KAMEROVÝ SYSTÉM S VLASTNÍ DIAGNOSTIKOU

Petr Písařík [1]***, Matouš Cejnek*** [1]

[1] ***Ústav přístrojové a řídící techniky, Fakulta Strojní, ČVUT v Praze***

Abstrakt: Tato práce popisuje tvorbu kamerového systému, který bude možné použít k monitorování objektů či procesů. Výběr vhodného hardware splňující požadavky. Schéma zapojení kamerového systému. Software kamerového systému je využívá pythonu3 a nástrojů linuxového prosředí Ubuntu Mate 18. Celkový postup implementace a testování.
Klíčová slova: IP-Cam, Python-3, Kamerový systém, Raspberry B3+, Ubuntu, Linux, ffmpeg, cron

Abstract: This work describes the creation of a camera system that can be used to monitor objects or processes. Selection of suitable hardware that meets the requirements. CCTV wiring diagram. The camera system is using Python3 and Linux environment Ubuntu Mate 18. Overall implementation and testing process.
Keywords: IP-Cam, Python-3, Cam system, Raspberry B3+, Ubuntu, Linux, ffmpeg, cron

1 Úvod

Kamerové systémy jsou nedílnou součástí života dnešních lidí. Na trhu je nepřeberná nabídka komerečních řešení již hotových kamerových systémů. V tomto článku jsou popsány některé aspekty návrhu low-cost domácího kamerového systému s důrazem na vlastní monitoring zdraví kamerového systému. V následujících sekcích je popsáno jak a proč byl zvolen použitý hardware a software. Jak byla navržena architektura samotného software řešení. Dále zde jsou také popsány různé klíčové problémy, na které je možné při tvorbě systému narazit, a jak je řešit.

2 Hardware

Použité zařízení, musí splňovat zadané požadavky, tedy být odolné proti venkovnímu prostředí. Umožňovat noční režim. Napájení pomocí PoE (Power over Ethernet) [1]. Poskytovat živý obraz pomocí RTSP (real time stream protokol) [2] a být dosažitelné z internetu. Spolehlivě pracovat i s delšími časovými úseky. Hardware kamerového systém je logicky rozdělen do tří částí : senzorické, síťové a výpočetní.

2.1 Senzorika

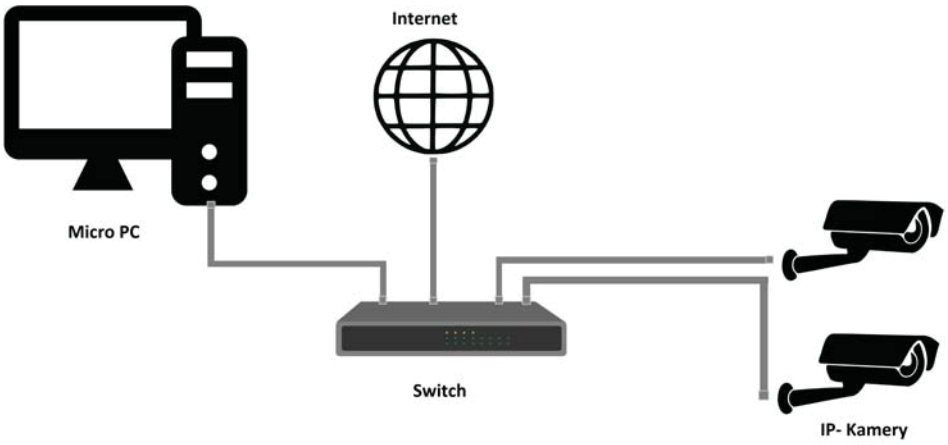
Jako kamera splňující požadavky se ukázala IP Kamera IPSU20HHF200 od firmy Cantonk.

2.2 Síťové připojení

Síťovou část propojující kamery s mikro počítačem a mikro počítač s internetem je realizována pomocí switche: TL-SF1008P [3]. Poskytující 4 POE porty, kterými je zajištěno napájení i datový přenos kamer.

2.3 Výpočetní zařízení

Jako výpočetní člen je použi mikro počítač Raspberry pi B3+. K mikro-PC je připojený externí HDD disk, jako prostor pro úložiště záznamů.



Obr. 1: Schéma zapojení jednotlivých komponent systému.

3 Software

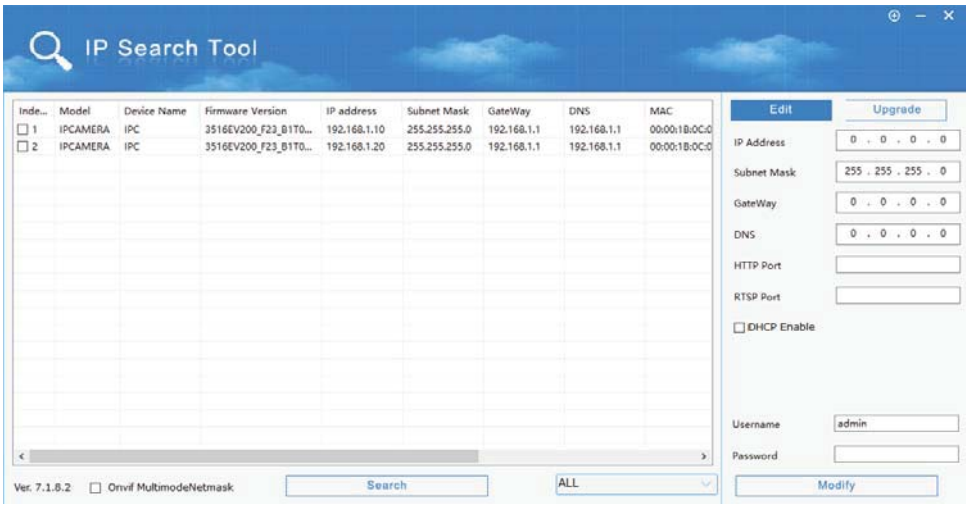
Systém je naprogramovaný v Python 3 [4] a ultilitách linuxu. Hlavním cílem je, aby systém umožňoval real-time náhled, ukládal záznam kamer ve smyčce a nejstarší část záznamu automaticky odstraňoval, aby se průběžně uvolňovalo místo na disku. Vše běží v linuxovém prostředí pro Raspberry Pi B3+ Ubuntu Mate verzi 18.04.

4 Implementace

Prvním krokem k zprovoznění systému je instalace operačního systému Raspberry Pi. Na oficiálních stránkách [5] jsem vybral nejnovější verzi operačního systému Ubuntu Mate. Ubumtu Mate jsem zvolil kvůli pohodlnému desktopovému přístupu. Dle instrukcí jsem vytvořil boot-SD card. Po ověření správného nainstalování systému následovala instalace Pythonu 3.8 společně s IDLE ze stránek [6].

4.1 Připojením kamer

Kamerový systém je zapojen podle schématu Obr.1. Raspbery je připojeno pomocí LAN k switch. Kamerové POE LAN zajišťuje zároveň napájení kamer a datový tok od kamer. V takto zapojeném systému jsem vyhledal připojená zařízení pomocí IP Search [7]. Mezi nalezanými zařízeními byly i IP kamery v tovární konfiguraci. Pro budoucí použití bylo nutné kamery vhodně překonfigurovat.



Obr. 2: Nalezení IP kamer dostupných v místní síti a jejich konfigurace

Ke konfiguraci jsem využil výše zmíněný program IP Serch Tool. U kamer bylo nutné nastavit IP adresu tak, aby odpovídala místní síti. Kromě IP adresy jsem nastavil i odpovídající Mask a Gate Way. Zapsal jsem si

označení RTSP portů, na kterých kamera vysílá stream. Dalším krokem bylo připojit se ke kamerám a změnit tovární přístupové údaje. Hardware kamerového systému byl těmito modifikacemi připraven k provozu.

4.2 Přístup a záznam RTSP streamu

Ve VLC player [8] jsem nejdříve ověřil existenci kamerových streamů a jejich dostupnost. V záložce síťových proudů jsem vyplnil IP adresu, číslo portu RTST a přihlašovací údaje. Následovala implementace v Pythonu. Využitím knihovny openCV jsem navázal spojení s kamerami a v živém modu zobrazil oba RTSP streamy. Problém nastal až při pokusech o ukládání záznamu. V pythonu jsem si vytvořil skript, který založil video soubor s datem a časem vytvoření. Do tohoto souboru se ukládala přijimaná data z kamery. Skript byl doplněn i o automatické mazaní souborů starších než předepsaný čas. Zkušební provoz odhalil, že takto běžící skript, který zobrazuje a ukládá živý obraz z jedné kamery využívá výkon mikro PC zkruba na 90%. Záznam z obou kamer vedl k přehřívání zařízení a ukončení běžících procesů. Laděním skriptu, snižováním kvality (snížení rozlišení a FPS (počet snímků za sekundu)) přijímaného videa se mi podařilo snížit výpočetní náročnost na cca 80%.Problémem byla také časová nespojitost mezi jednotlivými nahranými segmenty. Segmenty byly odděleny časovými prostoji v řádech několika sekund (2-13 sec). Pro záznam obou kamer byl výkon mikro PC stále nedostatečný.

Ukládání streamu pomocí pythonu v požadované kvalitě se ukázalo jako neuskutečnitelné. Z těchto důvodů jsem se rozhodl použít alternativní způsob záznamu RTSP streamu. Pro nahrávání jsem se rozhodl otesovat linuxové knihovny a programy umožňující live konverzi RTSP streamu (openRTSP), ffmpeg [9] a vlc [8]. Z testování se nejlépe vyšlo ffmpeg. OpenRTSPse ukázalo příliš náchylné na stabilitu a při drobném kolísání signálu docházelo ke kolapsu nahrávání. Také bylo složité přesně nastavit parametry aby došlo ke správnému záznamu. Automatický záznam pomocí cvlc bylco se týče předepisování parametrů nejjednodušší. Stačilo nastavit pouze adresu streamu, rozdělení na časové segmenty a jména souborů. Kvalita záznamu byla v porovnání s openRTSP horší. Docházelo zde ke krátkým usekům mezi jednotlivými segmenty kdy se nenahrávalo (1-3 sec). Rozlišení samotného záznamu bylo stejné jako v streamu(1920x1080), avšak záznam byl sekaný. Zaznamenaný soubor běžel s nahodilými záseky, kdy se na videu na několik sekund zastaví obraz a pak běží dál. Optimálním nástrojem pro záznam se ukázal ffmpeg.

4.3 FFMPEG

Ffmpeg je linuxová ultilita pro příkazový řádek pro konverzi video formátů. Umožňující konverzi streamování a nahravání digitálního obrazu a zvuku v reálném čase. Neméně důležité je znát i prarametry streamu jako je codec, FPS, rozlišení atd. Mnou použitý příkaz pro záznam je následující:

```
>>>ffmpeg -rtsp_transport tcp -i rtsp://login:password@IP_address:RTSP_port -codec:v
    copy -codec:a copy -map 0 -f segment -segment_time 28800 -segment_format mkv
    -strftime 1 -reset_timestamps 1 /path/name_%Y-%m-%d_%H-%M-%S.mkv
```

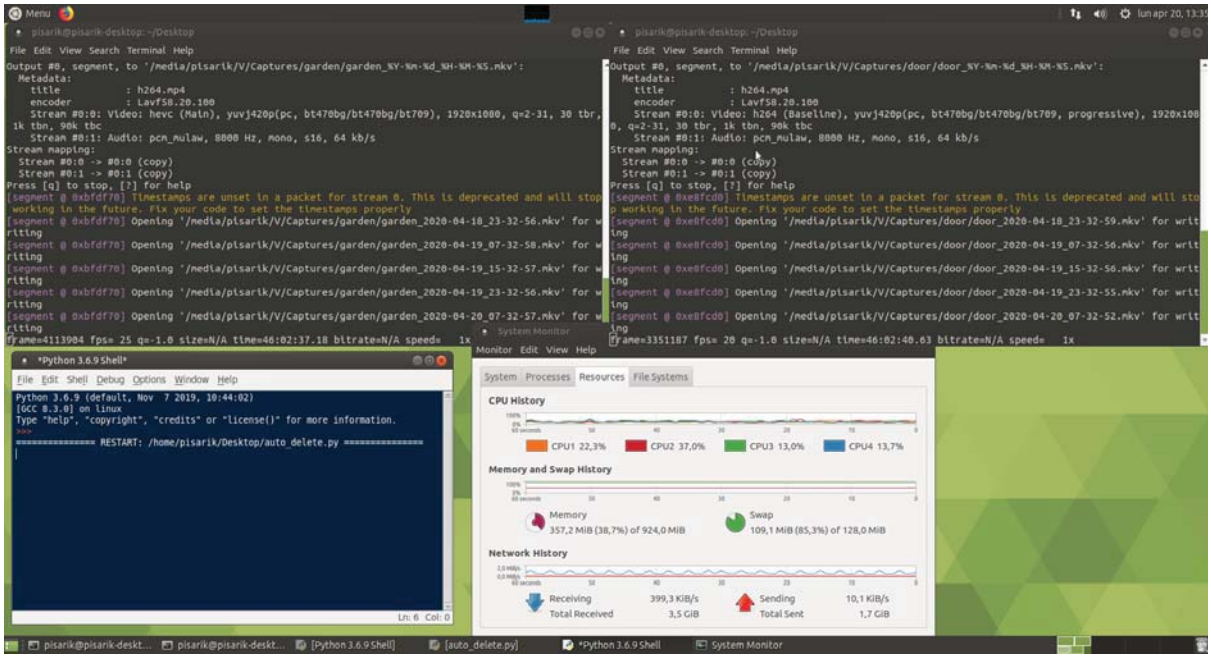
Kde *ffmpeg* je příkaz spouštějící ultilitu, *rtsp transport* spouští rtsp transportní protokoly, *tcp* volba TCP jako nižšího transportního protokolu,*i* aktivace nastavení vstupu (input), *rtsp://login:password@IP address:RTSP port* nastavení adresy streamu společně s přihlašovacími údaji a portem rtsp streamu, *-codec:v copy -codec:a copy* volba codecu automaticky podle zdroje, *map 0* mapování jednoho streamu,*f segment -segment time 28800* rozdělení záznamu na 8 hodinové úseky, *segment format mkv* volba výstupního formátu souboru. v našem případě je vhodné použít spíše formáty *mkv* a *mp4*. Použití formátu **avi** vedlo k potížím z důvodů nekonzistence přijímaného signálu. Formát *avi* nedokáže zpracovat dlouhodobě video ve kterém se občas ztratí data. Poslední část *-strftime 1 -reset timestamps 1 /path/name %Y %m %d %H-%M-%S.mkv* nastavuje cestu kam ukládat záznam a jeho jmého tvořené prefixem a aktuálním časem.

Stream ve formátu h264 je převáděn do mkv po 8 hodinových úsecích. Mkv formát umožňuje ztrátu učitého množství dat. Mezery mezi jednotlivými záznamy jsou zanedbatelné. Nahrávání je stabilní a může běžet v řádech dní.

Celkové výkonové požadavky zatěžují mikro PC 30% maximálního výkonnu, jak je vidět z (3). Takto zavedený systém tedy zaznamenává a automaticky odstraňuje staré soubory.

4.4 Automatické spuštění

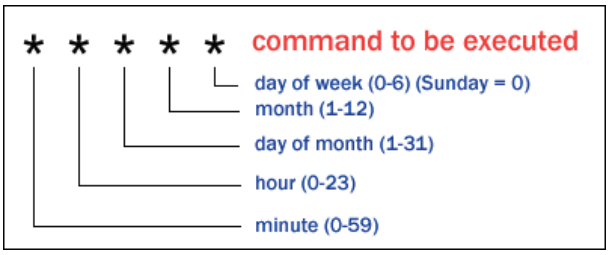
Pro spolehlivý chod kamerového systému je také nutné, aby se při výpadku automaticky spustil. Proto jsem nastavil automatické spuštění všech skriptů po rebootování systému. Do operačního systému jsem také zavedl pravidelné restartování mikro PC a kamer jedenkrát za týden. V definovaný čas dojde k uložení záznamu a vypnutí všech zařízení. Po spuštění systému se jako první spustí python skript automatického mazaní, který následují



Obr. 3: Nahrávání dvou kamer spolu s python skriptem, který automaticky odstraňuje soubory starší 5 dní.

skripty zaznamenávající kamery. Automatické spuštění je docíleno softwarovým démonem Cron [10], což je vlastně specializovaný systémový proces sloužící jako plánovač dějů. Pro automatické spuštění bylo nutné pomocí příkazů *sudo chmod -x /path/to/my/script.sh* nastavit,aby bylo skript možné spustit. Všechny takto spouštěné skripty bylo třeba doplnit o první řádek *#!/bin/sh*. Bez těchto úprav by nedošlo k jejich aktivaci.

```
@reboot         /path/to/my/script/auto_delete.sh
@reboot         /path/to/my/script/capture_cam_A.sh
@reboot         /path/to/my/script/capture_cam_B.sh
00 03 * * */3   /path/to/my/script/reboot.sh
```



Obr. 4: Syntax zápisu v Cron. *@reboot* znamená aktivaci vždy po zapnutí. Příkaz na poslední řádce znamená spuštění scriptu reboot každou středu ve tři hodiny v noci [11].

5 Vzdálený přístup

Pro kontrolu funkčnosti a přístupu k zánamům jsem na raspberry pi vytvořil FTP (File Transfer Protocol) servr. Kvůli většímu zabezpečení je vhodné použít zabezpečenější verzi SFTP (Secure File Transfer Protocol). Soubory přístupné pro klienty jsou pouze soubory umístěné na externím disku. Přihlásit se jde do tří různých uživatelských úrovní. Administrátorský přístup umožňuje správu a editaci souborů. Uživatelský přístup je pak pouze pro náhled bez možnosti editací souborů. Třetí úroveň povoluje spouštění skriptů a umožňuje nastavování systému.

6 Závěr

Vybral jsem hardware kamerového systému a instaloval jej na budovu. Pomocí raspberry pi jsem vytvořil systém jenž ukládá záznam dvou kamer do 8 hodinových segmentů. K mikro PC jsem připojil externí hard disk, sloužící jako prostor pro data. Jména video segmentů jsou tvořena označením kamery a časem kdy byla stopa založena. Automatické mazání skriptů je uskutečněno skriptem v pythonu, který odstaní soubory starší než předepsaný čas. Vše se spouští automaticky po startu systému Ubuntu mate.

Systém byl extenzivně testován 168 hodinama provozu při podmínkách, v kterých se bude běžně používat.

Poděkování

Hlavní autor článku by rád poděkoval společnosti Alphabet Inc. za provoz služby Google, bez které by tato práce nebyla v této míře uskutečnitelná.

Práce Matouše Cejnka byla podpořena z grantu *SGS18/177/OHK2/3T/12*.

Literatura

[1] *IEEE standard for Ethernet.* IEEE, New York, 2016. OCLC: 1017937152.

[2] Henning Schulzrinne, Anup Rao, and Robert Lanphier. Real time streaming protocol (rtsp). 1998.

[3] Download for TL-SF1008P. Library Catalog: www.tp-link.com.

[4] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009.

[5] Ubuntu MATE Team. Ubuntu MATE | For a retrospective future. Library Catalog: ubuntu-mate.org.

[6] IDLE — Python 3.8.2 documentation.

[7] Advanced IP Scanner - stáhněte si bezplatný nástroj na skenování sítě.

[8] VideoLan. VLC - Features - VideoLAN, 2020.

[9] Suramya Tomar. Converting video formats with ffmpeg. *Linux Journal*, 2006(146):10, 2006.

[10] Larry Reznick. Using cron and crontab. *Sys Admin*, 2(4):29–32, 1993.

[11] Suhesh KS. Crontab tips and tricks, June 2016. Library Catalog: www.eazylinux.com Section: Linux.

Selected article from
Tento dokument byl publikován ve sborníku

Nové metody a postupy v oblasti přístrojové techniky, automatického řízení a informatiky 2020

New Methods and Practices in the Instrumentation, Automatic Control and Informatics 2020

14. 9. – 16. 9. 2020, Zámek Lobeč

ISBN 978-80-01-06776-5

Web page of the original document:
http://iat.fs.cvut.cz/nmp/2020.pdf

Obsah čísla/individual articles:
http://iat.fs.cvut.cz/nmp/2020/

Ústav přístrojové a řídicí techniky, FS ČVUT v Praze, Technická 4, Praha 6