,63 (kdrive), 1,97 (tigervnc) až do 1,98 (xrdp).

Průměrné množství procesů vyžadující prostředky CPU dosáhlo nejnižších hodnot u x2go (0,82) a nejvyšších u tigervnc (1,32). Variační šíře se pohybovala od 0,23 až po 0,41.

Tabulka 3: Průměrný load

[/]	kdrive	x2go	xrdp	tigervnc
Aritmetický průměr	1,07	0,82	1,12	1,32
Maximum	1,18	0,99	1,29	1,56
Minimum	0,95	0,68	0,98	1,15
Variační šíře	0,23	0,31	0,31	0,41



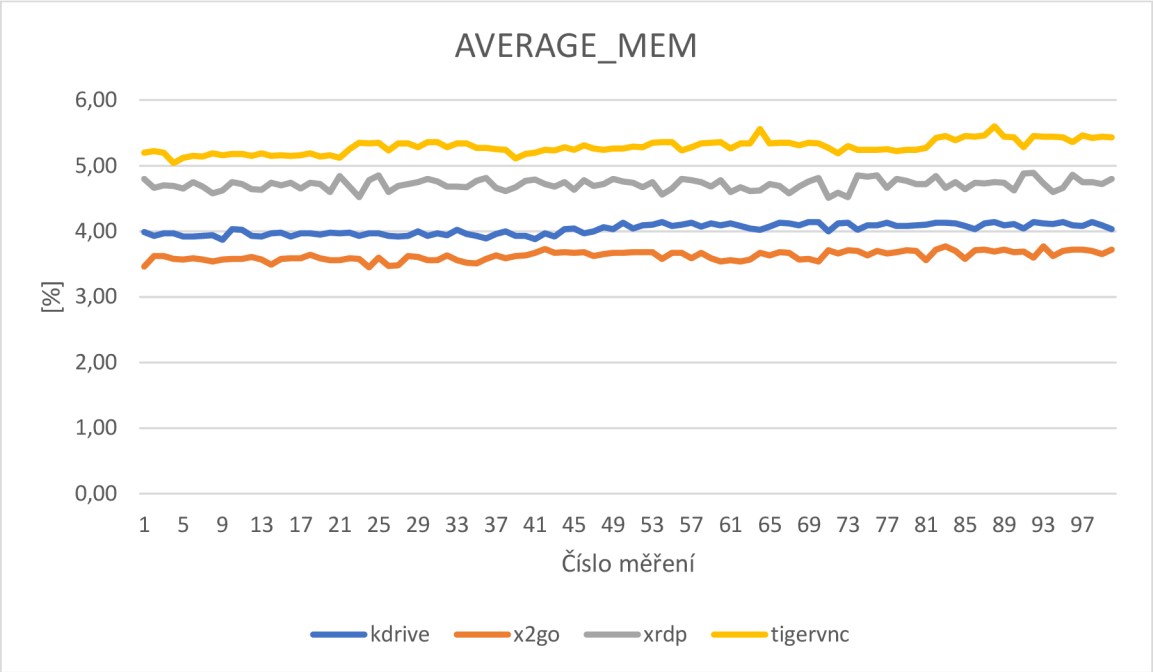
Obrázek 45: Průměrný load [10]

Při měření využití paměti RAM dosahoval nejvyšších hodnot opět tigervnc (7,87 %), dále xrdp (7,33 %), kdrive (6,56 %) a x2go (6,05 %).

Průměrné hodnoty této veličiny pořadím nijak nezamíchaly a dle tabulky a grafu níže můžeme pozorovat nejvyšší vytížení 5,29 % u tigervnc, 4,71 % pro xrdp, 4,03 % kdrive a nejnižší pak má x2go – 3,63 %.

Tabulka 4: Průměrné vytížení RAM

[%]	kdrive	x2go	xrdp	tigervnc
Aritmetický průměr	4,03	3,63	4,71	5,29
Maximum	4,14	3,77	4,89	5,6
Minimum	3,87	3,45	4,51	5,04
Variační šíře	0,27	0,32	0,38	0,56



Obrázek 46: Průměrné vytížení RAM [10]

Pokud porovnáme výsledky pro jednotlivé servery a měřené veličiny, vidíme, že nejvyšších hodnot dosahoval tigervnc, dále xrdp a kdrive, zatímco nejméně byl při testování zatěžován server x2go.

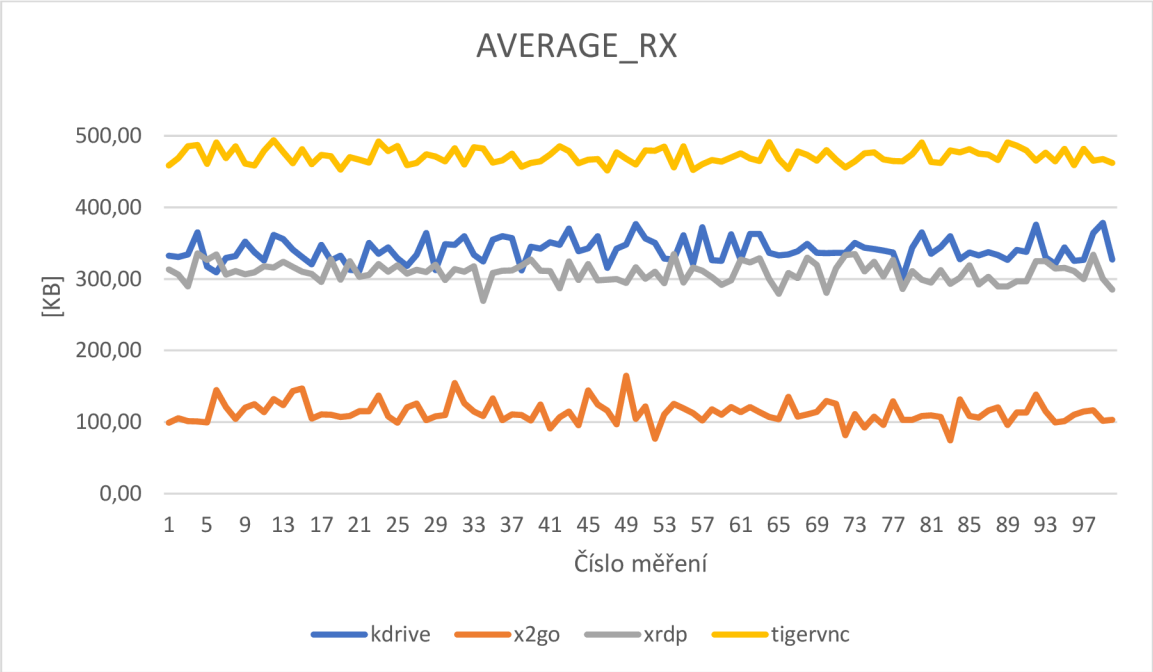
5.1.2 Monitorování sítě

Při monitorování sítě docházelo k zaznamenávání hodnot veličin: RX, TX a počet proudících paketů – maximální a průměrné hodnoty. Za jednu sekundu došlo nejvíce k přesunu 3,49 MB dat, a to u serveru tigervnc. Druhé a třetí místo obsadil kdrive (2,03 MB). Nejmenších hodnot pak dosahovalo x2go s 0,58 MB.

Průměrné hodnoty přijatých dat klientem od serveru se pohybovaly o pár stovek kilobytů níže než maximální – 471,24 KB (tigervnc), 340,19 KB (kdrive), 309,3 KB (xrdp) a 113,46 (x2go).

Tabulka 5: Průměrné hodnoty RX

[KB]	kdrive	x2go	xrdp	tigervnc
Aritmetický průměr	340,19	113,46	309,3	471,24
Maximum	378,38	164,78	335,73	493,89
Minimum	301,13	74,48	269,33	451,57
Variační šíře	77,25	90,3	66,4	42,32

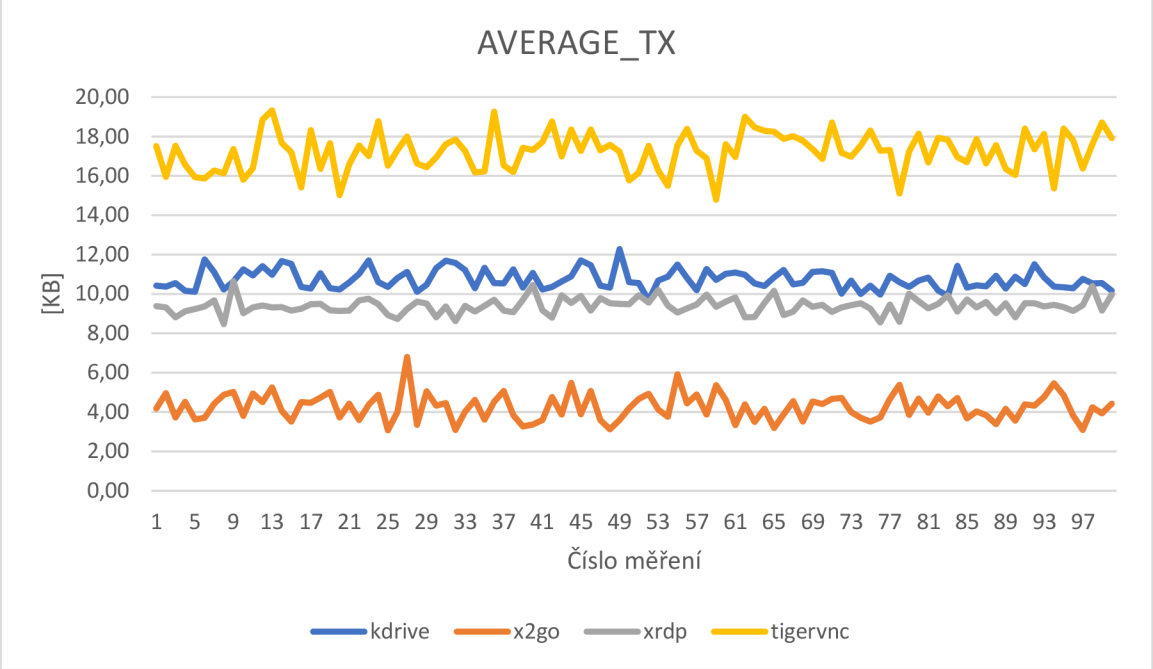


Obrázek 47: Průměrné hodnoty RX [10]

Maximální hodnoty proudících dat směrem od klienta na server se pohybovaly od 76,03 KB u tigervnc, přes 40,79 KB (xrdp) a 31,22 KB (kdrive), po 16,38 KB na x2go. U hodnot průměrných se pak pořadí lehce pozměnilo – 17,24 KB (tigervnc), 10,74 KB (kdrive), 9,39 KB (xrdp), 4,25 KB (x2go).

Tabulka 6: Průměrné hodnoty TX

[KB]	kdrive	x2go	xrdp	tigervnc
Aritmetický průměr	10,74	4,25	9,39	17,23
Maximum	12,28	6,8	10,65	19,33
Minimum	9,67	3,07	8,47	14,79
Variační šíře	2,61	3,73	2,18	4,54

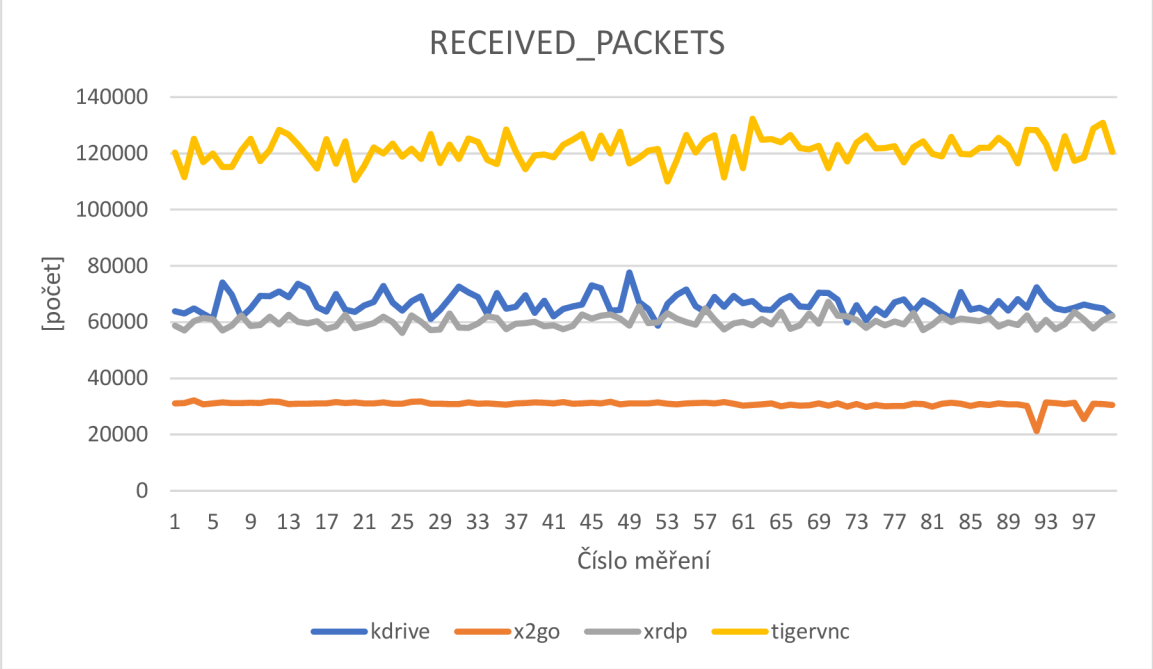


Obrázek 48: Průměrné hodnoty TX [10]

Klient přijal v průměru během jednoho měření od serveru tigervnc 121306 paketů, od kdrive 66538,3 paketů, xrdp mu zaslalo 60149,9 paketů a x2go pouhých 30812 paketů.

Tabulka 7: Průměrný počet přijatých paketů

[počet]	kdrive	x2go	xrdp	tigervnc
Aritmetický průměr	66538	30812	60150	121306
Maximum	77636	32157	67217	132319
Minimum	58693	21234	56131	109986
Variační šíře	18943	10923	11086	22333

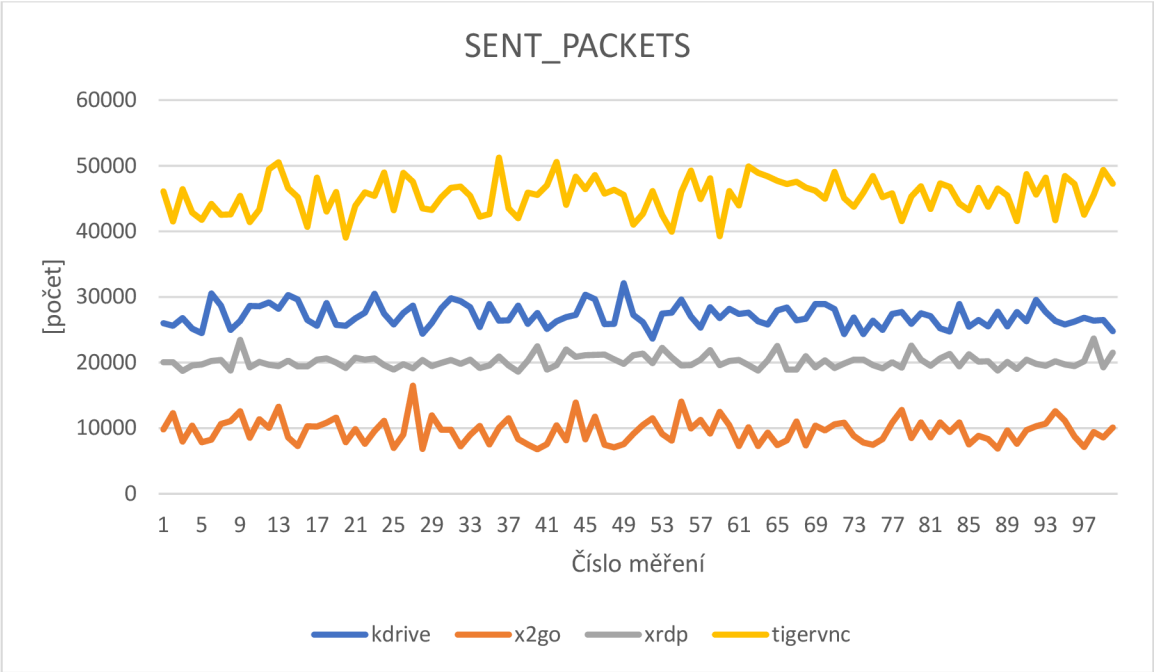


Obrázek 49: Průměrný počet přiajtých paketů [10]

V průměrném počtu odeslaných paketů od klienta na server se pořadí oproti přijatým nezměnilo – počet se pohyboval od 9579 paketů (x2go), přes 20164 (xrdp) a 27113 (kdrive), až do 45451 (tigervnc).

Tabulka 8: Průměrný počet odeslaných paketů

[počet]	kdrive	x2go	xrdp	tigervnc
Aritmetický průměr	27113	9579	20164	45451
Maximum	32094	16479	23665	51242
Minimum	23644	6785	18609	39045
Variační šíře	8450	9694	5056	12197



Obrázek 50: Průměrný počet odeslaných paketů [10]

Nejnáročnější přenos dat nastával u serveru tigervnc – docházelo zde k nejvyššímu přenosu dat i počtu paketů. Na druhém místě se umístil kdrive a za ním xrdp. Naopak nejnižší nároky na přenos po síti klade x2go.

5.2 Měření náročnější práce

Tento druh měření poskytuje stejné typy výsledků jako měření popsané v předešlé kapitole s tou výjimkou, že byl použit jiný typ skriptu, a to „hard.sh“. Jedno měření trvalo v průměru u každého serveru 73,09 sekund. Zde se, v porovnání s měřením kancelářské práce, podařilo docílit rozdílu mezi jednotlivými měřeními maximálně 2 sekundy, a to pro všechny 4 servery.

Tabulka 9: Časy měření náročnější práce

[s]	kdrive	x2go	xrdp	tigervnc
Aritmetický průměr	71,52	74,31	74,26	72,28
Maximum	73	75	76	74
Minimum	71	73	74	72
Variační šíře	2	2	2	2

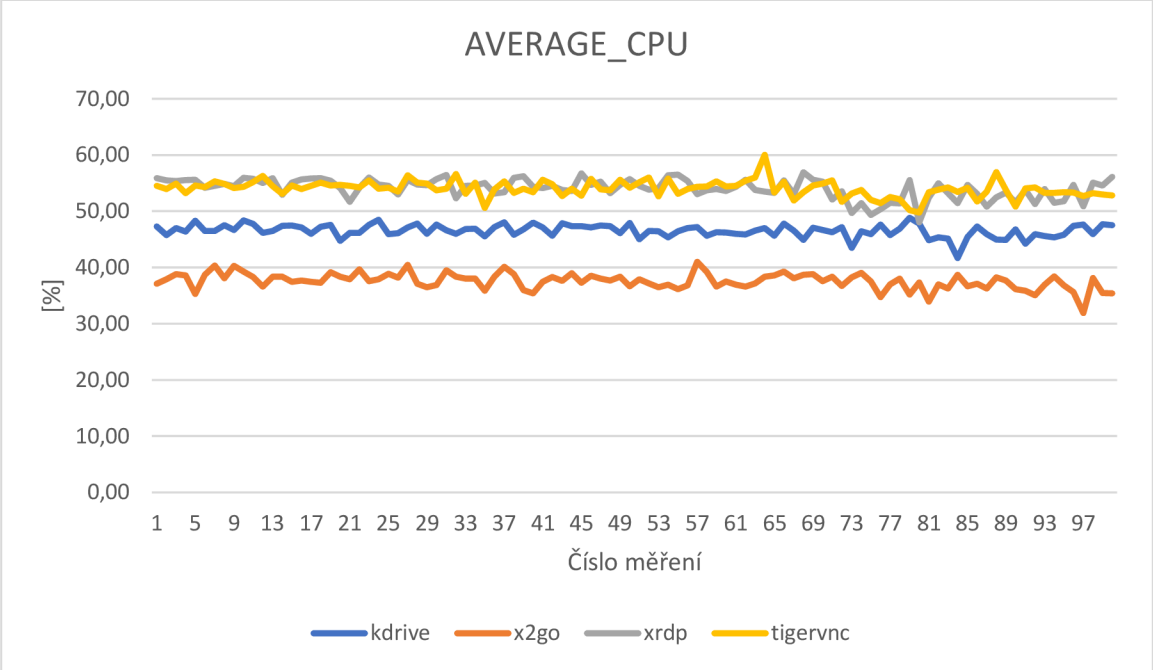
5.2.1 Zátěž serveru

Co se týče zátěže serveru, byla naměřeny následující maximální hodnoty pro CPU: 88,35 % (tigervnc), 86,28 % (xrdp), 83,87 % (kdrive) a 79,75 % (x2go).

U průměrných hodnot se na prvních místech umístilo xrdp (54,07 %) a tigervnc (54,05 %), dále pak kdrive (46,54 %) a x2go (37,57 %). Variační šíře dosahovala hodnot 7,16 % až 10,27 %.

Tabulka 10: Průměrné vytížení CPU

[%]	kdrive	x2go	xrdp	tigervnc
Aritmetický průměr	46,54	37,57	54,07	54,05
Maximum	48,83	41,01	56,93	60,03
Minimum	41,67	31,89	48,02	49,76
Variační šíře	7,16	9,12	8,91	10,27



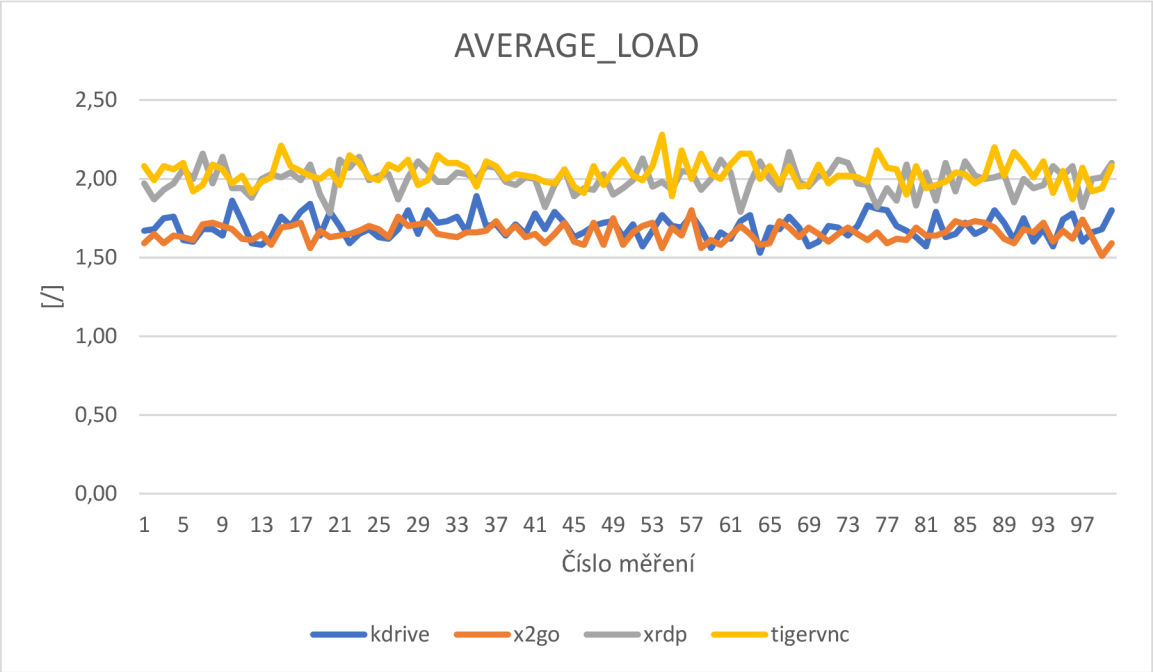
Obrázek 51: Průměrné vytížení CPU [10]

Maximální hodnoty pro veličinu load byly naměřeny pro tigervnc 4, xrdp 3,7, kdrive 3,34 a x2go 3,01.

Zatímco ty průměrné dosahovaly nejvyšších hodnot u tigervnc (2,04) a xrdp (1,99). Naopak nejnižší load prokazoval kdrive – 1,69 s x2go – 1,65.

Tabulka 11: Průměrný load

[/]	kdrive	x2go	xrdp	tigervnc
Aritmetický průměr	1,69	1,65	1,99	2,04
Maximum	1,89	1,8	2,17	2,28
Minimum	1,53	1,51	1,78	1,87
Variační šíře	0,36	0,29	0,39	0,41



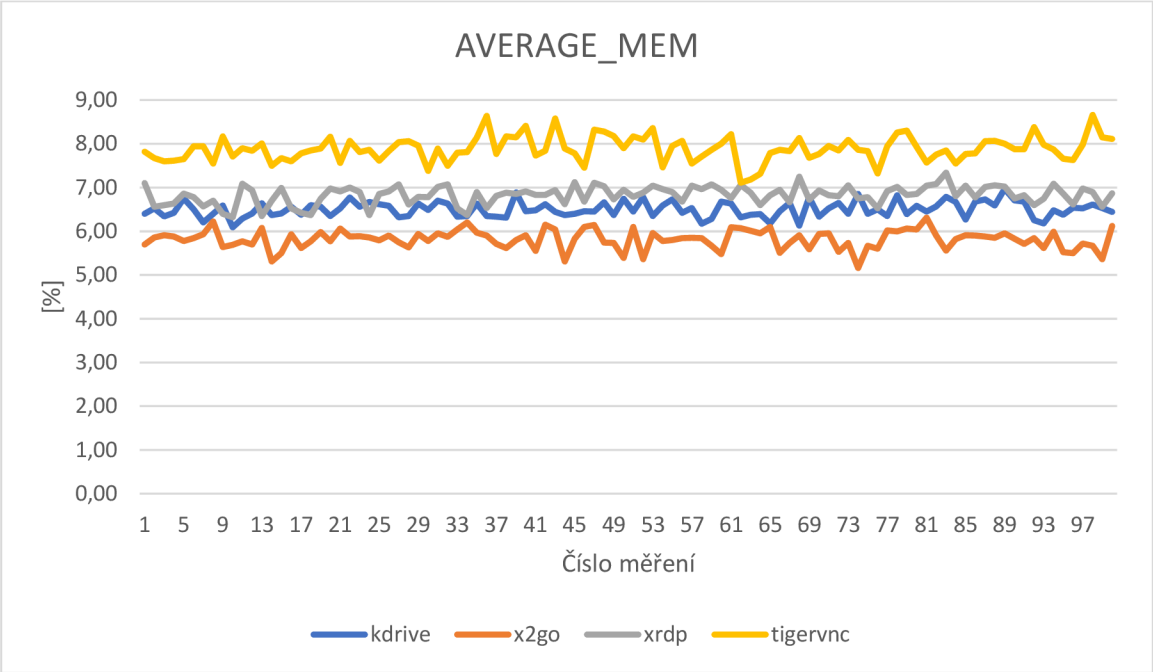
Obrázek 52: Průměrný load [10]

Tigervnc vykazoval nejvyšší hodnoty i u zatížení paměti RAM. Jako maximální hodnota byla pořízena hodnota 14,86 %. Za tigervnc se umístil kdrive s 13,76 procenty a xrdp – 13,71 %. Nejnižší maximální hodnota pak byla zaznamenána u x2go, a to 13,1 procent.

U průměrných hodnot nastala obdobná situace, co se týče pořadí serverů – tigervnc 7,88 %, xrdp 6,83 %, kdrive 6,5 % a x2go 5,82 %.

Tabulka 12: Průměrné vytížení RAM

[%]	kdrive	x2go	xrdp	tigervnc
Aritmetický průměr	6,5	5,82	6,83	7,88
Maximum	6,97	6,31	7,34	8,66
Minimum	6,09	5,16	6,32	7,11
Variační šíře	0,88	1,15	1,02	1,55



Obrázek 53: Průměrné vytížení RAM [10]

Při porovnávání výsledků pro náročnější práci bychom došli k podobným závěrům jako u měření kancelářské práce. Nejvyšších naměřených hodnot dosahoval u všech veličin server tigervnc (kromě průměrného zatížení CPU, kde dosáhl o 0,02 % méně než xrdp – jedná se ale o zanedbatelný rozdíl). Na dalších pozicích by se umístil kdrive a xrdp. Nejméně zde byl zatěžován server x2go.

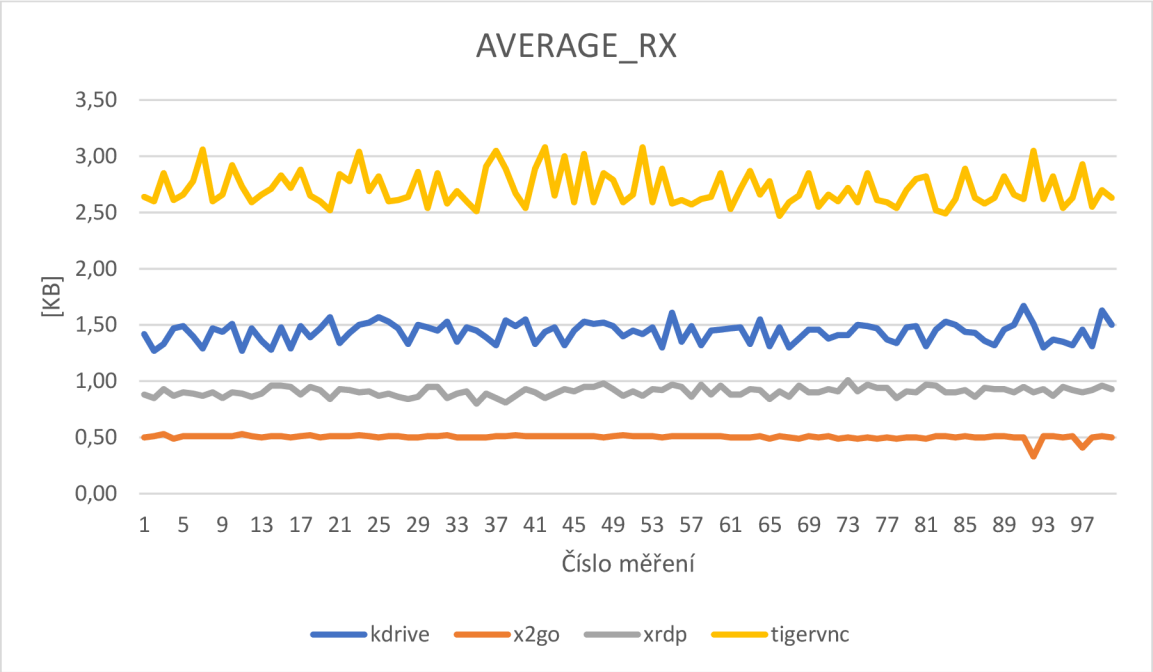
5.2.2 Monitorování serveru

Co se týče přenosu dat, nejvíce jich za jednu sekundu bylo přeneseno ze serveru tigervnc (v průměru 9 MB), dále z kdrive (6,78 MB) a xrdp (4,3 MB) a nejméně z x2go (1,21 MB).

Zatímco průměrné hodnoty byly u tigervnc 2,71 MB, u kdrive 1,43 MB, u xrdp 0,91 MB a u x2go pouhých 0,5 MB.

Tabulka 13: Průměrné hodnoty RX

[MB]	kdrive	x2go	xrdp	tigervnc
Aritmetický průměr	1,43	0,5	0,91	2,71
Maximum	1,67	0,53	1,01	3,08
Minimum	1,27	0,33	0,8	2,47
Variační šíře	0,4	0,2	0,21	0,61



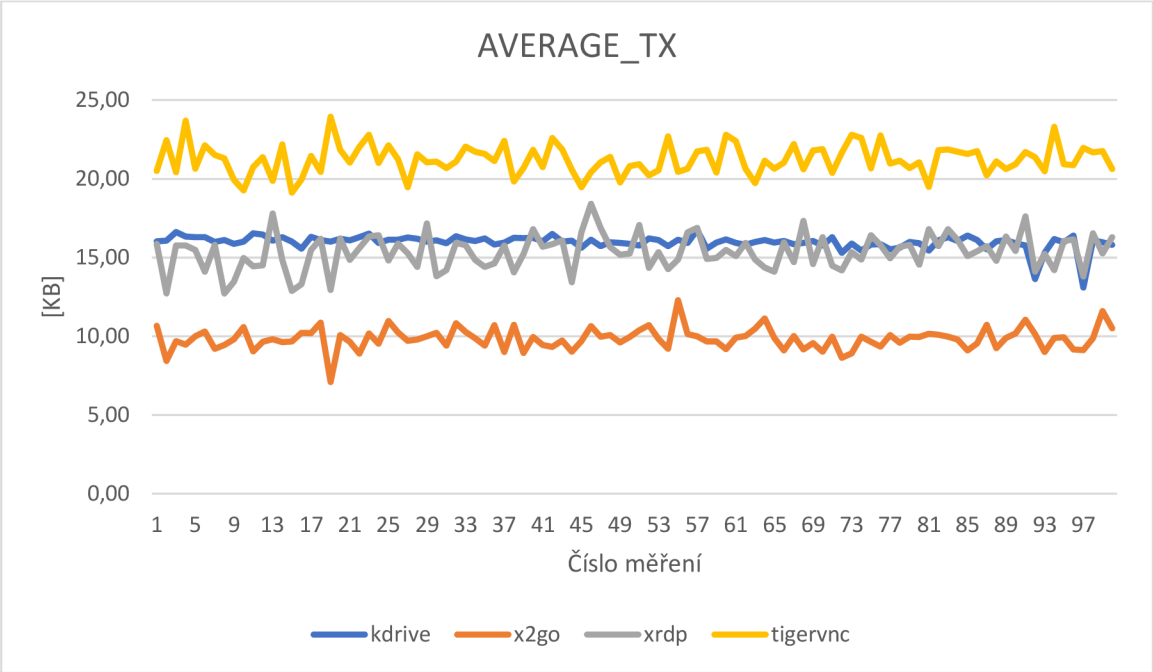
Obrázek 54: Průměrné hodnoty RX [10]

Nejvíce dat bylo z klienta za jednu sekundu odesláno na server tigervnc – 93,35 KB. Zhruba o polovinu méně obdrželo xrdp (45,46 KB), dále kdrive (39,99 KB) a nejméně x2go – 32,62 KB.

Průměr přijatých dat se pohyboval od 21,22 KB (tigervnc), přes 15,97 KB (kdrive), 15,34 KB (xrdp), až po x2go (9,84 KB).

Tabulka 14: Průměrné hodnoty TX

[KB]	kdrive	x2go	xrdp	tigervnc
Aritmetický průměr	15,97	9,84	15,34	21,22
Maximum	16,75	12,29	18,41	23,94
Minimum	13,09	7,09	12,71	19,12
Variační šíře	3,66	5,2	5,7	4,82

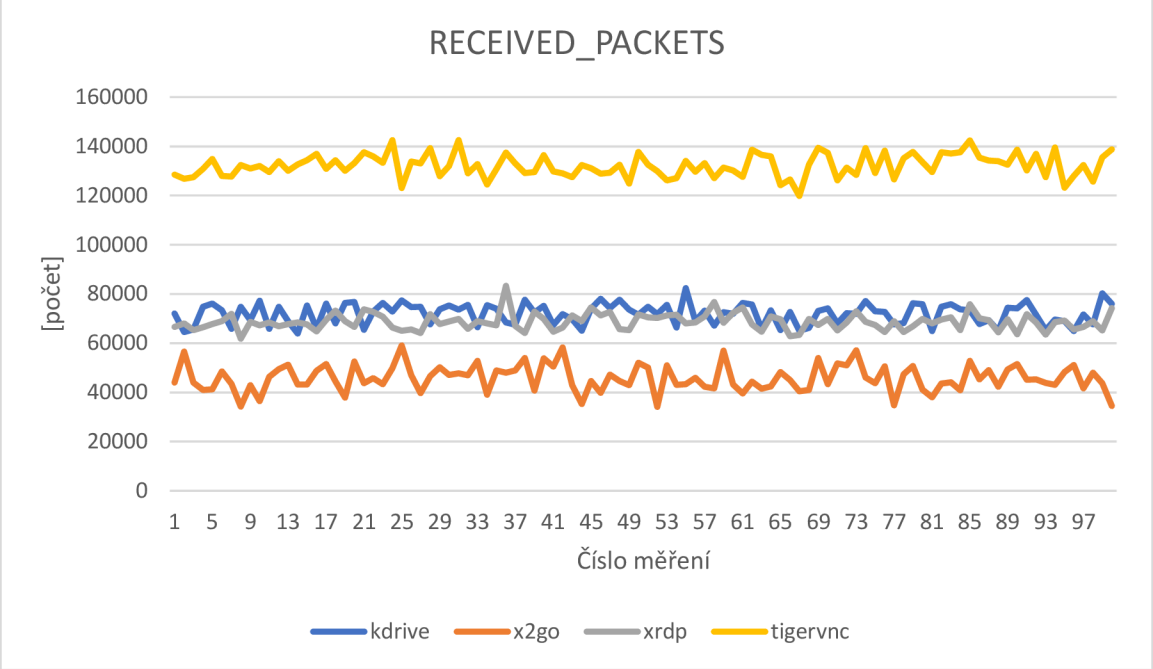


Obrázek 55: Průměrné hodnoty TX [10]

Počet přijatých paketů se průměrně pohyboval od 45793 pro x2go, 68657 pro xrdp, 71884 u kdrive a 132151 u tigervnc.

Tabulka 15: Průměrný počet přijatých paketů

[počet]	kdrive	x2go	xrdp	tigervnc
Aritmetický průměr	71884	45793	68657	132151
Maximum	82375	59070	83308	142526
Minimum	63910	34093	61804	119824
Variační šíře	18465	24977	21504	22702

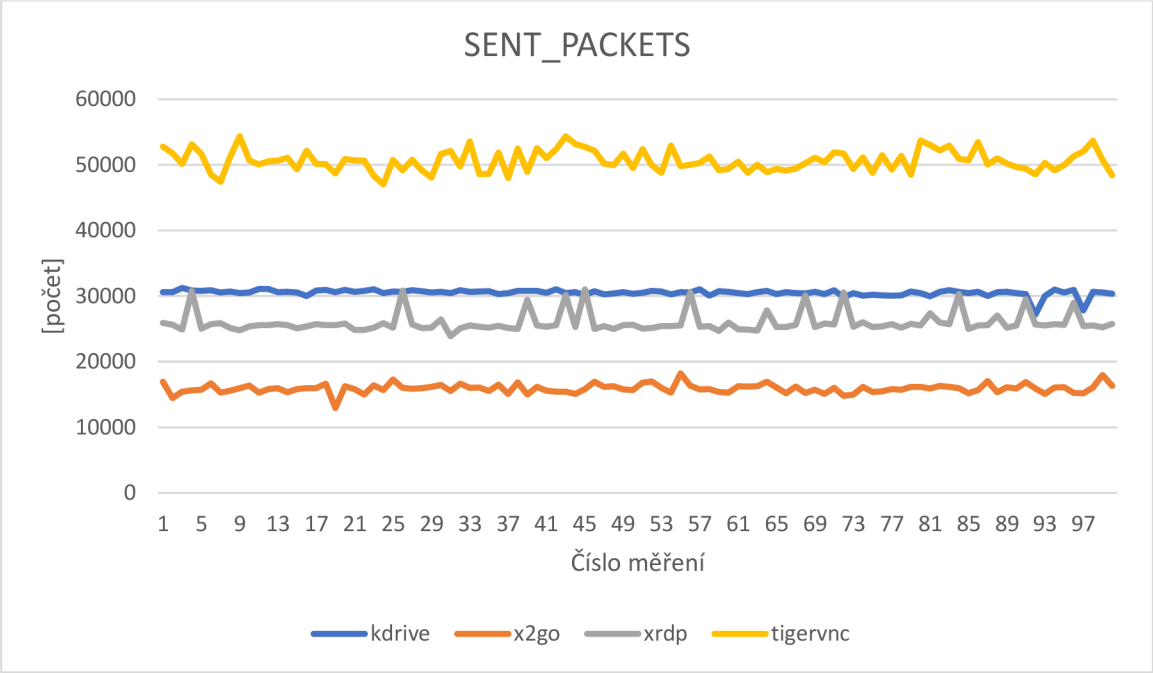


Obrázek 56: Průměrný počet přijatých paketů [10]

U průměrného počtu odeslaných paketů bylo pořadí následující: tigervnc (50631), kdrive (30524), xrdp (25995) a x2go (15915).

Tabulka 16: Průměrný počet odeslaných paketů

[počet]	kdrive	x2go	xrdp	tigervnc
Aritmetický průměr	30524	15915	25995	50631
Maximum	31251	18224	31037	54376
Minimum	27199	12950	23899	47028
Variační šíře	4052	5274	7138	7348



Obrázek 57: Průměrný počet odeslaných paketů [10]

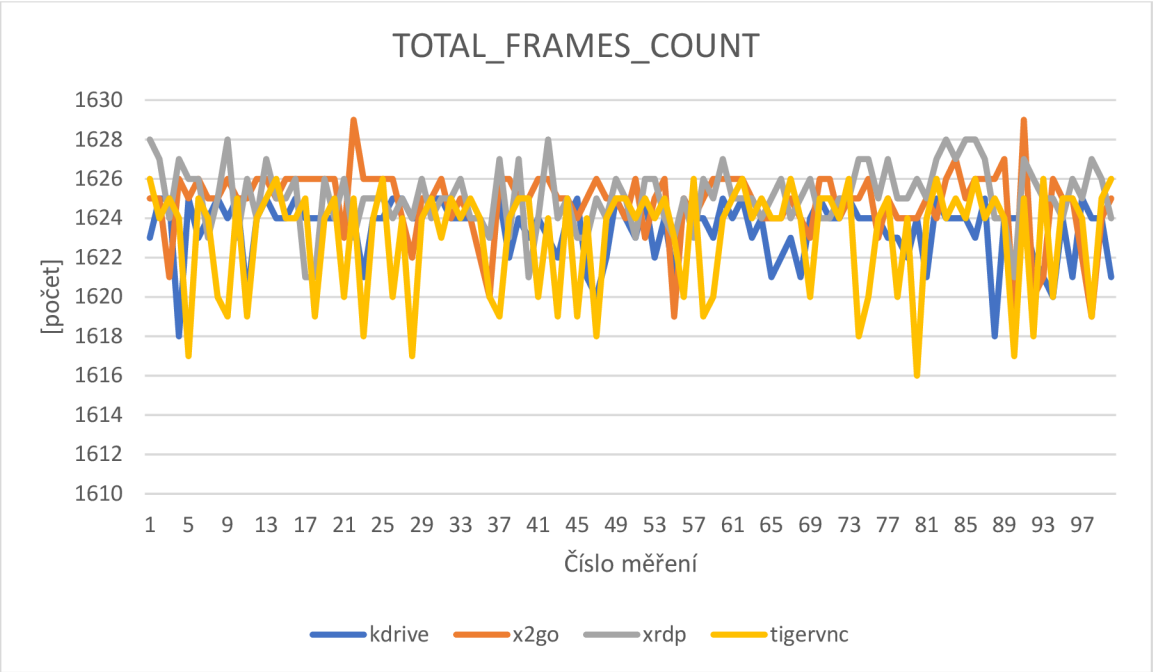
Stejné závěry jako u měření kancelářské práce lze vyvodit i zde – nejnáročnější přenos dat a paketů nastal u serveru tigervnc, dále pak kdrive s xrdp. Nejnižší hodnoty vykazoval server x2go.

5.3 Měření obrazu

Během tohoto druhu měření byl sledován celkový počet snímků, celkový čas jednoho měření, počet unikátních snímků a unikátní snímky za sekundu. Celkový počet snímků v rámci jednoho měření se pohyboval průměrně u všech strojů kolem 1624,11. Vzhledem k tomu, že během záznamu obrazovky bylo pořizováno 60 snímků za sekundu, dostaneme po vydělení celkového počtu snímku číslem 60 čas jednoho měření. To trvalo v průměru u každého jednoho serveru 27,07 sekund.

Tabulka 17: Celkový počet snímků

[počet]	kdrive	x2go	xrdp	tigervnc
Aritmetický průměr	1623,46	1624,78	1625,08	1623,11
Maximum	1625	1629	1628	1626
Minimum	1618	1619	1621	1616
Variační šíře	7	10	7	10

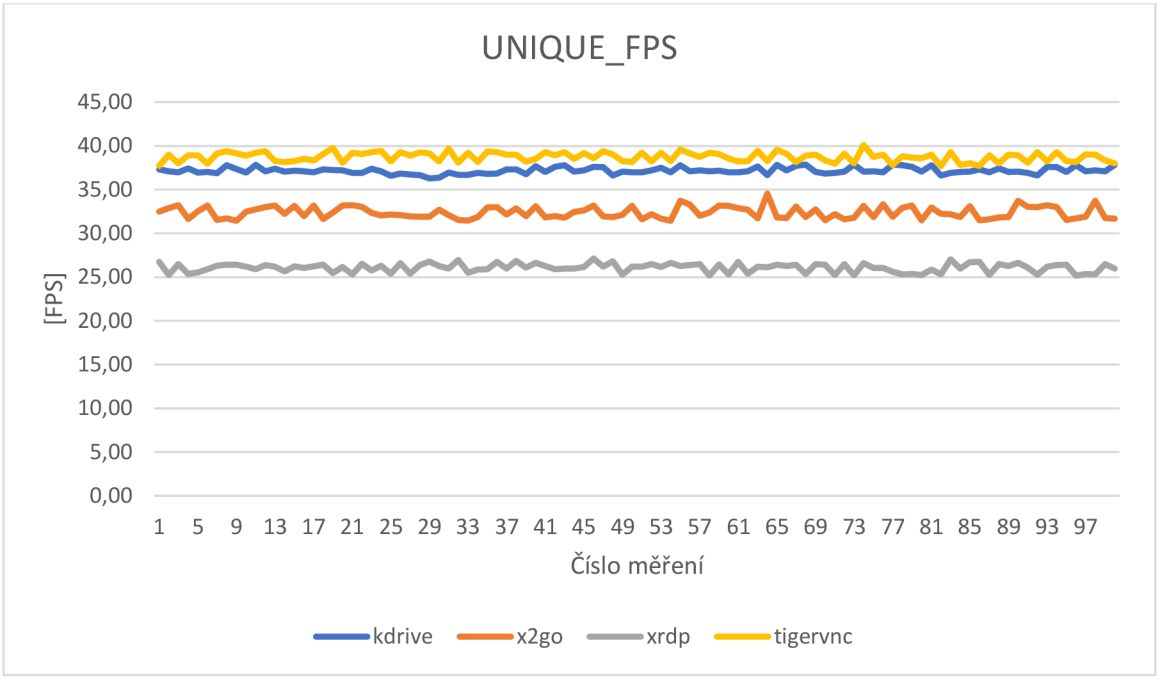


Obrázek 58: Celkový počet snímků [10]

Počet unikátních snímků byl po odstranění duplicitních snímků vypočten pro tigervnc – 1048,03, pro kdrive – 1005,43, pro x2go – 877,12 a pro xrdp – 706,31. Když tuto veličinu vydělíme časem běhu jednoho měření, získáme tím hodnoty pro veličinu počet unikátních snímků za sekundu – ta u tigervnc dosáhla v průměru 38,74 FPS a pro kdrive 37,16 FPS. U x2go to bylo o něco méně, a to 32,39 FPS a nejnižší hodnota vyšla pro server xrdp 26,08 FPS. Lidské oko je schopné vnímat obraz jako plynulý při minimální hodnotě 24 FPS, která byla dosažena při přenosu obrazu u všech čtyřech typů serveru.

Tabulka 18: Průměrný počet unikátních snímku za sekundu

[FPS]	kdrive	x2go	xrdp	tigervnc
Aritmetický průměr	37,16	32,39	26,08	38,74
Maximum	37,86	34,54	27,13	40,09
Minimum	36,28	31,44	25,17	37,68
Variační šíře	1,58	3,1	1,96	2,41



Obrázek 59: Průměrný počet unikátních snímků za sekundu [10]

6 Závěr

Hlavním cílem této diplomové práce bylo porovnání open source technologií pro vzdálenou plochu. Teoretická část popisovala technologie, které byly při samotném porovnávání využity. Na začátku praktické části byly sestrojeny skripty v jazyce bash, pomocí kterých byla simulována uživatelská aktivita. Nejprve pomocí skriptu pro kancelářskou práci a posléze i pro náročnější práci, ve kterém bylo zapojeno spouštění videa. V obou případech byla sledována činnost serveru a zatížení síťového připojení. Poté došlo k záznamu obrazu a sledování přenesených snímků za sekundu.

V případě monitorování zátěže serveru byl nejvíce zatěžován server tigervnc, a to jak během kancelářské práce, tak i během práce náročnějšího typu. Nejvyšší hodnoty byly naměřeny u všech tří sledovaných parametrů (CPU, load i paměť RAM). Nejnižších hodnot naopak dosahoval server x2go. Pokud disponujeme výkonnějším serverem, připadá v úvahu využití technologie tigervnc, v opačném případě je lepší sáhnout po softwaru, který bude klást menší nároky na výkon, tj. x2go, a to především v situacích, kdy se na daný server připojuje větší počet uživatelů.

Co se týče síťového přenosu, nejvyšší hodnoty byly vykazovány u všech veličin (RX, TX i počet paketů) na serveru tigervnc, zatímco ty nejnižší se naměřily u serveru x2go. U tigervnc došlo u náročnější práce k naměření průměrné rychlosti přenosu dat ze serveru na klienta 2,71 MB za sekundu, což odpovídá rychlosti stahování 21,68 Mbit/s. U rychlosti přenosu dat po síti by bylo dobré brát v potaz rychlost internetového připojení a počet uživatelů, kteří budou k serveru současně připojeni. Například při počtu 20 uživatelů současně využívajících tuto vzdálenou plochu by docházelo k přenosu rychlostí 433,6 Mbit/s, což by u nižší poskytované rychlosti internetu od providera nebylo možné reálně provozovat. U x2go u tohoto typu uživatelské aktivity dosáhla průměrná hodnota 0,5 MB/s, což jsou 4 Mbit/s – to při 20 uživatelích odpovídá 60 Mbit/s. Při nižších rychlostech internetového připojení nebo při nižším počtu uživatelů by tedy bylo vhodné sáhnout po technologii x2go, zatímco v síti s rychlejším připojením je možné provozovat tigervnc.

U zkoumání záznamu obrazu dosáhl nejlepších výsledků opět tigervnc s necelými 39 unikátními FPS, zatímco nejnižší hodnota 26 unikátních FPS patří serveru xrdp.

Z naměřených a následně zpracovaných dat lze usuzovat, že tigervnc se hodí spíše do prostředí s výkonnějším hardwarem a kvalitnějším internetovým připojením. Na druhou stranu je schopen za to nabídnout plynulejší překreslování obrazu než ostatní technologie. Nejnižší nároky na server i síť ze zkoumaných technologií naopak klade x2go, a se svými 32 unikátními FPS stále najde reálné využití.

7 Seznam použitých zdrojů

[1] The Complete History of Linux: Everything You Need to Know. [online]. [cit. 2023-04-01]. Dostupné z: https://history-computer.com/the-complete-history-of-linux-everything-you-need-to-know

[2] Nejoblíbenější distribuce Linuxu: TOP 10 nejpoužívanějších systémů. [online]. [cit. 2023-04-01]. Dostupné z: https://www.chip.cz/nejoblibenejsi-distribuce-linuxu-top-10-nejpouzivanejsich-systemu

[3] Co je to desktopové prostředí? [online]. [cit. 2023-04-01]. Dostupné z: https://www.root.cz/clanky/co-je-to-desktopove-prostredi

[4] Historie operačního systému GNU/Linux. [online]. [cit. 2023-04-01]. Dostupné z: https://www.root.cz/texty/historie-operacniho-systemu-gnulinux

[5] Unix. [online]. [cit. 2023-04-01]. Dostupné z: https://wiki.knihovna.cz/index.php/UNIX

[6] Obrázek grafického prostředí Debian GNU/Hurd. [online]. [cit. 2023-04-01]. Dostupné z: https://upload.wikimedia.org/wikipedia/commons/3/30/Debian-gnu-hurd-running-emacs.png

[7] Gnu Hurd. [online]. [cit. 2023-04-01]. Dostupné z: https://academic-accelerator.com/encyclopedia/gnu-hurd

[8] What is bash (Bourne Again Shell)? [online]. [cit. 2023-04-07]. Dostupné z: https://www.techtarget.com/searchdatacenter/definition/bash-Bourne-Again-Shell

[9] BLUM, Richard; BRESNAHAN, Christine. Linux command line and shell scripting bible. Indianapolis: John Wiley & Sons, 2015. ISBN 9781119209409.

[10] Vlastní zdroj

[11] Ubuntu Manpage: xdotool - command-line X11 automation tool. [online]. [cit. 2023-04-07]. Dostupné z: https://manpages.ubuntu.com/manpages/jammy/man1/xdotool.1.html

[12] Ubuntu Manpage: glxgears - ``gears" demo for GLX. [online]. [cit. 2023-04-07]. Dostupné z: https://manpages.ubuntu.com/manpages/jammy/man1/glxgears.1.html

[13] FFmpeg. [online]. [cit. 2023-04-07]. Dostupné z: https://github.com/FFmpeg/FFmpeg

[14] An introduction to using tcpdump at the Linux command line. [online]. [cit. 2023-04-07]. Dostupné z: https://opensource.com/article/18/10/introduction-tcpdump

[15] Understanding PSH ACK TCP Flags. [online]. [cit. 2023-04-07]. Dostupné z: https://www.howtouselinux.com/post/psh-ack-tcp-flags

[16] tshark(1). [online]. [cit. 2023-04-07]. Dostupné z: https://www.wireshark.org/docs/man-pages/tshark.html
[17] capinfos(1). [online]. [cit. 2023-04-07]. Dostupné z: https://www.wireshark.org/docs/man-pages/capinfos.html
[18] Client-Server Model. [online]. [cit. 2023-04-10]. Dostupné z: https://www.geeksforgeeks.org/client-server-model
[19] Obrázek modelu klient-server. [cit. 2023-04-10]. Dostupné z: https://www.tutorialspoint.com/data_communication_computer_network/images/client_server.jpg
[20] Difference between thin client and thick client. [online]. [cit. 2023-04-10]. Dostupné z: https://www.javatpoint.com/thin-client-vs-thick-client
[21] Klient/server na různé způsoby. [online]. [cit. 2023-04-10]. Dostupné z: https://www.earchiv.cz/a96/a611k150.php3
[22] Obrázek pěti variant modelu klient-server. [online]. [cit. 2023-04-10]. Dostupné z: https://www.earchiv.cz/a96/gifs/p611k151.gif
[23] Raspberry Pi. [online]. [cit. 2023-05-01]. Dostupné z: https://www.pcmag.com/encyclopedia/term/raspberry-pi
[24] Lekce 7 - Velká rodina Raspberry Pi - Přehled modelů a jejich funkcí. [online]. [cit. 2023-05-01]. Dostupné z: https://www.itnetwork.cz/hardware-pc/raspberry-pi/velka-rodina-raspberry-pi-prehled-modelu-a-jejich-funkci
[25] HALFACREE, Gareth. The Official Raspberry Pi Beginner's Guide: How to use your new computer (4th Edition). Cambridge: Raspberry Pi Press, 2020. ISBN 978-1-912047-73-4.
[26] Introduction to NX technology. [online]. [cit. 2023-05-01]. Dostupné z: https://web.archive.org/web/20121225051702/http://www.nomachine.com:80/documents/intr-technology.php
[27] How SSH tunneling works in NX 3.5.0. [online]. [cit. 2023-05-01]. Dostupné z: https://kb.nomachine.com/AR01C00137
[28] Linux in Government: Major Breakthrough in Linux Technology. [online]. [cit. 2023-05-01]. Dostupné z: https://www.linuxjournal.com/article/8342
[29] RDP a vzdálený přístup k počítačům. [online]. [cit. 2023-05-01]. Dostupné z: https://blog.jirivanek.eu/cs/2023/06/03/rdp-a-vzdaleny-pristup-k-pocitacum

[30] What Is RDP & How Do You Secure (or Replace) It? [online]. [cit. 2023-05-10]. Dostupné z: https://www.beyondtrust.com/blog/entry/what-is-rdp-how-do-you-secure-or-replace-it
[31] The Remote Framebuffer Protocol. [online]. [cit. 2023-05-11]. Dostupné z: https://www.rfc-editor.org/rfc/rfc6143.html
[32] Obrázek protokolu RFB. [online]. [cit. 2023-05-11]. Dostupné z: https://cam-orl.co.uk/images/pictures/rfb.gif
[33] Implementation and architecture. [online]. [cit. 2023-05-15]. Dostupné z: https://guacamole.apache.org/doc/gug/guacamole-architecture.html
[34] Obrázek architektury Apache Guacamole. [online]. [cit. 2023-05-15]. Dostupné z: https://guacamole.apache.org/doc/gug/_images/guac-arch.png
[35] Apache Guacamole: gateway pro vzdálený desktop (RDP a VNC). [online]. [cit. 2023-05-15]. Dostupné z: https://www.root.cz/clanky/apache-guacamole-gateway-pro-vzdaleny-desktop-rdp-a-vnc
[36] Obrázek nastavení parametrů připojení v Apache Guacamole. [online]. [cit. 2023-05-15]. Dostupné z: https://www.linux-magazine.com/var/linux_magazin/storage/images/issues/2020/237/guacamole/figure-3/770943-1-eng-US/Figure-3_reference.png
[37] Apple Remote Desktop User Guide. [online]. [cit. 2023-05-15]. Dostupné z: https://support.apple.com/cs-cz/guide/remote-desktop/welcome/mac
[38] KEDER, Daniel. Popis protokolu ARD. Brno, 2007. Bakalářská práce. MUNI.
[39] Obrázek Apple Remote Desktop. [online]. [cit. 2023-05-17]. Dostupné z: https://help.apple.com/assets/6244FACB41772A46DC570BA5/6244FACC41772A46DC570BAC/en_US/72dc8ce30f57ca467289d1bd66989b1a.png
[40] What is Citrix XenDesktop and Why Use It? [online]. [cit. 2023-05-17]. Dostupné z: https://www.parallels.com/blogs/ras/citrix-xendesktop
[41] Citrix Pricing for Virtual Desktops and Alternatives. [online]. [cit. 2023-05-17]. Dostupné z: https://www.parallels.com/blogs/ras/citrix-pricing
[42] PĚNKA, Michal. Terminálová učebna s využitím LTSP. Praha, 2009. Bakalářská práce. VŠE.
[43] NoMachine Enterprise Store. [online]. [cit. 2023-05-25]. Dostupné z: https://store.nomachine.com

[44] RealVNC® Connect. [online]. [cit. 2023-05-25]. Dostupné z: https://www.realvnc.com/en/connect
[45] What Is a Secured-Core PC for Windows 11? [online]. [cit. 2023-05-25]. Dostupné z: https://www.howtogeek.com/812013/what-is-a-secured-core-pc
[46] Understanding Windows Server 2022 Licensing. [online]. [cit. 2023-05-25]. Dostupné z: https://petri.com/understanding-windows-server-2022-licensing
[47] Přehled cen a licencování pro Windows Server 2022. [online]. [cit. 2023-05-25]. Dostupné z: https://www.microsoft.com/cs-cz/windows-server/pricing
[48] Remote Desktop Services. [online]. [cit. 2023-05-25]. Dostupné z: https://learn.microsoft.com/cs-cz/windows/win32/termserv/terminal-services-portal
[49] Remote Desktop Services roles. [online]. [cit. 2023-05-25]. Dostupné z: https://learn.microsoft.com/en-us/windows-server/remote/remote-desktop-services/rds-roles
[50] What is ThinLinc. [online]. [cit. 2023-05-25]. Dostupné z: https://www.cendio.com/thinlinc/what-is-thinlinc
[51] X2Go, Remmina and X2GoKdrive. [online]. [cit. 2023-05-25]. Dostupné z: https://sunweavers.net/blog/node/136
[52] TigerVNC/tigervnc. [online]. [cit. 2023-05-25]. Dostupné z: https://github.com/TigerVNC/tigervnc
[53] xrdp. [online]. [cit. 2023-05-25]. Dostupné z: https://www.xrdp.org
[54] sesman.ini(5) - Linux man page. [online]. [cit. 2023-05-25]. Dostupné z: https://linux.die.net/man/5/sesman.ini
[55] xrdp.ini(5) - Linux man page. [online]. [cit. 2023-05-25]. Dostupné z: https://linux.die.net/man/5/xrdp.ini
[56] X2Go Server. [online]. [cit. 2023-07-19]. Dostupné z: https://wiki.x2go.org/doku.php/wiki:advanced:x2goserver-in-detail
[57] X2Go KDrive. [online]. [cit. 2023-07-19]. Dostupné z: https://wiki.x2go.org/doku.php/wiki:advanced:x2gokdrive:start
[58] X2Go – Ein performanter Remote-Desktop, nun auch per Browser. [online]. [cit. 2023-07-19]. Dostupné z: https://kielux.de/content/downloads/vortrag/KOLT2022/x2go-kielux-2022-kdrive-html5-foliendownload.pdf
[59] Desktop Video Manual. [online]. [cit. 2023-10-01]. Dostupné z: https://documents.blackmagicdesign.com/UserManuals/DesktopVideoManual.pdf

[60] Obrázek UltraStudio Recorderu. [online]. [cit. 2023-10-01]. Dostupné z: https://images.blackmagicdesign.com/images/products/ultrastudio/techspecs/connections/ultrastudio-recorder-3g.png

8 Seznam obrázků, tabulek, grafů a zkratek

8.1 Seznam obrázků

8.2 Seznam tabulek

Přílohy

Přílohy.zip