VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

SPRÁVA FILTROVÁNÍ E-MAILŮ V ROUNDCUBE

EMAIL FILTERING ADMINISTRATION IN ROUNDCUBE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **TOMÁŠ ŠPANĚL**
AUTHOR

VEDOUCÍ PRÁCE **Ing. JAROSLAV DYTRYCH, Ph.D.**
SUPERVISOR

BRNO 2020

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačové grafiky a multimédií (UPGM) Akademický rok 2019/2020

Zadání bakalářské práce

Student: **Španěl Tomáš**
Program: Informační technologie
Název: **Správa filtrování e-mailů v Roundcube**
Email Filtering Administration in Roundcube
Kategorie: Web

Zadání:

1. Prostudujte prostředky pro správu filtrování e-mailů v systému Horde/IMP verze 4 a v systému Roundcube.
2. Seznamte se s jazykem popisu filtrování e-mailů v programu procmail.
3. Navrhněte vzhled a prototypy stránek konfiguračních formulářů pro správu a pravidla pro procmail pro realizaci navržené funkcionality.
4. Navrhněte a implementujte doplňkový modul pro systém Roundcube pro správu popisu filtrování e-mailů programem procmail, který bude umožňovat třídit příchozí e-maily podle kritérii do složek, nastavovat přesměrování a oznamování nepřítomnosti. Uživatel rovněž musí mít možnost vložit vlastní pravidla pro procmail přímo do generovaného souboru.
5. Porovnejte dosažený výsledek s existujícími možnostmi správy filtrování pomocí jazyka Sieve.

Literatura:

- Dle doporučení vedoucího.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Dytrych Jaroslav, Ing., Ph.D.**
Vedoucí ústavu: Černocký Jan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2019
Datum odevzdání: 28. května 2020
Datum schválení: 1. listopadu 2019

Abstrakt

Cílem této práce je vytvořit zásuvný modul pro webmail Roundcube, který umožňuje správu filtrování příchozí pošty a automatické oznamování nepřítomnosti, přičemž pro samotné filtrování je využit nástroj Procmail. V práci je dále prozkoumán způsob generování těchto filtrů/oznamování nepřítomnosti a jejich ucelené prezentace uživateli v rámci Roundcube. Výsledkem práce je funkční zásuvný modul a popis zajímavých pasáží z jeho implementace.

Abstract

The aim of this work is to create a plug-in for the Roundcube webmail, which allows the management of filters for incoming mail and automatic absence notification, where the Procmail tool is used for the filtering itself. The work also examines how to generate these filters and report the absence and their comprehensive presentation to the user within the Roundcube. The result of the work is a functional plug-in module and a description of interesting passages from its implementation.

Klíčová slova

Procmail, Roundcube, poštovní filtr, zásuvný modul, PHP, open source, webmail, automatická odpověď, Unix

Keywords

Procmail, Roundcube, mail filter, plugin, PHP, open source, webmail, automatic reply, Unix

Citace

ŠPANĚL, Tomáš. *Správa filtrování e-mailů v Roundcube*. Brno, 2020. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Jaroslav Dytrych, Ph.D.

Správa filtrování e-mailů v Roundcube

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Jaroslava Dytrycha, Ph.D. Další informace mi poskytl Ing. Petr Lampa. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Tomáš Španěl
11. června 2020

Poděkování

Rád bych poděkoval panu Jaroslavu Dytrychovi za vedení a pomoc, kdykoliv jsem ji potřeboval. Dále bych také rád poděkoval i panu Petru Lampovi za pomoc a druhý pohled na problematiku. Nakonec bych rád poděkoval také panu Bohumilu Michalovi za pomoc a rychlou zpětnou vazbu při prvotním uživatelském testování.

Obsah

Kapitola 1

Úvod

V dnešní době se člověk těžko vyhne nějaké formě komunikace pomocí elektronické pošty, ať už jde o firemní komunikaci, osobní korespondenci, reklamní sdělení či spam. Objem elektronické pošty z nejrůznějších zdrojů tedy roste, a někdy až zahlcuje. Vzniká tak přirozený požadavek tento poštovní tok nějak třídit – rozdělit tuto poštu do menších a stravitelných bloků.

Pro přístup ke své poště člověk použije nějaký program (třeba Outlook) nebo webové aplikace přístupné z prohlížeče (třeba Gmail). Některé z těchto programů či stránek mají už přímo v sobě funkčnost pro třídění pošty zabudovanou, jiné však ne, ale poskytují možnost, jak je rozšířit dalšími funkcemi. Jedním z nich je právě Roundcube.

Tento text se zabývá problematikou tvorby zásuvného modulu pro Roundcube, který jej rozšíří o funkce umožňující filtrování elektronické pošty a oznamování nepřítomnosti pomocí specializovaného nástroje Procmail. Samotný Roundcube slouží pro přístup k elektronické poště z prostředí prohlížeče, zatímco Procmail běží pod Linuxem. Tím, jak tyto nástroje fungují a jak je lze zkombinovat dohromady, se zabývá kapitola 2. V této kapitole jsou také vytyčeny požadavky na cílový modul.

Kapitola 3 poté popisuje samotný návrh zásuvného modulu s přihlédnutím k vytyčeným požadavkům a možnostem Roundcube a nástroje Procmail.

V kapitole 4 jsou popsány zajímavé pasáže ze samotné tvorby modulu a jeho výsledná struktura.

V kapitole 5 je následně uveden postup testování modulu a jeho nasazení a konfigurace. Současně je zde uvedeno, na jaké problémy jsem při testování narazil a jejich řešení.

V závěrečné kapitole 6 je shrnutí výsledků práce a vlastností vytvořeného modulu vůči vytyčeným požadavkům a zamyšlení nad možnými rozšířeními do budoucna.

Kapitola 2

Roundcube a filtrování pošty

V této kapitole bude blíže představen e-mailový klient Roundcube a filtrační nástroj Procmail. Poté budou přiblíženy cíle a požadavky na zásuvný modul vzhledem k možnostem těchto nástrojů a nakonec budou popsány další použité technologie.

2.1 Roundcube

Roundcube je webový klient s otevřeným zdrojovým kódem (open source) pro přístup k poště pomocí protokolu IMAP[1], dostupný pod licencí GPLv3 [20]. Nyní je již ve stabilní verzi 1.4 a práce na něm stále aktivně pokračují jak z rukou zakladatelů projektu, tak komunity v repozitáři projektu na platformě GitHub[2].

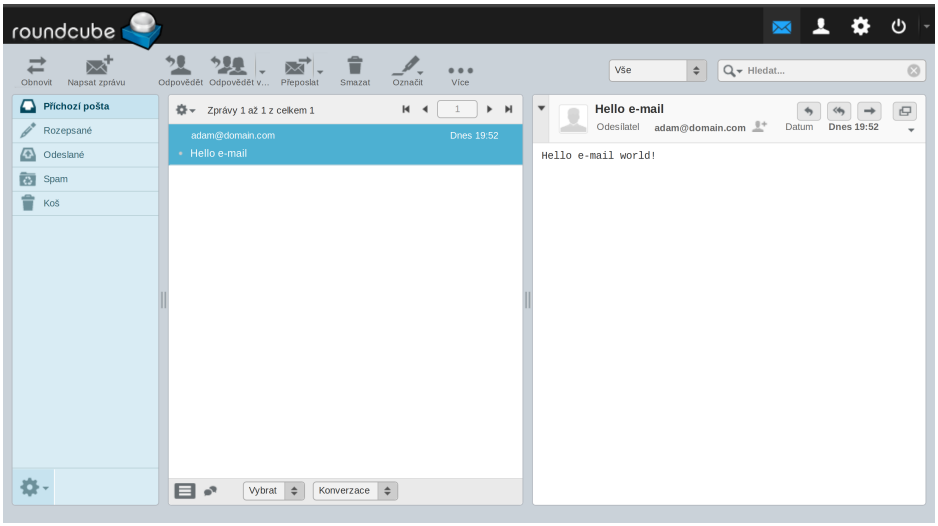
Webová aplikace Roundcube funguje jako plnohodnotná náhrada za desktopového poštovního klienta, která je přístupná z prohlížeče. Kromě čtení a odesílání zpráv nabízí mimo jiné tyto funkce:

- Zobrazení originální zprávy včetně hlaviček
- Správu kontaktů a jejich výběr z adresáře (například LDAP)
- Správu více identit pod jedním účtem
- Správu složek protokolu IMAP
- Vytváření předdefinovaných odpovědí
- Kontrolu pravopisu

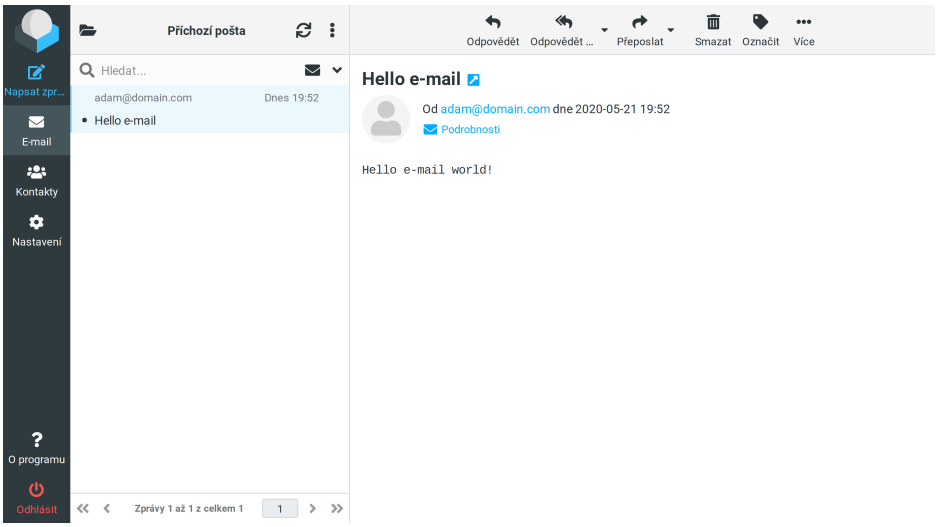
Roundcube také podporuje systém vlastních témat vzhledu. Poslední verze obsahuje už v základu dvě témata – starší *Larry* a modernější *Elastic* (od verze 1.4), které je responzivní a navrženo tak, aby s jeho využitím bylo možné Roundcube snadno ovládat i z mobilního zařízení s menší obrazovkou. Uživatel si pak může ve svých předvolbách zvolit, které z dostupných témat chce používat jako výchozí. Ukázka témat vzhledu je na obrázku 2.1.

Kromě vzhledové stránky lze i rozšiřovat nebo pozměňovat základní funkcionalitu pomocí zásuvných modulů (plugin). Ty lze mimo jiné získat z oficiálního repozitáře[3], který obsahuje desítky zásuvných modulů vytvořených komunitou kolem Roundcube.

[1] IMAP – Internet Mail Access Protocol https://tools.ietf.org/html/rfc3501
[2] https://github.com/roundcube
[3] https://plugins.roundcube.net/



(a) Téma Larry



(b) Téma Elastic

Obrázek 2.1: Porovnání základních témat vzhledu klienta Roundcube

Aplikační rozhraní pro tvorbu zásuvných modulů

Pro tvorbu zásuvných modulů je v Roundcube připraveno aplikační rozhraní (API), které umožňuje reagovat na důležité události v rámci aplikace pomocí tzv. *háků* (hook) [19]. Zásuvný modul zaregistruje u jádra aplikace funkci, která je poté vždy provedena, pokud nastane specifikovaná událost. Příkladem z wiki stránky projektu je třeba zachycení události při zobrazení e-mailu a náhrada textových verzí emotikon za jejich grafickou verzi.

Důležitou funkcionalitou API je možnost zobrazit uživateli vlastní stránky pomocí systému šablon, což jsou soubory v jazyce XHTML rozšířené o speciální značky z vlastní domény webmailu `<roundcube:* .../>` [21]. Tyto značky umožňují například vkládat lokalizované popisky či vkládat další soubory v jazyce XHTML nebo objekty renderované přímo zásuvným modulem, což je vhodné pro omezení redundance, pokud jsou některé prvky použity napříč více šablonami. Další dostupné značky umožňují například vložení

výsledku výpočtu, proměnné prostředí nebo podmínečně skrýt nebo vytisknout část výsledné stránky. Kromě vytváření stránek v jazyce XHTML pomocí šablon může zásuvný modul k výsledné stránce připojit také vlastní kaskádové styly (CSS) nebo skripty v jazyce JavaScript (JS).

Propojení stránky a logiky na serveru potom zajišťují *akce*, které může zásuvný modul také zaregistrovat u jádra aplikace. Argumenty, jako třeba data webového formuláře, jsou předávány pomocí standardních metod GET a POST protokolu HTTP.

2.2 Procmail

Procmail funguje jako LDA[4] a nástroj pro filtrování e-mailů v jednom. Jeho tvůrcem je Stephen R. van den Berg a pracoval na něm spolu s Philipem A. Guentherem. Poslední stabilní verze vyšla v roce 2001 a projekt je od té doby neudržovaný. Nástroj je dostupný pro systémy založené na Unixu [25].

Uživatel zadefinuje filtrovací pravidla[5] do souboru pomocí speciální syntaxe, tato pravidla poté Procmail sekvenčně prochází, dokud nenarazí na takové, které odpovídá vstupnímu e-mailu, a poté zpracování ukončí. V případě, že žádné z pravidel zprávě neodpovídá, je umístěna do výchozí složky.

Každé pravidlo se skládá z hlavičky, nula či více řádků s podmínkami a akce. Do hlavičky je možné vložit volitelné příznaky upravující standardní chování Procmailu, jako je třeba vytvoření kopie zprávy, která pak pokračuje dále pro zachycení dalšími pravidly, místo aby bylo provedením akčního řádku zpracování ukončeno.

Řádek s podmínkou začíná znakem * následovaným regulárním výrazem, který Procmail srovnává vůči vstupní zprávě. Před tento výraz lze vkládat speciální znaky, které mění význam podmínky, jako například vykřičník, který zneguje výsledek porovnání.

Po podmínkách následuje akce, která se provede pouze pokud platí všechny předchozí podmínky. Akce je buď cesta k cílové složce, kam se e-mail uloží, přeposlání e-mailu na jinou adresu nebo přesměrování e-mailu rourou do jiného programu. Akce však může být i vnořený blok, obsahující další pravidla. Lze tak vytvářet sady filtrů, které se porovnávají pokud pro ně platí společné podmínky. Takto vnořovat bloky lze neomezeně, což umožňuje vytvářet i komplexní sítě filtrů.

Kompletní popis lze nalézt v manuálové stránce pro Procmail, ze které jsem při popisu čerpal [4].

Formail

Formail je doprovodný nástroj Procmailu, umožňující základní manipulaci vstupního e-mailu, jako například extrahování/skládání hlaviček nebo generování hlaviček pro automatickou odpověď. Kompletní popis lze nalézt na manuálové stránce [3].

2.3 Požadavky a existující řešení

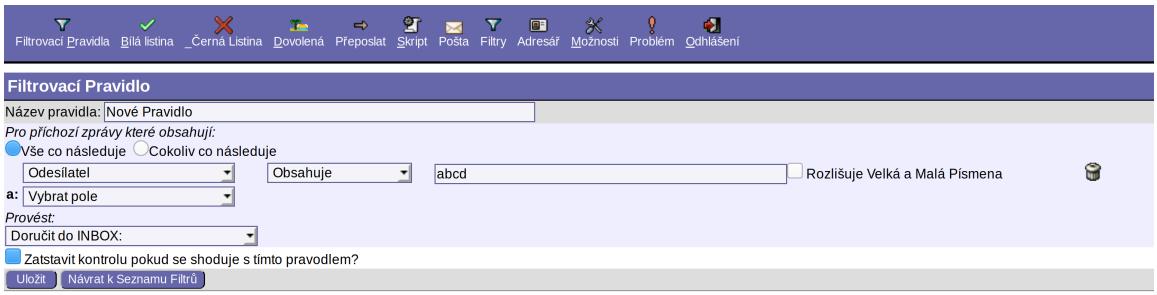
Na jedné straně tedy stojí Roundcube, webmail pro běžné uživatele bez vestavěné funkcionality pro automatické filtrování příchozí pošty, a na druhé Procmail, filtrační nástroj pro pokročilé uživatele bez vestavěné funkcionality pro vzdálenou správu.

[4] Local Delivery Agent – komponenta zodpovědná za doručení e-mailu do schránky lokálního uživatele
[5] V dokumentaci se též používá termín „recept“

Cílem práce je vytvořit zásuvný modul pro Roundcube, umožňující správu filtrů pomocí nástroje Procmail, fungující jako prostředník mezi těmito komponentami. Tyto filtry by měly umožňovat třídění příchozích e-mailů do složek, přesměrování pošty na jiné adresy či oznamovat nepřítomnost pomocí automatických odpovědí. Modul by však neměl uživateli znemožnit dále ručně přidávat do souboru vlastní pravidla.

V oficiálním repozitáři zásuvných modulů Roundcube lze nalézt pouze jediný modul, který podporuje správu filtrů pomocí nástroje Procmail. Plugin `manageprocmail`[6], jehož autorem je Ján Mochňak. Jedná se o předchozí bakalářskou práci na toto téma, která však nebyla dokončena a tento modul není plně funkční. Rozhodl jsem se práci pana Mochňaka nevyužít, neboť jeho návrh je dle mého názoru zbytečně zmatečný – využití nadstavby Nette pro tvorbu uživatelského rozhraní tam, kde není potřeba; práce nad databází tam, kde není potřeba; stavění kódu za použití zdrojových souborů zastaralého Horde (také webmail) atd.

I když přímo pro kombinaci Roundcube a Procmail hotové řešení ještě neexistuje, lze se inspirovat ze správce filtrů webového klienta Horde[7], který filtrování pomocí Procmailu podporuje. Snímek formuláře pro tvorbu filtru je na obrázku 2.2.



Obrázek 2.2: Snímek konfiguračního formuláře pro tvorbu filtru ve filtračním modulu Ingo pro webmail Horde

Cílová funkčnost

Všeobecně by tento zásuvný modul měl sloužit ke dvěma věcem – zadávání pravidel z prostředí webmailu a zjednodušení formátu pravidel tak, aby je pochopil i člověk technicky nezdatný nebo neznalý syntaxe pravidel.

První část v podstatě znamená zautomatizovat proces ukládání pravidel, aby se uživatel nemusel ručně připojit na vzdálený server, upravit soubor s pravidly a zase odpojit. Druhá část znamená, že skladba filtru je od implementace pomocí Procmailu abstrahována do formy prezentované uživateli pomocí termínů a prostředků, na které je z prostředí webmailu nebo všeobecně práce s e-mailem zvyklý.

U oznamování nepřítomnosti by měl uživatel mít možnost zadat vlastní odpověď místo nějaké generické zprávy generované modulem. Může tak dát například vědět, kde je nebo kdy se vrátí. Dále by bylo vhodné neodpovídat na automatické zprávy z ostatních systémů (odpověď se neočekává). Taktéž může být pro uživatele nežádoucí odpovídat na každou zprávu od jednoho odesílatele, ale pouze po vypršení určitého časového limitu mezi zprávami.

[6] https://plugins.roundcube.net/packages/mochja/manageprocmail
[7] https://www.horde.org/apps/ingo/

S přihlédnutím k ostatním filtračním řešením z toho vyplývá následující cílová funkčnost:

- Pojmenování filtrů
- Jednoduché podmínky – volba pole (odesílatel, předmět, ...), porovnávací operace (obsahuje, rovná se, ...) a porovnávaná hodnota
- Jednoduché akce – uložit do složky, přeposlat nebo zahodit
- Přidávání více podmínek nebo akcí v rámci jednoho filtru
- Možnost zvolit vztah mezi podmínkami – všechny platí, jakákoliv platí
- Možnost pokračovat ve filtrování nebo ukončení, pokud filtr splní podmínky
- Možnost filtry individuálně povolit nebo zakázat
- Možnost zadávat časový rozsah, vlastní zprávu a interval mezi odpověďmi u oznamování nepřítomnosti

2.4 Další použité technologie

Jelikož se zabýváme webovým klientem pro správu elektronické pošty, tak další použité technologie pokrývají právě požadavky na funkčnost s tím spojenou. Pro vývoj jsem si chtěl vytvořit vývojové prostředí podobné skutečné struktuře po nasazení, a aby zároveň bylo přenositelné. Přirozeně tak výběr padl na platformu Docker.

Docker

Docker je platforma pro běh kontejnerů, což je standardní jednotka softwaru, která obaluje kód a všechny jeho závislosti tak, že běží rychle a spolehlivě napříč různými výpočetními prostředími. Výhodou oproti virtualizovanému stroji je zejména menší výpočetní režie a celkově menší velikost [13].

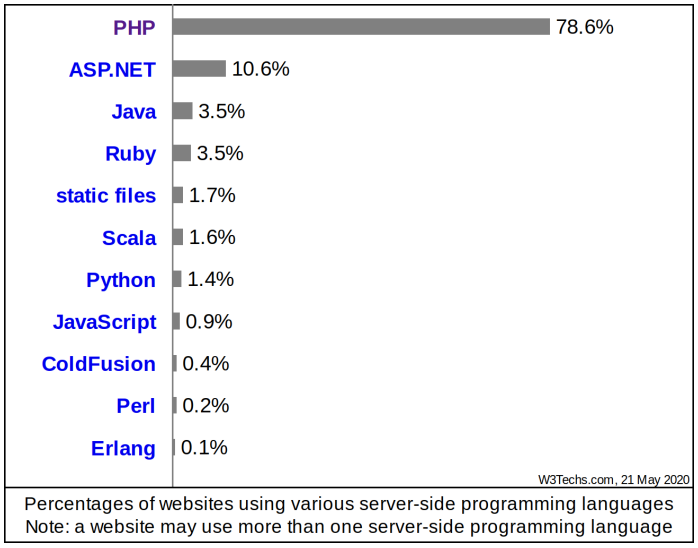
Při práci jsem jej využil pro vytvoření přenositelného a snadno konfigurovatelného vývojového prostředí. Všechny použité technologie byly pro vývoj nasazeny právě pomocí kontejnerů této platformy.

PHP

PHP je zkratka pro *PHP: Hypertext Preprocessor*. Jedná se o skriptovací jazyk vhodný pro vývoj webových aplikací s dynamickým obsahem [1].

Byl vytvořen Rasmusem Lerdorfem v roce 1994. Samotná zkratka původně měla jiný význam. Lerdorf vytvořil sadu skriptů *Personal Home Page Tools*, přezdívanou pouze jako *PHP Tools*. Tuto sadu používal na počítání přístupů na svou domovskou stránku. Odtud prošlo PHP za roky mnohými změnami, až do podoby jak ji známe dnes [2].

Jak lze vidět na obrázku 2.3, PHP je k datu 21. 5. 2020 nejpoužívanějším jazykem pro vytváření serverových webových aplikací s podílem 78,6 %. Statistika byla vytvořena na základě přibližně 10 milionů nejnavštěvovanějších stránek podle portálu Alexa v kombinaci s 1 milionem nejnavštěvovanějších stránek podle portálu Tranco [10]. Při vývoji bylo použito PHP ve verzi 7.3, což v době tvorby kontejneru byla nejnovější verze.



Obrázek 2.3: Porovnání využití jazyků pro tvorbu serverových aplikací k datu 21. 5. 2020, obrázek je přejat ze stránky W3Techs [9]

SFTP

SFTP v tomto případě znamená *SSL File Transfer Protocol*[8]. Protokol poskytuje zabezpečený přenos souborů či všeobecně práci nad vzdáleným souborovým systémem přes zabezpečený datový kanál poskytnutý například pomocí protokolu SSH2 [7]. Implementaci klienta využívající tento protokol poskytuje například rozšířený balík OpenSSH[9], který jsem použil při vývoji i já. Použitá verze byla 7.4.

Apache

Apache je rozšířený webový server s otevřeným zdrojovým kódem, který je dostupný zdarma. Webový server zpracovává příchozí požadavky na přístup k URL[10] a vrací požadovaný soubor nebo výstup programu (třeba právě skriptu jazyka PHP) [17].

Tento webový server jsem použil pro nasazení testovací instance Roundcube, použitá verze byla 2.4.

Postfix

Postfix je poštovní, zdarma dostupný server s otevřeným zdrojovým kódem pro odesílání a příjem elektronické pošty pomocí protokolu SMTP[11]. Jako software s otevřeným zdrojovým kódem byl zveřejněn v roce 1998, jeho tvůrcem je Wietse Venema, který v té době pracoval pod firmou IBM. Postfix vznikl jako modernější alternativa ke stárnoucímu programu Sendmail [6].

Při práci v rámci vývojového prostředí jsem použil verzi 3.1.12.

[8] Pozor na podobnost zkratky se *Simple File Transfer Protocol* (také SFTP) a *File Transfer Protocol Secure* (FTPS), jedná se o jiné protokoly

[9] https://www.openssh.com/

[10] URL – Uniform Resource Locator https://tools.ietf.org/html/rfc1738

[11] Simple Message Transfer Protocol https://tools.ietf.org/html/rfc5321

Dovecot

Dovecot je server s otevřeným zdrojovým kódem pro čtení pošty z úložiště pomocí protokolu POP3[12] nebo protokolu IMAP. První verze vyšla v roce 2002 a jeho tvůrcem je Timo Sirainen, který jej vytvořil, neboť nebyl spokojený s kvalitou kódu u tehdy dostupných alternativ poskytujících čtení pošty pomocí protokolu IMAP. Největší rozdíl mezi POP3 a IMAP je, že ten první stáhne poštu ze serveru, kde nenechá žádnou kopii, zatímco IMAP ano, a poštu tak lze číst z více míst a navíc podporuje třídění pošty do podsložek [12].

Tento server je použit i na Fakultě informačních technologií VUT, ve vývojovém prostředí jsem použil verzi 2.3.9.

HTML

Hypertext Markup Language (HTML) je značkovací jazyk pro tvorbu webových stránek, jehož tvůrcem je Tim Berners-Lee. První verze vznikla na začátku 90. let a obsahovala stručný seznam značek pro popis objektů stránky (záhlaví, zápatí, seznam, ...). V průběhu let se tento živoucí standard postupně vyvíjel a byl rozšiřován o další značky, dnes je již ve verzi HTML5.

Jelikož cílem práce je zásuvný modul pro webovou aplikaci, pochopitelně právě tento jazyk byl v kombinaci s CSS (viz níže) použit pro prezentaci uživatelského rozhraní. Informace o HTML jsem čerpal z knihy HTML5 a CSS3 [5].

CSS

Zatímco HTML popisuje strukturu stránky, vzhled řeší kaskádové styly – Cascading Style Sheets (CSS). Jedná se o jazyk pro popis stylu (vzhledu) stránek. Navrhl jej Håkon Wium Lie v roce 1994, když pracoval v CERN pod vedením Tima Bernerse-Lee, a byl postupně vylepšován a rozvíjen podobným způsobem jako HTML [18].

Dnes je dostupný již ve verzi CSS3.

ECMAScript (JavaScript)

ECMAScript (ES) je interpretovaný jazyk běžně používaný pro dodání dynamické funkčnosti jinak statickým stránkám v jazyce HTML. Dnes je standardní součástí všech webových prohlížečů – kód jazyka ECMAScript je vložen přímo ve zdrojovém kódu stránek a prohlížeč jej vykonává na straně klienta. Tento jazyk se původně nazýval JavaScript, jehož tvůrcem je Brendan Eich, poté byl však v roce 1997 standardizován ve specifikaci ECMA-262 a dostal jméno ECMAScript. Původní jméno je registrovanou značkou, která je nyní vlastněna firmou Oracle, a stalo se natolik zažitým, že je využíváno všeobecně jako synonymum i pro ECMAScript [11].

Při práci jsem využil funkcí jazyka ECMAScript6 vydaného v roce 2015, podpora ve všech běžných prohlížečích je od roku 2018[13].

Python

Python je interpretovaný programovací jazyk, jehož první verze vyšla v roce 1991. Vytvořil jej Guido van Rossum a jméno zvolil podle televizního pořadu Monthy Pythonův létající

[12] POP3 – Post Office Protocol (verze 3) https://tools.ietf.org/html/rfc1939

[13] https://www.w3schools.com/js/js_es6.asp

cirkus. Jeho výhodou je jeho velká rozšířenost (často je dostupný už ve výchozích instalacích distribucí Linuxu) a možnost snadno použít i kód napsaný v jiných jazycích pomocí podpůrných modulů. Jeho moduly `numpy` a `scipy` jsou zase často používány ve vědeckých sférách (např. trénování neuronových sítí) [23].

Při práci jsem použil Python ve verzi 3.6 pro dekódování hlaviček a těla e-mailu.

Kapitola 3

Návrh zásuvného modulu

Procmail nabízí mnoho funkcí, díky kterým lze vytvářet i komplexní filtrační scénáře. Už jenom možnost posílat e-mail rourou jako vstup do dalších programů dává možnosti takřka neomezené. Tím však vznikají závislosti na dalších nástrojích. Při návrhu jsem se tedy snažil dbát na to, aby těchto externích závislostí bylo ve výsledku nutných co nejméně.

3.1 Podmínky

Jak bylo řečeno v kapitole 2.2, podmínka v Procmailu je v podstatě regulární výraz, který se porovnává se vstupním e-mailem. Podmínka v Procmailu na pole *předmět* by pak mohla vypadat takto: `* Subject:.*abcd.*` (hledaná hodnota je abcd). Vzniká však problém, pokud je stejný řetězec obsažen i v těle zprávy a podmínka tedy v takovém případě platí i přesto, že cílem bylo pouze pole hlavičky.

Je tedy třeba rozdělit podmínky na dvě skupiny – pro hlavičku a pro tělo. Předsazením regulárního výrazu v podmínce řetězcem „`X ??`“ lze změnit výchozí chování tak, že se regulární výraz porovnává vůči hodnotě proměnné (v tomto případě `X`) místo e-mailu. V Procmailu jsou už v základu připraveny dvě speciální proměnné, které lze využít pro porovnávání pouze s poli hlavičky (`H ??`) nebo pouze s tělem (`B ??`). Upravená podmínka na předmět by pak vypadala takto: `* H ?? Subject:.*abcd.*`

Další problém však nastane ve chvíli, kdy je část e-mailu nějak kódována. Předmět e-mailu s hodnotou „Český předmět“ v kódování UTF-8 a formátu Base64[1] vypadá následovně: `Subject: =?UTF-8?B?xIxlc2vDvSBwxZllZG3Em3Q=?=` V rámci jedné hlavičky se může dokonce objevit i více kombinací takto kódovaných kusů textu (nebo nekódovaných vůbec). Procmail však neprovádí žádné dekódování, porovnání by tedy probíhalo nad nedekódovanou variantou a selhalo. Pro korektní porovnávání bude třeba nejdříve e-mail dekódovat a převést do kódování použitého u vytvářených pravidel. V rámci souboru s pravidly lze vytvářet i vlastní proměnné, do kterých lze kromě čistého textu přiřazovat i výstup systémového příkazu, na jehož standardní vstup Procmail automaticky posílá zpracovávaný e-mail. Lze tedy analogicky k Procmailu vytvořit proměnné pro hlavičku a tělo, např. `HEADER_DECODED` a `BODY_DECODED`, do kterých se přiřadí výstup dekódovacího příkazu a v podmínkách pak bude porovnávání probíhat nad dekódovanými variantami místo těch vestavěných.

[1] https://tools.ietf.org/html/rfc4648

Pole a operace

Uživateli budou prezentovány přednastavené možnosti pro výběr pole hlavičky, sestavené z často používaných položek. Jejich přehled je připraven v tabulce 3.1, pro jejich přesný popis lze nahlédnout do RFC 4021 [16].

	Hlavička
Od	(From|Reply-To|Return-Path):
Předmět	Subject:
Komu	To:
Cc	Cc:
List-Id	List-Id:

Tabulka 3.1: Přehled přednastavených polí a jejich ekvivalent ve formě regulárních výrazů

Podobně budou prezentovány také operace, aby uživatel nemusel zadávat regulární výraz ručně, avšak i tato možnost bude ponechána. Na regulární výrazy pro jednotlivé operace lze nahlédnout v tabulce 3.2. Použití dvojité stříšky u podmínek na tělo není překlepem, jedná se o rozšíření Procmailu, kterým se označuje začátek nebo konec celé víceřádkové sekce (jakou je právě tělo), zatímco jedna stříška a dolar označují začátek a konec řádku.

	Hlavička	**Tělo**
Obsahuje	^Subject: .*abcd.*$	*abcd*
Začíná na	^Subject: *abcd*.*$	^^*abcd*
Rovná se	^Subject: *abcd*$	^^*abcd*^^

Tabulka 3.2: Přehled přednastavených operací a jejich ekvivalent ve formě regulárních výrazů. Hledaná hodnota je *abcd*.

Konjunkce a disjunkce více podmínek

Uživatel si bude moci zvolit, zda se má akce filtru provést, pokud všechny podmínky platí – konjunkce, nebo pokud jakákoliv podmínka platí – disjunkce. Jelikož v Procmailu musí implicitně platit všechny podmínky, než je provedena akce, vytvoření podmínek v konjunkci v podstatě znamená transformaci každé podmínky na jeden řádek. Jak bude transformace vypadat lze vidět na obrázku 3.1, podmínka na hlavičku je označena $\boldsymbol{H_x}$ a podmínka na tělo $\boldsymbol{B_x}$, kde $\boldsymbol{x}$ je číslo podmínky.

$$H_1 \wedge B_2 \wedge !H_3 \wedge !B_4 \sim \begin{array}{l} :\texttt{0} \\ *\texttt{H}_1 \\ *\texttt{B}_2 \\ *\texttt{!H}_3 \\ *\texttt{!B}_4 \end{array}$$

Obrázek 3.1: Transformace podmínek v konjunkci na jejich Procmail ekvivalent

U disjunkce je situace jiná. V některých případech lze zredukovat podmínku pomocí znaku | (nebo) v regulárním výrazu, aby stačilo pouze jediné pravidlo: $H_1 \wedge H_2 \sim *\texttt{H}_1|\texttt{H}_2$.

Jakmile se však objeví podmínka s negací, nebo kombinace podmínek na tělo a na hlavičku, jedno pravidlo už přestane stačit.

Prostým rozdělením podmínek v disjunkci mezi více pravidel se stejnou akcí by mohla nastat situace, kdy platí podmínek více, a tedy akce by se také provedla víckrát, což je nežádoucí chování. Pro vyřešení tohoto problému lze v hlavičce pravidla použít příznak *E*. Tento příznak způsobí, že Procmail zpracuje pravidlo pouze tehdy, pokud předcházející pravidlo neprošlo. Pokud toto pravidlo uspěje, jsou všechna bezprostředně následující pravidla s tímto příznakem zakázána. Transformace podmínek v disjunkci je zobrazena na obrázku 3.2.

$$H_1 \vee B_2 \vee H_3 \vee !B_4 \sim \begin{array}{l} \texttt{:0} \\ *\,\texttt{H}_1|\texttt{H}_3 \\ \\ \texttt{:0E} \\ *\,\texttt{B}_2 \\ \\ \texttt{:0E} \\ *\,\texttt{!B}_4 \end{array}$$

Obrázek 3.2: Transformace podmínek v disjunkci na jejich Procmail ekvivalent

3.2 Akce

Uživatelé budou moci využívat tři akce – uložení do složky, přeposlání a zahození. Poslední řádek pravidla je akční řádek, pro uložení e-mailu do složky stačí pouze zadat její cestu. Zahození pak není nic jiného než „uložení“ e-mailu do virtuálního souboru `/dev/null`.

Pro přeposlání se na začátek akčního řádku musí umístit znak `!`, za ním poté následují e-mailové adresy pro přeposlání. V některých případech může toto jednoduché přeposlání stačit, otevírá se tím však možnost pro zacyklení. Pro řešení jsem se inspiroval u návodu pro přesměrování pošty od Fakulty informačních technologií VUT [8]. V návodu se jako příklad zacyklení uvádí nedoručitelnost e-mailu. Poštovní server vrátí chybu o nedoručitelném e-mailu klientovi, ten se jí pokusí znova odeslat atd. Jako řešení je uvedeno využití podmínky s vestavěným makrem Procmailu `FROM_MAILER`, které by mělo zachytávat e-maily přímo od poštovních serverů, jakými jsou právě třeba chyby při doručování. Ovšem k vytvoření cyklu může jednoduše dojít např. i v případě, že cílová adresa pro přeposlání přeposílá e-maily zase zpět. Vzhledem k tomu, že studenti FIT VUT mají 2 adresy (fakultní a univerzitní), z nichž jedna je ve výchozím nastavení přesměrována na druhou, může k této situaci dojít pouhým opomenutím změny výchozího nastavení druhé schránky. Druhá věc ke zvážení je, že někdy je žádoucí přeposílat i e-maily, které by jinak makro `FROM_MAILER` zachytilo, jako např. automatické zprávy o stavu monitorování poštovního provozu. Pro vyřešení obou problémů se nepoužije toto makro, ale přidá se vlastní hlavička do přeposílaného e-mailu a podmínka do přeposílacího pravidla, která bude kontrolovat, že tato vlastní hlavička ještě není přítomna.

V návodu se dále uvádí další problém – pokud je přeposílán hromadný e-mail, ale doručení selže, odesílateli je sice vrácena chyba doručení, ale ten neví, u koho konkrétně

k selhání došlo, neboť adresát je adresa po přeposlání, na kterou původně e-mail neodesílal. Zde jsem použil stejné řešení jako v návodu – pro identifikaci je k e-mailu připojena hlavička `Resent-From`, která identifikuje adresu, která provedla přeposlání. Odesílatel tak může z hlavičky e-mailu s chybou doručení původní problémovou adresu dohledat. Pokud nebude uživateli chování této bezpečné varianty přeposlání vyhovovat, vždy může použít „jednoduché“ přeposlání.

Více akcí

Uživatel v rámci filtru může specifikovat více akcí, například přeposlání na osobní adresu a zároveň uložení do příchozí pošty. Každé pravidlo v Procmailu nemusí mít žádné podmínky, ale vždy musí mít právě jednu akci, touto akcí však může být i vnořený blok obsahující další pravidla. Druhou částí řešení je využití příznaku *c* v hlavičce těchto pravidel. Tento příznak vytvoří kopii e-mailu, která pak pokračuje dále, jinak by se v akčním bloku zpracování zastavilo prvním pravidlem. Kombinaci akcí ukazuje příklad ve výpisu 3.1. Součástí příkladu je i zahození, i když nedává sémanticky smysl tuto akci kombinovat s ostatními.

```
:0
* ^Subject:.*abcd.*$
{
   # ulozeni do slozky
   :0c
   nejaka_slozka

   # jednoduche preposlani
   :0c
   ! nejaka_adresa@example.com

   # zahozeni
   :0
   /dev/null
}
```

Výpis 3.1: Příklad pravidel při použití více akcí pro jeden filtr

3.3 Chování po dokončení akcí

Uživatel bude moci zvolit chování po dokončení akcí. Normálně Procmail zpracování ukončí po nalezení prvního pravidla, které odpovídá vstupnímu e-mailu. Toto chování nemusí být vždy žádoucí a uživatel tedy bude mít na výběr ze tří možností – ukončit zpracování bez kopie (výchozí chování Procmailu), ukončit zpracování a uložit kopii do příchozí pošty a pokračovat dalším filtrem.

Ukončení s kopií přidá k filtru implicitní akci, která uloží kopii e-mailu do příchozí pošty. Za předpokladu korektní konfigurace Procmailu na serveru je cesta k této schránce uložena v proměnné prostředí Procmailu `$DEFAULT`.

Pokračování dalším filtrem lze docílit přidáním příznaku *c* do hlavičky pravidla, po provedení akcí pak bude kopie e-mailu pokračovat pro zpracování dalšími filtry.

3.4 Oznamování nepřítomnosti s podporou automatické odpovědi

Dalším funkčním blokem zásuvného modulu je oznamování nepřítomnosti. Uživatel definuje časové rozmezí, kdy bude nepřítomný, text automatické odpovědi a minimální interval mezi odpověďmi. Tato funkcionalita má sloužit například k naplánování dovolených v řádu dní, nikoliv k indikování bezprostřední absence z hodiny na hodinu.

Časové rozmezí

Omezení časového rozmezí bude zajišťovat podmínka na pole hlavičky *Date*. Pole obsahuje datum a čas, kdy odesílatel označil e-mail za hotový a připravený pro vstup do poštovního systému, ne nutně přesný čas odeslání [24]. Pro potřeby modulu je však dostačující, jak může potom podmínka vypadat, je ve výpisu 3.2.

```
* Date: (Mon|Tue|Wed|Thu|Fri|Sat|Sun), (((30|31) Jan)|((0?1) Feb) 2020)
```

Výpis 3.2: Příklad podmínky na datum pro rozmezí 30. 1. 2020 – 1. 2. 2020

Interval mezi odpověďmi

Pro vyřešení tohoto problému je třeba využít nějakou formu externí mezipaměti (cache) obsahující seznam adres, na které již byla odpověď odeslána, spolu s časovou značkou posledního odeslání. Tuto roli může plnit i jednoduchý textový soubor. Při přijetí e-mailu se adresa odesílatele a aktuální čas porovná s obsahem mezipaměti a výsledek tohoto porovnání určí, zda se automatická odpověď odešle a mezipaměť aktualizuje.

Odeslání odpovědi

Pro odeslání automatické odpovědi lze e-mail rourou poslat jako vstup pro program `Formail`, který ořízne z e-mailu tělo a transformuje hlavičky pro vytvoření odpovědi. K upravené hlavičce se poté připojí nové tělo e-mailu s textem zadaným uživatelem a samotné odeslání pak provede poštovní program, jehož cesta je dostupná z proměnné prostředí Procmailu – `$SENDMAIL`.

Zamezení zacyklení a ignorování automatických zpráv

Existuje potenciální nebezpečí, že při odeslání automatické zprávy přijde z druhé strany opět automatická odpověď a mohlo by tak dojít k zacyklení e-mailů. Podobně jako u přeposlání je řešením připojení vlastní hlavičky `X-Loop:` při odeslání a přidání podmínky na její nepřítomnost do pravidla. Všeobecně je nevhodné také odpovídat na automaticky generované zprávy, u kterých se zpravidla ani odpověď neočekává. V Procmailu je pro tento případ zabudované makro `^FROM_DAEMON`, které by mělo zachytit většinu zpráv od démonů[2].

[2] angl. *daemon* – dlouho žijící proces na pozadí bez kontrolního terminálu [14]

3.5 Struktura souboru a kontrola integrity

Sekce pravidel, které generuje zásuvný modul, bude třeba v souboru označit. Jelikož by měl modul umožňovat do souboru dále vkládat vlastní pravidla, nelze obsah souboru pokaždé jednoduše přepsat.

Sekci generovanou modulem bude obalovat hlavička a patička z komentářů (v Procmailu začínají znakem #), při prvním zapsání se tato sekce zapíše na konec souboru, při následujících zápisech bude držet svou pozici, tedy ručně půjde vkládat pravidla před i za sekci. Umožnění zápisu vlastních pravidel přímo do souboru s sebou samozřejmě nese riziko, že filtry poté nebudou fungovat dle očekávání. Využití kombinace filtrů z modulu a vlastních pravidel se tedy předpokládá pouze pro pokročilé uživatele, kteří ví, co dělají.

Hlavička sekce bude obsahovat kontrolní součet obsahu sekce. Tento součet bude sloužit ke kontrole integrity sekce, která by mohla být porušena nechtěnou úpravou souboru. Nebude sloužit jako zabezpečení proti záměrným úpravám.

Komentářů lze využít i pro pojmenování filtrů a zároveň pro ohraničení jednotlivých filtrů, které mohou být tvořeny i více pravidly. Další využití komentářů je pro zakázání filtrů nebo dovolených. Všechny řádky tvořící daný filtr nebo dovolenou lze předsadit znakem pro komentář, což je efektivně zakáže, ale nadále umožní jejich zpětné čtení. Návrh sekce je ve výpisu 3.3.

```
############################
# PLUGIN START #
# HASH:<kontrolni soucet> #
############################

#START:<jmeno>
<pravidla filtru>
#END:<jmeno>

<dalsi filtry>

###############
# PLUGIN END #
###############
```

Výpis 3.3: Návrh sekce zásuvného modulu v souboru s pravidly

Zpětné čtení pravidel

Uživatel bude mít přehled o již vytvořených filtrech, což mu bude umožňovat jejich úpravy, mazání či vypnutí. Bude třeba informace o těchto filtrech někde uložit. V zásadě existují dvě možnosti – ukládat informace o existujících filtrech v mezipaměti, nebo využít parser pro zpětné čtení pravidel ze souboru a jejich transformaci na objekty použitelné v rámci zásuvného modulu.

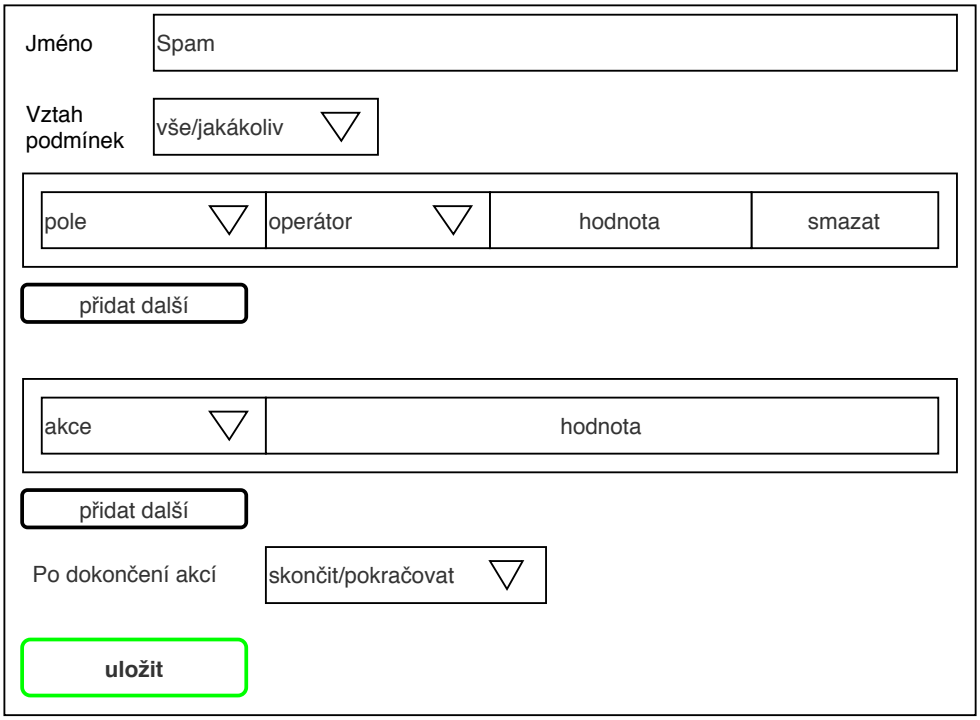
Uložení informací v mezipaměti by mohlo být realizováno například pomocí databázové tabulky nebo speciálního souboru. Sice se jedná o implementačně jednodušší řešení, avšak nezaručuje, že to, co uživatel v přehledu filtrů vidí, je opravdu to, co v souboru s pravidly je.

Lepší bude použít řešení využívající parser, které zaručí, že přehled filtrů opravdu odpovídá obsahu sekce v souboru s pravidly. Zároveň bude fungovat jako druhá vrstva (po kontrolním součtu) kontrolující integritu sekce. Toto řešení s sebou však přináší i nevýhodu zvýšené údržby, neboť změna formátu generovaných pravidel bude znamenat změnu i pro parser.

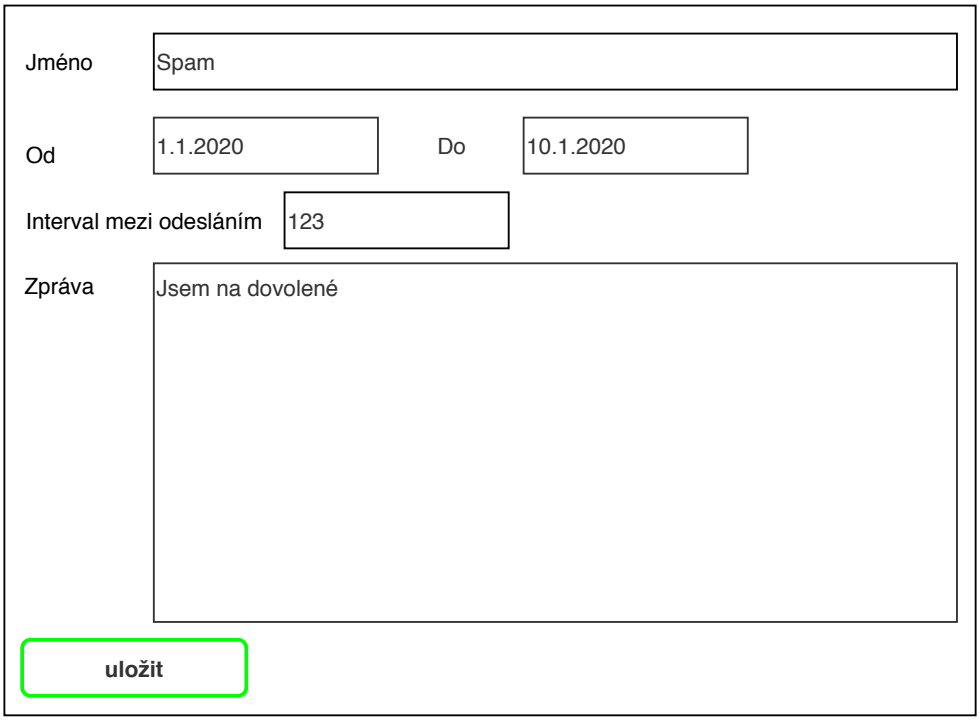
3.6 Prezentace uživateli

Roundcube nabízí rozhraní pro zobrazování vlastních stránek, které lze využít pro zobrazení konfiguračních formulářů zásuvného modulu. Konfigurace filtrů a dovolené bude umístěna na stránce nastavení. Díky systému šablon a rozhraní pro tvorbu prvků stránky na straně skriptu lze tyto formuláře lehce integrovat do webové aplikace tak, že se zdají být její přirozenou součástí a uživatel se tak pohybuje v prostředí, které je mu známé. Není tedy nutné přivádět do zásuvného modulu další závislosti v podobě nadstaveb pro tvorbu uživatelského rozhraní.

Návrh konfiguračních formulářů je na obrázku 3.3. Jednotlivé podmínky a akce budou přehledně zobrazeny jako řádky tabulky, kde je bude uživatel moci konfigurovat nebo smazat. Položky s přednastavenými hodnotami (v návrhu znázorněny obráceným trojúhelníkem) bude uživatel volit z vysouvacího seznamu, ostatní položky budou textová pole. O výsledcích akcí bude uživatel informován pomocí informačních zpráv tak, jak jsou standardně využívány u ostatních komponent Roundcube.



(a) Formulář pro tvorbu filtru



(b) Formulář pro tvorbu dovolené

Obrázek 3.3: Návrh konfiguračních formulářů zásuvného modulu

Kapitola 4

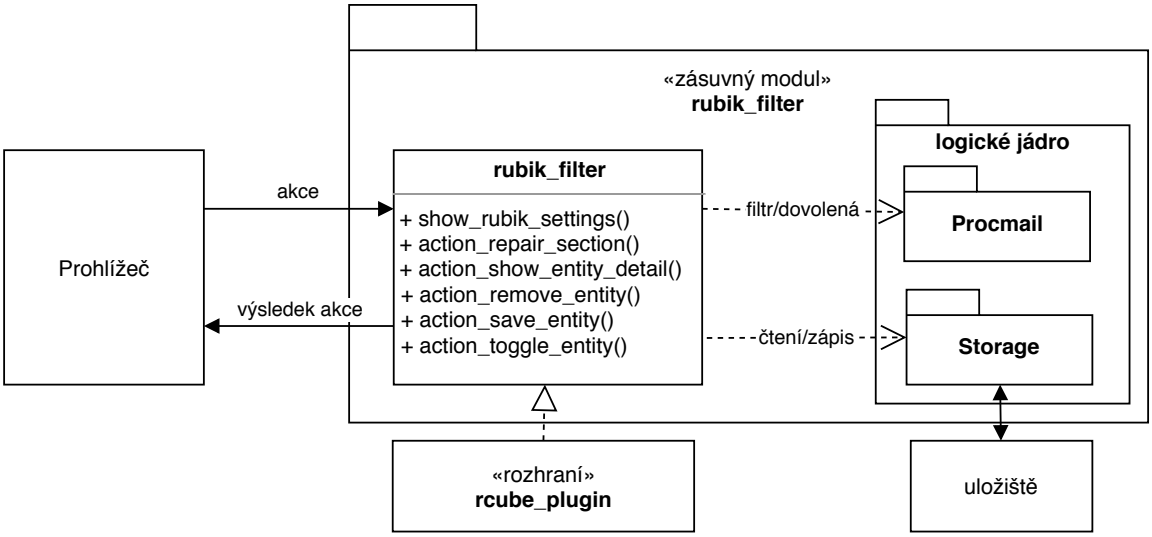
Implementace

Zásuvný modul jsem se rozhodl nazvat `rubik_filter` podle Rubikovy kostky. V mé představě představovaly nesetříděné e-maily nevyřešenou kostku a použitím správné sekvence pohybů (filtrů) vznikla kostka vyřešená (setříděná).

Na obrázku 4.1 je zobrazen diagram struktury zásuvného modulu. Každý zásuvný modul pro Roundcube musí obsahovat v kořenovém adresáři soubor se stejným názvem jako samotný adresář a s koncovkou `php`. Tento soubor musí obsahovat opět stejnojmennou třídu implementující rozhraní `rcube_plugin`. Toto je hlavní třída celého modulu a je tedy vstupním i výstupním bodem pro veškerou interakci s uživatelem – zpracovává a registruje dostupné akce a zobrazuje výsledek těchto akcí zpět uživateli.

Příkladem akce a výsledku může být např. požadavek o načtení sekce nastavení filtrů, výsledkem je pak zobrazení konfigurační stránky. Dalším příkladem je akce uložení filtru a výsledkem je úspěch nebo neúspěch prezentovaný pomocí informační zprávy v prohlížeči.

Zatímco hlavní třída poskytuje obsluhu a formální stránku zásuvného modulu, logické jádro obsahuje veškerou funkcionalitu spojenou s prací nad filtry či dovolenou (balík `Procmail`) a jejich čtením a zápisem (balík `Storage`). Oddělením této logiky od obsluhy modulu vznikla komponenta nezávislá na Roundcube a lze ji tedy využít jako knihovnu i v jiných projektech využívajících PHP.

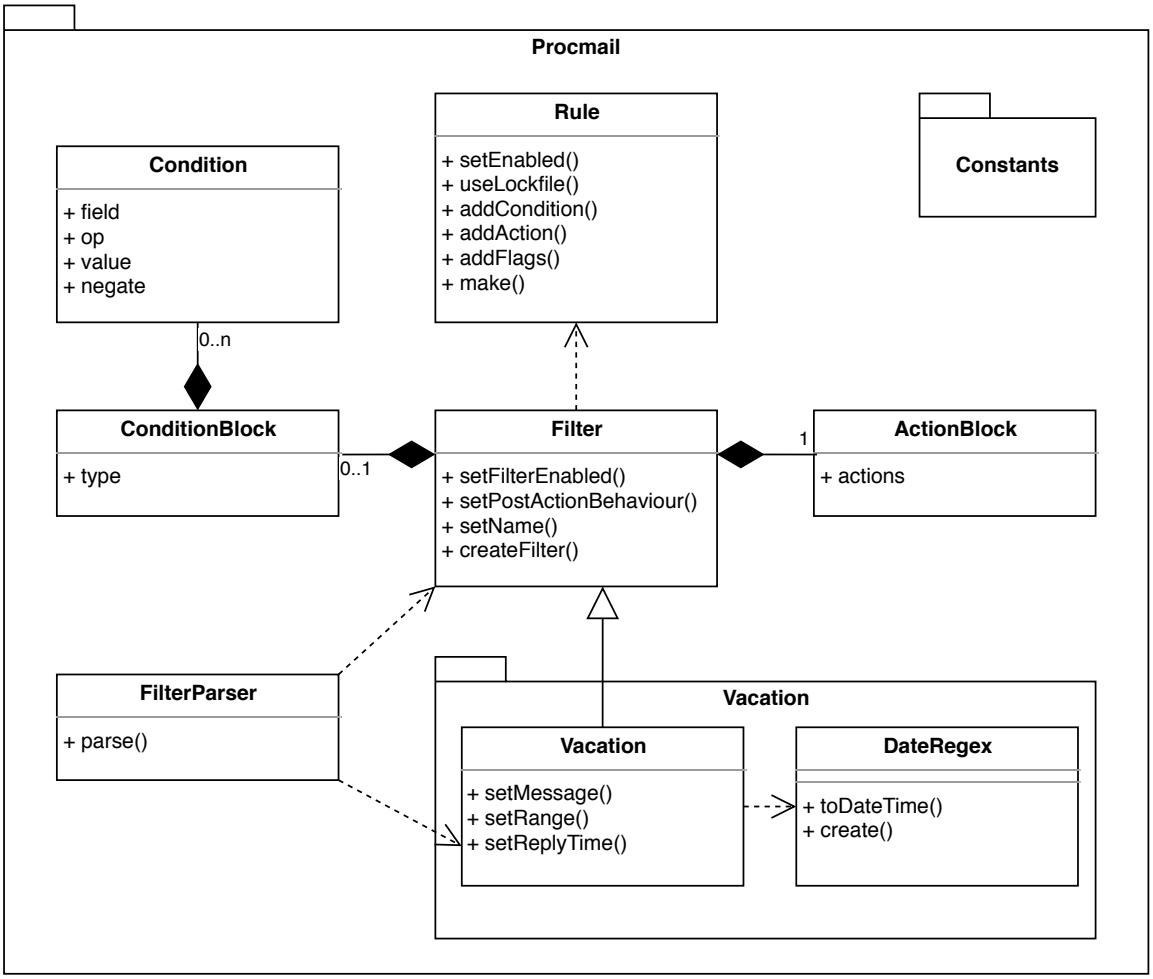


Obrázek 4.1: Diagram struktury zásuvného modulu

4.1 Logické jádro

Balík Procmail obsahuje logiku pro generování filtrů či dovolených, parser pro zpětnou transformaci pravidel ze souboru na filtry a řadu konstant popisujících přednastavené akce, operace a specifika pravidel (např. příznaky). Na obrázku 4.2 lze nahlédnout na strukturu tříd obsažených v balíku.

Nejnižší jednotkou je třída `Rule` (pravidlo), kterou lze využít pro generování libovolných pravidel dle syntaxe nástroje Procmail. Třída `Filter` pak představuje jeden filtr a funguje jako prostředník transformující množinu podmínek a akcí pomocí třídy Rule na ekvivalentní množinu pravidel nástroje Procmail. Podmínky jsou uloženy v bloku podmínek (třída `ConditionBlock`), který také drží informaci o typu vztahu mezi jednotlivými podmínkami (konjunkce/disjunkce). Samotná podmínka (třída `Condition`) je pak tvořena jedním z přednastavených polí, jednou z přednastavených porovnávacích operací, porovnávanou hodnotou a booleovskou hodnotou určující negaci. Jelikož i filtr bez podmínek je validní, nemusí obsahovat blok podmínek, nebo tento blok nemusí obsahovat žádné podmínky. Co však musí obsahovat, je neprázdný akční blok (třída `ActionBlock`), jinak není filtr validní. Ten obsahuje asociativní pole akcí, kde klíče jsou přednastavené akce. Kromě podmínek a akcí lze filtru nastavit i jméno, chování po vykonání akcí a zda je filtr povolen.



Obrázek 4.2: Diagram struktury tříd balíku Procmail z logického jádra

Třída Filter také obsahuje funkce pro vygenerování bloku s kódem pro dekódování e-mailu do proměnných `HEADER_D` a `BODY_D`. Příkazy, jejichž výstup je do těchto proměnných uložen, spouští krátký skript v jazyce Python, který provádí dekódování. Kód tohoto skriptu je vložen přímo v souboru s pravidly a pro dekódování je využito pouze modulů standardní knihovny jazyka Python. Pro korektní funkčnost těchto skriptů je třeba Python minimálně ve verzi 3.6. Ukázku dekódovacích skriptů v souboru s pravidly naleznete v příloze A.

V návrhu (viz 3.2) jsem uvedl, že by bylo vhodné poskytnout i bezpečnou variantu přeposlání, výsledné pravidlo poskytující tuto funkčnost je ve výpisu 4.1. Do proměnné `ODES` je uložena adresa původního odesílatele extrahovaná z hlavičky `Return-Path`. Dle RFC 2821 [15] je vyžadováno, aby tato hlavička byla vždy připojena při finálním doručení – e-mail opouští poštovní systém protokolu SMTP. V případě poštovních seznamů se zde může objevit adresa pro vrácení nedoručitelné pošty, která je jiná než původní odesílatel. Pokud tedy dojde k selhání u přeposlání, pošta je vrácena na tuto adresu.

Dále je pomocí nástroje Formail vložena k přeposílanému e-mailu hlavička `X-Loop-Rubik`, na jejíž nepřítomnost je v pravidle zároveň použita podmínka (proti zacyklení). Dále připojí hlavičku `Resent-From`, do které je vložena e-mailová adresa uživatele přidávajícího pravidlo (v příkladu xlogin@fit.vutbr.cz). Takto upravený e-mail je poté přesměrován do poštovního programu, který jej odešle jako původní odesílatel na cílovou adresu, kterou si už definuje sám uživatel (v příkladu xlogin.osobni@email.cz).

```
ODES=‘formail -x Return-Path‘

:0
* HEADER_D ??! ^X-Loop-Rubik: .*rubik.*$
| formail -a "X-Loop-Rubik: rubik" -a "Resent-From: xlogin@fit.vutbr.cz"
| $SENDMAIL -oi -f "$ODES" xlogin.osobni@email.cz
```

Výpis 4.1: Pravidlo poskytující bezpečné přeposlání e-mailu s kontrolou proti zacyklení (pro lepší čitelnost je akce rozložena na více řádku, v souboru je pouze na jednom)

Oznamování nepřítomnosti – dovolená

Třída `Vacation` (dovolená) je rozšířením třídy Filter o funkcionalitu potřebnou pro generování pravidel poskytujících oznamování nepřítomnosti – podpora pro podmínky na rozsahy dat skrze třídu `DateRegex` a pravidla pro kontrolu mezipaměti e-mailů a odesílání automatických odpovědí.

Samotnou mezipaměť je třeba někdy vyčistit, jinak by totiž mohla potenciálně růst donekonečna (respektive do zaplnění disku). V jeden čas může být aktivní pouze jedna dovolená, tedy nesmí se překrývat data jednotlivých dovolených, což také znamená, že do mezipaměti vždy zapisuje maximálně jedna probíhající dovolená. Další možnost zapisování do mezipaměti se tedy objeví s další dovolenou nebo popřípadě se změnou jedné z existujících dovolených, neboli při uložení dovolené. Právě v tomto momentě tedy zásuvný modul provede vyčištění mezipaměti.

Příkaz využívaný pro kontrolu mezipaměti lze vidět na výpisu 4.2. Nejdříve se pomocí nástroje Formail z e-mailu extrahuje hlavička s adresou odesílatele, ta se poté načte do proměnné `EMAIL`, do proměnné `NOW` se uloží aktuální časová značka a příkazem `touch` se vytvoří soubor mezipaměti, pokud ještě neexistuje. Zpracování mezipaměti probíhá po-

mocí programu `awk`, v parametru `diff` je předána časová značka jako rozdíl aktuálního času a uživatelem definovaného intervalu mezi automatickými odpověďmi. Program prochází mezipaměť po řádcích a pokud najde shodnou e-mailovou adresu a u ní zapsaná časová značka je novější než minimum definované proměnnou `diff`, ukončí se s chybovým kódem 1, kterým indikuje, aby se automatická zpráva neodeslala. Ostatní řádky se přepisují do dočasného souboru, který je poté přesunut místo trvalého souboru mezipaměti.

```
| formail -x "From:" | (
    read EMAIL;
    NOW=$(date +%s);
    touch ".rubik_vacation_cache";
    awk -v name="*$EMAIL" -v now="$NOW" -v diff=$(expr $NOW - 86400)
    '{
        if (index($0,name)) {
            if ($NF > diff) {exit 1}
        } else {
            print $0
        }
     }

     END {
        print name" "now
     }' ".rubik_vacation_cache" > ".rubik_vacation_cache.tmp"
    && mv ".rubik_vacation_cache.tmp" ".rubik_vacation_cache"
    || (rm ".rubik_vacation_cache.tmp" && exit 1)
)
```

Výpis 4.2: Akce pravidla kontrolujícího mezipaměť e-mailů (pro lepší čitelnost je rozložena na více řádku, v souboru je pouze na jednom)

Pokud příkaz kontroly vrátí nulovou hodnotu, provede se akce dalšího pravidla, která odešle automatickou odpověď, tento příkaz je ve výpisu 4.3. Použitím nástroje Formail a přepínače `-r` se e-mail transformuje na odpověď, tedy dojde k přidání *Re:* před předmět zprávy a k záměně příjemce a odesílatele. Poté je připojena vlastní hlavička `X-Loop`, která se používá při kontrole zacyklení, a dále hlavičky popisující typ a kódování textu odpovědi. Výstupem programu jsou hlavičky e-mailu odpovědi a za ně je připojený samotný text odpovědi v kódování Quoted-Printable[1]. Tím je sestaven kompletní e-mail odpovědi, který je poté přesměrován do programu pro odeslání pošty, jehož cesta je uložená v předdefinované proměnné prostředí `$SENDMAIL`. Kompletní příklad pravidel realizujících dovolenou lze nalézt v příloze B.

Třída `FilterParser` provádí zpětnou transformaci pravidel na instance tříd Filter a Vacation. Parser často používá regulární výrazy pro rozdělení větších bloků na menší – filtr na jednotlivá pravidla, pravidla na podmínky a akce atd. V případě, že řádek s podmínkou obsahuje více podmínek v disjunkci, je nutné manuálně rozdělit tento řádek podle párů závorek. Proto je nutné u přidávaných podmínek kontrolovat, že žádná nechybí nebo nepřebývá, což by stejně nebyl platný regulární výraz.

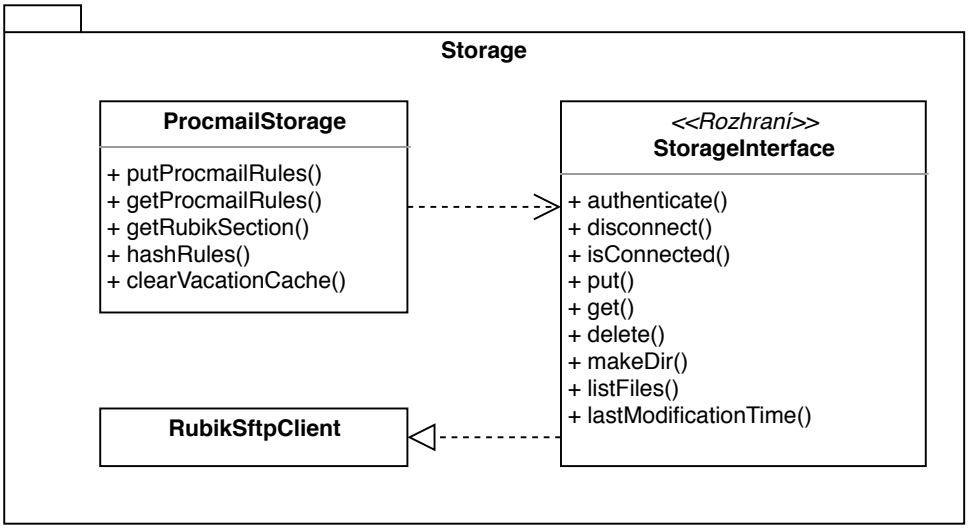
[1] https://tools.ietf.org/html/rfc2045#section-6.7

```
| (formail -rt
  -A "X-Loop: autoreply@rubik_filter"
  -i "Content-Transfer-Encoding: quoted-printable"
  -i "Content-Type: text/plain; charset=utf-8";
  echo -e "Jsem na dovolene,=0A=0Anapisu az se vratim.=0A=0AAdam";)
| $SENDMAIL -t -oi
```

Výpis 4.3: Akce pravidla odesílajícího automatickou odpověď (pro lepší čitelnost je rozložena na více řádku, v souboru je pouze na jednom)

Balík Storage

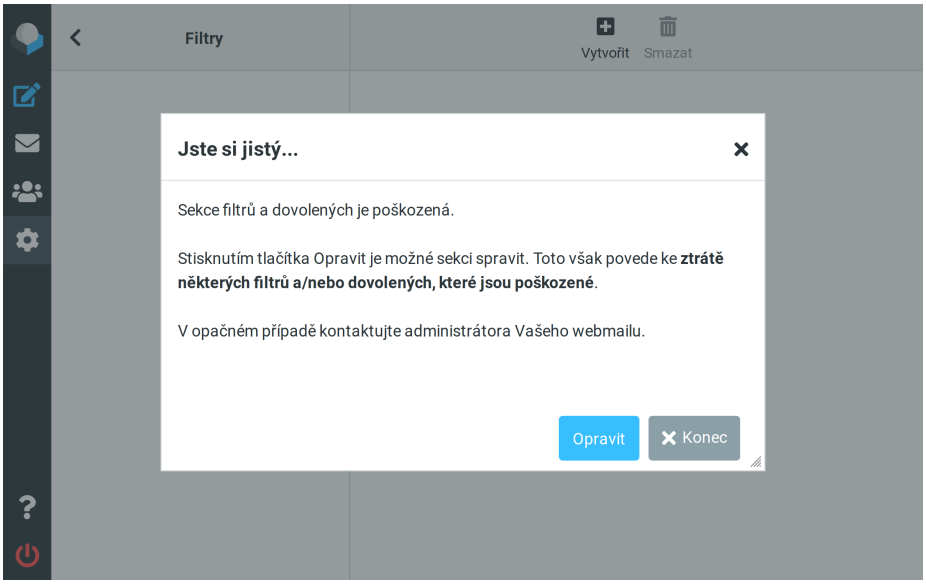
Struktura balíku Storage je na obrázku 4.3. Hlavní třídou je `ProcmailStorage`, která poskytuje funkcionalitu spojenou se čtením a zápisem pravidel a dále obsahuje důležité cesty, spravuje formát sekce v souboru a provádí kontrolu integrity pomocí kontrolního součtu. Pro kontrolní součet jsem využil algoritmus MD5, který pro vstupní data, v tomto případě text sekce pravidel včetně koncového znaku nového řádku, vytvoří 128 bitů dlouhý otisk, který je poté v souboru zapsán hexadecimálně. V případě, že otisk neodpovídá obsahu sekce nebo parser skončí chybou, se uživateli zobrazí dialog umožňující opravit sekci (viz obr. 4.4), což v podstatě znamená vymazání neplatných filtrů nebo i celé sekce.



Obrázek 4.3: Diagram tříd balíku Storage, poskytujícího funkcionalitu pro čtení a zápis pravidel

Třída `ProcmailStorage` pracuje nad rozhraním `StorageInterface`, které funguje jako klient poskytující funkcionalitu pro samotný zápis a čtení dat z úložiště a správu adresářů. Zásuvný modul se přihlašuje k úložišti se jménem a heslem právě přihlášeného uživatele Roundcube, není tak třeba vyžadovat manuální přihlášení při každé operaci nad filtry.

Využití rozhraní místo konkrétní třídy dává možnost jednoduše vyměnit implementaci tohoto klienta za jinou, bez nutnosti změn ve zbytku kódu. Jako výchozího klienta jsem vytvořil třídu `RubikSftpClient`, která poskytuje spojení s úložištěm pomocí protokolu SFTP

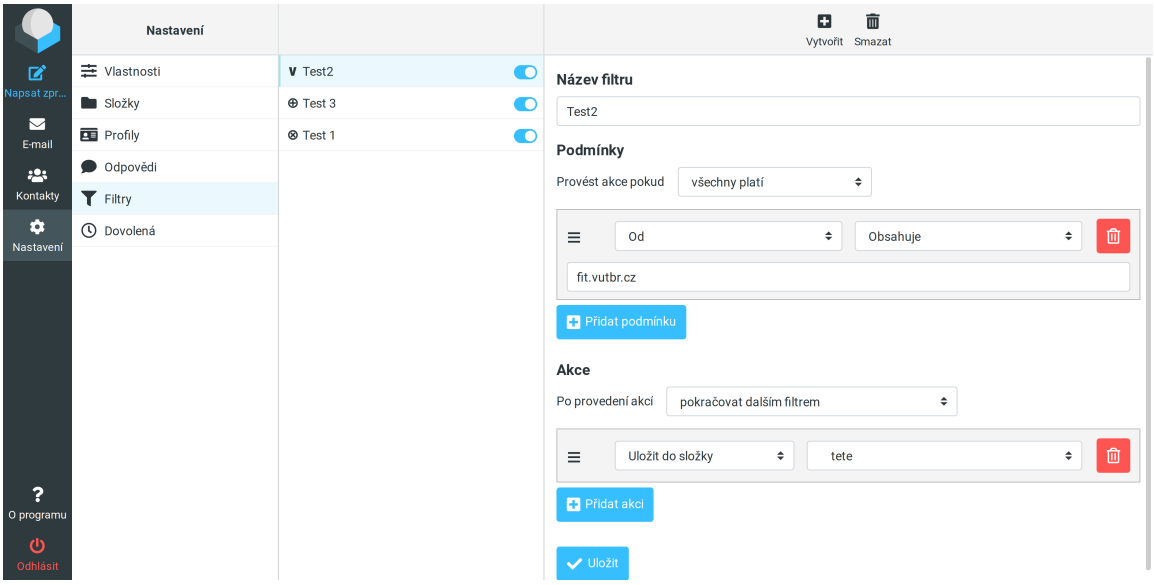


Obrázek 4.4: Chybový dialog, který je zobrazen, pokud dojde k chybě při načítání pravidel ze souboru.

(viz 2.4). Tato třída je vlastně jenom obal pro skutečnou implementaci klienta z knihovny `phpseclib`[2] dostupnou pod licencí MIT.

4.2 Uživatelské rozhraní modulu

Sekce pro konfiguraci zásuvného modulu je umístěna na stránce nastavení Roundcube, kde přibyly dvě nové položky – filtry a dovolená. Ukázka stránky s nastavením filtrů je na obrázku 4.5.



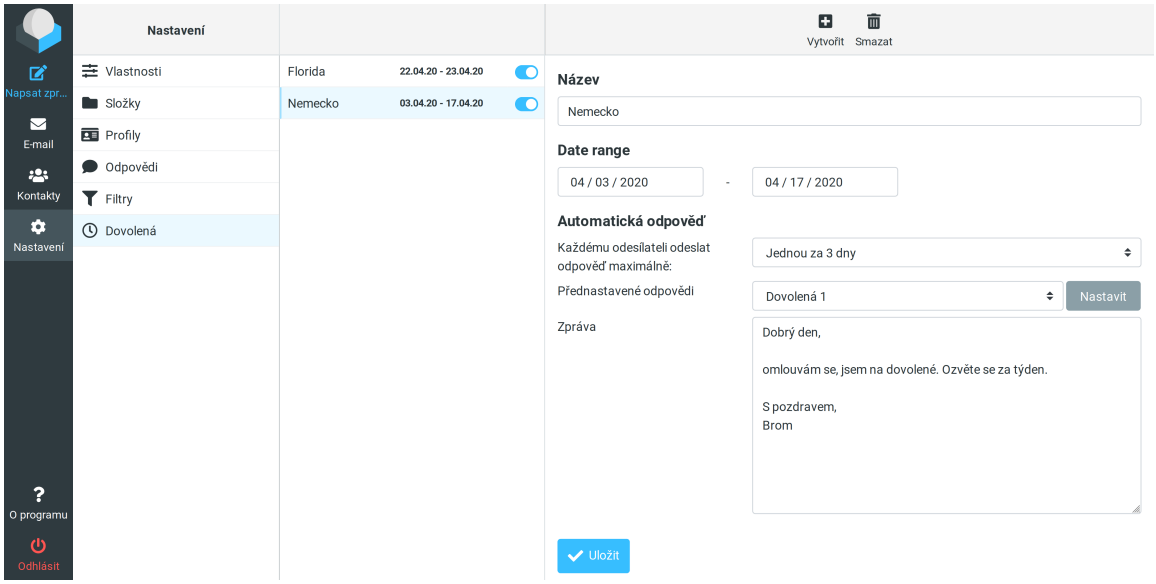
Obrázek 4.5: Snímek stránky s nastavením filtrů zásuvného modulu

[2] https://github.com/phpseclib/phpseclib

V seznamu filtrů je na pravé straně u každého řádku přepínač, kterým lze konkrétní filtr povolit nebo zakázat, na levé straně před jménem je znak z kódové sady Unicode 1.1 indikující jedno ze tří chování po provedení akcí. Filtry v tomto seznamu lze také přesouvat, jelikož na pořadí záleží kvůli sekvenčnímu zpracování pravidel. V rámci bloku podmínek nebo akcí lze jednotlivé položky také přesouvat, avšak jedná se pouze o vzhledovou záležitost, na funkcionalitu zde pořadí vliv nemá. Pro přesouvání jsem využil knihovny `SortableJS`[3] dostupné pod licencí MIT. Pokud uživatel zvolí jako akci uložení do složky, může si vybrat jednu z již existujících složek a nemusí vyplňovat jméno ručně.

Na obrázku 4.6 je stránka s nastavením dovolených. Stejně jako u filtrů lze dovolenou zakázat nebo povolit přepínačem a v seznamu je také u každé dovolené zobrazen její rozsah.

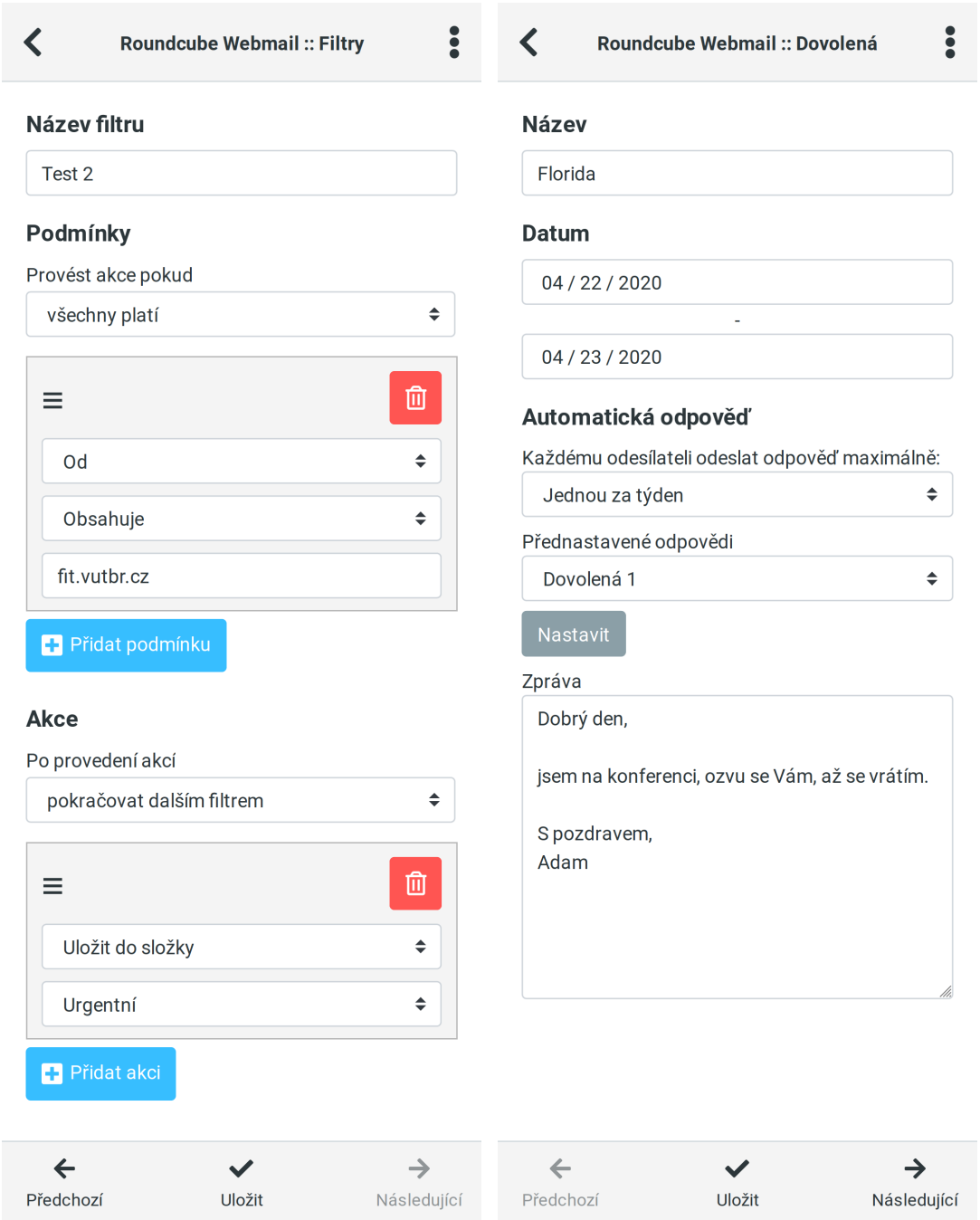
Volba rozsahu je realizována bez knihoven třetí strany pomocí vstupu typu `date`, který u většiny běžných prohlížečů (kromě Internet Explorer a Safari) umožňuje výběr data pomocí kalendáře, jinak je nutné zadávat datum ručně [22]. Interval mezi odpověďmi byl pro uživatele zjednodušen na výběr mezi několika možnostmi – jednou za 3 dny, jednou za týden a jednou za dovolenou. Co se týče odpovědí, může uživatel buď zadat vlastní zprávu nebo načíst jednu z přednastavených odpovědí, kterou si může definovat v rámci Roundcube[4].



Obrázek 4.6: Snímek stránky s nastavením dovolených zásuvného modulu

Zásuvný modul podporuje obě standardní témata vzhledu Larry i Elastic. Jelikož Elastic bylo navrženo jako responzivní téma, i konfigurační stránky modulu jsem implementoval jako responzivní a dají se tedy ovládat i z mobilního zařízení. Jak vypadá konfigurační stránka filtru a dovolené na mobilním zařízení, lze vidět na obrázku 4.7. V tomto užším zobrazení se vertikálnímu posunu uživatel zkrátka nevyhne.

[3] https://sortablejs.github.io/Sortable/
[4] jedná se o funkci samotného Roundcube, nikoliv o funkci zásuvného modulu



(a) Filtr (b) Dovolená

Obrázek 4.7: Snímek responzivního zobrazení filtru a dovolené na mobilním zařízení v tématu Elastic. V případě mobilního zobrazení se uživatel nevyhne vertikálnímu posouvání stránky, reálně uvidí v jednu chvíli zhruba polovinu (na výšku).

Kapitola 5

Testování a nasazení zásuvného modulu

V této kapitole je nejdříve popsán postup při testování modulu a zajímavé problémy, které jsem při testování objevil. Poté je krátce popsán způsob konfigurace a samotného nasazení modulu do Roundcube.

5.1 Testování

Při psaní kódu logiky jsem souběžně plnil i sadu automatických (jednotkových) testů pomocí nadstavby `PHPUnit`. Ty plnily dvojí účel – samozřejmě ověřovaly funkčnost kódu jako takového a zároveň ověřovaly moje pochopení fungování nástroje Procmail, jelikož část testů spouští Procmail přímo nad generovanými testovacími pravidly a potom ověřuje výsledek nad obsahem testovací schránky. Automatické testy byly zaměřeny pouze na logické jádro, kde pokrývají 90 % řádků kódu napříč 130 testy, které se mimo jiné zaměřují na následující položky:

- Vstupy podmínek – korektní ošetření únikových znaků, páry závorek, netisknutelné znaky, ...
- Generování a zpětná identifikace regulárních výrazů pro kontrolu rozsahů dat
- Korektní funkčnost jednotlivých operátorů a jednotlivých akcí
- Korektní generování bloků pravidel pro kombinace podmínek v disjunkci či konjunkci a akcí
- Korektní identifikace nejrůznějších kombinací akcí a podmínek při zpětném čtení a rozlišení filtru od dovolené
- Detekce překryvu rozsahu dat u dovolených
- Práce nad úložištěm – neplatný kontrolní součet nebo jinak deformovaná sekce, neplatné spojení, zápis do souboru s již existujícími pravidly a jejich zachování, tvorba zálohy apod.

Po otestování logického jádra přišlo na řadu napojení této logiky na API pro zásuvné moduly samotného Roundcube. Správné chování a vzhled jednotlivých prvků jsem ověřoval

ručně v rámci mého vývojového prostředí z kontejnerů platformy Docker. Dokumentace na wiki stránce projektu nebyla vždy úplná, často jsem musel ručně procházet zdrojový kód Roundcube a krokovat jej pomocí ladících nástrojů.

Po ověření, že lokálně se modul chová jak má, jsem pro jistotu ještě provedl ověření, jak se některá pravidla chovají přímo v prostředí FIT VUT. Pravidla generovaná zásuvným modulem jsem ručně vykopíroval na server a posíláním pošty z osobní adresy jsem ověřoval jejich správný výsledek, zejména pak u složitějších konstrukcí, jako např. bezpečné přeposlání, dekódování nebo dovolená.

Právě u dovolené se projevila chyba pouze v prostředí fakultního serveru, nikoliv však v prostředí vývojovém. V příkazu pro kontrolu mezipaměti používám program awk, kterému je předáván jako pojmenovaný argument odesílatel e-mailu. Program v prostředí operačního systému FreeBSD vždy vracel chybu kvůli neplatnému argumentu, ačkoliv jinde fungoval správně. Ukázalo se, že ve verzi programu awk, která je na serveru dostupná, je chyba[1], která se projeví, pokud hodnota argumentu začíná znakem `=`, což by měla být validní hodnota. Tento znak se do pole odesílatele dostane, pokud obsahuje kromě adresy i jméno a to je kódováno třeba pomocí UTF-8. Hodnota hlavičky pak začíná takto: `=?utf-8...` V tomto případě tedy příkaz kontroly mezipaměti chybně vrátí nenulovou hodnotu a automatická odpověď se nikdy neodešle. Řešením bylo přidání znaku `*` před hodnotu hlavičky odesílatele, což na funkčnost nemá vliv a lze tak zásuvný modul používat i s touto verzí programu awk.

Poslední částí testování bylo testování uživatelského rozhraní samotnými uživateli. Zásuvný modul byl nasazen do Roundcube na FIT VUT, kde ho nejdříve testovala užší skupina lidí. Výstupem tohoto testování bylo zjištění několika nedostatků – přednastavená pole byla pro pokročilejší uživatele nedostatečná, což jsem vyřešil přidáním možnosti zadat i vlastní hlavičku do podmínky.

Dále bylo odhaleno chybné kódování vstupní hodnoty podmínky s regulárním výrazem, ty jsou v PHP implementovány na základě jiného standardu než jaký je použit pro Procmail a mají tedy mírně odlišnou sadu metaznaků. Pro vyřešení problému jsem místo knihovních funkcí jazyka PHP použil pro ošetření metaznaků vlastní funkce pracující se správnou sadou a také jsem doplnil testovací sadu i o tyto případy.

U dovolených se objevil ještě další problém, který se projevil při přijetí e-mailu, u kterého neprošla kontrola pomocí mezipaměti e-mailů. Pravidlo s kontrolou vytvářelo kopii, která se odeslala, pokud byla kontrola úspěšná. V opačném případě se však odesílací pravidlo neprovedlo a kopie pokračovala dál, což mělo za následek dvojí doručení. Řešením bylo přidání záchytného pravidla, které tuto uniklou kopii zahodí. To lze vidět i na příkladu dovolené v příloze B.

5.2 Nasazení a konfigurace

Prvním krokem je konfigurace modulu, kterou čte ze souboru `config.inc.php`. Je tedy třeba nejdříve takto přejmenovat přiložený soubor `config.inc.php.dist`. Samotná konfigurace zásuvného modulu je minimální, obsahuje pouze jediný parametr – adresu úložiště. Ta může být zadána explicitně nebo lze využít jednoho ze zástupných symbolů podobně jako u konfigurace hostitele protokolu SMTP v Roundcube[2]:

- `%h` – název hostitele protokolu IMAP pro aktuálního uživatele

[1] https://bugs.freebsd.org/bugzilla/show_bug.cgi?id=217553
[2] https://github.com/roundcube/roundcubemail/wiki/Configuration#sending-messages-via-smtp

- `%n` – název hostitele Roundcube (např. mail.domain.com)
- `%t` – název hostitele Roundcube bez 1. části (např. domain.com)
- `%d` – doména hostitele z požadavku protokolu HTTP (může být alias)
- `%z` – doména hostitele protokolu IMAP

Instalace modulu je standardní – nakopírování modulu do složky `plugins/` a povolení modulu v konfiguraci Roundcube.

Důležitá je však především správná konfigurace samotného serveru, protože předpokladem pro správnou funkčnost modulu je, že adresa úložiště odpovídá úložišti účtů protokolu IMAP, server je přístupný protokolem SFTP, lze se na něj přihlásit stejnými údaji jako k Roundcube, kořenový adresář je domovský adresář uživatele a Procmail je správně nakonfigurovaný pro filtrování příchozí pošty. Pokud jsou tyto podmínky splněny, tak by měl zásuvný modul fungovat bez problémů.

Kapitola 6

Závěr

Cílem této práce bylo vytvoření zásuvného modulu pro Roundcube, umožňujícího správu filtrů pošty a automatické oznamování nepřítomnosti, to vše za pomoci filtrovacího nástroje Procmail. Tento cíl byl naplněn a zásuvný modul je již nyní nasazen přímo na instanci Roundcube na Fakultě informačních technologií VUT, kde je dostupný všem uživatelům.

Výsledný modul umožňuje vytvářet filtry, jakožto sady podmínek a akcí, které umožňují třídit poštu do složek, přeposílat na další adresy či poštu zahazovat. To vše je prezentováno uživateli pomocí uživatelského rozhraní přímo v Roundcube, jenž je vzhledově podobné a kompatibilní s oběma standardními tématy vzhledu. V případě responzivního Elastic jej může uživatel pohodlně používat i z mobilního zařízení. Kromě filtrování modul poskytuje i funkčnost pro oznamování nepřítomnosti (dovolená), umožňující naplánovat ji dopředu, připojit automatickou odpověď a nastavovat, jak často je automatická odpověď během této dovolené odesílána.

I když modul splňuje zadané požadavky a dosahuje kýženého výsledku, jsou stále možnosti, jak lze zásuvný modul rozšířit – zajímavou funkcí by byla například možnost připojení vícejazyčné automatické odpovědi u dovolené, která by odesílala automatickou odpověď v jazyce původního odesílatele. To by však vyžadovalo vymyslet mechanismus pro automatickou detekci jazyka příchozí zprávy.

Dalším zajímavým rozšířením by bylo vytvoření komponenty pro optimalizaci pravidel, která by dokázala pravidla zjednodušit nebo detekovat nemožné kombinace. Možnost zadávat podmínky pomocí regulárních výrazů přidává této úloze na složitosti a zajímavosti.

Tuto práci jsem si zvolil, neboť jsem neměl moc velkou zkušenost s tvorbou webových aplikací a s tím spojenými technologiemi. Bral jsem ji jako možnost naučit se něco nového v tomto odvětví a zároveň možnost udělat něco, co přímo zjednoduší každodenní život lidem používajícím Roundcube, což doufám, že bude platit i pro lidi nejen na fakultě, jelikož chci zásuvný modul zdarma nabídnout i širší komunitě kolem Roundcube a dále jej udržovat.

Literatura

[1] Achour, M., Betz, F., Dovgal, A. et al. *PHP: History of PHP – Manual* [online]. 2020 [cit. 2020-05-06]. Dostupné z: https://www.php.net/manual/en/history.php.php.

[2] Achour, M., Betz, F., Dovgal, A. et al. *PHP: Preface – Manual* [online]. 2020 [cit. 2020-05-06]. Dostupné z: https://www.php.net/manual/en/history.php.php.

[3] Berg, S. R. van den a Guenther, P. A. *Formail(1): mail formatter - Linux man page* [online]. 2020 [cit. 2020-05-11]. Dostupné z: https://linux.die.net/man/1/formail.

[4] Berg, S. R. van den a Guenther, P. A. *Procmailrc(5) – Linux man page* [online]. 2020 [cit. 2020-05-03]. Dostupné z: https://linux.die.net/man/5/procmailrc.

[5] Castro, E. a Hyslop, B. *HTML5 a CSS3*. Albatros Media a.s., 2015. ISBN 9788025144664.

[6] Dent, K. *Postfix: kompletní průvodce*. Grada Publishing, 2005. ISBN 9788024710297.

[7] Galbraith, J. a Saarenmaa, O. *SSH File Transfer Protocol*. Internet-Draft draft-ietf-secsh-filexfer-13. Internet Engineering Task Force, červenec 2006. Work in Progress. Dostupné z: https://datatracker.ietf.org/doc/html/draft-ietf-secsh-filexfer-13.

[8] Gaďorek, P. a Michal, B. *Přesměrování a třídění elektronické pošty* [online]. 2020 [cit. 2020-06-02]. Dostupné z: https://www.fit.vut.cz/units/cvt/net/procmail.php.cs.

[9] Gelbmann, M., Delamer, A. et al. *Usage Statistics and Market Share of Server-side Programming Languages for Websites, May 2020* [online]. 2020 [cit. 2020-05-06]. Dostupné z: https://w3techs.com/technologies/overview/programming_language.

[10] Gelbmann, M., Delamer, A. et al. *Web Technologies Statistics and Trends* [online]. 2020 [cit. 2020-05-06]. Dostupné z: https://w3techs.com/technologies.

[11] Grover, D. a Kunduru, H. *ES6 for Humans: The Latest Standard of JavaScript: ES2015 and Beyond*. Apress, 2017. ISBN 9781484226230.

[12] Heinlein, P. *Dovecot: POP3/IMAP Servers for Enterprises and ISPs*. CreateSpace Independent Publishing Platform, 2016. ISBN 9781534895706.

[13] Johnston, S., Morlhon, J.-L. de, Graham, J. et al. *What is a Container? | App Containerization | Docker* [online]. 2020 [cit. 2020-05-06]. Dostupné z: https://www.docker.com/resources/what-container.

[14] Kerrisk, M. The Linux programming interface : a Linux and UNIX system programming handbook. In:. San Francisco: No Starch Press, 2010, kap. 3. ISBN 978-1-59327-220-3.

[15] Klensin, D. J. C. *Simple Mail Transfer Protocol* [RFC 2821]. RFC Editor, 1. duben 2001. DOI: 10.17487/RFC2821. Dostupné z: https://rfc-editor.org/rfc/rfc2821.txt.

[16] Klyne, G. a Palme, J. *Registration of Mail and MIME Header Fields* [RFC 4021]. RFC Editor, březen 2005. DOI: 10.17487/RFC4021. Dostupné z: https://rfc-editor.org/rfc/rfc4021.txt.

[17] Laurie, B. a Laurie, P. *Apache: The Definitive Guide: The Definitive Guide, 3rd Edition*. O'Reilly Media, 2002. ISBN 9781449366544.

[18] Lie, H. a Bos, B. *Cascading Style Sheets: Designing for the Web, Portable Documents*. 3. vyd. Pearson Education, 2005. ISBN 9780132465731.

[19] Machniak, A., Benincasa, V. a Brüderli, T. *Plugin API · roundcube/roundcubemail Wiki* [online]. 2020 [cit. 2020-05-02]. Dostupné z: https://github.com/roundcube/roundcubemail/wiki/Plugin-API.

[20] Machniak, A., Benincasa, V. a Brüderli, T. *Roundcube Webmail* [online]. 2020 [cit. 2020-05-01]. Dostupné z: https://roundcube.net/.

[21] Machniak, A., Benincasa, V. a Brüderli, T. *Skin Markup · roundcube/roundcubemail Wiki* [online]. 2020 [cit. 2020-05-02]. Dostupné z: https://github.com/roundcube/roundcubemail/wiki/Skin-Markup.

[22] Mozilla Contributors. *<input type="date» – HTML: Hypertext Markup Language | MDN* [online]. 2020 [cit. 2020-05-18]. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/HTML/Element/input/date#Browser_compatibility.

[23] Nagar, S. *Introduction to Python for Engineers and Scientists: Open Source Solutions for Numerical Computation*. Apress, 2017. ISBN 9781484232040.

[24] Resnick, P. *Internet Message Format* [RFC 2822]. RFC Editor, 1. duben 2001. DOI: 10.17487/RFC2822. Dostupné z: https://rfc-editor.org/rfc/rfc2822.txt.

[25] Wikipedia contributors. *Procmail — Wikipedia, The Free Encyclopedia* [online]. 2020 [cit. 2020-05-02]. Dostupné z: https://en.wikipedia.org/w/index.php?title=Procmail&oldid=954413243.

Příloha A

Ukázka dekódovaní e-mailu v Procmailu za pomocí jazyka Python

```
#DECODE_BLOCK_START
LINEBUF=32000

HEADER_D=‘python3 -c "\‘cat <<EOF
import sys;import email;from email import policy;

mail = email.message_from_binary_file(sys.stdin.buffer,
    policy=policy.default)

for header, value in mail.items():
    print(header+’: ’, end=’’, flush=True)
    for text, encoding in email.header.decode_header(value):
        if encoding is None: encoding = ’utf-8’
        if not isinstance(text, str): text = str(text, encoding)
        text += ’\\n’
        sys.stdout.buffer.write(text.encode(’utf-8’))
    sys.stdout.flush()
EOF\‘"‘

BODY_D=‘python3 -c "\‘cat <<EOF
import sys;import email;from email import policy;

mail = email.message_from_binary_file(sys.stdin.buffer, policy=policy.
    default)

sys.stdout.buffer.write(mail).get_body().get_content().encode(’utf-8’))
print()
EOF\‘"‘
#DECODE_BLOCK_END
```

Příloha B

Příklad bloku pravidel tvořících dovolenou

```
#START:Egypt
:0c:.rubik.lock
* H ??! (^FROM_DAEMON *())
* H ??! (^FROM_MAILER *())
* H ??! (^(X-Loop): *(.*autoreply@rubik_filter.*) *$)
* H ?? (^(Date): *((Mon|Tue|Wed|Thu|Fri|Sat|Sun), ((((0?5|0?6|0?7|0?8) May)
    ) 2020).*))
{

:0Wc
| formail -x "From:" | (read EMAIL; NOW=$(date +%s); touch ".
    rubik_vacation_cache"; awk -v name="*$EMAIL" -v now="$NOW" -v diff=$(
    expr $NOW - 86400) '{if (index($0,name)) {if ($NF > diff) {exit 1}}
    else {print $0}} END{print name" "now}' ".rubik_vacation_cache" > ".
    rubik_vacation_cache.tmp" && mv ".rubik_vacation_cache.tmp" ".
    rubik_vacation_cache" || (rm ".rubik_vacation_cache.tmp" && exit 1))

:0Wa
| (formail -rt -A "X-Loop: autoreply@rubik_filter" -i "Content-Transfer-
    Encoding: quoted-printable" -i "Content-Type: text/plain; charset=utf
    -8" ; echo -e "Jsem na dovolene,=0A=0Anapisu az se vratim.=0A=0AAdam";)
     | $SENDMAIL -t -oi

:0
/dev/null

}
#END:Egypt
```