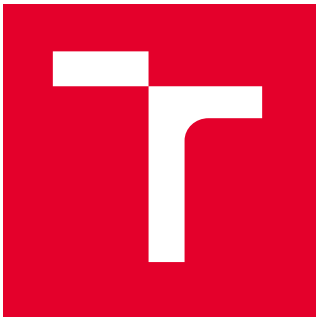


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ

DEPARTMENT OF TELECOMMUNICATIONS

AUTOMATIZOVANÝ DOHLEDOVÝ SYSTÉM PRO IT INFRASTRUKTURY

AUTOMATED MONITORING SYSTEM FOR IT INFRASTRUCTURES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Daniel Kunčický**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Marek Sikora**
SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Telekomunikační a informační systémy**

Ústav telekomunikací

Student: Daniel Kunčický ***ID:*** 230612

Ročník: 3 ***Akademický rok:*** 2022/23

NÁZEV TÉMATU:

Automatizovaný dohledový systém pro IT infrastruktury

POKYNY PRO VYPRACOVÁNÍ:

Obvyklé firemní či soukromé IT infrastruktury obsahují různé druhy prvků pro zajištění potřebných služeb. Může jít o hardwarové prvky jako záložní zdroje, NAS servery, Routery, Switche, nebo také o softwarové prvky jako např. zálohovací software, antivirové programy, poštovní servery, atd. Ve většině těchto zařízení lze definovat zasílání e-mailových zpráv, ať už jde o pravidelné reporty, nebo hlášení problémů.
Úkolem této bakalářské práce je vytvořit program pro automatizované vyhodnocování příchozích emailových notifikací. Program se bude schopen přihlásit do e-mailové schránky, vyhrazené pro tento účel,
a vykonávat třídění či značkování notifikací dle předem stanovené logiky. Dalším úkolem práce je obohatit program o funkce zasílání notifikací o průběhu, analýzy dochvilnosti, plánování automatického spuštění a export dat o dochvilnosti notifikací. Celé řešení bude navrženo a optimalizováno pro co nejjednodušší obsluhu administrátorem.

DOPORUČENÁ LITERATURA:

Podle pokynů vedoucího práce.

Termín zadání: 6.2.2023 ***Termín odevzdání:*** 26.5.2023

Vedoucí práce: Ing. Marek Sikora
Konzultant: Ing. Jan Sikora (APROSYS ICT s.r.o.)

prof. Ing. Jiří Mišurec, CSc.
předseda rady studijního programu

UPOZORNĚNÍ:

Autor bakalářské práce nesmí při vytváření bakalářské práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č.40/2009 Sb.

ABSTRAKT

Bakalářská práce se zabývá vývojem aplikace, která bude automatizovaně vyhodnocovat přijaté zprávy. V případě výskytu incidentu informuje administrátora o aktuálním stavu. Při svém periodickém spouštění bude zasílat informace o průběhu vyhodnocování zpráv. V rámci vyhodnocování zpráv také zkontroluje dochvilnost zpráv a příjem všech očekávaných zpráv. Pro správu programu byla vyvinuta grafická nadstavba. Aplikace byla vyvinuta v jazyce Python s možností převodu na exe soubor. Aplikace usnadňuje rutinní sledování stavu IT infrastruktury.

KLÍČOVÁ SLOVA

Dohledový systém, automatizované vyhodnocování, notifikace, analýza dochvilnosti, automatické spouštění, přístup do e-mailové schránky, autentizace.

ABSTRACT

The bachelor's thesis describes the development of an application that will automatically evaluate received messages. When an incident occurs, it informs the administrator about the current status. Application will send information about the progress of message evaluation when it runs periodically. As part of the message evaluation, it also checks the timeliness of the messages and the reception of all expected messages. A graphical dashboard was developed to manage the application. The application was developed in Python. The application facilitates the routine monitoring of the state of the IT infrastructure.

KEYWORDS

Monitoring system, automated evaluation, notifications, analysis of punctuality, automatic start, access to the e-mail box, authentication.

Vysázeno pomocí balíčku `thesis` verze 4.07; `http://latex.feec.vutbr.cz`

KUNČICKÝ, Daniel. *Automatizovaný dohledový systém pro IT infrastruktury*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2023, 60 s. Bakalářská práce. Vedoucí práce: Ing. Marek Sikora

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	Daniel Kunčický
VUT ID autora:	230612
Typ práce:	Bakalářská práce
Akademický rok:	2022/23
Téma závěrečné práce:	Automatizovaný dohledový systém pro IT infrastruktury

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářské práce panu Ing. Marku Sikorovi za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Obsah

Seznam obrázků

Úvod

Bakalářská práce se zabývá implementací monitorovacího systému, který bude periodicky analyzovat přijaté zprávy. Podle předem nastavených parametrů provede vyhodnocení všech zpráv od monitorovaných zařízení. Dále dohlíží na dochvilnost zpráv a hlídá příjem očekávaných zpráv (zda nějaká nechybí). V případě vzniku incidentu informuje formou e-mailové notifikace administrátora.

První kapitola vysvětluje pojem automatizované dohledové systémy, jejich složitost, možnosti a způsoby řešení. Dále pojednává o existujících, již hotových systémech na trhu, jak open-source/freeware, tak komerčních řešení. Zaměřuje se také na možnost úpravy těchto systému a implementaci vlastních externích skriptů. Dotýká se okrajově nadstaveb dohledových systémů, jak agregačních, tak vizualizačních. Využitelné jsou v různých dohledových centrech.

Konkrétní zadání směřuje k aplikaci, která má analyzovat zprávy na serveru Microsoft 365, proto následující kapitoly pojednávají především o možnostech řešení specifických subproblémů pro toto konkrétní prostředí.

Druhá kapitola popisuje dostupné autentizační / autorizační metody a protokoly, opět s důrazem na konkrétní prostředí. Jsou rozebrány jak historické a zastaralé, tak současné doporučované způsoby.

Třetí kapitola pojednává o způsobech přístupu do e-mailových schránek, opět s analýzou již zastarávajících způsobů a moderních, poskytovatelem služeb doporučených. Dále pojednává o podporovaných vlastnostech protokolů a jejich míru vhodnosti využití v rámci řešení zadání.

Čtvrtá kapitola se věnuje výběru nástrojů a prostředí, ve kterých byla aplikace vyvíjena. Zdůvodňuje zvolené technologie s ohledem na zabezpečení a omezení potřebným oprávnění aplikace na minimum.

Konkrétní řešení jádra aplikace pak popisuje kapitola pátá. Její členění odpovídá skladbě modulů aplikace.

Šestá kapitola řeší vývoj grafické nadstavby aplikace, která má za cíl uživateli zjednodušit správu a konfiguraci. Poskytuje také nástroje určené pro zpětné dohledání historie přijatých zpráv a export zpráv. Účelem je analýza anomálií přijatých zpráv mimo aplikaci.

1 Automatizované dohledové systémy

Důležitost automatizovaných dohledových systému ve firmách se neustále a průběžně zvyšuje. S růstem každé firmy a postupem doby přichází vyšší nároky na IT infrastrukturu nejen z důvodu nárůstu počtu uživatelů, ale i růstem komplexity systémů a procesů podporovaných či přímo řízených IT systémy. S tím, jak jsou firemní procesy více a více závislé na IT, se také zvyšují nároky na bezpečnost, důvěryhodnost, integritu, dostupnost a obnovitelnost IT prostředí a zdrojů. Od určitého počtu spravovaných zařízení a služeb už není pro administrátory možné periodicky ručně procházet a kontrolovat funkčnost každého zařízení/služby. Důležitá je také rychlost reakce na porušení integrity, a to jak z důvodu interního selhání, tak z důvodu případného útoku. Na řadu proto přicházejí dohledové systémy, které téměř nepřetržitě či v delších periodách vyhodnocují stav a funkčnost zařízení a služeb.

Na základě definovaných scénářů jsou tyto systémy schopny sbírat, koncentrovat a zasílat informace nebo výstrahy nadřízeným systémům. Případně jsou schopny samostatné předem definované nouzové reakce (typicky odpojení od sítě, zastavení služeb, restart). Velkou přidanou hodnotou některých systémů je agregace a vizualizace stavu zařízení, které administrátorovi umožňují rychlý a také strukturovaný přehled o stavu všech zařízení. Přechod k problému a jeho řešení může mít podobu "doklikání se stromem" a provedení předem definovaných akcí po stisknutí tlačítka. Systémy také umožňují zasílání výstrah na více míst, která mají problém řešit. Dohledový systém může být pasivní – přijímá a vyhodnocuje zprávy od zařízení (typicky email, SNMP), případně zareaguje na skutečnost, že zprávy nepřicházejí. Nebo aktivní, kdy systém posílá ověřovací dotazy a očekává relevantní odpověď, případně posílá aktivně dotazy simulující běžný provoz (heartbeat testování).

Dohledové systémy by svou komplexitou měly odrážet složitost, rozsah a kritičnost dohlíženého systému. Zejména typ a rozsah případné škody způsobené nefunkčností, což je také otázka ekonomická.

1.1 Existující řešení

Na trhu je v současnosti mnoho produktů, které nabízí i komplexní možnosti automatického monitoringu IT infrastruktury. S tím souvisí i velká počáteční náročnost na implementaci takového systému, případně vysoká cena. Systémy je nutné pro každou firmu individuálně upravit. To platí i pro různé open-source dohledové systémy. V případě, že firma nemá schopné IT oddělení, může se implementace externí dodavatelskou společností prodražit.

1.1.1 Komplexní systémy

Jedním z typických komplexních open-source systémů je Zabbix [1]. Podporuje monitorování sítě, zařízení i aplikací. Je schopen sledovat stavy síťových služeb, serverů a dalšího hardware, který umí komunikovat po síti. V některých případech je nutné dokoupit speciální hardware kartu, která zařízení do sítě připojí. Asi nejznámějším případem jsou záložní zdroje UPS, které v základní konfiguraci obvykle mají pouze USB nebo sériové rozhraní RS-232. Pro ukládání dat je použita relační databáze. Podporovány jsou například databáze MySQL nebo PostgreSQL [2].

Zabbix podporuje téměř jakékoliv monitorování. Pokud není k dispozici přímo vytvořený modul pro vlastní požadavky, je možné dodat vlastní. Jednou z možností je vytvoření externího Python skriptu, kde se provedou požadované úkony (třeba vyhodnocení e-mailů). Kontroly mohou ověřovat dostupnost služeb například pomocí pingu (očekává odpověď). Zabbix nabízí takzvané agenty, kteří se instalují do operačního systému sledovaného zařízení. Automaticky zařízení monitorují a odesílají informace na server. Pokud nelze na sledovaná koncová zařízení (server, UPS) instalovat další software, lze využít integrované protokoly SMTP nebo HTTP (pokud jsou implementovány). Umí se k zařízení připojit i přes protokoly pro vzdálenou správu. Jde o SSH (Secure Shell) či Telnet [2].

Reakce na vzniklé události závisí na konfiguraci. Může se pouze odeslat e-mail administrátorovi, případně se při vzniku nějaké události spustí definovaný skript. Pro větší firmy je zde implementována možnost eskalace. Pokud nastane událost na sledovaném zařízení a první kontaktní osoba v určitém časovém úseku nezareaguje, kontaktuje se další kontaktní osoba v pořadí [3].

Z komerčních balíčků je možné dále zmínit ManageEngine [4], SysAid [5] nebo Atera [6]. Model fungování a schopnosti jsou obdobné. Liší se v detailech, konfigurovatelnosti a samozřejmě cenou. Bohužel produkty tohoto typu přecházejí (nebo již přešly) na model předplatného. Pokud firma přestane platit, ztratí přístup k funkcionalitám.

1.1.2 E-mail parsery

Většina podnikových zařízení (a některé software) je schopna podávat zprávy o stavu formou e-mailu. Stačí mít zřízen u poskytovatele e-mailový účet, na který se zařízení bude přihlašovat. V cílové e-mailové schránce se tedy soustřeďují tyto zprávy o stavu. K základnímu rozřazení zpráv se dají využít pravidla přímo na serveru, která e-maily přesunou do složek (například podle odesílatele). Na pokročilou analýzu se používají specializované software "e-mail parsery".

Jsou to v podstatě jednodušší periodicky spouštěné programy, které procházejí schránku s přijatými maily. Na základě obsahu mailu, jeho příznaků nebo shody

zprávy s definovanou šablonou analyzovaný e-mail zatřídí nebo spustí definovanou akci. Typicky zašlou nějakých způsobem výstrahu dál, především administrátorovi.

1.1.3 Agregátory dat a vizualizace

Agregátory podávají ucelený a strukturovaný obraz o rozsáhlém dozorovaném prostředí. Vizualizační nadstavby pak podporují rychlou orientaci a většinou umožňují rychlý přechod ve struktuře k bodu, který vykazuje problém.

V centrálních dozorováních pracovištích se mohou sbírat data ze stovek zařízení. Není v silách operátora dohlížet na každý sledovaný bod individuálně. Proto se zavádí vizualizační nástroje (například Grafana [7]), které ve formě jednoduchých grafů a ukazatelů informují o stavu věci. Pokud se například sleduje síťový provoz a na grafu se neočekávaně objeví nárůst provozu, obsluha může začít zjišťovat důvody této anomálie.

2 Autentizační metody

Autentizace je proces ověření identity subjektu, jestli je autorizován (oprávněn) přistupovat k žádaným systémům a datům. Může se jednat o fyzického uživatele, ale i o různá propojení mezi službami a jejich infrastrukturou. V případě neúspěšného ověření identity musí být přístup zamítnut. Pro lepší ochranu před útočníky mohou být implementovány různé ochranné mechanismy, které mají zvýšit úroveň zabezpečení. Jde například o časové omezení počtu pokusů o přihlášení z jedné IP adresy. Tato omezení nemusí být vždy účinná, pokud útočník rozloží útok v čase [8].

2.1 Klasická autentizace

Klasická autentizace neboli basic authentication je zastaralý způsob autentizace a postupně se od něj ustupuje. V prostředí Microsoft 365 byla během roku 2022 zahájena postupná cílená definitivní deaktivace této metody. Během roku 2023 už je účinek deaktivace celosvětový a klasická autentizace už v prostředí Microsoft 365 nebude podporována [9]. Princip je velmi jednoduchý. Využívá pouze uživatelské jméno a heslo, které se zakóduje do formátu base64 a vloží do HTTP (Hypertext Transfer Protocol) hlavičky požadavku, který se posílá na server. Každý takto poslaný požadavek musí být opatřen touto hlavičkou, aby mohlo dojít k ověření.

Mnohé starší aplikace s implementovanou klasickou autentizací, včetně e-mailových klientů a různých automatizovaných skriptů, mohou používat nešifrovaný protokol HTTP [10]. Heslo je pak přenášeno v nešifrované podobě. Potenciální útočník tak může pomocí běžných nástrojů (např. Wireshark [11]) získat přihlašovací údaje a zneužít je. Velkou nevýhodou je nemožnost implementovat více faktorovou autentizaci MFA. V případě odchycení hesla je nutné ho změnit.

Další slabinou pak je možnost útoku Password Spray Attack. Jde o variantu útoku hrubou silou, kdy se útočník snaží přihlásit k velkému počtu účtů najednou s využitím databáze běžně používaných hesel. Tento útok je velmi složité detekovat, protože útočník přihlašovací jména neustále mění. Velmi často využívá botnety a má tak teoreticky neomezený počet IP adres. Útok na jedno konkrétní uživatelské jméno se rozprostře v čase. Proto nezafungují běžné obranné prostředky, které po pěti neúspěšných pokusech o přihlášení zablokují IP adresu. Pokročilejší ochrany vkládají do procesu přihlašování překážku, kterou zvládne překonat pouze člověk [12].

2.2 Moderní autentizace

Moderní autentizace, neboli modern authentication, je bezpečnější náhrada za zastaralou klasickou autentizaci. Jde o společné označení pro kombinaci různých autori-

začních a autentizačních metod mezi serverem a klientem či klientskou aplikací [13]. Místo hesel, která by se musela posílat v každém požadavku na server, se používají speciální tokeny. Token je mnohem delší než heslo, které si nastaví běžný uživatel. Většinou mají omezenou platnost, takže potenciál zneužití je menší. Pokud je moderní autentizace používána v různých aplikacích, nebude v jejich konfiguračních souborech uloženo čitelné heslo. Nemělo by tak dojít k jeho úniku. Pro každou aplikaci, která vyžaduje přístup k datům, se generuje vlastní token. V případě potřeby odebrání přístupu tak není nutné měnit heslo. Stačí pouze zrušit platnost tokenu. Pro moderní autentizaci není nezbytně nutné používat přihlašovací heslo. Lze například použít i různé hardwarové klíče. Výhodou je možnost používání vícefaktorové autentizace. Pro přihlášení tak nestačí pouze samotné heslo. Jde o další způsob zvýšení zabezpečení.

2.2.1 Oauth2

Oauth2 (Open Authorization) je otevřený standard, který slouží k autorizaci například API, které následně může za uživatele vystupovat a přistupovat k datům jiných API. Ke specifikaci povolených oprávnění se používají rozsahy, takzvané scopes [14]. Každé oprávnění má svůj rozsah (například mail.send pro odesílání e-mailů).

Primárně se používají dva druhy tokenů. První je přístupový, který slouží k přístupu a autentizaci oproti serveru (tedy náhrada hesla u klasické autentizace). Má omezenou platnost v řádu hodin nebo minut. V případě odchycení přístupového tokenu bude mít útočník pouze časově omezený přístup. Dále se používá obnovovací token. Slouží k získávání přístupových tokenů. Jeho platnost je také z bezpečnostních důvodů omezena na řádově týdny. Je tedy nutné tokeny relativně často obnovovat. Pokud aplikace zažádá s obnovovacím tokenem o token přístupový, server může v odpovědi odeslat zpět i nový obnovovací token. Pokud je známá maximální platnost obnovovacího tokenu, lze zajistit automatické obnovování tokenu periodicky. Jestliže aplikace bude často spouštěna, nemělo by dojít k vypršení platnosti tokenů [14]. Problém může nastat u sporadicky využívaných aplikací či zařízení. Při delší odstávce nutně dojde k vypršení obnovovacího tokenu a bude vyžadován zásah uživatele pro obnovu přístupu.

2.2.2 Přihlašování bez hesla

Metoda přihlašování bez hesla pro přihlášení využívá alternativní způsoby ověření identity. Místo hesel se interně používají certifikáty nebo tokeny. Některými společnostmi je tato metoda považována jako nejbezpečnější (jednofaktorová) existující metoda pro přihlašování [15]. Uživatel místo komplexního hesla, které si musí pamatovat, použije jiný faktor autentizace. Jedním z bezpečnějších faktorů autentizace

je hardwarový klíč s podporou standardu FIDO2. Uživatel vyplní svoje uživatelské jméno a připojí hardwarový klíč k počítači. Následně musí s klíčem provést interakci. Dražší klíče mají zabudovanou čtečku otisku prstů, zatímco levnější klíče mají pouze jednoduché kapacitní tlačítko, kterého je potřeba se dotknout a poté zadat PIN kód [16]. Možností je ale více. Existují mobilní aplikace, které při přihlašování zobrazí uživateli výzvu k potvrzení, zda je to opravdu on. Nevýhoda těchto aplikací je možnost útoku zvaného "přehlcení uživatele". Útočník se pokouší neustále přihlašovat, přičemž uživateli vyskakují notifikace, čímž ho zahltí. Aby se jich uživatel zbavil, přístup útočníkovi podvědomě potvrdí [17]. Případně lze využít biometrické čtečky umístěné přímo na zařízení. Jejich bezpečnost je na srovnatelné úrovni jako u hardwarových klíčů. Výhodou naopak je eliminace méně sofistikovaných phishing útoků, kdy útočník napodobí vzhled přihlašovacího webu a uživatele uvede v omyl. Ten následně zadá svoje přihlašovací údaje, které si útočník uloží. Při využití této přihlašovací metody uživatel žádné heslo nemá, a tak ho nemůže nikam zadat. Problém nastane v případě ztráty těchto ověřovacích zařízení. V případě ztráty jediného ověřovacího zařízení může v nejhorším případě dojít až k trvalému odmítnutí přístupu ke službě. Uživatelům je proto doporučeno zaregistrovat si těchto zařízení více. Nejméně však alespoň dvě zařízení.

2.2.3 Vícefaktorová autentizace

Vícefaktorová autentizace (Multi-factor authentication), označována také zkratkou MFA, je další vrstva ochrany ověření subjektu. Pro přihlášení k chráněnému rozhraní nestačí pouze znalost hesla. Uživatel musí poskytnout další faktor autentizace. Faktory autentizace se dělí na tři základní skupiny. Znalostní - založená na jedinečné tajné informaci, například heslo. Vlastnická - založená na výhradním přístupu k nějakému zařízení, kupříkladu hardwarový token, nebo mobilní telefon. Biometrická - založená na analýze fyzických charakteristik uživatele, například čtečka otisku prstů, sken obličeje, sken sítnice. Běžně používané služby často vyžadují nastavení dvou faktorů autentizace. Jde o heslo a další faktor. Při přihlašování s aktivovaným vícefaktorovým ověřením tedy proces postupuje následovně: (i) uživatel je vyzván k zadání hesla, následně ho aplikace či web vyzve k provedení zvoleného způsobu MFA. Může se jednat o aplikaci nainstalovanou v chytrém mobilním telefonu nebo hardwarový klíč [18].

Při volbě dalšího autentizačního faktoru je nutné se zamyslet nad úrovní jejich bezpečnosti. Vyhodnotit rizika a důležitost chráněného systému. Vždy půjde o kompromis mezi bezpečností, cenou a pohodlím uživatele. Hardwarové tokeny jsou velmi bezpečné, nicméně pořizovací cena může být podle použitého typu poměrně vysoká a uživatel ho musí mít u sebe. Mobilní aplikace, které zobrazí interaktivní výzvu

k potvrzení přístupu jsou zdarma, nicméně útočník může provést útok zahlcení uživatele. Uživatel je zahlcen vysokým počtem potvrzovacích notifikací, dokud jednu nepotvrdí. Dále existuje ověření kódem z SMS zprávy. Existuje zde riziko odchycení této zprávy přímo v mobilním telefonu záškodnickou aplikací, případně lze provést útok "SIM swap attack". Útočník získá kopii SIM karty uživatele a ověřovací zprávy přijdou přímo na útočníkem vlastněné zařízení [19]. Je vhodné si zaregistrovat více faktorů, aby v případě výpadku služby (např. výpadek SMS služby) či ztráty faktoru bylo možné se stále přihlásit. Biometrické faktory mohou znamenat nevhodný zásah do osobních dat uživatele, protože pro ověření je nejprve nutné odebrat a uložit srovnávací sadu biometrických charakteristik. Většinou jsou ukládány a porovnávány pouze hashe těchto biometrických dat, ale riziko zneužití stále existuje.

2.3 Autentizační knihovny

Složitost autentizace se spolu s moderní autentizací zvyšuje. K implementaci celého autentizačního řetězce je potřeba značná znalost prostředí. To enormně zvyšuje náklady na vývoj aplikací. Autentizační knihovny tento problém efektivně řeší. Zajišťují veškerou komunikaci se serverem a uživatel pouze zadává přihlašovací údaje. Riziko zanesení chyby do procesu autentizace se tak značně snižuje.

Autentizační knihovny mohou být vyvíjeny přímo dodavateli cloudových služeb. Součástí obvykle je i rozsáhlá dokumentace mapující funkci knihoven.

2.3.1 Knihovna ADAL

Knihovna ADAL (Active Directory Authentication Library) je autentizační knihovna podporující moderní autentizaci. Od roku 2020 se nachází v udržovacím stavu, žádné nové funkce nejsou přidávány. Na konci roku 2022 pak dosáhla konce životnosti EOL (End-of-life). Aplikace, které ji využívají, mohou přestat fungovat [20]. Je tedy zcela bezpředmětné se snažit o tvorbu vlastních aplikací s využitím knihovny ADAL.

2.3.2 Knihovna MSAL

Knihovna MSAL (Microsoft Authentication Library) je vyvíjena přímo společností Microsoft, která ji pod MIT [21] licencí nabízí k využití pro tvorbu vlastních aplikací. Poskytuje přístup především do cloudových služeb Microsoft 365 [22] (dříve Office 365). Slouží k získávání přístupových tokenů, které se poté využívají k přístupu k různým API. Může jít o API přímo od Microsoftu, případně i od třetích stran. Lze tedy vytvořit vlastní aplikaci, která podporuje jednotné přihlašování. Pomocí knihovny MSAL si ověří uživatele (například načtením jeho uživatelského profilu

s e-mail adresou) a provede přihlášení. Podporuje různé toky autentizace. Zejména jde o tok kódu zařízení a tok autorizačního kódu [23].

Podle přiložené konfigurace dokáže fungovat jako důvěrná klientská aplikace nebo i jako veřejná klientská aplikace. Podporuje ukládání tokenů, které v průběhu autentizace získala. Pro každého uživatele udržuje vlastní tokeny. Pokud API vrátí obnovovací token, může zajišťovat automatickou obnovu přístupového i obnovovacího tokenu. Limitem je životnost obnovovacího tokenu, a ta je defaultně nastavena na devadesát dnů. Od roku 2021 nelze životnost tokenu změnit [23].

Ke své funkci vyžaduje vytvoření podnikové aplikace, kde se nastavují oprávnění přístupu k API.

2.4 Podniková aplikace

Prostředí Microsoft 365 [22] pro úspěšně přihlášení vyžaduje využití moderní autentizace. Ta se implementuje využitím jedním z mnoha toků autentizace. Při vytváření vlastních aplikací s využitím autentizační knihovny MSAL je nutné vytvořit podnikovou aplikaci v administračním portále AZURE AD, která ji autorizuje. S takto vytvořenou cloudovou podnikovou aplikací následně komunikuje knihovna MSAL. Výhodou je snadné řízení přístupu. Administrátorovi tenanta (organizace) umožňuje kontrolovat přidělená oprávnění a zobrazit uživatele využívající tuto konkrétní aplikaci. Podle uvážení pak operativně přidává, případně maže přidělená oprávnění. Dále lze podrobně nastavit omezení přístupu buď na úrovni samotné aplikace nebo v závislosti na skupině, ve které se nachází přihlášený uživatel. Po každé změně oprávnění (scopes) je nutné se znovu přihlásit. Tuto technologii využívají i mnozí velcí výrobci softwaru, kteří chtějí využít SSO (Single Sign-on) přihlašování organizace oproti centrálnímu doménovému řadiči. Problém může nastat v případě, kdy administrátor tenanta zakáže uživatelům registraci k aplikacím, které nevytvořil ověřený vydavatel. Když se uživatel následně pokusí přihlásit k externí aplikaci, skončí to chybovou hláškou s důvodem zamítnutí přístupu [24].

2.4.1 Důvěrná klientská aplikace

Důvěrná klientská aplikace (confidential client application) je určena pro aplikace provozované na serveru. Jde například o webovou aplikaci nebo službu, která zajišťuje zálohování dat (je nutné přiřazení patřičných oprávnění). Servery nejsou běžně přístupné koncovým uživatelům a jejich úložiště je považováno za relativně bezpečné. Je proto možné ukládat tajné kódy (hesla) aplikace, aniž by došlo k úniku (neberou se v potaz různí útočníci a chybné konfigurace). Aplikace tedy může přes webové rozhraní poskytovat služby uživateli a na pozadí bezpečně komunikovat s různými

API. Nevýhodou je omezená platnost tajných kódů (hesel), která byla v minulosti omezena na maximálně dva roky. Poté je nutné vygenerovat nový tajný kód (heslo aplikace) a nějakým způsobem ho vyměnit v provozované aplikaci [25].

2.4.2 Veřejná klientská aplikace

Veřejná klientská aplikace (public client application) je určena k provozu na koncových zařízeních. Jde například o počítač nebo zařízení bez integrovaného webového prohlížeče. Není v nich možné zajistit dostatečnou důvěryhodnost, protože zařízení, na kterém je aplikace spouštěna, není pod kontrolou tvůrce. Koncový uživatel tak má přístup ke všem souborům v úložišti. Není proto z bezpečnostního hlediska možné ukládat tajné kódy (hesla aplikace) AZURE AD aplikace nebo její certifikát. To je zároveň i výhodou. Není nutné aplikaci po vytvoření dodávat žádné časově omezené tajné kódy (hesla). Aplikace tak může bez zásahu programátora fungovat teoreticky po celou dobu životnosti aplikace. Přístup je omezen pouze na uživatele, který se musí nejprve přihlásit a potvrdit aplikaci přístup ke svým datům [25].

2.4.3 Toky autentizace

Toky autentizace, označované také jako Authentication Flows, je hromadné označení pro způsoby autentizace vůči prostředí Microsoft 365 [22]. Jsou využívané knihovnou MSAL a tvůrce aplikace si podle zaměření aplikace může vybrat nejvhodnější tok autentizace. Lze je však implementovat i bez knihovny MSAL, nicméně to není doporučováno [26].

Tok kódu zařízení

Tok kódu zařízení, neboli Device code flow, je převážně určen pro zařízení bez klávesnice. Je dostupný pouze při použití veřejné klientské aplikace. Uživateli, který se chce přihlásit například k tiskárně se zobrazí pouze jednoduchý šestimístný kód (znaky A-Z a 0-9). Uživatel si ho zkopíruje a poté se přihlásí na svém primárním zařízení přes webový prohlížeč. Velká výhoda tohoto toku kódu zařízení je, že ke své funkci nepotřebuje žádné časově omezené expirující certifikáty nebo tajné kódy klienta (označováno také jako heslo aplikace) [27]. Při využití k vytváření vlastních aplikací může být výhoda, že uživatel je na svém počítači již přihlášen, a tak se tento způsob přihlašování jeví jako nejjednodušší. Nemusí také na cizím zařízení zadávat svoje přihlašovací heslo a tím pádem minimalizuje riziko úniku hesla.

Princip funkce je následující. Klient kontaktuje server, označován také jako authority / koncový bod. Serveru předá ID AZURE AD aplikace, ID tenanta a požadovaná oprávnění (scopes). Server pošle zpět kód zařízení a verifikační URL adresu.

Jde o údaje, které se zobrazí uživateli. Uživatel se následně přes webový prohlížeč přihlásí s použitím zobrazeného kódu. Klient následně opakovaně zkouší na koncovém bodu (/token) získat přihlašovací token. Získá ho pouze pokud se uživatel úspěšně přihlásí a potvrdí AZURE AD aplikaci přístup. V případě neúspěšného přihlášení nebo vypršení patnáctiminutového intervalu je nutné opakovat veškeré kroky znovu od začátku [27].

Tok přihlašovacích údajů vlastníka prostředku

Tok přihlašovacích údajů vlastníka prostředku, neboli Resource Owner Password Credentials, využívá jméno a heslo uživatele. Uživatel zadá svoje údaje a ty se zašlou v hlavičce HTTP/HTTPS na server. Ten vrátí přístupový a obnovovací token.

Využívání tohoto toku není doporučeno. Není kompatibilní s osobními účty, využít lze pouze v rámci organizace. Dalším limitujícím faktorem je nemožnost nastavit multifaktorovou autentizaci. Jedná se o dočasnou náhradu za klasickou autentizaci. V budoucnu může být tok zrušen. Jeho další existence tedy není zaručena a využití při tvorbě vlastních aplikací je sporné [28].

Tok autorizačního kódu

Tok autorizačního kódu, neboli Authorization Code Flow, je vhodný pro zařízení s webovým prohlížečem. Jde například o desktopového klienta synchronizačního cloudového poskytovatele. Vytvořená aplikace otevře uživateli vyskakovací okno, které ho vyzve k přihlášení. Má stejnou podobu jako běžné přihlašovací okno na webu "office.com" a uživatel může využít vícefaktorovou autentizaci [29].

Princip funkce je následující. Aplikace otevře vyskakovací okno, čímž vyzve uživatele k autentizaci. V případě úspěšné autentizace server vrátí autorizační kód pomocí přesměrování na URL (Uniform Resource Locator) uvedenou při tvorbě AZURE AD aplikace. Autorizační kód bude uveden v parametru URI (Uniform Resource Identifier) adresy ve formátu "/?code=aaAAbb". S pomocí autorizačního kódu aplikace zažádá o přístupový token. Server pošle zpět přístupový a obnovovací token. S těmito tokeny se následně provádí další požadavky na API [29].

3 Přístup ke zprávám

Zprávy (e-maily) jsou uloženy na serveru. Server tyto zprávy může zpřístupnit pomocí různých protokolů. Každá implementace serveru může poskytovat jak svoje vlastní proprietární protokoly, tak i protokoly určené standardy. Vzhledem k vybranému prostředí Microsoft 365 lze vybírat především z POP3, IMAP4, MAPI, Exchange Activesync a Graph API.

3.1 POP3

Protokol POP3 (Post Office Protocol) je velmi jednoduchý protokol co se týče práce se zprávami. Původně je z roku 1996. Příkazů, které slouží k práci s e-maily, je pět (STAT – zobrazí počet zpráv na serveru, LIST – zobrazení zpráv, RETR – stáhnutí zprávy, DELE – zpráva je označena k smazání, RSET – pokud byla zpráva označena ke smazání, tak se označení zruší). Je vhodný pro provoz převážně na jednom zařízení. Běžný uživatel má však zařízení obvykle více, a tak není doporučeno tento protokol používat. Podporuje jen jednostrannou synchronizaci, a to pouze stahování e-mailů ze serveru do zařízení. Stažené zprávy se následně mohou automaticky označit jako smazané příkazem DELE a po ukončení sezení příkazem QUIT server přejde do UPDATE stavu a zprávy odstraní [30]. To je možné brát i jako výhodu, protože některé hostingové firmy nabízí servery s omezenou kapacitou úložiště a rozšíření je zpoplatněno. Dlouhodobou archivaci historie a zálohování e-mailů musí řešit koncový uživatel.

Dále neumožňuje aktualizovat stav e-mailů na serveru (označit barevným štítkem). Přesouvání mezi složkami a ani nahrávání e-mailů na server není podporováno. Historie odeslaných e-mailů se tak ukládá pouze v zařízení odesílatele. Veškeré uživatelsky nastavené vyhodnocování a třídění pošty podle nastavených pravidel musí zajistit e-mailový klient nebo přímo server. Uživatelská nastavení zároveň není možné synchronizovat mezi všechna připojená zařízení.

3.2 IMAP4

Protokol IMAP4 (Internet Message Access Protocol) je novější náhrada za starší protokol POP3. Podporuje oboustrannou synchronizaci mezi serverem a teoreticky neomezeným počtem klientů. Zprávy jsou primárně vždy uloženy na serveru a klient si pouze ukládá lokální kopii do souboru. Jakákoliv změna, třeba označení e-mailu jako přečtený, se postupně propíše na všechna připojená zařízení. Na rozdíl od POP3 umí se serverem synchronizovat všechny složky, včetně odeslaných e-mailů. Pro zobrazení obsahu e-mailu slouží povinně podporovaný příkaz FETCH. Jde o protokol,

který je podporován většinou běžně dostupných e-mailových serverů. Označování e-mailů uživatelsky definovanými barevnými štítky lze dosáhnout pomocí FLAGS nebo KEYWORDS. Dle RFC3501 [31] je ale implementace této funkce volitelná. Není tak ihned zřejmé, jestli vybraný poskytovatel e-mailového serveru podporuje vyžadované funkce. Někteří výrobci serverového software se tak rozhodly je neimplementovat a nutí tím uživatele k používání svých proprietárních protokolů. Pro ověření, jestli server podporuje tyto volitelné funkcionality, se stačí připojit k serveru pomocí OpenSSL [32]. Po vybrání e-mail boxu příkazem SELECT INBOX server vrátí podporované FLAGS. Pokud se mezi nimi není hvězdička, tak uživatelsky definované flagy nejsou podporovány. Přesně tuto situaci zobrazuje obrázek 3.1, který ukazuje komunikaci přes IMAP4 s cloudovým serverem Exchange Online od společnosti Microsoft. Pro použití při tvorbě aplikace dle zadání je proto IMAP4 nevyhovující.

```
2 OK AUTHENTICATE completed.
2 SELECT INBOX
* 6 EXISTS
* 6 RECENT
* FLAGS (\Seen \Answered \Flagged \Deleted \Draft $MDNSent)
* OK [PERMANENTFLAGS (\Seen \Answered \Flagged \Deleted \Draft $MDNSent)] Permanent flags
* OK [UNSEEN 4] Is the first unseen message
* OK [UIDVALIDITY 14] UIDVALIDITY value
* OK [UIDNEXT 489] The next unique identifier value
2 OK [READ-WRITE] SELECT completed.
```

Obr. 3.1: Výpis podporovaných FLAGS na serveru, připojeno přes IMAP4

Výše zmíněný obrázek 3.1 byl pořízen při testovacím přihlášení k serveru Exchange Online [33], kde je od roku 2022 nutné využívat moderní přihlašování [34]. Nelze tedy použít jednoduchý způsob s využitím pouze jména a hesla, se kterým počítají mnohé starší knihovny programovacích jazyků. K přihlášení je nutné získat přístupový token, který je následně využit místo hesla. K jeho získání je nutné vytvořit AZURE AD aplikaci a následně využít například knihovnu MSAL. Nezbytné je přiřadit potřebná oprávnění. V tomto ukázkovém případě stačí přidat API oprávnění s názvem IMAP.AccesAsUser.All. Pouhé získání přístupového tokenu však není dostatečné. Pro úspěšnou autentizaci vůči serveru je nutné do formátu base64 zakódovat i uživatelské jméno. Po absolvování všech těchto kroků už je následně možné se k serveru připojit pomocí OpenSSL, využít příkaz AUTH=XOAUTH2 a přiložit zakódovaný řetězec. Pokud nikde v předchozích krocích nenastala chyba, dojde k úspěšnému přihlášení. Pomocí příkazů definovaných v normě RFC3501 je možné následně ručně procházet uložené zprávy na serveru.

3.3 MAPI

Messaging Application Programming Interface (MAPI) je rozhraní mezi OS Windows a aplikacemi. Těm umožňuje odesílat a načítat e-maily. Výměna dat mezi serverem a klientem probíhá pomocí běžného protokolu HTTP, což zvyšuje spolehlivost a snižuje chybovost spojení při vzájemné komunikaci. Velkou výhodou je funkce pozastavit a pokračovat. Pokud tedy uživatel převede svůj počítač do režimu spánku a následně se připojí z úplně jiného místa, tedy i IP adresy, pak při obnovení činnosti počítače se nemusí znovu navazovat celé spojení od nuly. Server si po určitou nastavenou dobu pamatuje kontext relace a při obnovení TCP spojení komunikace pokračuje tam, kde přestala před výpadkem připojení. Není nativně podporován v některých OS a nesplňuje tak požadavek na multiplatformní provoz [35].

3.4 Exchange ActiveSync

Exchange ActiveSync je proprietární protokol vyvíjený společností Microsoft. Spolupracuje pouze se servery Exchange. Určen je především pro mobilní zařízení. Je optimalizován pro pomalé internetové připojení. Pracuje s protokolem HTTP/HTTPS, přičemž požadavky mezi serverem a klientem se posílají ve formátu XML [36]. Nepodporuje sdílené e-mailové schránky. Pro vlastní implementaci nejsou k dispozici knihovny s vhodnou licencí. Tato omezení neúměrně zvyšují náročnost na vytvoření aplikace dle zadání. Proto byl vyřazen jako nevyhovující.

3.5 Microsoft Graph API

Microsft Graph je REST API[37], které umožňuje přístup k téměř všem uloženým datům a nastavením v rámci platformy Microsoft 365 [22] (původně Office 365), včetně e-mailového řešení Exchange Online [33]. Pracuje na principu požadavek-odpověď přes protokol HTTPS. Autentizace klienta vůči serveru se provádí za použití přístupového tokenu, který je nutné získat externím způsobem. Podporuje všechny požadavky na tvorbu aplikace dle zadání. Především tedy možnost zprávy číst, přesouvat, označovat uživatelsky definovanými štítky a odesílat. K dispozici je velmi rozsáhlá dokumentace i s příklady použití v programovacím jazyku. Obsahuje také příklady komunikace, tedy formát dat, co má klient odeslat a následně také očekávanou odpověď serveru [37]. Vzhledem k výše uvedeným vlastnostem se jeví využití MS Graph API jako nejlepší možný způsob pro přístup k serveru.

3.5.1 Webové rozhraní Graph Explorer

Webové rozhraní Graph Explorer [38] poskytuje přístup k API Microsoft Graph [37]. Je tak možné si dopředu vyzkoušet tvorbu dotazů a zároveň zobrazit strukturu odpovědi. Bez přihlášení obsahuje nachytané testovací prostředí. Pokud se však uživatel přihlásí, může operativně přidávat a odebírat příslušná oprávnění. Práce s webem je velmi jednoduchá. Obsluhovací část je rozdělena do přehledných oddílů. Každý oddíl reprezentuje část dotazu na API (tělo a hlavička požadavku). Jsou zde předpřipravené i testovací dotazy, aby bylo vidět, jakým způsobem lze zadávat příkazy.

Při testování příkazů je možné zobrazit soubor oprávnění s detailním popisem, které je možné použít k úspěšnému provedení dotazu. Programátor si tedy může zvolit nejlepší možné oprávnění pro danou situaci. Pokud ještě není v rámci vývoje aplikace implementovaná knihovna MSAL, lze Graph Explorer využít pro získání přístupového tokenu. Lze ho zde zobrazit a použít při testování ve vlastní aplikaci.

3.5.2 Delegovaná oprávnění

Delegovaná oprávnění (Delegated Permissions) nabízí omezení přístupu na jednotlivé uživatele. Z pohledu API, na které se aplikace snaží přistupovat, vystupuje jako přihlášený uživatel. To je vhodné pro aplikace, které potřebují mít přístup pouze k datům daného uživatele. K datům ostatních uživatelů, kteří jsou v rámci organizace také zaregistrováni, se bez jejich přihlášení a odsouhlasení přístupu nemůže vytvářená aplikace dostat [23]. Delegovaná oprávnění jsou tedy vhodná pro tvorbu aplikací typu e-mailový klient.

3.5.3 Oprávnění aplikace

Oprávnění na úrovni aplikace (Application Permissions) mají přístup k datům všech uživatelů v rámci organizace (podle nastavených oprávnění). Aby se zamezilo zneužití ze strany běžného uživatele, tak pro všechna oprávnění, která je možné aplikaci přidělit, je nutný souhlas správce organizace. Ten může zároveň vybrat okruh uživatelů, kterým přístup k aplikaci povolí. Oprávnění na úrovni aplikace je vhodné pro aplikace, které slouží například k zálohování dat všech uživatelů. Není nutné žádat o individuální souhlasy. Nicméně to nese s sebou rizika, kdy kompromitované tokeny jedné aplikace mohou pozměnit či smazat data kompletně celé organizace. Velmi také pomáhá s automatizací některých kroků. Jednotlivým uživatelům lze centrálně změnit nastavení v profilu [23].

3.5.4 Omezení souhlasu uživatelů

Defaultně je v prostředí Microsoft 365 povoleno uživatelům odsouhlasit přístup aplikacím i bez ověřeného vydavatele. To představuje značné riziko, kdy útočník může založit podvodnou aplikaci, uživateli poslat e-mail s falešným obsahem, který ho pomocí sociálního inženýrství donutí přístup potvrdit. V rámci zvýšení zabezpečení organizace nebo tenanta je doporučeno uživatelům omezit možnost potvrzovat přístupy bez souhlasu správce [39].

Zamezení odsouhlasení přístupu uživatelem lze nastavit v AZURE AD portálu v podnikových aplikacích (enterprise applications). Zde jde nastavit omezení buď pouze pro aplikace od ověřených vydavatelů nebo úplná restrikce "nepovolit souhlas uživatelů" kdy dojde k celkovému zákazu potvrdit jakoukoli aplikaci bez souhlasu správce. Dále je potřeba se zaměřit na souhlas vlastníka skupiny a odsouhlasení také zakázat. V rámci zachování možnosti přihlašování k aplikacím je možné nastavit notifikace vybraným administrátorům, kteří situaci zváží a aplikace povolí [39].

4 Návrh vlastního řešení

Výběr dílčích technologií se odvíjel podle požadavků zadání. Jednotlivé požadavky byly pečlivě rozděleny na menší části, aby mohly být řešeny samostatně. V některých případech existuje více možností řešení. Vyhodnocení výhod a nevýhod každého řešení bylo nezbytné pro výběr té nejvíce vhodné technologie vzhledem k vyžadovanému prostředí.

4.1 Volba programovacího jazyka

Základní požadavek při výběru programovacího jazyka bylo zajistit, aby jádro aplikace bylo připraveno na multiplatformní provoz. Z tohoto důvodu byl vybrán programovací jazyk Python [40]. Ten je podporován na všech nejrozšířenějších i některých minoritních platformách. Jsou pro něj aktivně komunitou vyvíjeny různé knihovny, které usnadňují práci a programátor se může soustředit pouze na podstatu řešeného problému. Další výhodou je poměrně jednoduchá práce s listy a slovníky, které budou v architektuře aplikace využity ve velké míře. K vývoji byla použita starší verze 3.7 pro zachování určité zpětné kompatibility. Oproti novější verzi 3.9 totiž nepodporuje některé nově přidané funkce pro lepší práci s textem [41]. Není však problém aplikaci spustit v novějších verzích. Výsledná aplikace bude dostupná ve dvou variantách.

4.1.1 Varianta exe

Pro prostředí Microsoft Windows [42] byla zvolena forma přímo spustitelné aplikace, tedy exe soubor. Všechny potřebné soubory a knihovny budou zabaleny do jednoho exe souboru. Uživatel nebude muset pro správnou funkci aplikace dodatečně nic instalovat.

Parametry budou předávány jednak ve formě konfiguračního souboru, jednak jako argument při spouštění aplikace. Tímto argumentem bude možné vybrat název konfiguračního souboru, který má být použit. Jedna aplikace může vyhodnocovat teoreticky neomezený počet e-mailových schránek. Aplikace může být periodicky spouštěna za pomoci vestavěné funkce OS Windows plánovačem úloh. Případně bude možné aplikaci spouštět ručně dvojklikem, alternativně přes příkazový řádek. Může být také spouštěna bez přihlášeného uživatele a pracovat tak na pozadí. Nevyžaduje žádná zvýšená oprávnění. V případě automatického provozu není nutné při spuštění aplikace zobrazovat okno s výpisem. Omezí se tím vyrušení uživatele v případě provozu na běžném počítači.

Pro snadnější obsluhu a konfiguraci bude v oddělené aplikaci vytvořena grafická nadstavba. Ta bude využívat funkcí jádra aplikace, se kterým může komunikovat.

4.1.2 Varianta py

Tato varianta je určena především pro OS Linux, kde je Python [40] podporován. Některé distribuce jej obsahují defaultně. Aplikace také může být spouštěna na mini PC s nízkou spotřebou. Například na Raspberry PI nebo na VPS u hostingové firmy. I tato varianta bude kompatibilní s grafickou nadstavbou. Na rozdíl od varianty exe, spuštění této varianty vyžaduje přípravu operačního systému. Jedná se o instalaci všech potřebných knihoven. Pro méně zdatného uživatele to může představovat značný problém.

4.2 Autentizace a autorizace aplikace

Vytvářená aplikace bude pro veškerou autentizaci proti cloudovým serverům využívat knihovnu MSAL [21]. Ta zajistí všechny potřebné operace související s úspěšnou autentizací. Bude také spravovat získané tokeny. Po přijetí tokenu ho vždy ihned uloží do souboru v interním formátu. Automaticky vyřeší obnovu přístupového tokenu, pokud od minulého použití vypršela jeho platnost. K tomu využije obnovovací token. Pokud vyprší platnost obnovovacího tokenu, bude uživatel vyzván k opětovnému přihlášení. Maximální doba platnosti obnovovacího tokenu je pevně stanovena na 90 dnů. Každý uživatel, který bude chtít nechat kontrolovat a vyhodnocovat svoji schránku, se bude muset individuálně přihlásit. Je potřeba zdůraznit omezení tohoto řešení. Pokud nebude aplikace delší dobu využívána, dojde k vypršení obnovovacího tokenu a uživatel bude nucen se znovu přihlásit. Aplikace je navržena tak, že má stejný přístup k datům jako přihlašovaný uživatel. Pouze tedy s omezením nastavených v oprávnění pro API.

4.2.1 AZURE AD podniková aplikace

Autentizační knihovna MSAL [21], která využívá moderní metodu přihlašování, vyžaduje ke své funkci vytvořit podnikovou aplikaci [44] (autorizace). Po zvážení všech ostatních možností bude doporučeno povolit přihlašování pouze z účtů náležících pod stejného tenanta. Tenant je jiné interní označení pro název jedné organizace nebo domény dané firmy. Metoda ověřování byla zvolena "tok kódu zařízení" (Device Code Flow). S tím také souvisí volba delegovaných oprávnění. Vytvářená aplikace tedy bude mít přístup pouze k datům přihlášeného uživatele. Asi nejdůležitějším důvodem byla absence nutnosti využívat expirující tajné kódy (hesla aplikace) klienta.

Jejich maximální platnost totiž není možné nastavit na více než dva roky. Poté je nezbytné vygenerovat nové tajné kódy klienta a upravit konfigurační soubor aplikace. Mohlo by se stát, že obsluha zapomene tento tajný kód včas vyměnit. V rámci návrhu byla snaha eliminovat problémy tohoto typu a tak byly ostatní toky vyřazeny jako nevhodné.

Při počáteční fázi návrhu funkčnosti aplikace byla zjištěna potřeba získat pouze minimální oprávnění a to "User.Read", "Mail.ReadWrite" a "Mail.Send". Při kompromitaci tokenů jednoho uživatele tedy nemůže dojít k úniku jiných dat, než ke kterým byl povolen přístup. V tomto případě jde pouze o přístup k mailboxu, zobrazení jména uživatele a odesílání e-mailů jménem uživatele.

4.3 Přístup a manipulace s daty

Pro přístup a úpravě zpráv bylo zvoleno rozhraní Microsft Graph API [37], které umožňuje přístup k téměř všem uloženým datům a nastavením v rámci platformy Microsoft 365 [22] (původně Office 365). Pracuje na principu požadavek odpověď přes protokol HTTPS, což značně zjednodušuje vlastní implementaci aplikace. Vyhovuje všem požadavkům na tvorbu aplikace dle zadání. Především jde tedy o možnost zprávy číst, přesouvat, označovat uživatelsky definovanými štítky. Toto API je vhodné i pro budoucí rozšiřování funkcí aplikace. Bude využito i k odesílání notifikačních zpráv. Samotné zpracování dat zajistí jádro vytvářené aplikace. V průběhu vyhodnocování bude jádro aplikace následně přes Microsoft Graph API provádět úpravy zpráv na serveru.

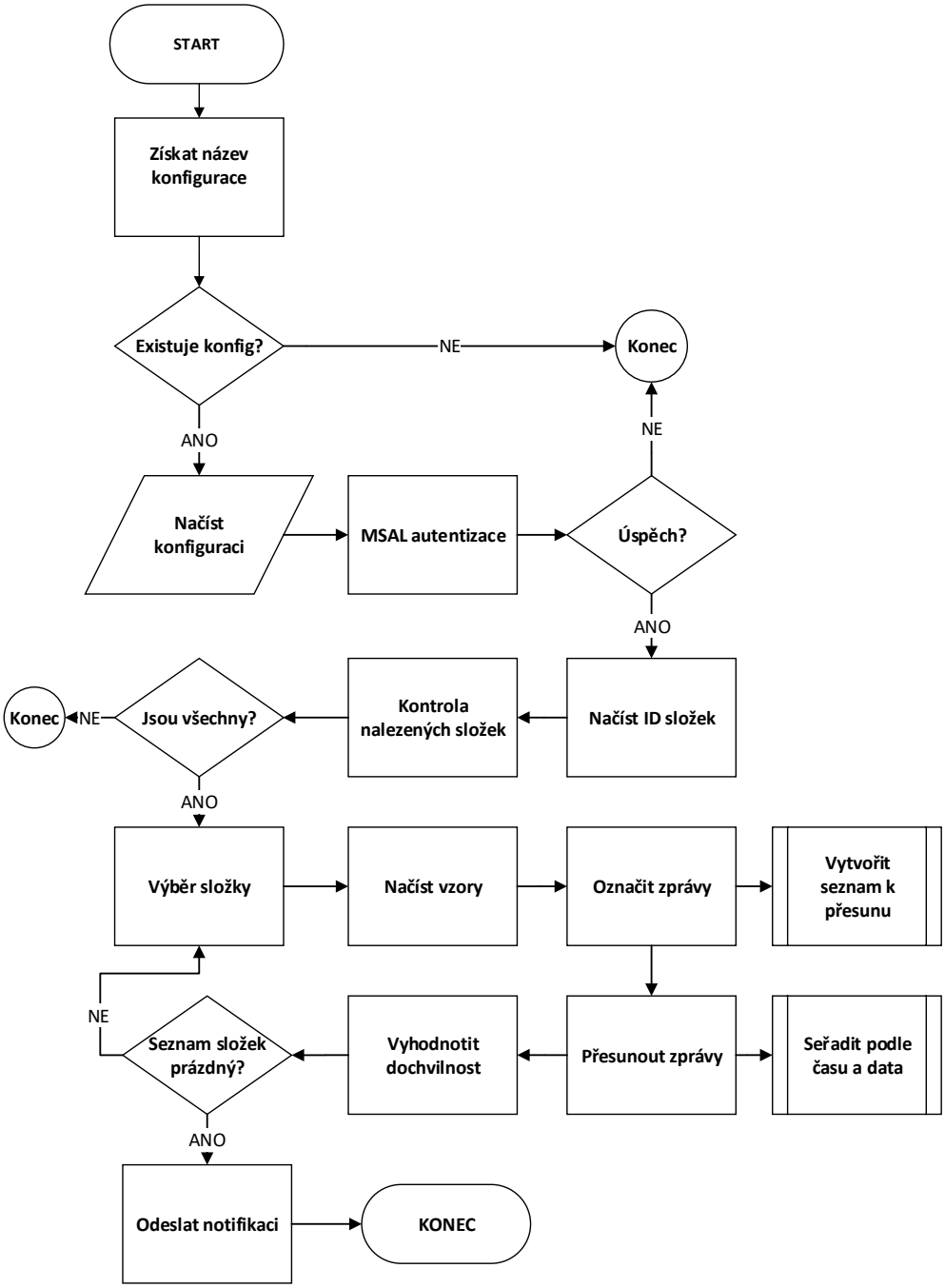
4.4 Grafická nadstavba

Grafická nadstavba má za cíl zjednodušit správu aplikace. Obsahovat bude veškerá potřebná nastavení a úpravu konfiguračních souborů. Bude vytvořena formou webové stránky za využití frameworku Flask. Výhodou je snadná úprava vzhledu a rozvržení ovládacích prvků. Určena je pouze pro lokální provoz na jednom počítači. Nebude tedy podporovat žádné přihlašování.

Součástí bude i nástroj pro dohledání historie zpráv. V případě, že dojde k anomálii v podobě zpožďování zpráv, bude možné zobrazit celou historii. Výsledkem bude seznam, kde administrátor může vyhodnotit četnost opakování nežádoucího stavu. Pro další analýzu mimo aplikaci bude nástroj umožňovat export zpráv do textového souboru.

4.5 Struktura jádra aplikace

Jádro aplikace je složeno z celkem deseti hlavních modulů. Pro snadnější pochopení základních funkcí aplikace byl obrázek 4.1 zjednodušen pouze na průchod a vyhodnocení zpráv.



Obr. 4.1: Zjednodušená struktura jádra aplikace

5 Implementace jádra aplikace

Jádro aplikace obstarává automatizované bezobslužné monitorování e-mailových zpráv. Podle nastavených pravidel zprávy vyhodnocuje, třídí a při vzniku incidentu (příjem mimořádné chybové zprávy) informuje o této skutečnosti administrátora.

Aplikace funguje zcela samostatně (není závislá na grafické nadstavbě) a díky využití jazyka Python umožňuje multiplatformní provoz. Vnitřní struktura aplikace byla rozdělena na individuální moduly. Každý modul zpracovává svoji přiřazenou část dat, která zpracuje a následně výsledek své činnosti předá navazujícímu modulu.

Při běhu aplikace mohou vznikat různé neočekávané chyby, které mohou zapříčinit pád aplikace. Proto byly chyby rozděleny podle závažnosti. Při nízké závažnosti (například nenastavený příjemce notifikací) daný modul vypíše chybu a aplikace pokračuje dále. Při závažné chybě (například chybná odpověď od API) se aplikace automaticky ukončí. Sníží se tak pravděpodobnost vzniku nekonzistence dat. Pokud to povaha problému dovoluje, tak se aplikace snaží o těchto mimořádných událostech informovat správce.

5.1 Kontrolní modul

Při spuštění aplikace (ať už ručně nebo plánovaně) se zkontroluje existence elementárních konfiguračních souborů. Jako další se kontroluje existence přístupového tokenu. Pokud soubory neexistují anebo není k dispozici vhodný token, dojde k ukončení aplikace. V tomto případě nedochází k varování administrátora pomocí e-mailu.

5.2 Konfigurační soubory

Konfigurační soubory obsahují informace nutné pro provoz aplikace. Pro jednoduchou správu těchto informací byla vybrána knihovna ConfigParser [45]. Vyznačuje se úhledným ukládáním a načítáním informací z běžného textového souboru. Data se ukládají ve formátu "klíč = data", což se při vyčítání dat interpretuje ve formátu slovníku, který je obsažen v jazyce Python. Výsledný formát "[klíč:data]"je vhodný pro další zpracování jako je přidávání, odebírání a editace dat. Velkou výhodou je především snadná čitelnost výsledných textových souborů uživatelem.

Konfigurační soubory se načítají výhradně ihned po spuštění aplikace. Na pozdější úpravy není brán zřetel. Pro lepší přehlednost využívá aplikace více konfiguračních souborů. Jejich celkový počet závisí na počtu vytvořených úloh. Samotná aplikace vyžaduje hlavní konfigurační soubor s názvem "MNinternal.cfg", který obsahuje globální a výchozí hodnoty pro ostatní moduly. Každá úloha vytvořená přes

grafickou nadstavbu pak generuje dva soubory (nastavení úlohy a nastavení dochvilnosti). V jedné složce zároveň s aplikací tak může být více konfiguračních souborů s rozdílným nastavením. Efektivně se tím vyřešil i problém s příliš obsáhlým konfiguračním souborem. V případě poškození dílčích souborů je snadnější nahradit jeden konkrétní soubor než nahrazovat jeden kompletní soubor.

5.2.1 Funkce konfiguračního modulu

Při spouštění aplikace je možné definovat název vyžadovaného konfiguračního souboru argumentem. V potaz se bere pouze první parametr. Pokud není specifikován žádný název konfiguračního souboru, použije se výchozí název "MNuser.cfg". Dalším krokem je zjišťování existence souboru s daným názvem ve složce config. V případě nenalezení žádného souboru s hledaným názvem, případně chybně nastaveném parametru v plánovači úloh se vyvolá kritická chyba. V důsledku se aplikace sama ukončí. Při úspěšném načtení souboru tento modul z daného souboru načte požadované informace. Ty jsou pak dispozici ostatním modulům, které s daty pracují. V případě vložení dalších klíčů uživatelem do konfiguračních parametrů, které nejsou v aplikaci uvedeny, se tyto přeskočí.

5.3 Modul autentizace

Modul autentizace obstarává přístup k API serveru. Získává a obnovuje přístupové a obnovovací tokeny. Implementována byla metoda přihlašování pomocí toku kódu zařízení (Device Code Flow) s využitím veřejné klientské aplikace [46]. Modul je v odděleném souboru, což zajistí jeho modulárnost a nahraditelnost. V případě budoucího požadavku na jinou metodu přihlašování se celý tento modul nahradí adekvátní verzí. Hlavní částí modulu jsou obslužné kódy, které komunikují s knihovnou MSAL [21]. Knihovně se z konfiguračního souboru předají nezbytné informace. Jde o e-mail adresu, ID tenanta, ID aplikace, ověřovací autoritu (authority) a žádaná oprávnění (scopes). Po zavoláni modulu autentizace se knihovna MSAL prvně podívá do své mezipaměti, která je uložena v souboru my_cache.bin. V dalším kroku ověří, jestli nemá z dřívějšího spuštění uložen obnovovací token pro zvolený účet. Pokud ne, kontaktuje ověřovací autoritu a získá náhodně vygenerovaný kód, který následně zobrazí uživateli společně s URL adresou. Ten pak v jakémkoli moderním externím webovém prohlížeči otevře uvedenou adresu a zadá tento vygenerovaný kód. Přihlásí se a server pošle knihovně nazpět přístupový i obnovovací token. Pokud už se zvolený účet v minulosti přihlásil, načte si knihovna MSAL z mezipaměti obnovovací token, přes který získá přístupový token a ten vrátí zpět jádru aplikace. Nově získané tokeny a mezipaměť si následně ihned ukládá do souboru. Pokud před

ukončením aplikace došlo ke změně tokenů, uloží je znovu. V jednom mezipaměťovém souboru může být uloženo více uživatelských účtů. V případě neúspěšného přihlášení se aplikace nuceně ukončí. Bez úspěšné autentizace totiž nemá žádný přístup k datům na serveru. Je proto bezpředmětné pokračovat k dalším krokům.

Pokud je využita grafická nadstavba, pak jádro aplikace nezajišťuje získání tokenu ani přihlášení.

5.4 Načtení ID složek podle jejich názvu

Tato část (DisplayNameToID) obstarává zpracování parametru folders z konfiguračního souboru. Jde o to, že Microsoft Graph API [37] neumí přímo pracovat se zobrazovanými názvy složek (Display Name). V jedné e-mailové schránce může reálně existovat mnoho složek se stejným názvem. Každá složka má však svoje unikátní ID se kterým se následně dále pracuje. Pro správnou funkci modulu je ale potřeba, aby schránka v žádném případě neobsahovala složky se stejným názvem. Tento modul je nezbytný i z důvodu dalšího omezení Microsoft Grapf API, které i podle oficiální dokumentace [47] neumí najednou vypsat celou stromovou strukturu složek. Vždy lze zobrazit pouze nadřazenou složku (parent folder) a její podřazené složky o jednu úroveň níže (child folders). Pokud má podřazená složka další podsložky, musí modul do této složky vstoupit a znovu zažádat o vypsání podřazených složek. Výsledkem je nutnost postupně procházet všechny složky a jejich podsložky a průběžně si ukládat shody. Nevýhodou je zbytečná zátěž na API. Pro částečnou eliminaci zbytečného zatížení API byl implementován konfigurační parametr “do_not_scan“, který automaticky vyřadí všechny další podsložky ve stromové struktuře uvedených složek. Každý mailbox obsahuje vždy několik interních složek, které nejsou zobrazovány. Pro českou lokalizaci schránky proto byl vytvořen výchozí seznam nepotřebných složek.

Tento modul si tedy postupně vyžádá od Microsoft Graph API [37] výpis složek, které zpracuje. Postupuje stromovou strukturou a pokud složka obsahuje další podsložky, uloží si její ID do pracovního seznamu, který postupně prochází, dokud není prázdný. V případě shody názvu složky s hledaným názvem se do finálního seznamu uloží unikátní ID a zobrazovaný název složky (Display Name).

Výsledky, které modul získá, jsou dostupné pouze za běhu aplikace. Po jejím ukončení se automaticky smažou. Zabraňuje se tím tak různým nesrovnalostem, které může uživatel způsobit například přesunutím složky, kdy se její ID může automaticky změnit. Značnou nevýhodou tohoto řešení je nutnost při každém spuštění znovu procházet všechny složky a podsložky dané schránky. Podle rozsáhlosti schránky může hledání unikátních ID trvat přes třicet sekund.

5.4.1 Kontrola nalezených složek

Modul kontrola nalezených složek (checkfolders) před předáním seznamů zpět do jádra aplikace provede kontrolu, že byly nalezeny všechny hledané složky. Pokud se nějaká složka nenajde, tedy seznam hledaných a nalezených není stejný, spustí se vyhledání názvů nenalezených složek. Ty se následně vypíší uživateli a aplikace se nuceně ukončí. Ochrání se tak konzistence dat na serveru. Problém by mohl nastat v modulu přesouvání, který vyžaduje existenci zdrojové a cílové složky.

Existenci složek pro přesun testuje zároveň i modul procházení složek. Modul kontrola nalezených složek totiž slouží i pro manuální servisní mód aplikace, kde není nutné, aby toto kontroloval.

5.5 Procházení složek

Procházení složek je stěžejní součástí jádra aplikace. Od předešlého modulu (DisplayNameToID) dostane seznam finálních názvů a ID složek, které má za úkol zpracovat. Postupně prochází seznam složek. Nejdříve zkontroluje, zda se v seznamu nachází cílová složka pro přesun. K názvu složky tedy přidá podtržítko a následně se snaží najít shodu. Pokud ji nenajde, složka se přeskočí a uživateli se vypíše chyba. Aplikace se neukončí a pokračuje s další složkou. Pokud je kontrola cílové složky pro přesun úspěšná, zažádá Microsoft Graph API [37] o výpis všech zpráv v dané složce. V případě, že odpověď obsahuje následný link (next link), musí žádat opakovaně, protože API vrací padesát záznamů najednou (počet vracených záznamů lze konfigurovat). V této opakované žádosti využije dodaný následný link. Po každé přijaté odpovědi pošle data k vyhodnocení do modulu načtení vzoru. Až dojde na konec, tedy odpověď od API už neobsahuje následný link, začne znovu procházet všechny zprávy. Odpověď od API ale posílá do modulu označování. Tímto je zajištěno, že se nejdříve najdou všechny vzorové zprávy. Vzorová zpráva je rozpoznána podle přiřazené kategorie (výchozí hodnota je _TEMPLATE). Až poté dojde k samotnému vyhodnocení všech zpráv dané složky. Po dokončení předchozích operací se spustí modul přesouvání. Poslední akcí je smazání sdílených seznamů, jejichž obsah je pro každou složku specifický a není nutné je nikam ukládat.

5.6 Načtení vzoru

Modul načtení vzoru (workONtemplate) automaticky prochází všechny zprávy dané složky. Hledá zprávy, které se shodují se vzorem. Název kategorie tohoto vzoru je jasně definován v konfiguračním souboru. V případě, že je zpráva označena kategorií (barevným štítkem) s názvem vzoru (například _TEMPLATE) se do sdíleného

seznamu LoadedTemps ukládá předmět, odesílatel zprávy a názvy všech kategorií zprávy.

Pokud je zpráva označena pouze vzorovou kategorií, zpráva se přeskočí a do seznamu neuloží. Je to považováno za chybu uživatele, protože cílem je zprávu označit nějakou kategorií. Zprávy zůstanou nezpracované. Omezí se tak zbytečná zátěž na API a ostatní moduly. Zároveň se zrychlí zpracování v modulu přesouvání.

5.7 Označení zpráv

Modul označování (SetCaregories) dostane od modulu procházení odpověď od API, kterou zpracuje. Má k dispozici sdílené seznamy s jejichž pomocí zjišťuje, zda je stejný odesílatel a stejný předmět. Když narazí na zprávu, která má vzorovou kategorii, přeskočí ji a pokračuje na další zprávu. Pokud odesílatel a předmět souhlasí s některým vzorem, načtou se potřebné kategorie. Z nich je odstraněna vzorová kategorie (například _TEMPLATE). Všechny ostatní kategorie se společně s unikátním ID zprávy pošlou přes API k označení. Zároveň se zpráva označí jako přečtená. Její unikátní ID, datum přijetí, předmět a cílová složka se vloží do sdíleného seznamu ListOFtobemoved. Tento sdílený seznam využije další modul.

5.8 Přesouvání zpráv

Modul přesouvání zpráv (moveMessage) porovnává a přesouvá zprávy do určených složek. Nejprve ze sdíleného seznamu ListOFtobemoved, který naplní modul označování, vybere jednu skupinu zpráv. Skupina zpráv je rozeznávána podle stejného předmětu a odesílatele. Zprávy ve skupině následně seřadí podle data a času přijetí. Poslední přijatou zprávu ponechá v původním umístění a ostatní zprávy přesune do cílové složky (název původní složky + podtržítko). Zároveň do konzole vypisuje, která zpráva je zrovna přesouvána.

Přesun zprávy na serveru Microsft 365 [22] je interně implementován tak, že nejdříve se původní zpráva zkopíruje do cílového umístění a následně se v původním umístění smaže [48]. Výsledkem je, že každá zpráva ztratí svoje původní unikátní ID, což prakticky vylučuje možnost kontroly přesunutých zpráv v cílové složce. Pokud totiž zpráva při přesunu zmizí, nelze s tím z pohledu aplikace nic udělat. Je nutné se tedy spolehnout na správnou funkci API. Rychlost přesouvání také závisí na stavu API. Každá zpráva se totiž přesouvá zvlášť.

5.9 Analýza dochvilnosti zpráv

Analýza dochvilnosti zpráv je důležitou součástí jádra aplikace. Při provozu vyhodnocuje, jestli se zpráva pouze zpozdila, anebo nepřišla vůbec. V případě monitorování kritických systémů je velmi důležité získat informaci o vznikajícím problému brzy. Při sledování zálohovacích systému může chybějící zpráva znamenat, že zálohovací software je mimo provoz.

Modul, který dochvilnost vyhodnocuje, se spouští až po vyhodnocení zpráv. Obsahuje dvě funkce, které obstarávají samotné zpracování.

5.9.1 Detekce zpoždění zprávy

Při implementaci modulu bylo nezbytné vyřešit několik zásadních problémů. Především šlo o nastavení akcí v případě nečekaných situací. Při výpadku spojení na zařízení se může stát, že se zprávy nahromadí. Po obnově připojení se všechny takto shromážděné zprávy odešlou zároveň. Výsledkem bude hromada zpráv, které přišly mimo očekávané časové sloty. Musí proto proběhnout kontrola počtu zpráv, jestli nepřesahuje určitou přijatelnou mez. Modul musí správně vyhodnotit všechny možné rozestupy zpráv. Ať už jde o krátké časové úseky v řádu minut, tak si musí poradit i s delší časovou periodou (řádově týdny). Chybějící zpráva značí další problém, se kterým modul musí počítat.

Detekce zpoždění zprávy pracuje pouze s nově přijatými zprávami. Z konfiguračního souboru se načte definice kategorie, aby bylo zřejmé, jaký očekávaný rozestup každá kategorie má. Následně se podle aktuálního času a času nejstarší zprávy vygeneruje seznam očekávaných zpráv.

Po vygenerování seznamu se modul snaží každou zprávu přiřadit k jednomu časovému slotu. Pokud by na jeden slot připadalo více zpráv, znamená to zpoždění mimo všechny limity. Tento stav se modul pokusí vyřešit, aby bylo možné vyhodnotit ostatní zprávy. Vyřešením stavu se rozumí vyhodnocení pravděpodobnosti, která zpráva nebyla očekávána a narušila tak očekávaný řetězec zpráv. Toho je docíleno seřazením zpráv a srovnáním času. Ta zpráva, která se nejvíce blíží očekávanému času přijetí, je označena jako pravděpodobně očekávaná. Zbylá zpráva (případně více zpráv) je vyhodnocena jako chyba a vygeneruje se text upozornění pro notifikační modul. Může také jít o chybu uživatele, který nesprávně definoval rozestup mezi zprávami.

Přiřazení více než dvou zpráv k jednomu časovému slotu značí zásadní problém. Mohlo dojít k záseku zařízení, které nyní všechny čekající zprávy odeslalo najednou. Zasažený slot se ihned označí jako chyba a žádná pravděpodobnost se nevyhodnocuje. Jestliže je vhodné vyhodnotit dochvilnost ostatních slotů, tak se vyhodnotí.

V opačném případě se do modulu notifikací zařadí zpráva informující o incidentu.

Pokud na jeden slot připadá právě jedna zpráva, přistoupí se k vyhodnocení dochvilnosti. Každý vzor může mít definovanou vlastní maximální odchylku dochvilnosti, která má přednost před globální hodnotou. Nastavení individuálních časů dochvilnosti je volitelné. Vždy je k dispozici globální hodnota, která se případně využije. V případě překročení limitu se informace o zpoždění zapíše do zásobníku v notifikačním modulu. Administrátor tak je informován o stavu.

Překročení maximální odchylky dochvilnosti je zjištěno přičtením hodnoty odchylky k očekávanému času přijetí zprávy. Reálný čas přijetí zprávy a očekávaný čas plus odchylka se porovnají. Z důvodu technologického omezení se nevyhodnocují sekundy.

5.9.2 Detekce chybějící zprávy

Detekce chybějících zpráv pracuje na principu porovnání počtu zpráv přijatých a počtu zpráv očekávaných. Pokud bylo přijato stejně zpráv, jako bylo očekáváno, pravděpodobně žádná zpráva nechybí. Nicméně je stále nutné prověřit, jestli opravdu nepřišly nějaké neočekávané zprávy. To by tuto kontrolu jednoduše vyřadilo. Proto se čeká na vyhodnocení dochvilnosti, kde tyto informace budou zjištěny. Rozsah pracovní kontroly musí být velký. Tím je myšlen velký rozestup mezi starou a novou zprávou, který může v hraničních případech dosahovat délky několika let.

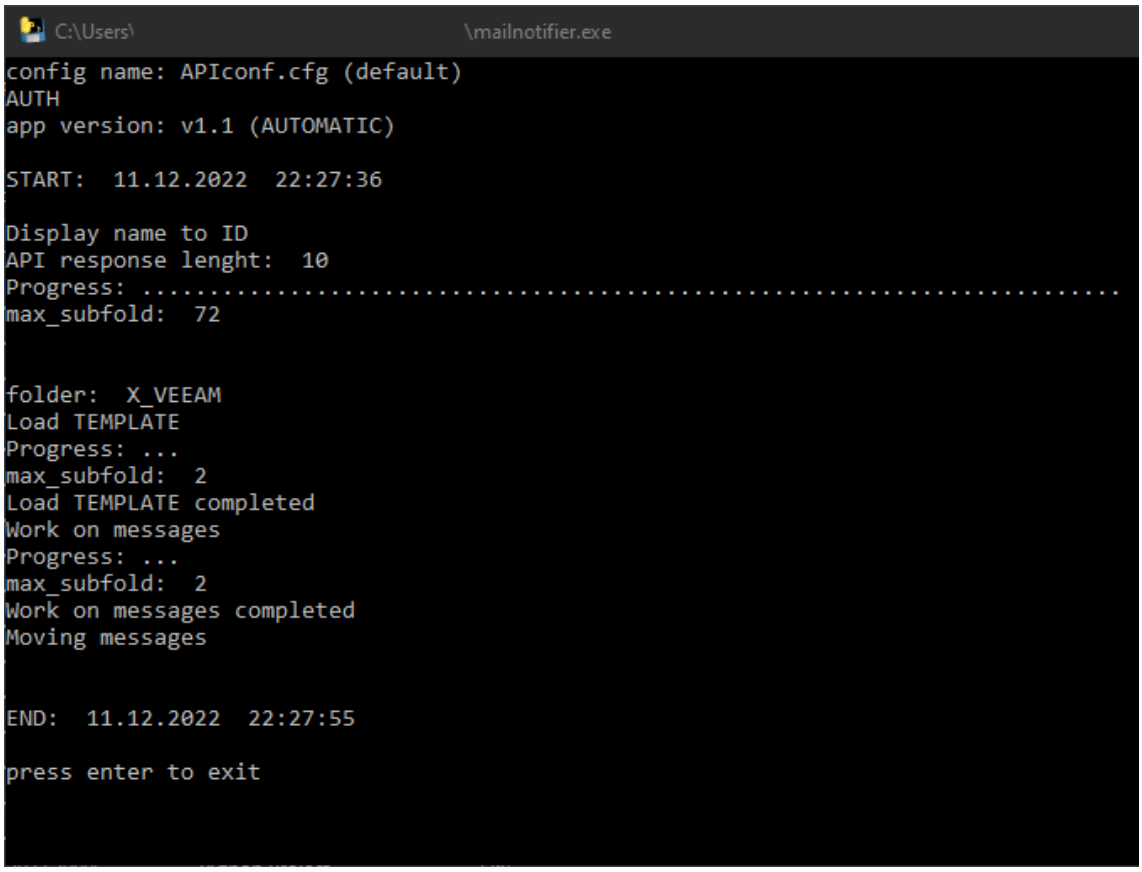
Opět pracuje pouze s přijatými zprávami. Vychází to z předpokladu, že již vyhodnocené a zpracované zprávy v této fázi není nutné znovu načítat. Vyhodnocení chybějících zpráv již proběhlo v předcházejícím běhu.

V případě detekce chybějící zprávy se do zásobníku notifikačního modulu uloží zpráva s informací o očekávaném čase, předmětu zprávy a odesílatele. Bude tak možné zjistit, která zpráva chybí a nepřišla vůbec.

5.10 Výpis postupu

Vytvářená aplikace může podle velikosti schránky, tedy počtu zpráv ve složce, provádět jednotlivé operace v delším časovém úseku. Uživatel by mohl nabýt dojmu, že se aplikace zasekla a nic nedělá. Proto byl v jednotlivých modulech implementován výpis postupu. Aplikace vždy vypíše složku, kterou zrovna bude analyzovat a zpracovávat. Následně se přihlásí aktuálně spuštěný modul, který vypíše svůj název. Pod ním se postupně vypisuje postup ve formě teček. Každá tečka, která v řádku přibude, znamená jednu sadu proběhlých operací. Pokud by se aplikace zasekla, tečky přibývat nebudou. Poté se informačně vypíše stav ochrany (max_subfold), jaké hodnoty dosáhla, aby bylo možné ji případně v konfiguračním souboru navýšit.

Až modul dokončí zpracování složky, informuje uživatele o této skutečnosti názvem modulu a hláškou completed. Pro kontrolu, jak dlouho aplikace běžela se při startu a před ukončením aplikace vypisuje aktuální datum čas. Příklad spuštěné aplikace zobrazuje obrázek 5.1.



Obr. 5.1: Spuštěná aplikace

5.11 Zasílání notifikací

Aplikace po dokončení všech úkonů, anebo v případě incidentu (výskytu chyby) při běhu zašle na předem definované adresy notifikace o průběhu. Každá notifikace by měla obsahovat pouze obsahově důležitá data, aby byla zachována určitá přehlednost. Předmět zprávy obsahuje název úlohy, která byla naposledy spuštěna, a uživatelem nakonfigurovaný text. Uživatelský text není nutné zobrazovat, lze jej vypnout v konfiguraci.

Samotné získávání obsahu notifikace funguje na principu naplňování zásobníku zprávami. Každý modul, který byl vyhodnocen jako důležitý (například přesouvání zpráv), má možnost do zásobníku zpráv vložit svoji zprávu. Mírnou nevýhodou může

být výskyt zdánlivě opakujících se zpráv, pokud se stejná chyba vyvolá několikrát v řadě za sebou. Text se bude lišit pouze v názvech složek a úloh.

5.11.1 Struktura notifikace

Z důvodu zvýšení přehlednosti došlo k rozdělení notifikačních zpráv na skupiny. Na první pohled bude patrná závažnost zprávy. Některé slouží pouze jako informace o správné funkci aplikace a jako statistický přehled. Pokud nastane závažná chyba, bude zpráva informující o této skutečnosti zařazena na patřičné místo (skupina chyby).

Struktura notifikace se skládá z názvu úlohy, času spuštění aplikace, informací o vyhodnocení zpráv, čas ukončení běhu aplikace, informací ohledně dochvilností zpráv a případných chyb.

Informace o vyhodnocení zpráv se skládají z počtu přesunutých emailů pro každou složku, seznam zpracovaných vzorů ve složce a počet nepřečtených zpráv. Nepřečtená a tedy nezpracovaná zpráva znamená pravděpodobně chybu. Nebyl pro ní nalezen žádný vhodný vzor.

Informace o dochvilnosti vypisují případné zpožděné nebo chybějící zprávy. Součástí je očekávaný čas přijmutí a vzor zprávy. V případě vzniku chyby (přiřazeno více zpráv na jeden časový slot) se vypíše pouze informace o vzniku chyby.

Pokud nastane závažná chyba, která znemožní pokračovat ve vyhodnocování zpráv, bude odeslána speciální notifikace pouze s chybou. Musí však být splněn technologický předpoklad pro odeslání notifikace (funkční notifikační modul). Za závažnou chybu se považuje například ztráta spojení s API. Do sekce chyby se vloží text "CHYBA - popis chyby" s popisem chyby, případně i informace pro který vzor chyba nastala.

5.12 Podpůrné servisní moduly

Pro účely testování některých funkcí obsahuje aplikace servisní moduly. V automatickém módu nebudou využity. V běžném provozu nejsou potřebné.

V manuálním módu jsou některé přímo přístupné pod číselnou volbou z menu. Nicméně manuální spouštění modulů způsobí odlišné chování jednotlivých modulů. Oproti automatu je nutné dodržovat pořadí spouštěných modulů. Není tedy doporučeno používat manuální mód bez seznámení se s jeho funkcí. Běžný uživatel by nikdy neměl narazit na potřebu využít manuální mód.

5.12.1 Zpětné nastavení složky

Tento modul velmi usnadňuje opakované testování aplikace. Pro předem definovanou složku provede její zpětné nastavení do původního stavu. Všechny zprávy ve složce (kromě vzoru) označí jako nepřečtené a odstraní všechny přiřazené kategorie (barevné štítky). Pokud vnikne potřeba opakovaně testovat funkci aplikace, lze využít právě tento modul.

5.12.2 Výpis sdílených seznamů

Slouží pro účely hledání chyb či ověření správné funkce logické stránky aplikace. Vypisuje aktuální stav všech sdílených seznamů, které aplikace vytváří. Je tak možné kontrolovat správnou funkci modulů, protože se většinou po určitých událostech seznamy mažou a znovu naplňují jinými daty. Data ve sdílených seznamech jsou relevantní pouze pro právě zpracovávanou složku. Výpis je obsáhlý a do konzole vypisuje komplexní data, tak jak jsou uložena v seznamu.

5.12.3 Výpis zpráv ve složce

Po spuštění vypíše všechny zprávy v předem zvolené složce. Ke každé zprávě vypíše odesílatele, předmět, kategorie (barevné štítky) a zjednodušený obsah těla zprávy bez HTML formátování. Je si tak možné ověřit reálný stav zprávy na serveru v případě, kdy e-mailový klient ukazuje již přesunuté a tudíž neexistující zprávy.

6 Implementace nadstavby aplikace

Grafická nadstavba jádra aplikace má za cíl zjednodušit správu aplikace. Některé moduly vyžadují ke své funkci více konfiguračních souborů a mohou na sebe mít návaznost. Nadstavba všechny tyto souvislosti má implementovány a zajišťuje tak celistvost konfiguračních souborů.

Při implementaci bylo důležité navrhnout ovládací prvky tak, aby rozvržení aplikace bylo pro uživatele srozumitelné. Jednotlivé prvky jsou barevně odlišeny.

Nadstavba je vytvořena formou webové stránky. Využívá framework Flask [49], který společně s podpůrnými moduly zpracovává požadavky od prohlížeče (HTML GET a POST). Jednotlivé stránky pak dle zaměření obsahují prvky JavaScriptu. Samotné zpracování dat probíhá přímo na straně serveru aplikace. Lokální instanci serveru lze spustit na běžném počítači. Nadstavba nevyžaduje instalaci ani další přípravu závislostí v případě využití varianty exe.

6.1 Úvodní stránka

Úvodní stránka se automaticky otevře po spuštění aplikace. Zobrazuje všechny dostupné funkce a nastavení. Je rozdělena do sekcí, aby bylo na první pohled zřejmé, které funkce spolu souvisí.

Barevné schéma bylo navrhnuto s ohledem na přehlednost. Každá barva vyjadřuje nějaký stav nebo volbu. Všechny ovládací prvky dodržují navrhnuté barevné schéma. Nevhodné kombinace barev, jako je žlutá s bílou, nejsou využity. Zhoršují čitelnost textu.

Červená barva znamená problém, anebo destruktivní volbu. Červeně podbarvená tlačítka mažou formuláře a úlohy. V případě informačního panelu znamená červená problém, například nepřihlášení k serveru. Zároveň se červeně podbarvují vypnuté a zakázané moduly.

Světle zelená znamená provozní stav a splnění předpokladů pro funkci navazujících modulů. Příkladem je přihlášení k serveru. Po úspěšném přihlášení se ve stavovém panelu tato informace podbarví světle zelenou.

Tyrkysová barva informuje o důležitých nastaveních, která je nutné na první pohled zachytit. Na stavovém panelu je takto podbarvena informace o nastavené složce plánovače úloh.

Šedá barva značí přístup k funkcím, neoznačuje žádný mimořádný stav. Šedě podbarvené tlačítko značí, že je funkce v provozu a zapnutá.

Tmavě zelená je určena pro prvky interagující s uživatelem a také jako ukazatel aktivní stránky v navigační liště. Barvu využívají všechny formuláře a tlačítka, která ukládají data nebo načítají data.

Hnědá barva odkazuje na spuštění externí funkce nebo programu, který může být z aplikace spouštěn.

6.2 Přihlášení k podnikové aplikaci

Stránka přihlášení k podnikové aplikaci vyzve uživatele, aby zadal přihlašovací údaje (ID tenanta, ID aplikace, e-mail). Údaje se uloží a aplikace si na pozadí vyžádá token. Uživatel musí potvrdit přístup. Zadá kód zařízení na nově otevřeném okně a odsouhlasí přístup. V případě zadání špatných údajů se pouze zobrazí upozornění a proces přihlášení dále nepokračuje.

Pro získání tokenu se využívá knihovna MSAL, která zajišťuje komunikaci s cloudovým serverem. Získané tokeny ukládá do souboru v interním formátu. Při spuštění aplikace se automaticky ověří existence a platnost tokenu. Pokud existuje platný token, změní se podbarvení stavového boxu na úvodní stránce a povolí se moduly, které vyžadují ke své funkci token.

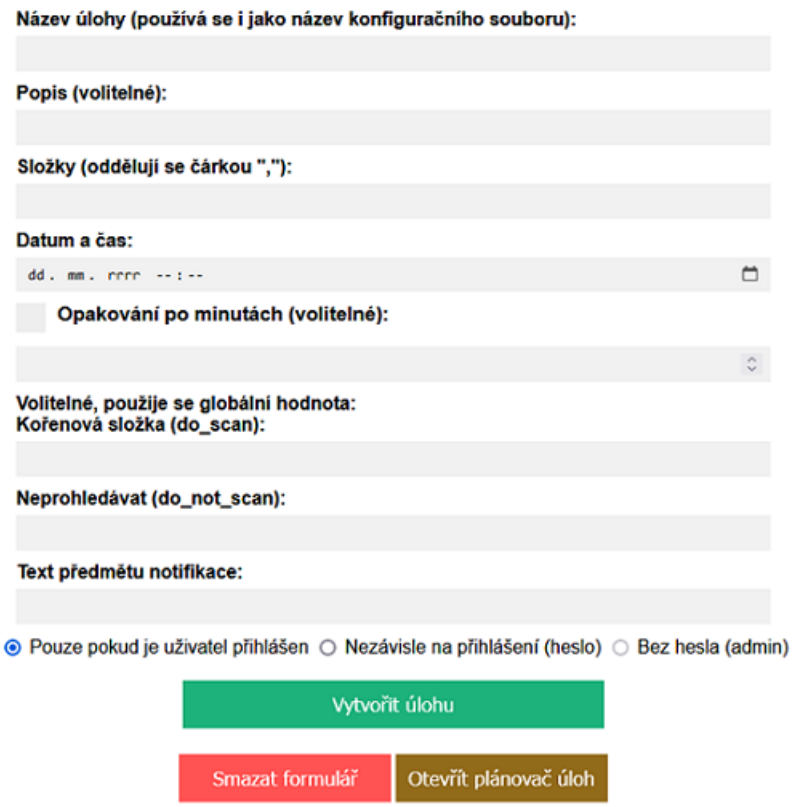
6.3 Správa úloh

Správa úloh je souhrnné označení pro veškerou manipulaci s úlohami, které aplikace nabízí. Obsahuje pouze nezbytné ovládací prvky, aby uživatel nebyl zahlcen množstvím nepotřených nastavení.

6.3.1 Tvorba úloh

Stránka vytvoření úlohy zobrazí uživateli formulář, kam vloží požadované parametry. Jedná se o název úlohy, popis, monitorované složky, datum a čas, opakování po minutách, volitelně složky neprohledávat a kořenová složka. Volba možnosti zabezpečení nastavuje spuštění aplikace s ohledem na provoz i bez přihlášeného uživatele. Je nutné případně zadat uživatelské jméno a heslo počítače. Akce se potvrdí tlačítkem uložit. Aplikace uživatele informuje o stavu, jestli úloha vytvořena byla, anebo nebyla a proč. Formulář kontroluje, jestli úloha se zadaným názvem již neexistuje. Pokud existuje, nedovolí úlohu vytvořit a zabrání tak přepsání konfiguračních souborů existující úlohy. Vyplněné hodnoty se nesmažou a uživatel může název upravit.

Vnitřní struktura modulu spolupracuje s integrovanou funkcí systému Windows, plánovačem úloh. Ten řídí automatické periodické spouštění jádra aplikace. Data, která uživatel zadá do formuláře, se po vyhodnocení uloží do patřičných konfiguračních souborů. Data, která se předávají plánovači úloh, je nutné převést do formátu XML. Aplikace si načte vzorový XML soubor, upraví všechna potřebná nastavení a hodnoty. Následně se přes příkazový řádek provede importování úlohy.



Obr. 6.1: Formulář úloh

Formulář 6.1 obsahuje ještě tlačítko smazat formulář, které ihned smaže všechny vyplněné hodnoty. Druhé tlačítko otevírá plánovač úloh. Uživatel si tak může nastavit i pokročilejší parametry, které ale ve většině případů nejsou relevantní.

6.3.2 Editace úloh

Stránka editace úloh využívá komponenty ze stránky vytvoření úlohy. Jediný rozdíl spočívá ve výběru názvu úlohy. Názvy úloh se při otevření stránky načtou a uživatel ze seznamu vybere editovanou úlohu. Formulář je jinak shodný s obrázkem 6.1.

Data se musí načítat jak z plánovače úloh pomocí formátu XML, tak z konfiguračního souboru. Uložení změněných dat pak probíhá úplně stejně, jako v případě vytváření úlohy. Pouze je nutné plánovači úloh příkazem force sdělit, že má přepsat starou úlohu.

Aby se předešlo smazání pokročilejších nastavení, která si uživatel mohl nastavit mimo aplikaci, ukládá se XML z plánovače do mezipaměti. V případě editace přes aplikaci se pak upraví pouze vyžádané parametry. Zbytek se ponechá v původním stavu a znovu naimportuje.

6.3.3 Seznam úloh

Stránka seznam úloh zobrazuje přehlednou tabulku s vypsanými úlohami. Obsahuje pouze název úlohy, příští spuštění, stav (úloha povolena / úloha zakázána) a tlačítko zobrazit podrobnosti.

Tlačítko zobrazit podrobnosti vypíše aktuální údaje k úloze. Editace dat na této stránce není možná. Formulář s podrobnostmi je shodný s obrázkem 6.1, pouze tlačítka jsou pozměněna.

Na této stránce se nachází i volba smazat úlohu. Po zvolení úloh ke smazání se automaticky odstraní konfigurační soubory a smaže se z plánovače úloh.

6.4 Dochvilnost zpráv

Konfigurace dochvilnosti zpráv se skládá ze tří stránek. Společně poskytují potřebné údaje pro jádro aplikace. Pro lepší přehlednost se načítají data pouze pro vybranou úlohu a složku úlohy (pokud nemají data globální význam).

Vyhodnocování dochvilnosti lze vypnout. První volbou je vypnout vyhodnocování pro celou kategorii. Druhou volbou je vypnutí na úrovni složky úlohy. Ve výchozím stavu není vyhodnocování dochvilnosti povoleno.

6.4.1 Definice kategorií

Definice kategorií převádí slovní název na číselnou hodnotu. Uživatel si ve schránce vytvoří a slovně pojmenuje kategorii. Aplikace nemůže převést název "10-minutly" na 10 minut. Proto se uživateli zobrazí tabulka s výpisem existujících kategorií. Jako první definuje rozestup mezi zprávami a tím definuje očekávaný čas mezi příjmem další zprávy. Další buňka obsahuje nastavení výchozího času maximální odchylky. Ta se využije pokud individuální vzor zprávy nemá nastavenu odchylku vlastní.

Název kategorie	Rozestup mezi zprávami (m)	Maximální odchylka (m)	VYP/ZAP		Smazat
Modrá kategorie	0	0			
hourlydd	0	0			

Obr. 6.2: Definice kategorií

6.4.2 Individuální nastavení vzorů

Vzory, které se automaticky načítají při běhu jádra aplikace, mají svoje vlastní nastavení dochvilnosti zpráv. Každý načtený vzor má nastavení pro vlastní odchylku zprávy (jak moc se může zpozdit) a nastavení pro příjem první zprávy od půlnoci dne. Pro vzory s očekávaným příjmem delším než jeden den je zvolen odlišný postup vyhodnocení dochvilnosti. Do kolonky se tak vyplní očekávaný čas v běžném formátu. Místo 19:43 se zapíše 1943.

Smazání vzoru a jeho nastavení se provede přepnutím přepínače. Nedochází k automatickému mazání definicí.

Odesílatel	Předmět	Kategorie	První od 00:00 (m)	Odchylka (m)	Smazat
J @I cr	????????????	['*weekly']	0	1	
J @(r	SYNOLOGY success	['dailydd', 'hourlydd']	0	0	

Obr. 6.3: Nastavení vzorů

6.5 Globální nastavení

Aplikace spravuje globální hodnoty, které se používají jako výchozí. Pokud jsou stejné hodnoty vyplněny v individuální úloze, dostanou přednost. Po uložení hodnot se znovu načte vizuální ukazatel na úvodní stránce, aby reflektoval změny. Proto není nutné po uložení změn restartovat aplikaci.

První hodnota "maximální počet průchodů" poskytuje ochranu před zacyklením. Při zpracovávání zpráv v jádru aplikace může nastat překročení limitů požadavků na API. Pokud dojde k překročení limitu, aplikace se ukončí. Defaultně je nastavena vyzkoušená optimální hodnota 500, která dostačuje pro obvyklý počet průchodů.

Následují hodnoty "Odkládací složka" a "Název vzoru". Slouží pro nastavení výchozí odkládací složky, kam se budou přesouvat zprávy a pro definování vzorové kategorie zprávy.

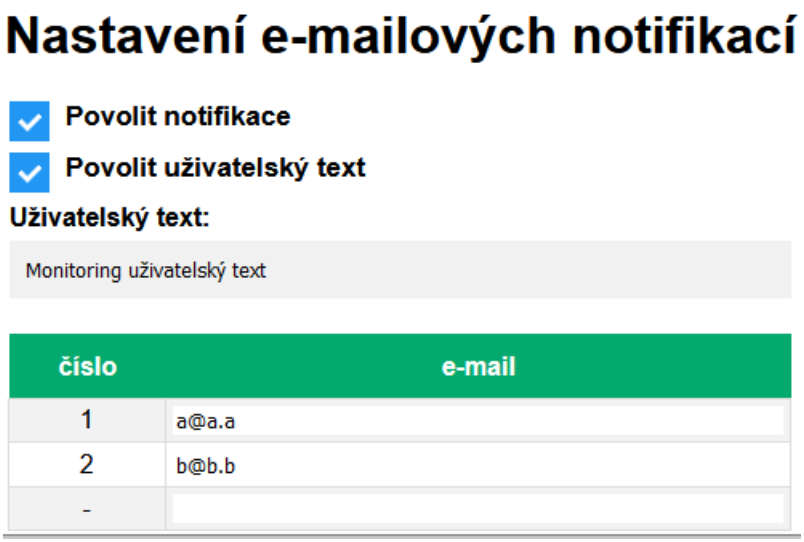
Hodnota "Složka plánovače úloh" nastavuje využitou složku v plánovači úloh. Budou se do ní ukládat vytvářené úlohy. Hodnota musí být nastavena, jinak není zaručena správná funkčnost aplikace.

Poslední hodnoty "Kořenová složka" a "Neprohledávat" nastavují výchozí hodnotu pro parametry doScan a DoNotScan. Každá individuální úloha může tyto parametry ignorovat, pokud má nastaveny svoje vlastní.

6.6 Nastavení notifikací

Notifikace, které se odesílají po vyhodnocení zpráv jádrem aplikace, obsahují v předmětu název úlohy. Pro lepší přehlednost se do předmětu může přidat i uživatelský text. Globální hodnota se využije pouze v případě, kdy individuální úloha nemá nastaven vlastní uživatelský text.

Počet adresátů je téměř neomezený. Stránka vykresluje přehlednou tabulku 6.4, která zobrazuje všechny uložené adresáty a jeden prázdný řádek. Při ukládání adresátů se vždy přidá další prázdný řádek. Editovat lze všechny uložené e-maily zároveň.



Obr. 6.4: Nastavení notifikací

6.7 Historie a export zpráv

Nástroj historie a export zpráv se využívá při analýze dochvilnosti zpráv. Obsluha si načte a exportuje okruh zpráv, které ručně nebo externím programem prohledá.

Prvotní načtení informací pro vybranou úlohu zajišťuje jádro aplikace. Nadstavbě dodá seznam vzorů pro každou složku společně s unikátním ID složky. Aplikace nevyužívá žádnou mezipaměť a data vždy načítá znovu. Nevýhodou je časová prodleva, která může přesáhnout třicet sekund. Využití mezipaměti by znamenalo možné nesrovnalosti v seznamu ID složek v případě změn ve schránce. Proto bylo zvoleno řešení bez mezipaměti.

Výběr složky a následně vzoru už zpracovává přímo nadstavba. Volbu složky a vzoru je možné průběžně měnit. Načítání samotných zpráv probíhá přímo přes API a tak je čas načtení v řádu sekund. Pro účely analýzy konkrétního úseku zpráv

je přidáno nastavení časového rozmezí. Načítají se pouze zprávy, které vyhovují nastavenému rozmezí.

Export zpráv kopíruje zobrazenou tabulku a ukládá zprávy do textového souboru. Vytvořený soubor si uživatel může uložit. Každý řádek obsahuje pouze jednu zprávu včetně času přijetí a předmětu.

Datum a čas	Předmět	Odesílatel
2023-02-04 20:33:49	??????????	J @0 .com
2023-02-04 20:33:49	??????????	J @0 .com
2023-02-04 20:33:49	??????????	J @0 .com

Obr. 6.5: Historie zpráv

Závěr

Práce se zabývala návrhem a vývojem aplikace (a její nadstavby) pro automatizované dohlížení nad IT infrastrukturami. Aplikace je úzce zaměřena na prostředí Microsoft 365 [22], ke kterému se připojuje.

Aplikace prochází zadané složky, načítá vzorové zprávy a následně vyhodnocuje nově přijaté zprávy. Administrátora informuje o stavu pomocí notifikací. Notifikace obsahují informace o chybách, ale i o úspěšném vyhodnocení zpráv, jejich počtu a další nezařazené informace. Na základě přijatých notifikací může administrátor podniknout nápravné kroky směřující k vyřešení nežádoucí situace.

Nadstavba, která se používá pro správu a konfiguraci aplikace, byla vytvořena formou webové stránky. Obsahuje nástroj pro zobrazení a export historie zpráv. Veškerá konfigurace se provádí právě přes nadstavbu.

Výsledkem práce je funkční aplikace včetně grafické nadstavby. Zprovoznění vyžaduje přípravu prostředí na straně serveru, kde je nutné vytvořit podnikovou aplikaci. Veškeré kroky popisující zprovoznění a konfiguraci aplikace jsou k dispozici v uživatelském manuálu.

Při vývoji se nepodařilo použít klasický a relativně jednoduchý přístup protokolem IMAP4. Prostředí Microsoft 365 jej totiž nemá implementováno v rozsahu, který by umožnil splnit zadání. Výsledkem tedy je, že aplikace spolupracuje pouze s konkrétním prostředím jednoho výrobce.

Literatura

[1] *ZABBIX*, Zabbix.com [online]. [cit. 2022-12-11]. Dostupné z: <https://www.zabbix.com/>

[2] IT-SLOVNIK.CZ TEAM. *Co je to Zabbix?* [online]. [cit. 2022-12-12]. Dostupné z: <https://it-slovnik.cz/pojem/zabbix>

[3] *ZABBIX NOTIFICATIONS* [online]. [cit. 2022-12-11]. Dostupné z: <https://www.zabbix.com/notification>

[4] *ManageEngine*, ManageEngine.com [online]. [cit. 2022-12-11]. Dostupné z: <https://www.manageengine.com/>

[5] *SysAid*, SysAid.com [online]. [cit. 2022-12-11]. Dostupné z: <https://www.sysaid.com/>

[6] *Atera*, Atera.com [online]. [cit. 2022-12-11]. Dostupné z: <https://www.atera.com/>

[7] *Grafana*, Grafana.com [online]. [cit. 2022-12-11]. Dostupné z: <https://grafana.com/>

[8] OKTA, INC. *What is Authentication?* [online]. [cit. 2022-12-12]. Dostupné z: <https://auth0.com/intro-to-iam/what-is-authentication>

[9] HENDRICKSON, Joanne, Chris DAVIS a kol. *Deprecation of Basic authentication in Exchange Online* [online]. [cit. 2022-12-12]. Dostupné z: <https://learn.microsoft.com/en-us/exchange/clients-and-mobile-in-exchange-online/deprecation-of-basic-authentication-exchange-online>

[10] SMARTBEAR SOFTWARE. *Basic Authentication* [online]. [cit. 2022-12-12]. Dostupné z: <https://swagger.io/docs/specification/authentication/basic-authentication/>

[11] *Wireshark.org* [online]. [cit. 2022-12-12]. Dostupné z: <https://www.wireshark.org/>

[12] THE EXCHANGE TEAM. *Use Authentication Policies to Fight Password Spray Attacks* [online]. 2022 [cit. 2022-12-12]. Dostupné z: <https://techcommunity.microsoft.com/t5/exchange-team-blog/use-authentication-policies-to-fight-password-spray-attacks/ba-p/3643487>

[13] VICE, Kelley, Ashok LOBO a kol. *Hybrid modern authentication overview* [online]. [cit. 2022-12-12]. Dostupné z: <https://learn.microsoft.com/en-us/microsoft-365/enterprise/hybrid-modern-auth-overview?view=o365-worldwide>

[14] OKTA, INC. *What is OAuth 2.0?* [online]. [cit. 2022-12-12]. Dostupné z: <https://auth0.com/intro-to-iam/what-is-oauth-2>

[15] HALL, Justin, Frank BOYLAN a kol. *What authentication and verification methods are available in Azure Active Directory?* [online]. 09/2022 [cit. 2022-12-12]. Dostupné z: <https://learn.microsoft.com/en-us/azure/active-directory/authentication/concept-authentication-methods>

[16] SINHA, Gargi, Janice RICKETTS a kol. *Plan a passwordless authentication deployment in Azure Active Directory* [online]. 11/2022 [cit. 2022-12-12]. Dostupné z: <https://learn.microsoft.com/en-us/azure/active-directory/authentication/howto-authentication-passwordless-deployment?culture=en-us&country=US>

[17] ABRAMS, Lawrence. MFA Fatigue: Hackers- new favorite tactic in high-profile breaches. *Bleepingcomputer* [online]. 09/2022 [cit. 2022-12-12]. Dostupné z: <https://www.bleepingcomputer.com/news/security/mfa-fatigue-hackers-new-favorite-tactic-in-high-profile-breaches/>

[18] GAŠPARÍK, Petr. *Vícefaktorová autentizace* [online]. SecurityWorld, 11/2014 [cit. 2022-12-12]. Dostupné z: <https://ami.cz/novinka/vicefaktorova-autentizace/>

[19] ADAMU, haroun. *How to protect yourself from a SIM-swap attack* [online]. 11/2022 [cit. 2022-12-12]. Dostupné z: <https://www.androidpolice.com/how-to-protect-yourself-from-a-sim-swap-attack/>

[20] SOUMI, a kol. *When are ADAL and Azure AD Graph reaching end of life?* [online]. 3/2022 [cit. 2022-12-12]. Dostupné z: <https://learn.microsoft.com/en-us/answers/questions/768833/when-is-adal-and-azure-ad-graph-reaching-end-of-li.html>

[21] MIROSOFT CORPORATION. *Microsoft Authentication Library (MSAL) for Python* [online]. [cit. 2022-12-12]. Dostupné z: <https://github.com/AzureAD/microsoft-authentication-library-for-python>

[22] MICROSOFT CORPORATION. *Microsoft 365* [online]. [cit. 2022-12-12]. Dostupné z: <https://www.office.com/>

[23] WERNER, Chris, Marsh MACY a kol . *Overview of the Microsoft Authentication Library (MSAL)* [online]. 10/2022n. l. [cit. 2022-12-12]. Dostupné z: <https://learn.microsoft.com/en-us/azure/active-directory/develop/msal-overview>

[24] OMONDI, Jackline, Alex BUCK a kol. *What is application management in Azure Active Directory?* [online]. 12/2022 [cit. 2022-12-12]. Dostupné z: <https://learn.microsoft.com/en-us/azure/active-directory/manage-apps/what-is-application-management>

[25] WERNER, Chris, Marsh MACY a kol. *Public client and confidential client applications* [online]. 10/2022 [cit. 2022-12-12]. Dostupné z: <https://learn.microsoft.com/en-us/azure/active-directory/develop/msal-client-applications>

[26] WERNER, Chris, Marsh MACE a kol. *Authentication flow support in MSAL* [online]. 10/2022 [cit. 2022-12-12]. Dostupné z: <https://learn.microsoft.com/en-us/azure/active-directory/develop/msal-authentication-flows>

[27] LUDWIG, Nick, Owen RICHARDS a kol. *Microsoft identity platform and the OAuth 2.0 device authorization grant flow* [online]. 11/2022 [cit. 2022-12-12]. Dostupné z: <https://learn.microsoft.com/en-us/azure/active-directory/develop/v2-oauth2-device-code>

[28] LUDWIG, Nick, Dickson MWENDIA a kol. *Microsoft identity platform and OAuth 2.0 Resource Owner Password Credentials* [online]. 8/2022 [cit. 2022-12-12]. Dostupné z: <https://learn.microsoft.com/en-us/azure/active-directory/develop/v2-oauth-ropc>

[29] MURRAY, David, Rudy SCHOCKAERT a kol . *Microsoft identity platform and OAuth 2.0 authorization code flow* [online]. 9/2022n. l. [cit. 2022-12-12]. Dostupné z: <https://learn.microsoft.com/en-us/azure/active-directory/develop/v2-oauth2-auth-code-flow>

[30] MYERS, J., Carnegie MELLON a M. ROSE. *Post Office Protocol - Version 3* [online]. Květen 1996, [cit. 2022-12-07], RFC1939, DOI 10.17487/RFC1939, Dostupné z URL: <https://www.rfc-editor.org/rfc/rfc1939>

[31] CRISPIN, M. *INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1* [online]. Březen 2003, [cit. 2022-12-07], RFC 3501, DOI 10.17487/rfc3501, Dostupné z URL: <https://www.rfc-editor.org/rfc/rfc3501>

[32] *OpenSSL* [online]. [cit. 2022-12-11]. Dostupné z: <https://www.openssl.org/>

[33] *O365* [online]. [cit. 2022-12-11]. Dostupné z: <https://outlook.office.com/mail/>

[34] PRAKASH, Siva, Mike WHITTY Ahmed HASHAD a kol. *Authenticate with Oauth* [online]. [cit. 2022-12-11]. Dostupné z: <https://learn.microsoft.com/en-us/exchange/client-developer/legacy-protocols/how-to-authenticate-an-imap-pop-smtp-application-by-using-oauth>

[35] HENDRICKSON, Avatar Joanne, Chris DAVIS Dabid COULTER a kol. *MAPI over HTTP* [online]. 03/2022 [cit. 2022-12-11]. Dostupné z: <https://learn.microsoft.com/en-us/Exchange/clients/mapi-over-http/mapi-over-http?redirectedfrom=MSDN&view=exchserver-2019>

[36] HENDRICKSON, Joanne, Yulia USENKO Matt PENNA a kol. *Exchange ActiveSync in Exchange Server* [online]. 2/2022 [cit. 2022-12-11]. Dostupné z: <https://learn.microsoft.com/en-us/exchange/clients/exchange-activesync/exchange-activesync?view=exchserver-2019>

[37] OLASON, Lily, Linda CAPUTO a Daniela MONTERO a kol. *Overview of Microsoft Graph* [online]. 6/2022 [cit. 2022-12-11]. Dostupné z: <https://learn.microsoft.com/en-us/graph/overview>

[38] *Graph Explorer* [online]. Microsoft Corporation [cit. 2022-12-11]. Dostupné z: <https://developer.microsoft.com/en-us/graph/graph-explorer>

[39] SIGNORET, Philippe, Jackline OMONDI a kol . *Manage consent to applications and evaluate consent requests* [online]. 7/2022 [cit. 2022-12-12]. Dostupné z: <https://learn.microsoft.com/en-us/azure/active-directory/manage-apps/manage-consent-requests>

[40] PYTHON SOFTWARE FOUNDATION. *Python* [online]. [cit. 2022-12-12]. Dostupné z: <https://www.python.org/>

[41] LANGA, Łukasz, ed. *What-s New In Python 3.9* [online]. PYTHON SOFTWARE FOUNDATION. 10.2.2023 [cit. 2023-05-23]. Dostupné z: <https://docs.python.org/3.9/whatsnew/3.9.html>

[42] MICROSOFT CORPORATION. *Windows* [online]. [cit. 2022-12-12]. Dostupné z: <https://www.microsoft.com/en-us/windows>

[43] *LINUX* [online]. [cit. 2022-12-12]. Dostupné z: <https://www.linux.org/>

[44] OMONDI, Jackline, Marsh MACY a kol . *Quickstart: Add an enterprise application* [online]. 6/2022n. l. [cit. 2022-12-12]. Dostupné z: <https://learn.microsoft.com/en-us/azure/active-directory/manage-apps/add-application-portal>

[45] *CONFIGPARSER* [online]. [cit. 2022-12-11]. Dostupné z: <https://docs.python.org/3/library/configparser.html>

[46] LUO, Ray a Abhi DNYA. *Device_flow_sample.py* [online]. [cit. 2022-12-12]. Dostupné z: <https://github.com/AzureAD/microsoft-authentication-library-for-python/blob/dev/sample/device_flow_sample.py>

[47] DAS, Abheek, Vincent BIRET Faith OMBONGI a kol. *List mailFolders* [online]. [cit. 2022-12-11]. Dostupné z: <https://learn.microsoft.com/en-us/graph/api/user-list-mailfolders?view=graph-rest-1.0>

[48] DAS, Abheek, Faith OMBONGI a Jason JOHNSTON a kol. *Message move* [online]. [cit. 2022-12-11]. Dostupné z: <https://learn.microsoft.com/en-us/graph/api/message-move?view=graph-rest-1.0&tabs=http>

[49] *Flask* [online]. [cit. 2023-05-22]. Dostupné z: <https://flask.palletsprojects.com>

Seznam symbolů a zkratek

AD	Active Directory
API	Programovací rozhraní aplikace – Application Programming Interface
FIDO2	Fast Identity Online
HTML	Značkovací jazyk – HyperText Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ID	Identita – Identifier
IMAP	Poštovní protokol – Internet Message Access Protocol
IT	Informační technologie – Information Technology
MAPI	Messaging Application Programming Interface
MFA	Více faktorová autentizace – Multi Factor Authentication
MSAL	Autentizační knihovna –Microsoft Authentication Library
PIN	Personální identifikační číslo – Personal Identification Number
POP	Poštovní protokol – Post Office Protocol
REST	Převod reprezentativního stavu – Representational State Transfer
RS-232	Doporučený Standard 232 – Recommended Standard 232
SMTP	Jednoduchý e-mailový protokol – Simple Mail Transfer Protocol
SNMP	Protokol pro správu sítě – Simple Network Management Protocol
SQL	Strukturovaný dotazovací jazyk – Structured Query Language
SSH	Zabezpečená vzdálená konzole – Secure Shell
UPS	Záložní zdroj – Uninterruptible Power Supply
URI	Identifikátor zdroje – Uniform Resource Identifier
URL	Adresa – Uniform Resource Locator
USB	Univerzální sériová sběrnice – Universal Serial Bus

VPS Virtuální privátní server – Virtual Private Server

XML Extensible Markup Language

A Obsah elektronické přílohy

Elektronická příloha obsahuje manuály a aplikaci s nadstavbou. Dostupné jsou ve variantách exe a py. Provoz testován na Python verze 3.7 a 3.9 (nutné doinstalovat závislosti requirements.txt), případně využít variantu exe.

- / ..... kořenový adresář přiloženého archivu
  - návod_tvorba_AZUREAD_aplikace.pdf
  - uživatelský manuál.pdf
  - LICENSE.txt ..... licence použité knihovny MSAL
  - společné soubory ..... společné soubory variant exe a py
    - MNinternal.cfg ..... globální konfigurační soubor
    - MNuser.cfg ..... výchozí uživatelský konfigurační soubor
    - timeouts.txt
    - vzor_denni.xml
    - Xtemp.txt
    - config ..... složka pro ukládání konfiguračních souborů
    - watchdog ..... složka pro ukládání nastavení dochvilnosti
  - varianta exe
    - mailnotifier.exe ..... vyhodnocovací program
    - app.exe ..... nadstavba
  - varianta py
    - requirements.txt
    - mailnotifier v1.5 ..... vyhodnocovací program
      - mailnotifier.py
      - CORE.py
      - GETURL.py
      - MSAL.py
      - SETUP.py
      - notifications.py
    - app ..... nadstavba
      - app.py
      - HISTORY.py
      - MNconnector.py
      - SETGET.py
      - TASKS.py

B Manuál

Příloha obsahuje uživatelský manuál k aplikaci Mailnotifier v1.5 a popisuje kroky nutné ke zprovoznění.

Uživatelský manuál k aplikaci Mailnotifier v1.5

Manuál popisuje konfiguraci a význam kolonek jednotlivých formulářů. Na některých stránkách má nadstavba aplikace rozklikávací nápovědu s vysvětlením konfigurace.

1. **Předpoklady pro správnou funkci aplikace**
   - Ve schránce se nevyskytují složky, podsložky a kategorie se stejným názvem
   - Je připravena podniková aplikace v AZURE AD
   - Cílové složky pro přesun zpráv jsou v odkládací složce již vytvořeny (SYN = > _SYN)
   - Je vytvořená složka v plánovači úloh
   - V předmětu zpráv se nenachází znaky: & = + , ;

2. **Příprava souborů a složek**
   - Do složky, ve které chcete provozovat aplikaci, zkopírujte obsah složky „společné soubory“
   - Ve složce s aplikací se musí nacházet prázdné složky „config“ a „watchdog“
   - Podle zvolené varianty (py nebo exe) zkopírujte do složky s aplikací jejich obsah
   - Variantu exe lze nyní běžně spustit
   - Pro varianta py nainstalujte Python (vyzkoušené verze 3.7 a 3.9)
   - Přes příkazový řádek nainstalujte potřebné knihovny ze souboru requirements.txt příkazem „pip install -r requirements.txt“.
   - Nyní lze spustit samotnou aplikaci příkazem app.py (ve složce, kde je aplikace umístěna)

3. **Přihlášení k podnikové aplikaci**
   - V navigační liště klikněte na „Přihlášení k AZURE AD“
   - Vyplňte všechny údaje podnikové aplikace a klikněte na získat kód
   - Klikněte na „Zkopírovat kód a přesměrovat“
     - Pokud se kód nezkopíruje automaticky, zkopírujte ho ručně
   - Na nově otevřeném okně se přihlaste dle zobrazených instrukcí a povolte přístup
   - Aplikace by měla být úspěšně přihlášena
   - Přihlášení zkontrolujte na úvodní stránce (zmizí červeně zabarvené boxy)

4. **Nastavení globálních hodnot**
   - Sekce nastavení \ Globální nastavení
   - max_subfold – změňte pouze v případě příliš velké schránky (aplikace upozorní na problém)
   - Odkládací složka – název složky, ve které budou cílové složky pro přesun (SYN => _SYN)
     - Příklad cesty: _TEMP _SYN => nastavte _TEMP
   - Název vzoru – název vzorové kategorie (zprávy takto označené jsou považovány za vzor)
   - Složka plánovače úloh – vložte název vytvořené složky v plánovači úloh
   - Kořenová složka – vložte název globální kořenové složky (většinou Doručená pošta)
     - Příklad cesty: Doručené/monitoring/UPS => nastavte „Doručené“
   - Neprohledávat – složky (a její podsložky) s vloženým názvem se nebudou monitorovat
   - Změny uložte

5. **Nastavení notifikací (adresáti a globální předmět notifikace)**
   - Sekce nastavení\Notifikace
   - Povolit notifikace – zapne nebo vypne zasílání notifikací
   - Povolit uživatelský text – zapne nebo vypne uživatelský text do předmětu notifikace
   - Uživatelský text – globální uživatelský text, každá individuální úloha může mít vlastní
   - Seznam e-mailů – postupně vkládejte a ukládejte další e-maily, vždy se objeví nový řádek

6. Definice kategorií (nastavení rozestupu zpráv)

Nastavení kategorií (barevných štítků), které jsou na serveru již vytvořeny.

- Sekce nastavení\Definice kategorií
- Rozestup mezi zprávami – čas v minutách mezi očekávaným přijetí další zprávy
  - Zpráva přichází každých 10 minut => nastavte 10
- Maximální odchylka – globální hodnota maximálního zpoždění zprávy pro danou kategorii
- VYP/ZAP – vypnutí nebo zapnutí vyhodnocování dochvilnosti u dané kategorie
- Smazat – přepnutím doprava a uložením změn dojde k výmazu záznamu

7. Vytváření úloh

- Sekce Úlohy \ Vytvoření úlohy
- Název – vyplňte unikátní název (nesmí se opakovat)
- Popis – popis úlohy v plánovači úloh
- Složky – seznam složek, který chcete vyhodnocovat (záznamy oddělujte čárkou „a,b,c“)
- Datum a čas – datum a čas spuštění úlohy (spustí se každý den v tento čas)
- Opakování po minutách – spouštění úlohy opakovaně po minutách
- Kořenová složka (volitelné) – přepíše globální hodnotu
  - Příklad cesty: Doručené/monitoring/UPS => nastavte „Doručené“
- Neprohledávat (volitelné) - složky s vloženým názvem se nebudou monitorovat
- Text předmětu notifikace – individuální předmět pro vytvářenou úlohu
- Volba spouštění:
  - Pouze pokud je uživatel přihlášen – úloha se spustí pouze za této podmínky
  - Nezávisle na přihlášení – je nutné vyplnit jméno a heslo účtu počítače
  - Bez hesla – nyní není podporováno (pouze zobrazení)

8. Editace úloh

- Sekce Úlohy \ Editace úloh
- Vyberte název úlohy a načtěte ji
- Popis formuláře je shodný s vytvářením úloh

9. Seznam úloh (mazání úloh)

Seznam úloh vypisuje všechny vytvořené existující úlohy. Kliknutím na zobrazit podrobnosti se vypíšou podrobnosti o úloze, včetně volby **smazat úlohu.**

- Sekce Úlohy \ Seznam a stav úloh
- Zobrazená tabulka vypisuje seznam a stav úloh
- Tlačítkem zobrazit podrobnosti se vypíšou podrobnosti o úloze
- Tlačítkem smazat v podrobnostech lze úlohu smazat

10. **Nastavení dochvilnosti zpráv**
    - Sekce Dochvilnost zpráv \ Nastavení složek dochvilnosti
        - Načtěte úlohu
        - Do seznamu povolených složek napište seznam složek, kde chcete dochvilnost vyhodnotit
        - Složky oddělujte čárkou (musí jít o složky dané úlohy) a změny uložte
        - Spusťte úlohu (plánovačem nebo ručně mailnotifier.exe úloha.cfg)
            - Ve složce na e-mailovém serveru se musí nacházet 2 a více zpráv u daného vzoru
            - Mailnotifier načte vzory (načítá je průběžně s každým spuštěním)

    - Přejděte do sekce Dochvilnost zpráv \ nastavení dochvilnosti vzorů
        - Vyberte a načtěte úlohu
        - Vyberte a načtěte složku úlohy
        - Nastavte jednotlivé vzory
            - První od 00:00
                - Rozestup zpráv je méně než den – nastavte příchod první zprávy od 00:00 v minutách. Zpráva přijde v 01:10 => nastavte 70.
                - Rozestup zpráv je více než den – nastavte čas přijmu zprávy, 19:43 => 1943
            - Odchylka – maximální individuální odchylka zpoždění zprávy (timeout v minutách)

        - Volba smazat – smaže definici vzoru
        - Změny uložte

11. **Historie přijatých zpráv**
    - Sekce Nástroje \ Historie a export zpráv
    - Vyberte název úlohy
    - Vyplňte časové rozmezí načítaných zpráv
    - Načtěte úlohu
        - Načítání trvá dle okolností přes 30 sekund!
        - Pokud se načítání nezdaří, je vhodné aplikaci restartovat
    - Vyberte vzor ze seznamu a načtěte zprávy
    - Zobrazí se tabulka s výpisem zpráv
    - Kliknutím na kolonku datum a čas se zprávy seřadí
    - Zprávy lze exportovat a uložit do textového souboru tlačítkem „Exportovat zprávy“