VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

MONITOROVÁNÍ SÍŤOVÉHO PROVOZU S IDENTIFIKÁTORY OBSAHUJÍCÍMI NÁRODNÍ ABECEDY

INTERNATIONAL ALPHABETS IN NETWORK TRAFFIC MONITORING

DIPLOMOVÁ PRÁCE
MASTER´S THESIS

AUTOR PRÁCE
AUTHOR

Bc. Irena Talašová

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. Libor Polčák, Ph.D.

BRNO 2018

Abstrakt

Tato práce se zabývá implementací podpory síťových identifikátorů obsahujících národní znaky v software projektu SProbe, který je navržen pro monitorování síťových toků. Bylo nutno otestovat a zhodnotit stav vybraných síťových aplikací pracujících nad protokoly SMTP, POP3, IMAP, FTP nebo SIP. Na základě získaných informací bylo navrženo rozšíření stávajícího software projektu SProbe o modul pro podporu monitorování síťového provozu se znaky národních abeced. Navržené řešení bylo následně naimplementováno a důkladně byla otestována funkčnost celého výsledného systému. Dále je v této práci navržena a implementována úprava testovacího prostředí pro rychlejší a efektivnější testování systému se zaměřením na testy obsahující diakritiku.

Abstract

This work deals with the implementation of support for network identifiers containing national characters in the SProbe software project, which is designed to monitor network flows. It was necessary to test and evaluate the status of selected network applications working over SMTP, POP3, IMAP, FTP or SIP protocols. On the basis of the information obtained, an extension of the current software project SProbe was proposed - a module to support network traffic monitoring with national alphabet characters. The proposed solution was then implemented and the functionality of the entire resulting system was tested. In addition, this work proposes and implements a modification of the test environment for faster and more efficient testing of the system focusing on tests containing diacritics.

Klíčová slova

Monitorování, síťový provoz, UTF-8, národní abeceda, počítačová síť, IDN, EAI, SIP, SMTP, POP3, IMAP, FTP.

Keywords

Monitoring, network traffic, UTF-8, national alphabet, computer network, IDN, EAI, SIP, SMTP, POP3, IMAP, FTP.

Citace

TALAŠOVÁ, Irena. Monitorování síťového provozu s identifikátory obsahujícími národní abecedy. Brno, 2018. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Libor Polčák, Ph.D.

Monitorování síťového provozu s identifikátory obsahujícími národní abecedy

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracovala samostatně pod vedením Ing. Libora Polčáka, Ph.D. Uvedla jsem všechny literární prameny a publikace, ze kterých jsem čerpala.

……
Irena Talašová
1.5. 2018

Poděkování

Ráda bych poděkovala svému vedoucímu, Ing. Liboru Polčákovi, Ph.D., za jeho odborné rady a trpělivou pomoc během práce na diplomové práci.

Obsah

1 Úvod

Současné tendence směřují k tomu, aby uživatelé informačních technologií mohli používat znaky své národní abecedy v síťových protokolech. Změny jsou patrné nejen v RFC specifikujících používané síťové protokoly, ale i v samotných aplikacích. Jelikož lze RFC chápat spíše jako soubor doporučení a ne závazné pokyny pro implementaci programů, často existuje rozdíl v podpoře diakritiky protokolů v rámci RFC a v rámci aplikací pracujících s tímto protokolem. Pro monitorování síťového provozu zahrnujícího znaky národních abeced je důležité zjistit tento rozdíl a navrhnout takové řešení rozšíření stávajícího monitorovacího systému, které pokryje a nalezne co největší část hledaných identifikátorů s diakritikou.

Cílem této práce je zjistit, jaký je aktuální stav podpory národních abeced v aplikacích pracujících se síťovými protokoly a navrhnout takové rozšíření, které bude mít co nejmenší nároky na paměť a výpočetní výkon systému a zároveň zachytí co největší počet identifikátorů s diakritikou, které se mohou v síti objevit.

V druhé kapitole práce jsou uvedeny základní principy a charakteristiky síťových protokolů, které jsou význačné pro tuto práci.

Třetí kapitola pojednává o podpoře diakritiky a národních znaků ve specifikacích RFC vybraných síťových protokolů a to z historického i aktuálního hlediska. Jelikož se velmi často vývojáři síťových aplikací nedrží doporučení RFC, je třeba především provést vlastní rešerši a zjistit stav aktuálního relevantního software, a to jak serverového, tak i klientského; u některých protokolů také služeb poskytovatelů.

Čtvrtá kapitola práce se zabývá softwarovou architekturou projektu SProbe, jehož rozšíření o modul pro podporu zachytávání síťových identifikátorů se znaky národních abeced je hlavním cílem a výsledkem této práce. Je zde uveden princip fungování jednotlivých částí systému, komunikace mezi hardware a software a také způsob, jakým je přidán nový odposlech a jak dojde k jeho konfiguraci na sondu.

Pátá kapitola představuje konkrétní návrh na rozšíření systému o podporu síťových identifikátorů. Budou rozebrány jednotlivé části systému a nutné změny, které se v nich budou muset provést.

Šestá kapitola práce je věnována implementaci navrženého rozšíření. Popisuje detailnější způsob řešení a je doprovozena fragmenty kódu implementační části.

Sedmá kapitola se zabývá testováním. Zmiňuje dva druhy testování. První testování probíhalo na PCAP souborech s reálnými pakety obsahujícími diakritiku. Druhý způsob testování bylo testování pomocí automatických testovacích skriptů, u kterých bylo nutné pro vyšší efektivitu testování navrhnout změny a následně implementovat.

V osmé kapitole je shrnutí této diplomové práce. Jsou zde uvedeny dosažené výsledky a následně nastíněny možnosti dalšího vývoje software.

2 Komunikační a síťové protokoly

V této kapitole budou představeny principy některých nezabezpečených komunikačních a síťových protokolů se zaměřením na ty, které budou dále v této práci zkoumány a testovány. Nejprve budou popsány dva telefonní protokoly, protokol H.323 a SIP, dále ze souborových protokolů protokol FTP, nakonec budou popsány e-mailové protokoly, konkrétně protokoly SMTP, POP3 a IMAP.

2.1 Telefonní protokoly

Voice over Internet Protocol (VoIP) je technologie umožňující přenos digitalizovaného hlasu v těle paketů rodiny protokolů UDP/TCP/IP prostřednictvím počítačové sítě. Využívá se pro telefonování prostřednictvím Internetu nebo jiné datové sítě [1].

IP telefonie musí poskytovat dostatečné přenosové pásmo, kvalitní, dostupný a bezpečný přenos hlasu a v neposlední řadě také integrovatelnost s veřejnou telefonní sítí PSTN. Musí být schopná převést hlas na data, řídit komunikaci buď přes ústřednu nebo mezi telefony a připojit se do veřejné telefonní sítě. S řízením komunikace úzce souvisí schopnost registrovat účastníky, adresovat, směrovat a vytvářet hovory. Uplatňují se přenosové protokoly, které lze rozdělit do dvou skupin, a to signalizační (SIP, H.323, SCCP a další) a transportní (RTP/RTCP) [2].

Nutnou podmínkou pro srozumitelné a spolehlivé VoIP telefonní spojení je zajištění tzv. kvality služby, zkráceně označované QoS. Přestože existuje více různých telefonních protokolů, v následujících podkapitolách se zaměříme pouze na protokoly H.323 a SIP, které jsou pro tuto práci nezbytné.

2.1.1 Protokol SIP (Session Initiation Protocol)

Existuje mnoho internetových aplikací vyžadujících vytvoření a řízení spojení, kde spojením rozumíme výměnu dat mezi skupinou účastníků. Implementace takovýchto aplikací je ztížena chováním účastníků. Uživatelé se mohou připojovat z různých míst a různých koncových zařízení, mohou být adresováni rozličnými adresami a také mohou vést paralelně několik hovorů.

SIP svou funkcí doplňuje protokoly sloužící pro přenos multimediálních dat, jako například hlas nebo video. Dovoluje nalézt v síti koncová zařízení, která spolu chtějí komunikovat a ustavit mezi nimi spojení, které bude využito pro přenos dat. Jedná se tedy o síťový protokol pracující na síťové vrstvě L7. Dokáže navázat, modifikovat a ukončit spojení [3].

SIP pracuje nad UDP a je používaný pro signalizaci VoIP. Jeho základními funkcemi je registrace účastníků, navázání spojení, směrování hovorů a adresování pomocí SIP URI. SIP URI je SIP identita účastníka, kterou lze považovat za typ URI (Uniform Resource Identifier). SIP adresa je podobná e-mailové adrese. Obsahuje uživatelskou část a doménovou část za použití znaku „@“ jako oddělovače.

Základní prvky SIP architektury tvoří server UAS (User Agent Server) a uživatelský agent UAC. Zatímco adresování probíhá pomocí SIP URI, směrovací informace jsou uloženy v SIP hlavičce (*Via*, *Route* a další). Ve zprávách SIP typu *Invite* a *Ok* je přenášena data protokolu SDP obsahující informace potřebné pro přijímání hlasového toku. Pro transport samotných hlasových či obrazových dat se používá protokol RTP [4].

2.1.2 Protokol H.323

H.323 je podobně jako SIP protokol určený pro nastavení, správu a ukončení mediální relace. Jedná se o soubor standardů z ITU-T (mezinárodní telekomunikační unie), který definuje rozsáhlou sadu protokolů zajišťující zvukovou a vizuální komunikaci prostřednictvím počítačové sítě. Jedná se o poměrně starý protokol, který byl převážně nahrazen protokolem SIP (Session Initiation Protocol). H.323 je na rozdíl od SIP složitější a horší pro údržbu a rozšiřitelnost [5].

Pod záštitou jména H.323 rozumíme sadu protokolů. Jmenujme například signalizační protokol H.225, protokol H.235 pro zabezpečení nebo protokol RTP používající se pro přenos dat.

2.2 Souborové protokoly

Termínem přenos souborů rozumíme přenos souborů přes počítačovou sít, nejčastěji se jedná o Internet, ale data se mohou přenášet také například v rámci Intranetu. Pro přenos souborů existuje mnoho metod a protokolů, avšak zde bude zmíněn pouze jeden zástupce, pravděpodobně nejznámější, protokol FTP. Systém zajišťující přenos souborů se nazývá souborový server. Jeho protějškem na straně uživatele je souborový klient, který žádá souborový server o stažení nebo nahrání souboru na server.

2.2.1 Protokol FTP (File Transfer Protocol)

Protokol FTP se používá pro sdílení a přenos dat. Primárně je určen pro použití specializovanými programy, ale stejně dobře lze ovládat i prostřednictvím terminálu. Jedná se o jeden z nejstarších protokolů, který je určen pro sdílení vzdáleného souborového systému. Při jeho vývoji byl kladen důraz na jednoduchost a snadnost použití [6].

Jedná se o klient/server model, kde klient inicializuje spojení na portu 21 pomocí protokolu TCP. Po úspěšné autentizaci klient pomocí příkazů zasílaných přes vytvořené řídící spojení prochází vzdálený souborový systém. Samotný datový přenos následně probíhá pomocí plně duplexního datového kanálu. Typ přenášeného souboru může být ASCII, EBCDIC nebo binární. Datový kanál běží v aktivním režimu na portu 20 a iniciuje jej server, zatímco řídící kanál na portu 21 běží v pasivním režimu a je zahájen klientem [7].

2.3 E-mailové protokoly

E-mailové protokoly slouží pro přenos a doručení elektronické pošty mezi koncovými uživateli. Hlavními požadavky je především bezpečné doručení e-mailu, který může obsahovat text nebo jiná data, především dokumenty, obrázky nebo audio.

Architektura e-mailové komunikace se skládá z e-mailových serverů, které mezi sebou komunikují a během doručování si předávají e-maily pomocí protokolu SMTP. Tyto servery obsahují uživatelské e-mailové schránky, do kterých jsou e-maily doručovány. Druhým hlavním prvkem architektury jsou klientské aplikace a servery, které pomocí protokolu POP3 nebo IMAP manipulují s e-maily a tím umožňují uživateli přístup ke svým zprávám [8][9].

2.3.1 Protokol SMTP (Simple Mail Transfer Protocol)

Cílem protokolu SMTP je účinný a efektivní přenos e-mailů. Důležitým rysem SMTP protokolu je schopnost přenést a doručit e-mail napříč různými sítěmi. SMTP model zahrnuje klienta a server. V okamžiku kdy má klient zprávu, kterou by rád doručil, kontaktuje server a ustaví se obousměrný přenosový kanál. Pomocí příkazů klient přenese zprávu na SMTP server. Každý server může být buď cílovým serverem nebo pouze jedním ze spojů na cestě e-mailu.

Směrování e-mailu mezi servery probíhá díky hlavičce e-mailu. Adresa e-mailové schránky se skládá z uživatelské (lokální) části a doménové části oddělených znakem „@“. SMTP zpráva se skládá z adresy odesílatele, adresy příjemce (nebo adres příjemců) a priority. Samotná e-mailová zpráva se pak skládá z hlavičky a těla zprávy.

SMTP je textový protokol, avšak e-maily mohou obsahovat přílohy, tedy netextové části. Pro jejich přenos se používá kódování MIME, které se kromě protokolu SMTP uplatňuje také v protokolu HTTP.

Přenos zprávy probíhá ve čtyřech krocích. Nejprve dojde k navázání spojení tím, že se klient připojí na server. Následuje jeho identifikace a samotný přenos zprávy pomocí definovaných příkazů. Po ukončení komunikace je zpráva nahraná na SMTP server a začíná její putování mezi dalšími SMTP servery až k tomu cílovému, který obsahuje e-mailovou schránku adresáta [10].

2.3.2 Protokol POP3 (Post Office Protocol Version 3)

POP3 je internetový protokol zajišťující příjem pošty. Používá se pro stahování e-mailových zpráv ze serveru na klienta. Pro tuto komunikaci využívá TCP/IP spojení na portu 110. Naváže se TCP spojení mezi klientem a serverem a následně probíhá komunikace formou výměny příkazů a odpovědí.

Při použití tohoto protokolu se stáhnou všechny zprávy do klienta a na serveru obvykle dojde k jejich smazání. Uživatel si může prohlížet zprávy i poté, co se odpojí od serveru. Pro svou činnost požaduje výlučný přístup ke schránce [8].

2.3.3 Protokol IMAP (Internet Message Access Protocol)

IMAP je internetový protokol určený taktéž pro práci s e-mailovou schránkou. Od protokolu POP3 se liší hlavně tím, že je během práce trvale připojen pomocí TCP na portu 143 k poštovní schránce a umožňuje s jejím obsahem manipulovat z kteréhokoliv místa.

Změny se provádí přímo na serveru a do počítače jsou stahovány jen nejnutnější věci. Podporuje práci offline se zprávami a následnou synchronizaci. Komunikace mezi klientem a serverem probíhá podobně jako v případě protokolu POP3, tedy prostřednictvím příkazů a odpovědí na ně [9].

3 Podpora diakritiky v síťových protokolech a v reálných aplikacích

V této kapitole budou k vybraným síťovým protokolům popsány možnosti podpory diakritiky v doporučeních RFC. Velká část této kapitoly bude věnována také analýze reálných aplikací, jak na straně serverových aplikací, tak i na straně aplikací klientských, jakou měrou podporují práci s diakritikou, a poznatkům, které jsem touto analýzou shromáždila. Vznikne tak ucelený pohled na rozdílný stav podpory doporučené v RFC oproti podpoře v reálných aplikacích.

Získané informace budou pro tuto práci důležité, neboť je třeba vědět, jaké možnosti dnešní aplikace uživatelům poskytují, a tudíž jaké možnosti uživatelé mají pro to, aby skryli svou síťovou komunikaci před monitorovacími nástroji.

3.1 Definice základních pojmů

IDN (Internationalized Domain Names) je zkratkou označující mezinárodní doménová jména. Jedná se o metodu a standard, který umožňuje lidem na celém světě používat doménová jména ve svém rodném jazyce. Znaky doménového jména jsou kódovány jako UTF-8 řetězec [11].

Punycode je název pro kódovací syntaxi, umožňující překlad řetězce UTF-8 znaků do podoby ASCII řetězce. S tímto kódováním se lze setkat často v počítačových sítích například v DNS dotazech nebo e-mailových adresách.

Díky zavedení EAI (Email Address Internationalization) do poštovního softwaru je možné používat e-mailové adresy s národními znaky v lokální části e-mailové adresy [12]. Tyto dva základní pojmy a jejich zkratky budou používány v následujících kapitolách.

3.2 SIP

V následujících kapitolách bude uvedena podpora národních znaků v protokolu SIP dle příslušného RFC a dále stručně popsány poznatky o skutečném stavu implementace podpory v rámci některých testovaných SIP klientů, serverů a nejznámějších volně dostupných SIP poskytovatelů.

3.2.1 Podpora v SIP URI dle RFC

SIP je definován v RFC [3]. SIP je textově orientovaný protokol používající UTF-8 kódování. Formát polí hodnot hlavičky je definován jménem hlavičky. Jedná se o sekvenci UTF-8 oktetů nebo kombinaci bílých znaků, oddělovačů a řetězců ohraničených uvozovkami. Těla zpráv mohou používat různá kódování. Toto kódování by mělo být uvedeno v hodnotě pole hlavičky Content-Type.

SIP zprávy mohou obsahovat binární těla. Pokud odesílatel neuvede jiné, předpokládá se kódování UTF- 8.

UTF-8 znaky mohou být použity pouze pro hodnoty polí s popisným charakterem (například jméno a příjmení) a u těch polí, u kterých se nepředpokládá interpretování parserem zpráv [3] [4]. Jména volajících jsou kódována jako UTF-8 - lze tedy používat národní znaky. Doménová část URI musí obsahovat pouze vybrané ASCII znaky. Toto vyplývá z uvedené syntaxe SIP URI [3]. Uživatelská část SIP URI musí obsahovat pouze vybrané ASCII znaky případně sekvenci hexadecimálních znaků [3]. Jedná se o systém, kde se namísto znaku, který není ASCII, napíše řetězec "%xx", kde x je hexadecimální číslice označující příslušný byte původního znaku v UTF-8. Povolené ASCII řetězce pro doménovou i uživatelskou část lze nalézt v [3] (sekce 25.1).

V praxi ale každý SIP klient s adresou obsahující diakritikou naloží jinak.

3.2.2 Podpora na úrovni SIP serverů

Testovala jsem jeden opensource SIP server (Asterisk 14.5.0) a jeden komerční SIP server (Switchvox 6.1.2).

Prvním testovaným opensource nástrojem je server Asterisk[1]. Asterisk implementuje telefonní ústřednu (PBX – Private Branch Exchange) pomocí běžného počítače s OS Unix nebo Linux. Jedná se pravděpodobně o jedno z nejrozšiřitelnějších a nejflexibilnějších řešeních. Díky své nízké náročnosti a vysoké konfigurovatelnosti může běžet i na slabším HW zařízení. Výhodou je i to, že jej lze propojit s adresářovými službami a tím získat další funkcionalitu.

Druhým z testovaných severů je komerční nástroj Switchvox[2]. Jedná se o plně funkční systém sjednocené komunikace (UC – Unified Communications) založený na Asterisku. Je určen pro malé až střední podniky. Jde o levnější variantu oproti konkurenčním UC systémům, které jsou založené na proprietárních technologiích. Z hlediska administrátora a uživatelů je rozdělen do třech webových portálů - uživatelský, administrátorský a switchboard.

Switchvox byl testován prostřednictvím demo verze volně dostupné skrze internetový prohlížeč po určitý čas od registrace. Oproti tomu Asterisk byl nainstalován do virtuálního počítače se systémem Ubuntu a pomocí změn v konfiguračních souborech byla testována jeho podpora diakritiky v SIP adresách a účtech. K testování byl použit klient Zoiper, který byl připojen na účet vytvořený na Asterisku.

Prvním testem bylo vyzkoušení založení účtu uživatele s diakritikou a přihlášení pomocí tohoto účtu. Switchvox umožňuje zadat jméno a příjmení uživatele s diakritikou, avšak jeho číslo (extension) se může skládat pouze z číslic libovolného počtu dle nastavení v uživatelském portálu.

1 URL http://www.asterisk.org/

2 URL https://www.digium.com/products/business-phone-systems

Na Asterisku lze bez problému vytvořit jakéhokoliv uživatele. Pro testovací účely jsem založila účet „talašová@10.0.2.15“. Po úspěšném vytvoření účtu jsem se pokusila přihlásit tímto uživatelským účtem pomocí klienta Zoiper. Přihlášení proběhlo v pořádku.

Switchvox bez problému podporuje diakritiku ve jménech a příjmeních uživatelů. Proto i jejich vyhledávání a zobrazování v seznamu kontaktů nepředstavuje problém.

Další testovanou vlastností byla schopnost přijímat a provádět hovory ze založeného účtu s diakritikou. Pokusila jsem se tedy volat z účtu „talašová@10.0.2.15“ na účet bez diakritiky. Hovor probíhal v pořádku, stejně jako při volání na účet s diakritikou „odesílatel@10.0.2.15“. Přenášely se UTF-8 znaky.

Závěrečným testem bylo vyzkoušet jaké znaky se budou přenášet v závislosti na tom, jakým způsobem se zadá adresa volaného do klienta Zoiper. Nejdříve jsem vyzkoušela zavolat z účtu „odesílatel“ na účet „talašová“, do klienta zadáno jako „tala%c5%a1ov%c3%a1“, a posléze jako „tala %c5%a1ová“. V obou dvou případech Asterisk hovor přijal a diakritika se přenášela tak, jak byla zadána – tedy v hexadecimální podobě nebo v kombinaci s UTF-8 znaky.

Ze získaných poznatků je vidět, že zatímco komerční nástroj Switchvox diakritiku nepodporuje, Asterisk si s ní poradil ve všech případech. Na závěr je uvedena přehledová tabulka získaných výsledků. Níže v tabulce 3.1 je shrnutí provedených analýz, detaily analýz lze nalézt v příloze 1.

	Jména uživatelů s diakritikou	**Vyhledávání kontaktů s diakritikou**	**Založení účtu uživatele s diakritikou**	**Volání na účet s diakritikou**
Switchvox	Ano	Ano	Ne	Ne
Asterisk	Ano	-	Ano	Ano

Tabulka 3.1: Shrnutí analýz podpory diakritiky na úrovni SIP serverů

3.2.3 Podpora na úrovni SIP klientů

Tato podkapitola obsahuje informace nashromážděné analýzou podpory diakritiky u některých volně dostupných SIP klientů. Jedná se o klienty Zoiper (verze 3.3), Ekiga (verze 4.0), Linphone (verze 4) a SFLPhone (verze 1.3.0). Testování probíhalo za pomoci serveru Asterisk a uživatelského účtu „talašová@10.0.2.15“ na něm vytvořeného.

Prvním testem analýzy bylo otestování schopnosti přihlásit se pomocí účtu s diakritikou. Zatímco klient Zoiper s tímto neměl problém, stejně jako klient SFLPhone; Ekiga a Linphone nedokáží takový požadavek zpracovat.

Druhým testem analýzy bylo otestování chování klienta při volání na adresu obsahující v uživatelské části diakritiku. Chování klienta Zoiper bylo zmíněno v předchozí podkapitole při

analýze serveru Asterisk. Ekiga se chová poměrně jednodušeji. Je jedno, jestli zadáme adresu jako UTF-8 znaky, hexadecimální znaky nebo jejich kombinaci, vždy se chová stejně. Znaky mimo ASCII převede do hexadecimální podoby a uskuteční hovor. Obdobně se choval i SFLPhone a Linphone. Problém potom představují klienti jako Zoiper, kteří adresu kódují různě podle toho, jak ji zadá uživatel, a tím pádem existuje spousta způsobů jak ji zapsat pomocí kombinací UTF-8 znaků a jejich hexadecimální podoby.

Třetím testem analýzy bylo zjištění chování klientů při volání na adresu s diakritikou v doménové části adresy. Zoiper naložil se znaky podobně jako s těmi v uživatelské části. S uskutečněním hovoru neměl problém a zasílal DNS dotaz na adresu ve formátu, jak ji zadal uživatel. Adresa „talaš“ se přenáší jako „ tala\305\241“, „xn—tala-j6a.net“ jako „ xn—tala-j6a.net“ a „tala%c5%a1ová“ jako „tala%c5%a1ov\303\241“. Chování klienta Ekiga je jiné. Při všech způsobech zadání dojde k překódování do UTF-8 podoby a následné zaslání DNS dotazu. Pouze při zadání doménové části ve tvaru Punycode tento zůstane. Obdobně také Linphone a SFLPhone.

Čtvrtým a posledním testem analýzy bylo vytváření kontaktů s diakritikou v adresáři a jejich následné vyhledávání. Vytvářet takové kontakty zvládli všichni klienti, ale s vyhledáváním měl problém klient Ekiga a Linphone.

Práce klientů s diakritikou a národními znaky je různá. Je patrné, že na rozdíl od doporučení ve specifikaci RFC, podpora u klientů a serverů je poměrně velká. Níže v tabulce 3.2 je shrnutí provedených analýz, detaily analýz lze nalézt v příloze 2.

	Přihlášení účtem s diakritikou	**Volání na adresu s diakritikou**	**Vytváření jmen kontaktů s diakritikou**	**Vyhledávání v adresáři s diakritikou**
Zoiper	Ano	Ano	Ano	Ano
Ekiga	Ne	Ano	Ano	Ne
Linphone	Ne	Ano	Ano	Ne
SFLPhone	Ano	Ano	Ano	Ano

Tabulka 3.2: Shrnutí analýz podpory diakritiky na úrovni SIP klientů

3.2.4 Podpora u bezplatných SIP poskytovatelů

Velká část uživatelů protokolu SIP získala vlastní SIP URI na základě požadavku a registrace u některého z bezplatných SIP poskytovatelů. Je důležité zjistit, jestli nejznámější SIP poskytovatelé nabízí možnost vytvořit SIP URI s diakritikou nebo národními znaky, aby bylo dále možné určit řetězce, které se při monitorování síťového provozu mohou objevit. Za tímto účelem jsem u některých z poskytovatelů zkoušela vytvořit účet s diakritikou v uživatelské i doménové části.

Prvním testem bylo zvolení doménové části adresy s diakritikou. Ve všech testovaných případech, tedy u IPTel, Ekiga, SIP2SIP, Antisip, nešlo doménovou část zvolit vůbec. Bylo to možné

pouze u Onsip, kde ale diakritika nešla zadat. Adresa obsahuje omezení na malé znaky anglické abecedy a čísla.

Žádný z testovaných bezplatných SIP poskytovatelů nenabízel možnost založit si adresu s uživatelskou částí obsahující diakritiku. Výjimkou je Ekiga, kde se, předpokládám, jednalo spíše o chybu. Ostatní klienti vždy uplatňují omezení, nejčastěji na písmena anglické abecedy, čísla a občas také pomlčku, tečku a podtržítko. Ve výsledku lze tedy říci, že testovaní bezplatní SIP poskytovatelé diakritiku neumožňují. Jednotlivé testy lze nalézt v příloze 3.

3.3 FTP

V následujících podkapitolách bude rozebrána podpora národních abeced v RFC pro protokol FTP a především jeho podpora v testovaných FTP serverech a klientech. Při testování reálných aplikací se zaměříme na použití diakritiky v názvech souborů přenášených pomocí FTP klientů a serverů a také na použití diakritiky v uživatelských jménech a heslech uživatelů FTP.

3.3.1 Podpora v protokolu FTP dle RFC

File Transfer Protocol je jeden z nejstarších a nejpoužívanějších protokolů internetu. Původní znaková sada tohoto protokolu byla 7-bit ASCII. V dnešní době se však vyskytuje potřeba používat národní znaky. Původní FTP specifikace byla rozšířena o podporu různých znakových sad. Pro přenos textových souborů obsahujících národní znaky (UTF-8 kódování) se nepoužívá ASCII mód, ale binární mód. UTF-8 znaky uvnitř souboru tedy nejsou pro FTP problém. RFC doporučuje, aby FTP klienti i servery používali kódování UTF-8 pro přenos názvů souborů a cest k nim. Díky tomuto opatření by také nedocházelo k nedorozuměním mezi různými FTP klienty a servery zapřičiněným odlišným kódováním [13].

3.3.2 Podpora na úrovni FTP serverů

Testované FTP servery (kromě serveru Wedos) byly nainstalovány do virtuálního počítače a následně testovány pomocí různých klientských FTP aplikací. Testovala jsem servery eFTP server (verze 3.2.3.112), Cerberus FTP server (verze 8), FileZilla sever (verze 0.9.60.2) a pro zajímavost jsem také otestovala jeden webový hosting od Wedosu[3].

Prvním testem bylo nahrání a následné stažení souboru s diakritikou („řčš.txt“) na testovaný server. Pro server eFTP jsem použila klientskou aplikaci eFTP klient. Upload i download souboru se zdařil a výsledné kódování bylo správné. Se stejným výsledkem dopadly testy Cerberus FTP server a Cerberus FTP klient, FileZilla server a FileZilla klient a hosting Wedos v kombinaci s klienty gFTP, FileZilla klient, Dolphin, FireFTP, Konqueror, Total Commander a další.

Druhým testem byla možnost přihlásit se na server pomocí účtu a hesla obsahující diakritiku. S tímto neměly problém servery eFTP, FileZilla a Cerberus, pouze hosting Wedos toto neumožňuje.

3 URL https://hosting.wedos.com/cs/

Posledním a třetím testem byl pokus nahrát a následně stáhnout soubor s diakritikou pomocí klienta od jiného tvůrce než je tvůrce serveru. Při použití Cerberus FTP serveru a klienta FileZilla se nevyskytl žádný problém ani v kódování souboru. Avšak při použití FileZilla serveru a eFTP klienta měly soubory špatné kódování.

FTP servery obecně nemají problém přenášet soubory obsahující diakritiku. Přestože jsem nenašla server, který by s tím problém měl, teoreticky existovat může. Stejně tak je tomu u FTP klientů. Avšak často se vyskytuje problém nestejného kódování souborů při použití odlišeného FTP klienta a FTP serveru, jako tomu bylo například u testu použití FileZilla server a eFTP klienta. Zde byl eFTP klient nainstalován na OS Windows a název souboru byl eFTP klientem přenesen jako ANSI, zatímco FileZilla měla nastavené kódování UTF-8. Je dobré používat stejný FTP server a stejného FTP klienta (FileZilla server + FileZilla klient, eFTP server + eFTP klient, Cerberus server + Cerberus klient atd.) a hlídat, že se vše kóduje UTF-8, je-li to možné.

Shrnutí podpory diakritiky jednotlivých FTP serverů je k nalezení v tabulce 3.3. Detaily jednotlivých testů lze nalézt v příloze 4.

	Přenos souboru s diakritikou	**Přihlášení pomocí účtu s diakritikou**
eFTP	Ano	Ano
Cerberus	Ano	Ano
FileZilla	Ano	Ano
Wedos	Ano	Ne

Tabulka 3.3: Shrnutí analýz podpory diakritiky na úrovni FTP serverů

3.3.3 Podpora na úrovni FTP klientů

Testovala jsem klienty Dolphin (verze 4.14.3), FileZilla (verze 3.26.2), FireFTP (verze 2.0), gFTP (verze 2.0.19), Konqueror (verze 4.14.3), Total Commander (verze 9.0a), WS_FTP (verze 12.5), e- FTP (verze 3.2.3.112) a Cerberus (verze 8). Testování probíhalo tak, že klienti byli připojeni na FTP server od Wedosu nebo na vlastní eFTP server. Na server jsem nahrávala a naopak stahovala z něj různé soubory obsahující diakritiku v názvu souboru. Cílem bylo zjistit, jestli přenos souborů proběhne úspěšně a jaké bude kódování souborů.

Prvním testem klientů bylo, jestli dovolují zadat uživatelské jméno a heslo s diakritikou. S tímto nebyl problém u žádného testovaného klienta, tedy Dolphin, FileZilla, FireFTP, gFTP, Konqueror, Total Commander, WS_FTP, eFTP a Cerberus.

Pomocí výše zmíněných klientů byl na FTP server od Wedosu nahrán a poté stažen soubor „talašová.txt“ všechny pokusy dopadly úspěšně a soubory měly správné kódování.

Přenos souboru s diakritikou v názvu nebyl problém u žádného z klientů, pokud byl daný soubor vytvořen a následně přesouván za pomoci toho samého klienta. Chyby v kódování se občas

vyskytly, pokud bylo použito různých klientů (například pomocí jednoho dojde k uložení souboru na server a pomocí druhého zobrazím či stáhnu tento soubor).

Klienti obsahují podporu pro různá kódování názvů souborů. Jak bude soubor kódován, je rozhodnuto různými způsoby. Klient gFTP například detekuje kódování ze seznamu nastavených znakových sad vzdáleného názvu souboru[4], zatímco FireFTP nabízí funkci Server Encoding pro změnu kódování souboru, dokud si se serverem nerozumí[5]. Obecně se tedy jedná o manuální nebo automatické nastavení kódování. Výchozí a doporučené kódování by však mělo být UTF-8, což je třeba pohlídat, pokud používáme různé FTP klienty a servery. Shrnutí provedených testů lze nalézt v tabulce 3.4. Detaily jednotlivých provedených testů jsou uvedeny v příloze 5.

	Přenos souboru s diakritikou	**Přihlášení pomocí účtu s diakritikou**
Dolphin	Ano	Ano
FileZilla	Ano	Ano
FireFTP	Ano	Ano
gFTP	Ano	Ano
Konqueror	Ano	Ano
Total Commander	Ano	Ano
WS_FTP	Ano	Ano
eFTP	Ano	Ano
Cerberus	Ano	Ano

Tabulka 3.4: Shrnutí analýz podpory diakritiky na úrovni FTP klientů

3.4 SMTP

Následující podkapitoly se věnují problematice podpory národních abeced v protokolu SMTP. Bude zmíněna podpora podle RFC i aktuální implementační stav některých testovaných SMTP serverů a v závěru také freemailových služeb.

3.4.1 Podpora v protokolu SMTP dle RFC

Původní specifikace SMTP jako u mnohých jiných protokolů nepodporovala jiné než ASCII znaky. Změnu přineslo až SMTP rozšíření [14], které dovoluje UTF-8 znaky v e-mailových hlavičkách a adresách. Servery, které implementují toto rozšíření, dávají tuto skutečnost najevo odpovědí SMTPUTF8 po zaslání příkazu EHLO. Podle specifikace se nesmí zasílat hlavičky obsahující UTF-8 řetězec serveru, který nepodporuje SMTPUTF8 rozšíření. E-mail je zahozen, pokud nelze nalézt

4 URL https://www.gftp.org/changelog.html

5 URL http://fireftp.net/help.html

vhodný SMTP server pro doručení [14]. UTF-8 znaky jsou za těchto podmínek podporovány jak v uživatelské části adresy, tak v doménové části. Doménová část se při formulaci DNS dotazu převede na řetězec bez diakritiky pomocí Punycode. Punycode je algoritmus převodu UTF-8 řetězce na řetězec obsahující pouze ASCII znaky [15].

3.4.2 Podpora na úrovni SMTP serverů

SMTP servery se liší stavem implementace SMTPUTF8 podpory. Analyzovala jsem některé volně dostupné servery a výsledné poznatky jsou shrnuty v této podkapitole. Mezi testované servery se řadí Postfix (verze 3.1.0), Exim (verze 4.88), Qmail (verze 1.06), Sendmail (verze 8.15.2). Častou metodou testování bylo odeslání emailu na adresu jøran@blåbærsyltetøy.gulbrandsen.priv.no. Jedná se o e-mailový automatický odpovídač s podporou znaků unicode. Podle slov majitele schránky Arnta Gulbrandsena, lze testovat v libovolném množství. Na začátku testování odpovídač nefungoval správně. Po dotázání se Arnta na chybu, ji rychle a ochotně opravil.

Prvním testovaným serverem byl Postfix[6]. Postfix support for Email Address Internationalization (EAI) a SMTPUTF8 extension jsou rozšíření pro podporu diakritiky v SMTP protokolu [14]. Použití tohoto rozšíření je: MAIL FROM: <address> SMTPUTF8. Princip doručení je takový,že server předá zprávu dalšímu serveru s podporou SMTPUTF8. Může se stát, že takový server nebude nalezen a tudíž zpráva nebude doručena. Podpora UTF-8 funguje v Postfixu od verze 2.12.

Druhým testovaným serverem byl Exim[7]. Exim4 by měl EAI podporovat. Jelikož je třeba program přeložit s parametrem SUPPORT_I18N a *libidn* knihovnou, nelze použít automatickou instalaci programu, ale program ručně přeložit. Exim Internationalisation[8] pojednává o podpoře diakritiky a národních abeced. Podpora UTF-8 funguje od verze 4.86, ale je pouze experimentální.

Třetím testovaným serverem byl Qmail[9]. Ten EAI nepodporuje a od roku 2005 ani nevycházejí nové verze. Existují však neoficiální patch soubory, se kterými by SMTPUTF8 podpora měla fungovat[10].

Čtvrtým testovaným serverem byl Sendmail. Jedná se o poměrně starý open source nástroj, stále však poměrně dost používaný na internetu. Bohužel EAI a IDN nepodporuje.

První věcí, kterou jsem vyzkoušela, bylo odeslat e-mail z adresy s diakritikou na adresu bez diakritiky. Tedy „MAIL FROM: talašová@talašová.cz“ a „RCPT TO: talasova.irena@gmail.com“. Za použití podpory SMTPUTF8 nebyl problém odeslat takovýto e-mail na serveru Postfix a Exim. Oproti tomu u Sendmailu a Qmailu nelze zadat takovouto adresu odesílatele.

6 URL http://www.postfix.com/

7 URL http://www.exim.org/

8 URL http://www.exim.org/exim-html-current/doc/html/spec_html/ch-internationalisation.html

9 URL http://www.qmail.org/

10 URL https://github.com/gentoo/gentoo/pull/2839

Druhým testem bylo odeslat e-mail na adresu s diakritikou z adresy bez diakritiky. Tedy „MAIL FROM: talasova@talasova.cz“ a „RCPT TO: jøran@blåbærsyltetøy.gulbrandsen.priv.no“. Obdobně jako v předchozím případě byla zpráva odeslána na serverech Postfix a Exim. Postarší Sendmail a Qmail ani tentokrát nepřekvapil, zpráva nelze odeslat.

Z uvedených analýz vyplývá, že moderní SMTP servery SMTPUTF8 rozšíření, potažmo diakritiku, v e-mailových adresách podporují. Starší servery, pro které nevychází aktualizace, toto rozšíření většinou nepodporují. Souhrn provedených analýz je níže v tabulce 3.5, podrobnosti lze nalézt v příloze 6.

	Postfix	**Exim**	**Qmail**	**Sendmail**
Podpora EAI	ANO - s podporou SMTPUTF8	ANO - s podporou SMTPUTF8, avšak s ruční kompilací	NE	NE
Podpora IDN	ANO - s podporou SMTPUTF8	ANO - s podporou SMTPUTF8, avšak s ruční kompilací	NE	NE

Tabulka 3.5: Shrnutí analýz podpory diakritiky na úrovni SMTP serverů

3.4.3 Podpora v rámci freemailových služeb

Pro účely testování podpory diakritiky v rámci jednotlivých freemailových služeb byl použit Postfix server verze 3.1.0 nainstalovaný do systému Ubuntu. Z tohoto SMTP serveru byly odesílány testovací e-maily s adresou odesílatele „talašová@talasova.cz“. Aby bylo doručení e-mailu ze strany tohoto serveru správné, je nutné použít SMTPUTF8 podporu. V takovém případě server předá e-mail pouze serveru, který UTF-8 podporuje. Pokud UTF-8 podpora není při odesílání zprávy použita, server e-mail neodešle korektně. Toto chování bylo ověřeno prostřednictvím schránky GMail, který EAI (Email Address Internationalization) podporuje (uvedeno dále v této kapitole). Testování probíhalo také za pomoci automatických odpovídačů s podporou znaků unicode jøran@blåbærsyltetøy.gulbrandsen.priv.no a českého odpovídače testmail@háčkyčárky.cz. Testovanými schránkami byly schránky od seznam.cz, centrum.cz, atlas.cz, volny.cz, eposta.cz, tiscali.cz, yahoo.com, gmail.com a hotmail.com.

Prvním testem bylo odeslání e-mailu z Postfixu s adresou „MAIL FROM: talašová@talasova.cz“ na adresu e-mailových schránek od seznam.cz, centrum.cz, atlas.cz, volny.cz, eposta.cz, tiscali.cz, yahoo.com, gmail.com a hotmail.com. Ve všech těchto případech nebyl e-mail doručen do schránky. Jediná výjimka byla schránka od gmail.com, kam byla odeslaná zpráva v pořádku doručena.

Druhým testem bylo odeslání e-mailu z výše uvedených schránek na adresu s diakritikou v doménové části tedy IDN. Pro testovací účely byl zvolen český automatický odpovídač „testmail@háčkyčárky.cz“. V případě správného doručení e-mailu do této schránky dojde k automatické odpovědi „IDN OK“. U schránky seznam.cz taková zpráva nejde odeslat, stejně jako

u centrum.cz, atlas.cz, volny.cz, tiscali.cz a yahoo.com. Ale v mobilních verzích centrum.cz, atlas.cz a volny.cz byla zpráva úspěšně odeslána. Vrátila se i úspěšná potvrzující odpověď serveru "IDN OK". Přičemž adresa odesílatele se zobrazila jako „testmail@xn—hkyrky-ptac70bc.cz“. Při odeslání e-mailu ze schránky u eposta.cz a hotmail.com se zpráva jevila jako úspěšně odeslána, avšak nevrací se potvrzující email "IDN Ok". Schránka gmail.com umožňuje odeslat tento testovací e-mail a rovněž se vrátila úspěšná potvrzující odpověď "IDN OK", adresa odesílatele je dokonce i ve správném tvaru s diakritikou.

Třetím testem bylo odeslání zprávy z jednotlivých schránek na adresu s diakritikou v uživatelské části „talašová@talasova.cz“. Tedy test EAI. Z žádné schránky ani z jejich mobilních verzí, kromě gmail.com nešlo odeslat takovýto e-mail. Vetšinou vše skončilo chybou „Adresát je neplatný“. Při testování gmail.com, v domnění, že jde o jeden z dalších automatických odpovídačů, jsem odeslala e-mail na adresu مثال@مثال.السعودية. Odepsal mi Arnt Gulbrandsen s ochotou pomoci. Email tedy dorazil v pořádku a jediná tato schránka podporuje EAI.

U žádného z testovaných poskytovatelů nelze založit schránku s diakritikou, dokonce ani u gmail.com, kde je podporováno jak IDN tak EAI. Souhrn provedených analýz je níže v tabulce 3.6, detaily lze nalézt v příloze 7.

	IDN	**EAI**	**Založení schránky s diakritikou**
Seznam	Ne	Ne	Ne
Centrum, Atlas, Volny	Ne	Ne	Ne
Mobilní verze Centrum, Atlas, Volny	Ano	Ne	Ne
Eposta, Tiscali	Ne	Ne	Ne
Mobilní verze Tiscali	Ne	Ne	Ne
Gmail	Ano	Ano	Ne
Yahoo	Ne	Ne	Ne
eHotmail	Ne	Ne	Ne

Tabulka 3.6: Shrnutí analýz podpory diakritiky v rámci freemailových služeb

3.5 POP3 a IMAP

V následujících podkapitolách bude nejprve shrnuto, jakou podporu diakritiky mají protokoly POP3 a IMAP v RFC; následně budou rozebrány provedené analýzy podpory diakritiky a národních abeced na úrovni serverů a klientů.

3.5.1 Podpora v protokolu POP3/IMAP dle RFC

Původní RFC specifikace internetového protokolu POP3 nepočítala s možnými UTF-8 znaky či řetězci. Toto bylo změněno přidáním podpory UTF-8 prostřednictvím příkazu „UTF8“. Po zaslání příkazu „UTF8“ dojde k přepnutí z ASCII režimu komunikace do režimu UTF-8. Mezinárodní znaky jsou podporovány v uživatelských jménech, heslech, e-mailových adresách, hlavičkách zpráv a těle textu [16].

Stejně jako u protokolu POP3 i u protokolu IMAP byla podpora mezinárodních znaků do RFC specifikace přidána až dodatečně. IMAP server deklaruje schopnost zpracovávat UTF-8 řetězce ve jménech, heslech, e-mailových adresách a hlavičkách pomocí odpovědi "UTF8=ACCEPT" nebo "UTF8=ONLY" [17].

3.5.2 Podpora na úrovni POP3/IMAP serverů

Stejně jako SMTP servery se POP3 a IMAP servery navzájem liší stavem implementace podpory diakritiky a národních znaků. Moderní a stále vyvíjené servery většinou zahrnují UTF-8 podporu, zatímco starší a již neaktualizované servery tuto podporu neimplementují vůbec. Testovanými servery jsou Dovecot (verze 2.2.29), MS Exchange Server (verze 2016), Cyrus (verze 2.4) a Courier (verze 0.78.2).

První z testů se zaměřil na vyhledání a manipulaci s e-maily obsahujícími diakritiku v hlavičkách e-mailů. Servery Dovecot[11], MS Exchange Server[12] a Courier[13] tímto testem prošly bez problému. Jediný Cyrus vyhledání e-mailů nedokázal.

Druhým testem bylo vytvoření účtu s diakritikou. Snažila jsem se vytvořit účet „talašová@talasova.cz“. Courier, Cyrus a Dovecot tento účet dokázali vytvořit. Jediný MS Exchange server vytvoření účtu zamítl kvůli nepovoleným znakům.

Třetím testem bylo přihlášení se k serveru pomocí vytvořeného účtu z předchozího testu. Na serverech Dovecot, MS Exchange Server a Courier jsem se dokázala takovýmto účtem přihlásit. Pouze na serveru Cyrus se přihlášení nezdařilo.

Z uvedených testů vyplývá, že obvyklým stavem implementace podpory diakritiky v POP3/IMAP serverech je situace, kdy bez problému funguje manipulace se zprávami a vyhledávání zpráv obsahujících diakritiku, avšak přihlášení se pomocí účtu s diakritikou nebo jeho vytvoření není často úspěšné. Výjimkami jsou Cyrus, který UTF-8 nepodporuje vůbec, a Courier, jehož podpora byla nejlepší z testovaných serverů. Souhrn analýz je uveden v tabulce 3.7, detaily v příloze 8.

11 URL https://www.dovecot.org/

12 URL https://products.office.com/cs-cz/exchange/microsoft-exchange-server-2016

13 URL http://www.courier-mta.org/

	IDN	EAI	Založení schránky s diakritikou
Dovecot	Ano	Ano ve vyhledávání e-mailů, Ne v uživatelských jménech	Ano
MS Exchange server	Ano	Ano ve vyhledávání e-mailů, Ne v uživatelských jménech	Ne
Cyrus	Ne	Ne	Ano
Courier	Ano	Ano	Ano

Tabulka 3.7: Shrnutí podpory diakritiky na úrovni POP3/IMAP serverů

3.5.3 Podpora na úrovni POP3/IMAP klientů

Podpora diakritiky POP3/IMAP klientů byla testována především pomocí odesílání e-mailů z adresy talašová@talašová.cz nebo odesílání e-mailů na tuto adresu. Také jsem využila automatického odpovídače testmail@háčkyčárky.cz pro otestování podpory diakritiky v doménové části adresy (IDN). Probíhalo testovaní klientů Thunderbird (verze 45.8.0), Roundcube Webmail (verze 1.2.4), aplikace Pošta z Microsoft store (verze 17.8104.42377.0), Outlook 2016 (verze 17.8104.42377.0) a Mutt (verze 1.5.24).

Nejprve jsem otestovala, zda lze z klientů odesílat e-mail na adresu bez diakritiky z adresy s diakritikou (diakritika se tedy nachází pouze v poli FROM). Klienti Thunderbird, Outlook 2016 a Mutt dovolili odeslání takovéhoto e-mailu. Oproti tomu v klientu Roundcube Webmail takováto zpráva nelze odeslat.

Ve druhém testu jsem se zaměřila na otestování podpory IDN. Pokoušela jsem se odeslat zprávu na adresu „testmail@háčkyčárky.cz“ (e-mailová adresa automatického odpovídače). U Thunderbirdu, Roundcube Webmailu, Pošty, Outlooku i Muttu došlo k úspěšnému odeslání e-mailu.

Třetí a poslední test byl zaměřen na testování podpory EAI, tedy podpory adresy s diakritikou v uživatelské části. Snažila jsem se odeslat zprávu na adresu „talašová@talasova.cz“. Odeslání bylo v Thunderbirdu neúspěšné. Stejně tak nejde zpráva odeslat v Roundcube Webmail a Poště. Klienti Mutt a Outlook 2016 dovolili takovouto zprávu odeslat.

Stav podpory UTF-8 řetězců je u různých klientů odlišný. Všichni testovaní klienti podporovali diakritiku v doménové části adresy. UTF-8 znaky v lokální části jsou však většinou problém, a to především v poli „TO“, tedy v rámci adresy příjemce zprávy. Souhrn provedených analýz je v tabulce, detaily lze nalézt v příloze 9.

	IDN	EAI	Poznámka
Thunderbird Mail	Ano	Ano/Ne	Ano v poli „FROM“, ne v poli „TO“
Roundcube Webmail	Ano	Ne	-
Pošta (Windows)	Ano	Ne	-
Outlook 2016	Ano	Ano	-
Mutt	Ano	Ano	-

Tabulka 3.8: Shrnutí podpory diakritiky na úrovni POP3/IMAP klientů

4 Architektura software vznikajícího v rámci projektu SProbe

Tato kapitola poskytne krátký úvod do problematiky zákonných odposlechů. Bude představen jeden konkrétní systém vyvíjený v rámci projektu SProbe; jeho základní bloky a architektura a další nástroje využitelné pro zákonné odposlechy. Blíže se zaměřím na uvedení principu fungování těch části software, které budou během návrhu a implementace rozšíření o podporu národních znaků modifikovány.

Projekt SProbe vzniká na Fakultě informačních technologií VUT s cílem vytvořit síťové sondy pro zákonné odposlechy až do úrovně aplikační vrstvy. Jedná se o kombinaci hardware a software s cílem využití pro orgány činné v trestním řízení. Pro dosažení požadovaného výkonu je využit koncept softwarově definovaného monitorování a výpočetní platforma FPGA SoC. Sondy se zaměřují nejen na analýzu a filtraci provozu, ale také poskytují statistické informace, informace o kvalitě měřených dat a identifikují šifrovaný provoz.

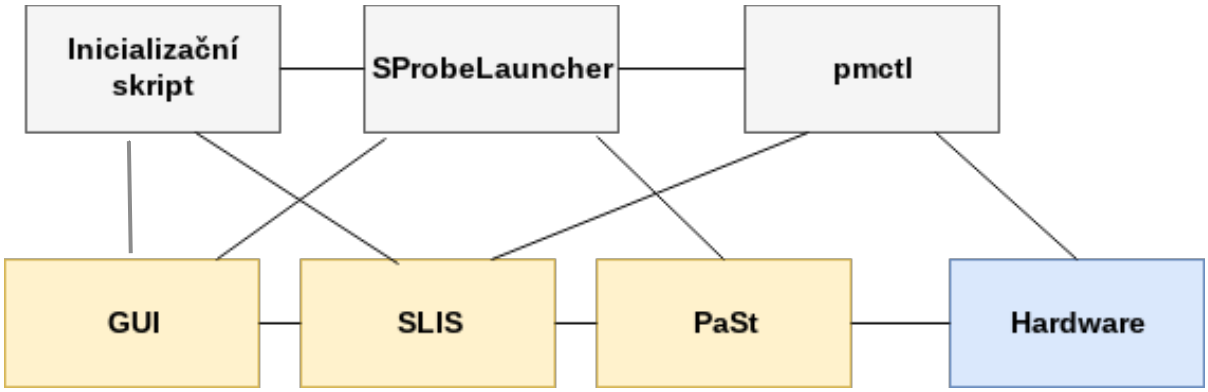
4.1 Úvod do systémů pro zákonné odposlechy

Cílem zákonných odposlechů je umožnit získat důkazní materiál o závažných trestných činech pomocí informací získaných z počítačových sítí. Zákonné odposlechy byly povoleny Radou Evropské unie [18] a následně standardizovány ETSI (European Telecommunications Standards Institute). Orgány činné v trestním řízení spolupracují s provozovateli počítačových sítí (AP – Access Provider) a poskytovateli služeb (Service Provider) za účelem získávání důkazních informací. Sběr dat v síti obstarává systém LIS (Lawful Interception System), který je instalován do odposlouchávané sítě. Požadavky na zahájení odposlechu musí být schváleny soudem. Získaná data podstupují další analýzu a slouží jako podklad pro jednání orgánů trestního řízení (Law Enforcement Agency – LEA, dále jen LEA) [19].

Network Identifier (NID) - síťový identifikátor je používaný v síťových protokolech k označení účastníků komunikace nebo k označení konkrétního spojení (např. TCP). Systém LIS je schopen označit uživatele na základě MAC adresy, statické IPv4 a IPv6 adresy nebo obecně rozsahem adres definovaným adresou sítě a maskou. Na úrovni aplikačních protokolů je možné identifikovat uživatele resp. jeho komunikaci na základě emailové adresy, SIP adresy nebo loginu některého z protokolů pro předávání zpráv v reálném čase [19].

4.2 Blokové schéma architektury SProbe

Software vyvíjený v rámci projektu SProbe je možno pomyslně rozdělit do několika logických celků, z nichž některé komunikují s hardware. Pro lepší představu lze znázornění jednotlivých softwarových bloků vidět na obrázku 1.



Obrázek 1: Architektura software pro SProbe

Každá z komponent má specifickou funkci:

- Inicializační skript slouží pro nastavení sondy po startu. Spouští například webové rozhraní sondy, systém SLIS a nastavuje další důležité parametry jako šifrování SD karty a další.
- SProbe Launcher uchovává nastavení sondy a na jeho základě spouští potřebné služby. Dále také umožňuje komunikaci mezi hardware a webovým rozhraním. Při změně konfigurace sondy je třeba spoustu nástrojů, které spouští a nastavují různé služby, restartovat. Proto vznikl SProbe Launcher (SPL), který komunikuje s GUI a spravuje služby na sondě. SPL je spuštěn po nabootování OS na sondě a běží až do vypnutí sondy. SPL zahrnuje několik služeb. Tyto služby jsou implementovány pomocí shell skriptů. Kromě konfigurace služeb, jako nastavení hardwarových komponent, exportu dat a dalších, SPL také umožňuje nastavení času na sondě. Novou konfiguraci automaticky zapíše do RTC modulu.
- Pmctl je nástroj konfigurující jednotku provádějící pattern match (PM) a umožňující komunikaci mezi systémem SLIS a hardwarem sondy.
- GUI, nebo-li webové rozhraní, slouží pro interakci systému s uživatelem. Jeho prostřednictvím lze konfigurovat sondu a přidávat, upravovat nebo mazat odposlechy. Detailnější popis je uveden v podkapitole 4.3.
- SLIS je systém vycházející ze systému LIS zmíněný v podkapitole 4.1. Spravuje odposlechy a vytváří pravidla pro PaSt. Více o tomto systému lze nalézt v podkapitole 4.4.
- PaSt provádí rozpoznávání aplikačních protokolů, zachytávání odposlouchávaných řetězců a export zachycených dat. Detailnější informace jsou uvedeny v podkapitole 4.5.

Z pohledu této práce jsou nejdůležitější tři části výše uvedeného software. Zaprvé GUI, které

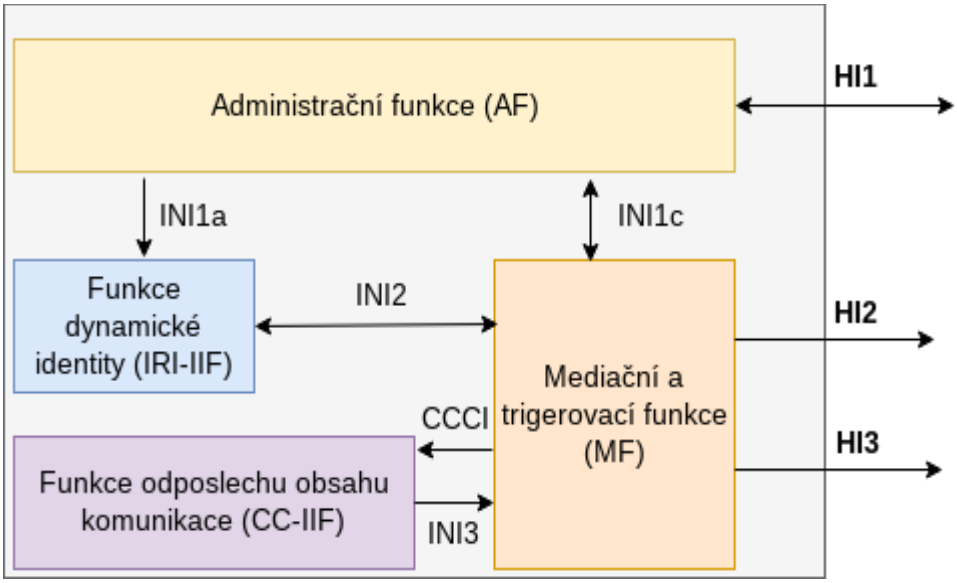
musí umožňovat zadat uživateli NID s diakritikou. Zadruhé je to systém SLIS, který bude vyžadovat nejvíce úprav pro zavedení podpory diakritiky a třetí nejdůležitější částí software SProbe z pohledu implementace podpory diakritiky je PaSt, kde je nutno řádně otestovat chovaní a schopnost práce s pakety a konfiguračními řetězci obsahující diakritiku.

4.3 GUI

Webové rozhraní sondy poskytuje především jednoduché a uživatelsky přívětivé prostředí pro konfiguraci sondy. Proti neoprávněnému přístupu je chráněno uživatelským jménem a heslem. Přihlášený uživatel může z logovacích záznamů zjistit, kdo a z jaké IP adresy se na sondu přihlašoval a zda bylo přihlášení úspěšné. Zobrazují se také další záznamy o provozu sondy a o manipulaci s odposlechy. GUI také umožňuje měnit parametry exportu dat jako cílovou IP adresu a port a zobrazit další uživatelsky zajímavé věci jako jsou různé statistiky o provozu na jednotlivých rozhraních sondy. Jednou z nejdůležitějších vlastností GUI je schopnost, za použití funkce *hi1.py* jako prostředníka mezi GUI a SLIS, přidávat, odebírat, pozastavovat a znova spouštět odposlechy dle zadaných parametrů od uživatele.

4.4 SLIS

Systém SLIS (Sec6Net Lawful Interception System) je konkrétní implementace systému LIS vytvořená v rámci projektu Sec6Net na Vysokém učení technickém v Brně [19]. Architektura tohoto systému je znázorněna na obrázku 2.



Obrázek 2: architektura systému SLIS [19].

Administrační funkce (AF) přijímá vstupní požadavky na odposlech prostřednictvím rozhraní HI1. Tato funkce provede kontrolu zadaných položek jako například jedinečnost LIID (Lawful Interception Identifier). Správně zadaný odposlech je zařazen do fronty čekajících odposlechů. AF musí inicializovat tento odposlech, to znamená správně nastavit ostatní části systému pro zachytávání patřičných dat stejně tak jako přijmout nový odposlech. AF musí být schopná provést změny existujícího odposlechu či odposlech smazat a patřičně překonfigurovat nastavení dalších bloků. Ke komunikaci s ostatními bloky dochází prostřednictvím rozhraní INI1a a INI1c.

AF funkce pomocí rozhraní INI1a komunikuje s blokem Funkce dynamické identity (IRI-IIF), jehož úkolem je detekce zpráv v síti vázající se k dynamické změně identity uživatele (např. díky DHCP). Každou takovouto novou skutečnost pak tato funkce propaguje do dalších části SLIS. Tato funkce také vytváří metadata o začátcích a koncích odposlouchávaných spojení a předává je pomocí rozhraní INI2 do Mediační a trigerovací funkce.

Mediační a trigerovací funkce (MF) spravuje zachycená data. Zpracovává metadata od bloku IRI-IIF a obsah komunikace od bloku CC-IIF (Funkce pro odposlech a komunikaci). Výsledná zpracovaná data zasílá prostřednictvím rozhraní HI2 a HI3 na výstup.

Blok Funkce odposlechu obsahu komunikace (CC-IIF) podle požadavků na začátek a konec odposlechů zachytává patřičná data a posílá je zpět do MF pomocí rozhraní INI3 [19].

4.5 Packet Stack (PaSt)

PaSt je software spravující informace o tocích detekovaných v hardware. Dále provádí detekci protokolů a export dat do sběrného místa Law Enforcement Monitoring Facility (LEMF). Jedním z důležitých požadavků na PaSt je portabilita a obecná správa dat, jelikož je použita jak na sondě s rychlostí 1G tak na sondě s rychlostí 10G. Každá z variant obsahuje OS Linux, ale je založena na jiné architektuře CPU. Dalším nárokem na PaSt je rychlé zpracování dat. Rychlost linky je natolik velká, že je nemožné důkladně parsovat a zpracovávat data. Z tohoto důvodu jsou využity regulární výrazy pro zachycení charakteristických rysů jednotlivých zachytávaných protokolů. Zachytávaná data jsou předzpracována v hardware, který pomocí regulárních výrazů nalezne zachytávané NIDy (síťové identifikátory) a také identifikuje protokoly, které musí být kvůli kompletnosti zachytávány dříve, než se objeví požadovaný NID. HW a SW codesign umožňuje rychlé zpracování dat s tím, že jen některá data jsou zpracována podrobně.

PaSt je implementována v jazyce C++. Díky tomu je program přenositelný a umožňuje vysokoúrovňové programování s možností specifikovat časově kritické operace nízkoúrovňově na úrovni jazyka C. V PaSti je odděleno získávání dat a zpracování dat, aby bylo možno ji použít ve všech následujících případech:

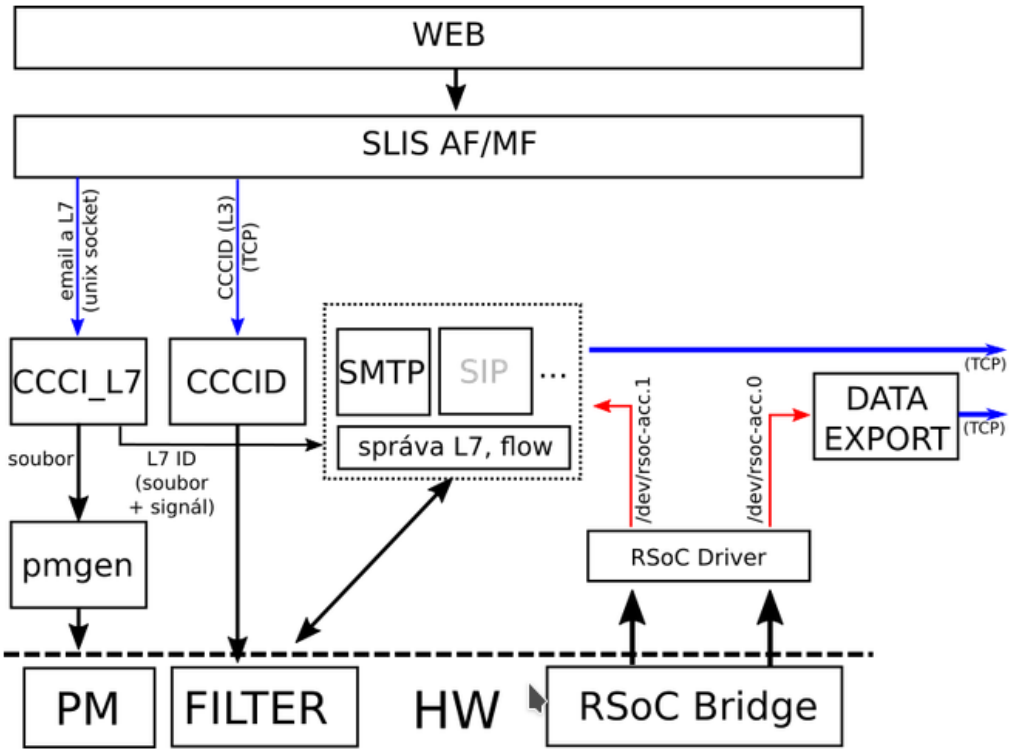
- čistě softwarová verze: testování nad soubory PCAP,

- provoz na sondě SProbe 1G s rozhraním RSoC[14],
- provoz na sondě SProbe 1G s rozhraním DPDK[15],
- provoz na sondě SProbe 10G,

Pro testování základní funkcionality PaSti bez sondy SProbe je možno PaSt zkompilovat pro použití s knihovnou libpcap. Druhou možností je PaSt zkompilovat pro sondu SProbe pracující s rozhraním RSoC. V tomto případě je funkčnost kompletní a nechybí HW pro detekci NIDů ani okamžiků potenciálních začátků odposlechů.

Program PaSt se spouští s parametry definující rozhraní RSoC a konfiguračními soubory pro každý podporovaný protokol. Po startu si PaSt z těchto konfiguračních souborů načte všechny hledané NIDy. Je-li specifikováno LEMF (Law Enforcement Monitoring Facility), pak se s ním PaSt spojí pomocí TCP. Pro každý jednotlivý podporovaný protokol je vytvořen speciální parser, který má za úkol vyhledávat NIDy uvedené v konfiguračním souboru příslušného protokolu. Po každé změně v odposleších, potažmo v konfiguračních souborech pro PaSt, je PaSti zaslán signál SIGUSR1 nebo SIGHUP sloužící k znovunačtení konfigurace.

Blokové schéma systému s detailnějším zaměřením na PaSt je znázorněno na obrázku 3.



Obrázek 3: Blokové schéma PaSti.

14 VIKTORIN, Jan. *HW/SW Codesign for the Xilinx Zynq Platform*. Brno, 2013. Dostupné z: http://www.fit.vutbr.cz/study/DP/DP.php?id=14453. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. 2013-06-19. Vedoucí práce Korček Pavol.

15 https://dpdk.org/

CCCID očekává prostřednictvím TCP soketu CCCID zprávy od SLISu. Podle typu se buď přidá, nebo odebere L3 pravidlo do FILTERu.

CCCI_L7 zajišťuje jednotnou konfiguraci PaSti a PM. Volá pmgen, který zajišťuje, aby se jednotka nekonfigurovala paralelně několikrát a aby se pmgen nespouštěl více než jedenkrát současně. Zároveň mění konfigurační soubory pro PaSt a každou změnu ohlašuje PaSti pomocí signálu.

Pattern Match (PM) jednotka slouží pro vyhledávání L7 identifikátorů. Jednotka Protocol Identifier slouží k vyhledání začátku datového toku zájmového provozu používajícího některý z definovaných protokolů předtím, než jednotka Pattern Match nalezne v toku identifikátor.

4.5.1 Průchod paketu PaStí

Srdcem programu je smyčka pcap_loop(), která:

1. nejprve načte nový/další paket ze vstupu,
2. poté se vyhodnotí maska příslušných parserů uložená v Flow Table (FT),
3. aplikační parsery paket zpracují,
4. vyhodnotí se, co se má s daným paketem stát – export, uložení do tabulky toků a další,
5. Je-li paket nalezen PI, potom se uloží do FT, dokud se nezachytí NID pomocí PM nebo dokud hardware nevydá rozhodnutí o odstranění.

Pro každý aktivní tok zpracovaný PaStí se vytvoří záznam v tabulce toků obsahující hlavně IP adresy, porty, TCP/UDP a masku parserů. S každým novým paketem PaSt zkontroluje, zda již má pro daný tok uloženy nějaké informace v tabulce toků. Pokud ne, založí pro něj nový záznam. Pokud hardware vyhodnotí tok jako potenciální začátek odposlechu jsou první pakety uloženy pro případné další použití. Pokud HW v toku nalezl některý z hledaných NIDů, pak se v rámci PaSti ověřuje přítomnost hledaného NIDu pomocí regulárních výrazů specifických pro konkrétní aplikační protokol. HW dokáže pomocí regulárních výrazů vyhledávat sice rychleji, než software, ale tyto výrazy mají pouze omezenou složitost. Z tohoto důvodu se dále vyhledává i v software, což má upřesňující charakter.

V okamžiku, kdy PaSt rozhodne, že zpracovávaný paket obsahuje nakonfigurovaný NID, a tedy je určen pro odposlouchávání, uloží informaci o této skutečnosti do tabulky toků a dojde k exportu paketu spolu se všemi uloženými pakety od potenciálního začátku odposlechu. Pokud PaSt rozhodne, že paket nemá být zachycen, NID se například vyskytl v jiném než hledaném kontextu, je zpracování paketu ukončeno a pravidlo pro záchyt je z hardware odstraněno. Pokud byl detekován potenciální začátek odposlechu, je možné, že tok se ještě stane předmětem odposlechu, a proto není ihned smazán z filtru. Toky, u kterých se zjistí se, že nemají být odposlouchávány a ani potenciální začátek odposlechu neoznačuje data pro export, jsou odstraněny z hardware i software.

Čtení ze vstupu a export dat do LEMF má na starost třída *dispatcher*. Při běhu dispatcheru mohou nastat dvě situace. Buď se všechny pakety podařilo úspěšně odeslat anebo se přerušilo spojení s LEMF. Pokud se přeruší spojení s LEMF, *dispatcher* se pokusí spojení opět navázat. Pakety určené

pro export jsou průběžně bufferovány. V okamžiku, kdy se spojení znovu naváže, dispatcher se nejdříve pokusí odeslat první paket z bufferu a až poté odesílá pakety se vstupu. Cílem je průběžně odesílat nashromážděná data bez omezení zpracování vstupu.

4.5.2 Podporované aplikační protokoly

Mezi podporované aplikační protokoly patří:

1. **SMTP** – v PaSti je implementován záchyt dat od „MAIL FROM“ pro každý e-mail, ve kterém se nachází hledaný NID buď v poli „MAIL FROM“ nebo „RCPT TO“. Při výskytu nakonfigurované e-mailové adresy v rámci „MAIL FROM“ je e-mail exportován ihned. Pro zajištění exportu celé e-mailové obálky i při záchytu na základě některého z adresátů se pro všechny e-maily používá odložené rozhodnutí o začátku odposlechu. V okamžiku, kdy se zachytí hledaný NID v poli „RCPT TO“ dojde k exportu uložených dat od „MAIL FROM“.
2. **POP3/IMAP** - PaSt implementuje záchyt celého sezení protokolu POP3/IMAP i záchyt jednotlivých e-mailů. Nakonfigurovaná e-mailová adresa pro odposlech je rozpoznána i v rámci komunikace POP3. Jelikož je příkaz „USER“ u protokolu POP3 i FTP stejný, je implementována třída pro jejich odlišení.
3. **FTP** – je implementován pouze záchyt kontrolního kanálu, kde je možné zachytit buď celé sezení, nebo „RETR“ a „STOR“.
4. **SIP** - PaSt umožňuje zachytávání celého sezení SIP, zachytávání konkrétního hovoru a přenos zpráv. Parser SIPu se snaží ověřovat párování dotazů a odpovědí, čímž je parsování méně náchylné k chybám oproti protokolům SMTP, POP3, IMAP a FTP. Problémy vznikají v okamžiku, kdy sonda vidí pouze část komunikace.

4.6 Softwarové řízení hardware

Prefiltrace provozu v hardware je založena na tom, že hardware musí být schopen identifikovat potenciálně zajímavý provoz a předávat do software veškerá data toku, počínaje detekcí zájmového toku. Software poté datový tok podrobí detailnější analýze. Výsledkem je konečné rozhodnutí o tom, zda je tento tok určený k odposlechu a v tom případě dojde k jeho exportu pomocí LEMF, nebo je tok zahozen a pravidlo v hardwarovém filtru pro předávání je zrušeno.

Průchod paketu obsahující odposlouchávaný NID je takový, že ze sítě nejprve putuje přes hardware do PM/PI jednotky, která je konfigurována pomocí nástroje pmctl. Dále putuje do filtru, který je konfigurován jednotkami PM/PI a následně je PaSti předán. PaSt provede export datového toku. PaSt také může odebrat pravidlo z hardwarového filtru a to pokud:

- v PaSti se zjistí, že se nejedná o podporovaný aplikační protokol,
- komunikace byla ukončena pomocí FIN paketů v obou směrech,
- aplikační parser po rozpoznání aplikačního protokolu detekoval konec odposlouchávaného celku anebo je filtr přeplněn,

5 Návrh na rozšíření softwaru SProbe pro identifikaci provozu s národními abecedami

V této kapitole bude uveden návrh rozšíření stávajícího systému o podporu diakritiky. Tento problém lze rozdělit do několika částí, uvedených v následujících podkapitolách, v nichž je třeba otestovat a ověřit chování stávajícího systému a následně implementovat rozšíření.

5.1 Kódování UTF-8

UTF-8 je jedním z nejčastěji používaných kódování. UTF znamená "Unicode Transformation Format" a "8" znamená, že v kódování se používají 8-bitová čísla.

Uvědomme si, že:

- je-li kódová hodnota znaku Unicode menší než 128, potom je znak reprezentován v UTF-8 odpovídající hodnotou bytu,
- pokud je kódová hodnota Unicode mezi 128 a 0x7ff, znak je reprezentován dvěma byty mezi 128 a 255,
- kódové hodnoty Unicode větší než 0x7ff jsou převedeny na tříbytové nebo čtyřbytové sekvence, kde každý byte je mezi 128 a 255.

V této práci používám kódování UTF-8, které má několik výhodných vlastností [22]:

- dokáže pracovat s jakýmkoli Unicode znakem,
- řetězec Unicode znaků se změní na řetězec bytů, který neobsahuje žádné vložené nulové byty,
- řetězec ASCII znaků je také platný UTF-8 řetězec,
- UTF-8 je poměrně kompaktní; většina znaků zabírá pouze 1 nebo 2 byty,
- pokud jsou byty poškozeny nebo ztraceny, je možné určit počátek další sekvence kódované kódem UTF-8 a resynchronizovat. Je také nepravděpodobné, že náhodné 8-bitové údaje budou vypadat jako platné UTF-8 kódy.

5.2 Podpora UTF-8 znaků v celém systému SProbe

Prvním důležitým předpokladem pro implementaci podpory národních znaků v projektu SProbe je skutečnost, že všechny komponenty software dokáží pracovat s UTF-8 znaky. Do GUI musí být možné přidat například odposlech na emailovou adresu „talašová@talašová.cz“, SLIS musí být schopen z tohoto řetězce vytvořit regulární výrazy a PaSt musí být schopna zachytávat pakety obsahující tento výraz a následně je exportovat.

Přidání nového odposlechu se provádí přes grafické uživatelské rozhraní implementované v jazyce PHP. V aktuálním stavu software není možné zadat odposlech obsahující diakritiku. Nejprve tedy bude nutné zjistit, kde je chyba a tu opravit. Problém pravděpodobně bude ve funkci `escapeshellarg()`, která odstraňuje znaky, které nejsou ASCII, z UTF-8 řetězce. Tomuto odpovídá chování GUI, které vypouští znaky s diakritikou.

Informace o nově zadaném odposlechu z GUI dále putuje do systému SLIS. Ani tento systém nepodporuje diakritiku. Systém SLIS je implementován v jazyce Python 3. Tato verze podporuje UTF-8 řetězce, UTF-8 je dokonce implicitní způsob kódování, přesto kód reprezentující funkci HI1 nedokáže UTF-8 znaky zpracovat.

Jakmile nový odposlech projde přes funkci HI1, je třeba zpracovat jeho NID (network identifier) a vytvořit z něj podle definovaných pravidel regulární výraz. Vytvořené regulární výrazy budou podmnožinou všech výrazů, které by mohly být vytvořeny. Je však nutno vybrat pouze ty, které se při testování aplikací v kapitole 3 vyskytovaly nejčastěji kvůli omezeným paměťovým schopnostem hardware (PM/PI jednotek). Ani tyto funkce nedokáží zpracovat UTF-8 znaky. Bude třeba je modifikovat tak, aby diakritika byla zapsána správně do konfiguračních souborů pro PaSt.

Z vytvořených konfiguračních souborů s diakritikou je PaSt i pattern match jednotka správně inicializována a dochází již k bezchybnému záchytu i exportu paketů obsahujících hledané řetězce s diakritikou. V tomto okamžiku, kdy je odposlech obsahující diakritiku schopen projít od GUI přes SLIS a PaSt až na export dat, je vhodné začít uvažovat, jaké řetězce obsahující národní znaky v souvislosti s jednotlivými zkoumanými síťovými protokoly se mohou vyskytnout. Tato řešení budou prezentována v následujících podkapitolách a vycházejí ze skutečností zjištěných v kapitole 3 o doporučeném stavu implementace síťových aplikací prostřednictvím RFC i z jejich zjištěného skutečného stavu.

5.3 SIP

Jak již bylo zmíněno v kapitole 3.2, SIP URI se používá pro identifikaci účastníků během síťové komunikace uživatelů dle protokolu SIP. Právě adresa SIP URI se používá jako jeden ze síťových identifikátorů charakterizující odposlech. Tento NID je zpracován a je z něj vytvořen výraz, který bude zachytáván.

Myšlenkou implementace podpory diakritiky do tohoto systému je vytvoření více regulárních výrazů ze SIP URI, které se budou odposlouchávat. Různé výrazy budou konstruovány podle toho, zda se diakritika nachází v uživatelské části, doménové části nebo obou částí adresy. Výsledné řetězce byly navrženy dle aktuální implementace SIP klientů, a to s ohledem na paměťové nároky. Během tvorby konfiguračních výrazů pro PaSt může nastat několik situací:

- V případě, že adresa neobsahuje žádnou diakritiku, vytvoří se pouze regulární výraz odpovídající zadané adrese. Je sice možné, že některý znalý uživatel by se mohl pokusit překódovat adresu neobsahující diakritiku například do hexadecimálního tvaru, a tím se pokusil obejít odposlech této adresy, avšak jak již bylo zmíněno dříve, z důvodu omezeného paměťového prostoru nelze generovat všechny možné řetězce. Nemá tedy smysl generovat další řetězce.
- Pokud se diakritika nachází v doménové části adresy, je mimo UTF-8 znaky potřeba počítat také s tím, že tato část může být zadána ve formě Punycode, jako tomu bylo například u klienta Zoiper. Zde se tedy kromě pravidla, které používá UTF-8 znaky, vytvoří také pravidlo převádějící tuto část do Punycode. Uveďme konkrétní příklad: pro SIP adresu „sip@háčkyčárky.cz“ by konfigurační SIP soubor pro PaSt měl vypadat, jak je uvedeno v ukázce 1.

```
sip@háčkyčárky\.cz
sip@xn—hkyrky-ptac70bc\.cz
```

Ukázka 1: Konfigurační soubor, diakritika v doméně.

- Národní znaky v uživatelské části se mohou stejně jako v testovaných klientech (například klient Zoiper) vyskytovat v hexadecimální podobě. Vytvoří se tedy regulární řetězec obsahující místo UTF - 8 znaků hexadecimální znaky a druhý vytvořený řetězec používá jen UTF - 8 kódování. Pro SIP adresu „talašová@ekiga.cz“ by se měly do konfiguračního souboru SIP parseru zapsat řetězce z ukázky 2.

```
talašová@ekiga\.cz
tala%c5%a1ov%c3%a1@ekiga\.cz
```

Ukázka 2: Konfigurační soubor, diakritika v uživatelské části.

- Je-li diakritika přítomna jak v doménové, tak v uživatelské části adresy, je vytvořeno nejvíce řetězců určených pro odposlech. Jedná se o kombinace výše uvedených. Tyto řetězce mohou vzniknout buď úmyslným pokusem útočníka skrýt svoji identitu nebo nestandardní implementací některého SIP klienta. Vytvoří se tedy řetězec pouze s UTF-8 znaky. Následuje řetězec s UTF-8 znaky v uživatelské části a Punycodem v doménové části. Dále je vytvořen řetězec s překódovanými hexadecimálními znaky v obou částech adresy a také řetězec s hexadecimálními znaky v uživatelské části a Punycodem v doménové části. Uvažujme jako

příklad adresu mámerádi@háčkyčárky.cz, potom do konfiguračního SIP souboru by se v tomto případě měly přidat řetězce z ukázky 3. Z ukázky 3 je patrné, že se nevytváří odposlech s adresou „mámerádi@*h%c3%a1%c4%8dky%c4%8d%c3%a1rky*\.*cz*“. Tento řetězec nebyl přidán mezi generované řetězce za účelem uspoření paměťového místa pro výrazy. Tato varianta se při testování SIP aplikací a serverů v kapitole 3.2 jevila jako méně pravděpodobná, než ostatních pět výrazů. Pokud nebude nutno v budoucnu řešit tato hardwarová omezení, bylo by vhodné přidat i tento odposlech do seznamu odposlouchávaných adres.

```
mámerádi@háčkyčárky\.cz
mámerádi@xn--hkyrky-ptac70bc\.cz
m%c3%a1mer%c3%a1di@xn--hkyrky-ptac70bc\.cz
m%c3%a1mer%c3%a1di@háčkyčárky\.cz
m%c3%a1mer%c3%a1di@h%c3%a1%c4%8dky%c4%8d%c3%a1rky\.cz
```

Ukázka 3: Konfigurační soubor, diakritika v celé adrese.

5.4 FTP

Zde se zaměřujeme na případné zachytávání názvu souborů. Klienti i servery by měl i dbát doporučení a přenášet soubory v UTF - 8 kódování. Velká většina se pravděpodobně tímto řídí a tím pádem odposlech názvu souboru zadaného pomocí UTF - 8 znaků pokryje vysoké procento přenášených souborů. Existují však výjimky, které používají kódování jiné a i takto soubor přenášejí, avšak tímto se nemá smysl zabývat a vytvářet nová pravidla, neboť by jich bylo velké množství, což by neúměrně zatěžovalo paměť a přínos by nebyl velký. Zde se tedy spokojíme s tím, že systém dokáže zpracovat UTF - 8 řetězec a i ho v proudu dat zachytit. V tomto případě budou řetězce přidávané do konfiguračních souborů jednoduché. Například pro FTP login s názvem „pčr“ se přidá řetězec „pčr“ do konfiguračního souboru a pro soubor s názvem „Nový.txt“ řetězec „Nový\.txt“.

5.5 E-mailové protokoly

V této práci mezi e-mailové protokoly řadíme protokol POP3, IMAP a protokol SMTP. Nejprve se zaměříme na protokol POP3 a IMAP. Zde testovaní klienti bez problémů využívali UTF-8 kódování, žádné jiné se během testování nevyskytlo. Proto i zde jsem otestovala, že systém dokáže zachytit UTF-8 znaky v POP3 a IMAP loginech a tvorba dalších řetězců není žádoucí.

V případě protokolu SMTP je situace obdobná jako u protokolu SIP, avšak s tím rozdílem, že u SMTP jsem neobjevila ani klienty ani servery, které by se striktně nedržely doporučení uvedených ve standardech, jako tomu bylo u SIP klientů. Pro diakritiku v doménové části e-mailové

adresy je charakteristický Punycode. Punycode se v tomto případě reálně využívá jako tomu bylo při testu autoresponderu od sdružení CZ.NIC, „testmail@háčkyčárky.cz“. U tohoto protokolu tedy může vzniknout několik situací:

- Pokud adresa neobsahuje diakritiku, pak je vytvořen pouze jeden řetězec totožný se zadanou adresou.
- Pokud se diakritika nachází pouze v uživatelské části, použije se pro tuto část UTF-8 kódování znaků. Například pro e-mailovou adresu „talašová@anerii.eu“ se do konfiguračního souboru přidá řetězec z ukázky 4.

```
talašová@anerii\.eu
```

Ukázka 4: Konfigurační soubor, diakritika v uživatelské části.

- Diakritika v doménové části znamená kromě pravidla obsahující UTF-8 znaky za znakem „@“ také vytvoření pravidla obsahující Punycode na tomto místě. V případě e-mailové adresy „testmail@háčkyčárky.cz“ bude konfigurační soubor obsahovat řetězce zapsané v ukázce 5.

```
testmail@háčkyčárky\.cz
testmail@xn—hkyrky-ptac70bc\.cz
```

Ukázka 5: Konfigurační soubor, diakritika v doménové části.

- Diakritika v celé e-mailové adrese znamená vytvoření dvou pravidel. První je složeno pouze z UTF-8 znaků a druhé kombinuje UTF-8 znaky v uživatelské části a Punycode v části doménové. Obdobně jako v předchozím případě, při zadání odposlechu na e-mailovou adresu „talašová@talašová.cz“ vzniknou dva nové odposlouchávané řetězce zapsané v ukázce 6.

```
talašová@talašová\.cz
talašová@xn—talaov-uta53h\.cz
```

Ukázka 6: Konfigurační soubor, diakritika v celé adrese.

Tvary vytvořených pravidel by měly pokrýt většinu forem výskytů identifikátorů e-mailových adres. Přestože se výjimky mohou v síti objevit, je to silně nepravděpodobné a jednalo by se spíše o pokus zmást zařízení pro odposlech. Navíc by v takovém případě téměř jistě nedošlo k doručení e-mailu.

5.6 Úprava testovacího prostředí

Důležitým krokem při vývoji a implementaci podpory diakritiky bude testování. Rozsáhlejší testování implementovaného systému bude prováděno až po dokončení implementace všech částí pro ověření

kompletní funkčnosti rozšíření. Testy však budou využívány také jako pomocné nástroje při testování aktuálního stavu software a při přidávání jednotlivých funkčních testů. Z toho plyne vysoká důležitost testovacích skriptů a vznikají na ně nové požadavky, které bude nutno brát v potaz.

Pro systém SProbe je navržen systém automatických testů. Pro každý jednotlivý podporovaný aplikační protokol je napsána samostatná sada nejrůznějších testů v jazyce Python. Lze spustit sadu testů pouze pro určitý protokol nebo pomocí skriptu autotesting.py dojde k postupnému otestování všech podporovaných protokolů. Testovací skripty jsou založeny na nástroji Scapy[16]. Scapy je nástroj pro manipulaci s pakety. Jedná se o nástroj napsán v jazyce Python. Umožňuje zachytávání paketů a odesílání paketů na specifikované rozhraní, vytváření paketů nejrůznějších protokolů a jejich dekódování.

V současné době fungují testy tak, že je nejprve třeba, aby uživatel manuálně nastavil pomocí GUI sondy všechny odposlechy na požadované NIDy, které se v testech objevují. Zde se však objevuje první problém aktuální implementace testů. Na sondu je možno nakonfigurovat jen omezený počet odposlechů. Lze počítat s tím, že sonda zvládne asi 16 odposlechů zároveň. Toto číslo není nikterak vysoké, obzvlášť když si představíme, že při použití síťových identifikátorů s diakritikou by většinu místa v paměti zabrala pravidla pro NIDy obsahující UTF-8 znaky. Toto je první místo testů, které bude třeba upravit.

Přínosná by byla implementace testů, kde by docházelo k automatickému přidávání odposlechů před začátkem běhu testů a po ukončení testů automatické smazaní těchto nově přidaných odposlechů. To vše by se mělo dít bez účasti uživatele a plně automaticky. Tím pádem bude nutno do každého testovacího skriptu pro každý podporovaný aplikační protokol přidat část kódu, která bude komunikovat se sondou, nakonfiguruje potřebné odposlechy a poté se zase od sondy odpojí.

Jedna z možností, jak manipulovat s odposlechy z testů spouštěných na lokálním počítači, je s pomocí vloženého bash skriptu. Bude nutno se přihlásit přes SSH na sondu a zde za pomoci funkce hi1.py pro manipulaci s odposlechy pozastavit, přidat nebo smazat odposlechy. Jelikož by ale uživatel který na sondě má nakonfigurované „své“ odposlechy o ně jistě nerad přišel, ještě před automatickou konfigurací nových potřebných odposlechů pro úspěšný běh skriptu se pozastaví všechny aktuální odposlechy. Díky tomu bude uživatel schopen po doběhnutí testů znova spustit a pokračovat v odposleších se kterými pracoval před spuštěním testů.

Tímto způsobem bude docíleno toho, že pro každý podporovaný aplikační protokol se budou potřebné odposlechy nakonfigurovávat zvlášť, a tím pádem jejich počet může být vyšší než v případě, kdy se přidají všechny odposlechy najednou.

Dalším úkolem bude navrhnutí nových testů, které budou přidány k těm původním tak, aby byly co nejlépe otestovány situace, kdy NIDy obsahují diakritiku. První možností je rozšířit původní množinu testů o nové testy speciálně zaměřené víceméně pouze na testování toho, zda sonda dokáže zpracovat odposlech s diakritikou, identifikovat takový síťový tok a potřebná data exportovat. Je to jednoduché řešení, avšak s nevýhodou toho, že by se zbytečně opakovaly podobné testy pro NIDy

16 URL https://scapy.net/

s diakritikou a bez diakritiky. Ve své podstatě by bylo užitečnější, pokud by se všechny již naimplementované testy daly využít i pro NIDy obsahující diakritiku. Tyto již vytvořené testy testují širokou škálu problémů, které se během záchytu mohou objevit. Na druhou stranu není nutné, a z časových důvodů vhodné, úplně všechny testy v každém testovacím souboru spouštět jak pro NID obsahující diakritiku, také pro NID klasický bez diakritiky. Z tohoto důvodu bude nejlepším řešením vytvořit dekorátor, pomocí něhož budeme schopni označit vybrané metody, potažmo testy, které se mají spouštět pro více předem zvolených NIDů.

5.7 Shrnutí úprav prováděných v této práci

Pro zavedení podpory diakritiky do software projektu SProbe je nutno modifikovat jednotlivé komponenty tak, aby všechny byly schopné přijímat a zpracovávat odposlechy obsahující diakritiku. Po provedení analýzy a testování současného systému jsem zjistila, že je nutno provést úpravy zejména v těchto částech software:

- GUI – aktuálně není možné zadat odposlech obsahující diakritiku. Veškerá diakritika v síťovém identifikátoru je odstraněna funkcí `escapeshellarg()` ještě před předáním odposlechu funkci `hi1.py`.
- SLIS – Zde je třeba zaměřit se na podporu kódování UTF-8 a nastavení tohoto kódování jako výchozího. Dále je třeba provést změny ve vytváření konfiguračních souborů pro PaSt. Jelikož použití diakritiky v síťových aplikacích je poměrně variabilní, je třeba z jednoho síťového identifikátoru obsahující diakritiku vytvořit více identifikátorů shodného obsahu, ale rozdílného zápisu. Zaměříme se na síťové identifikátory e-mailových protokolů, protokolů FTP a SIP.
- Následně je třeba provést úpravy ve stávajícím testovacím prostředí. Testy je třeba rozšířit o automatické přidávání a odmazávání odposlechů podle spouštěného testu. Dále je třeba vytvořit testy pro otestování diakritiky. Výhodným řešením je úprava již existujících testů pomocí dekorátoru tak, aby byly uživatelem definované testy spouštěny jak pro NID obsahující diakritiku, tak pro NID bez diakritiky.

6 Implementace

Při implementaci řešení navrženého v kapitole 5 jsem postupovala shora dolů, tedy od obecných částí softwaru k těm specifičtějším. Nejdříve jsem se zaměřila na to, aby webové rozhraní sondy bylo schopno přijmout uživatelský požadavek na přidání odposlechu se síťovým identifikátorem obsahujícím diakritiku. GUI následně tento požadavek předá funkci hi1.py, která v současném implementačním stavu také nedokáže pracovat s UTF-8 znaky.

Toto bude následující krok v implementaci podpory diakritiky. Funkci hi1.py předá uživatelský požadavek systému SLIS, který přijatý NID vyhodnotí a vytvoří konfigurační soubor pro každý jednotlivý podporovaný aplikační protokol obsahující řetězce, které budou využity PaStí pro zachytávání sítových toků. Právě systém SLIS implementuje logiku rozšíření systému SProbe o podporu monitorování síťového provozu se znaky národních abeced.

Nakonec bude třeba ověřit, že PaSt dokáže na základě vytvořených konfiguračních souborů zachytávat definované síťové toky a pakety obsahující diakritiku exportovat.

Posledním krokem bude testování. Testování bude prováděno dvěma způsoby: jednak zasíláním provozu z různých PCAP souborů na příslušné porty sondy a kontrola správnosti exportovaných nebo neexportovaných PCAP souborů; druhak použitím vytvořených automatických testovacích skriptů.

V této kapitole bude popsán způsob implementace podpory diakritiky do GUI, hi1.py a systému SLIS.

6.1 GUI

Webové rozhraní k sondě je napsáno v jazyce PHP. Nyní se budeme soustředit pouze na schopnost GUI manipulovat s odposlechy. Cílem je zpracování formulářových polí a formulování příkazu pro zaregistrování nového odposlechu. Uživatel vepíše informace o nově přidávaném odposlechu do formuláře a odešle. Požadované informace jsou LIID (identifikátor zadávaného odposlechu), NID (síťový identifikátor) a doba zahájení a ukončení odposlechu. Každý NID začíná předponou definující aplikační protokol. Předpona „mail:“ pro protokol SMTP, „pop3:“ pro protokol POP3, „imap: pro protokol IMAP, „sip:“ pro protokol SIP, „ftp:“ pro zadání FTP loginu uživatel a předpona „filename:“ slouží k definování názvu souboru.

Jakmile uživatel vepíše hodnoty do formulářových polí a formulář odešle, backend logika GUI zjistí, zda se jednalo o požadavek na smazání, zastavení, znovu spuštění či přidání nového odposlechu. Z hlediska implementace podpory diakritiky je důležitým okamžikem vkládání, protože pokud lze odposlech do systému vložit, i ostatní operace proběhnou bez problému. Při vkládání nového odposlechu se získají hodnoty formulářových polí, které jsou předány jako parametry funkci hi1.py, která zajistí „probublání“ nově přidaného odposlechu dále do systému. Parametry této funkce však musí být před jejím zavoláním upraveny tak, aby byly použitelné jako argumenty v shellu. To

zahrnuje přidání uvozovek na začátek a konec řetězce a escapování některých symbolů. Pro tyto účely se v kódu používá funkce `escapeshellarg`(). Po jejím detailnějším prozkoumání jsem zjistila, že pracuje pouze s ASCII znaky. Tomu odpovídá chování, které vykazovalo GUI, kdy po přidání nového odposlechu s diakritikou došlo k odstranění těchto znaků. Například NID „testmail@háčkyčárky.cz“ se vložil jako „testmail@hkyrky.cz“. Takovéto chování funkce `escapeshellarg()` by se mělo změnit po přidání řádku z ukázky 7.

```
setlocale(LC_CTYPE, "en_US.UTF-8");
```

Ukázka 7: Nastavení jazyka.

Nakonec však byla v kódu zvolena jiná varianta řešení, kdy funkce `escapeshellarg()` byla použita pouze na připravení ostatních argumentů (LIID, start time, end time) a samotný NID je upraven pomocí regulárního výrazu. Proměnná `$nid` obsahuje uživatelem zadaný odposlech. Připravené argumenty jsou předány funkci HI1.py pomocí následujícího kódu v ukázce 8.

```
exec("cd /opt/slis/ && ./hi1.py insert $liid \"". $nid ."\"
$start $end 2>&1", $output, $r);
```

Ukázka 8: Spuštění skriptu hi1.py.

Díky těmto změnám je GUI schopno pracovat i s NIDy obsahujícími diakritiku. Přidaný odposlech s diakritikou je vidět na obrázku 4.

Obrázek 4: Vložení odposlechu obsahující diakritiku v GUI.

6.2 funkce hi1.py

Aby funkce hi1.py mohla dělat prostředníka mezi webovým rozhraním a systémem SLIS, musí i ona být schopna pracovat s UTF-8 znaky. Princip kódování znaků v UTF-8 bylo vysvětleno v podkapitole 5.1.

Při implementaci bylo využito tzv. *surrogateescape encoding*. Toto kódování nazývá sám tvůrce UTF-8b[17] kódování. Tento způsob kódování funguje správně pouze pokud se data převedou zpátky na byty také pomocí tohoto kódování. Použití jiného kódování vyvolá výjimku. Kódování UTF-8b bylo ve funkci hi1.py použito pro kódování a dekódování přijatého NID řetězce. Konkrétní použití je znázorněno v ukázce 9.

```
args.NID = args.NID.encode("UTF-8", "surrogateescape")
                    .decode("UTF-8", "surrogateescape")
```

Ukázka 9: Použití „surrogateescape encoding“.

6.3 SLIS

V systému SLIS je potřeba udělat nejvíce změn ze všech ostatních částí SProbe software z pohledu implementace podpory diakritiky. Právě tato část software zpracovává NIDy zadané uživatelem a přijaté prostřednictvím funkce hi1.py. Z jednotlivých síťových identifikátorů jsou vytvořeny řetězce, které jsou následně zapsány do konfiguračních souborů pro jednotlivé aplikační parsery. Soubor nid.py obsahuje třídy jednotlivých síťových identifikátorů. Tyto třídy slouží k úpravě přijatého NIDu a k vytvoření konfiguračního řetězce. Mezi podporované NIDy a příslušné třídy patří:

- třída NIDEmailAddress pro e-mailové adresy (protokol SMTP),
- třída NIDIMAPLogin pro protokol IMAP,
- třída NIDSIP pro protokol SIP,
- třída NIDPOP3Login pro protokol POP3,
- třída NIDFTPLogin pro FTP login,
- třída NIDFilename pro název souboru.

Nadále nás budou zajímat pouze NIDy a příslušné třídy pro protokoly SMTP, POP3, IMAP, FTP a SIP.

6.3.1 NIDPOP3Login, NIDIMAPLogin

Jelikož v návrhu rozšíření systému v kapitole 5 bylo zmíněno, že v případě POP3 a IMAP loginu není třeba systém rozšiřovat o nějaké další vzniklé konfigurační řetězce a systém zachytává pakety s UTF-8 znaky správně, implementace tříd pro POP3 a IMAP login je velmi jednoduchá. Například pro POP3 třída NIDPOP3Login je implementována jako je zobrazeno v ukázce 10. NIDIPLimitedSprobe je třída implementující důležité metody jako `__init__` nebo

17 URL https://www.python.org/dev/peps/pep-0383/

`getPaStRegexes` pro specifické odposlechy obsahující odposlouchávanou adresu. Návratová hodnota metody `getParserList` charakterizuje protokol NIDu.

```
class NIDPOP3Login(NIDIPLimitedSProbe):
     def __init__(self, value):
          NIDIPLimitedSProbe.__init__(self, value, "pop3")
     @staticmethod
     def getType():
          return "POP3 login"
     @staticmethod
     def getParserList():
          return [3]
```

Ukázka 10: Implementace NIDPOP3Login.

6.3.2 NIDFTPLogin

U aplikačního protokolu jsou vytvořeny dvě třídy NIDFTPLogin a NIDFilename pro rozlišení, zda síťový identifikátor obsahuje FTP login uživatele nebo název souboru, který je přenášen pomocí protokolu FTP. Ani v tomto případě není třeba systém rozšiřovat o další vytvářené řetězce, a proto třídy zůstávají v podobě, kterou představuje ukázka 11.

```
class NIDFTPLogin(NIDIPLimitedSProbe):
    def __init__(self, value):
        NIDIPLimitedSProbe.__init__(self, value, "ftp")
    @staticmethod
    def getType():
        return "FTP login"
    @staticmethod
    def getParserList():
        return [6]
```

Ukázka 11: Implementace NIDFTPLogin.

6.3.3 NIDEmailAddress

E-mailové adresy obsahující UTF-8 znaky v doménové části adresy bývají klientskými aplikacemi překódovány pomocí Punycodu do řetězce pouze ASCII znaků v doménové části. Je tedy třeba v případě adresy s diakritikou v doménové části vytvořit konfigurační řetězce dva. Jeden obsahující pouze UTF-8 znaky v celé e-mailové adrese a druhý obsahující UTF-8 znaky v uživatelské části adresy a Punycodem v doménové části.

Třída `NIDEmailAddress` při inicializaci zkontroluje, zda je NID ve správném tvaru. E-mailová adresa nesmí začínat a končit znakem „@“ a musí obsahovat právě jeden tento znak. Tato třída také definuje dvě metody a to `getPaStRegex` a `getPMRegex`, které se starají o vytvoření konfiguračních souborů ze zadaného NIDu.

Metoda `getPaStRegex` zkontroluje, zda e-mailová adresa na začátku nebo na konci neobsahuje znak „*“, který značí, že na tomto místě se může nacházet jakýkoli řetězec a případně je k NIDu místo toho znaku přidán regulární výraz označující jakýkoli řetězec. Dále se u NIDu zjistí, zda obsahuje diakritiku v doménové části adresy. Pokud ano, vytvoří se další řetězec s překódováním této části do Punycodu. Seznam vzniklých řetězců je předán zpět volající metodě. Analogickou implementaci jako metoda `getPaStRegex` má metoda `getPMRegex`, proto následující ukázka 12 zachycuje pouze metodu `getPaStRegex`. `Regex_escaper` má na starosti escapování znaků v adrese.

```
def getPaStRegex(self):
    prefix = ""
    postfix = ""
    v = self._value
    res = []
    if v[0] == "*":
       v = v[1:]
       prefix = r"[^ \t\r\n@]*"
       if v[-1] == "*":
          v = v[:-1]
          postfix = r"[^ \t\r\n@]*"
    x = v.find('@') + 1
    domain = v[x:]
    user = v[:x]
    domain = domain.encode('idna')
    address = user.encode() + domain
    res.append(prefix + regex_escaper(v) + postfix)
    if address.decode() != v:
       res.append('\n' + prefix +
           regex_escaper(address.decode()) + postfix)
    return ''.join(str(elem) for elem in res)
```

Ukázka 12: Implementace getPaStRegex v NIDEmailAddress.

6.3.4 NIDSIP

Pravidla tvorby řetězců vzniklých ze SIP adresy obsahující UTF-8 znaky byla popsána v návrhu rozšíření v kapitole 5. Podobně jako u NIDu e-mailové adresy se nejprve v třídě `NIDSIP` zkontroluje použití znaku „@“. Rovněž se kontroluje použití znaku „*“ který je případně nahrazen regulárním výrazem označujícím jakýkoliv řetězec, jak naznačuje ukázka 13.

```
def getPaStRegex(self):
    prefix = ""
    postfix = ""
    v = self._value
    res = []
    if v[0] == "*":
       v = v[1:]
       prefix = r"[^ \t\r\n@]*"
    if v[-1] == "*":
       v = v[:-1]
       postfix = r"[^ \t\r\n@]*"
```

Ukázka 13: Implementace getPaStRegex v NIDSIP.

Následně se ze SIP adresy extrahuje doménová část, která je překódována do Punycodu. Nově vzniklý řetězec obsahující původní uživatelskou část adresy a doménovou část zakódovanou pomocí Punycodu se zapíše do výsledného seznamu řetězců. Dále je tento řetězec porovnán s původní SIP adresou a pokud se jedná o rozdílné řetězce, zapíše se i tento původní vzor do seznamu řetězců, jak je napsáno v ukázce 14. Proměnná v obsahuje odposlouchávanou e-mailovou adresu.

```
x = v.find('@') + 1
domain = v[x:]
user = v[:x]
domain = domain.encode('idna')
address = user.encode() + domain
res.append(prefix + regex_escaper(v) + postfix)
if address.decode() != v:
    res.append('\n' + prefix +
               regex_escaper(address.decode()) + postfix)
```

Ukázka 14: Kódování řetězce do Punycodu.

Poté se zjistí, zda je uživatelská část adresy složena pouze z ASCII znaků. Pokud není, je uživatelská část adresy překódována do hexadecimální podoby znaků a vzniklé řetězce jsou téže uloženy do seznamu výsledných řetězců. Kód je pro tuto část zahrnut v ukázce 15.

```
try:
     user.encode('ascii')
except:
     user = user.encode('utf-8')
     user = str(user)
     user = user[2:]
     user = user[:-1]
     user = user.replace(r"\x", "%")
     res.append('\n' + prefix + regex_escaper(user +
                domain.decode()) + postfix)

     if domain.decode() != v[x:]:
          res.append('\n' + prefix +
                     regex_escaper(user + v[x:]) + postfix)
          domain = v[x:].encode('utf-8')
          domain = str(domain)
          domain = domain [2:]
          domain = domain [:-1]
          domain = domain .replace(r"\x", "%")
          res.append('\n' + prefix +
                     regex_escaper(user + domain) + postfix)

     return ''.join(str(elem) for elem in res)
```

Ukázka 15: Implementace překódování uživatelské části adresy

6.3.5 SLIS.py

Vytvořené konfigurační řetězce se ukládají do souboru pomocí metody `updateFiles` v souboru `slis.py`. Tato metoda musí otevírat soubory pro zápis také v režimu UTF-8. Pro každý typ NIDu je zavolána příslušná metoda a otevřen odpovídající konfigurační soubor, do kterého se provede zápis připraveného seznamu konfiguračních řetězců, to je zobrazeno v ukázce 16.

```
def updateFiles():
    index = 0
    config_file = {}
    res = []
    pimask = 0
    with open(fileSmtp + '.tmp', 'w', encoding="utf-8") as
                    config_file["email"], \
         open(fileImap + '.tmp', 'w', encoding="utf-8") as
                    config_file["imap"], \
         open(fileSip + '.tmp', 'w', encoding="utf-8") as
                    config_file["sip"], \
         open(filePop3Logins + '.tmp', 'w',
              encoding="utf-8") as config_file["pop3"], \
         open(fileFtpLogins + '.tmp''w',encoding="utf-8")
                    as config_file["tp"], \
         open(fileFtpFilenames+'.tmp','w',encoding="utf-8")
                    as config_file["filename"]:
    for nidstr, attrs in intercept_table.items():
        config_file[attrs.nid_type].write(attrs.nid.getPaSt
           Regex() + '\n')
```

Ukázka 16: Implementace updateFiles.

Konfigurační řetězce pro Pattern Match jednotku se ukládají všechny do jednoho souboru za využití metody z ukázky 17.

```
with open(filePM + '.tmp', 'w', encoding="utf-8") as
config_file:
      for nidval, attrs in intercept_table.items():
                    res = attrs.nid.getPMRegex()
                    for elem in res:
                         key_pm = '/^' + elem + '/i#'
                         +attrs.nid.getPMConfigString()
```

Ukázka 17: Implementace použití funkce getPMRegex.

Vytvořené soubory slouží jako informace pro PaSt o tom, jaké řetězce obsažené v paketech se mají zachytávat.

6.4 Testovací prostředí

Po dokončení implementace podpory diakritiky v jednotlivých částech software jsem přistoupila k testování pomocí automatických testů vytvořených pro systém SProbe. Jak již bylo zmíněno v kapitole 5.6, bylo třeba automatické testy upravit, neboť tak, jak byly implementovány dříve nebylo možné efektivně spustit velké množství testů.

Nejprve bylo třeba odstranit závislost všech testů na NID „gurpartap@patriots.in“. Tato e-mailová adresa byla využita pro všechny testy pro otestování, zda byla komunikace se sondou řádně navázána a tato úspěšně exportuje data. Kvůli této závislosti musel být nakonfigurován odposlech na tuto e-mailovou adresu i v případě, že se spouštěly testy například na protokol FTP. Jednalo se tedy o zbytečně zabrané místo pro uložení odposlechu. Popisem řešení tohoto problému se zabývá podkapitola 6.4.1.

Druhým krokem bylo změnit kód tak, aby docházelo k automatické konfiguraci odposlechů na sondu ještě před spuštěním testů a po jejich skončení se přidané odposlechy odstranily. Toto téma je diskutováno v podkapitole 6.4.2.

Třetím a posledním krokem úprav bylo vytvoření dekorátoru metod, který dokáže některé metody testů spustit vícekrát pro různé NIDy. Tímto způsobem lze velmi elegantně jednou napsaný kód použít pro otestování více síťových identifikátorů. Řešení je popsáno v podkapitole 6.4.3.

6.4.1 Změna signatury dle protokolu

Cílem bylo, aby se použila pro testování spojení se sondou signatura, která odpovídá testovanému protokolu, a tak odpadla potřeba neustále konfigurovat odposlech na e-mail „gurpartap@patriots.in“. Nejprve jsem do metody zajišťující spuštění testů přidala parametr, který identifikuje protokol pro který jsou testy spouštěny. Například kód pro otestování e-mailové komunikace prostřednictvím protokolu SMTP je zapsán v ukázce 18, kde je vidět přidání parametru „smtp“, na základě kterého budou na sondu nakonfigurovány všechny e-mailové adresy objevující se v testech.

```
genericTest("smtp", TestsSMTP, verbosity=2, failfast=False)
```

Ukázka 18: Spuštění testu pro SMTP protokol.

Při inicializaci třídy `SprobeConnection`, vytvářející spojení se sondou, dojde ke zvolení signatury na základě prvního parametru metody `genericTest`, jak je vidět v ukázce 19.

```
#set signature for connection
if (protocol == "smtp") or (protocol == "auto")
   or (protocol == "imap") or (protocol == "ipv6")
   or (protocol == "pop3"):
     self.signature = "MAIL FROM:<gurpartap@patriots.in>\r\n"
if (protocol == "ftp"):
     self.signature = "RETR Topeni.wav\r\n"
if (protocol == "sip"):
     self.signature = "MAIL FROM:<gurpartap@patriots.in>\r\n"
```

Ukázka 19: Implementace rozlišení protokolu.

Tato zvolená signatura se následně využije v metodě `__create_INI3_socket()`, vytvářející spojení se sondou, kde se použije pro vytvoření paketu, který se odesílá sondě, a následně se čeká, až sonda tento paket exportuje.

Stejně tak je tato signatura použita v metodě w`ait_until_ready` pro vytvoření paketu. Po odeslání paketu se čeká, dokud sonda tento paket neexportuje a tím nepotvrdí, že se nachází v definovaném stavu.

6.4.2 Automatická konfigurace odposlechů

Původně bylo automatické testování implementováno tak, že před spuštěním testovacího skriptu bylo nutno pro úspěšný běh testů manuálně, například přes GUI, zadat NIDy, které se v testech vyskytují. Kvůli malé paměti však lze zadat pouze asi 16 takovýchto řetězců, což je pro testovací účely velmi málo a ruční zadávání odposlechů je časově velmi neefektivní. Proto jsem navrhla rozšíření testů o automatickou konfiguraci testů. Principem je ještě před samotným spuštěním testů pozastavit všechny aktuálně probíhající odposlechy a přidat nové odpovídající testovací množině. Po proběhnutí testů jsou všechny tyto nově přidané odposlechy odstraněny a sonda je připravena na další testovaní. I zde se využívá nově přidaná signatura identifikující množinu testů, jak je vidět z ukázky 18 v podkapitole 6.4.1.

Přidávání odposlechů je nutno provést ještě před vytvářením spojení se sondou, neboť do doby, než se přidají požadované odposlechy, není zaručeno, že by sonda exportovala data v metodách `__create_INI3_socket` a `wait_until_ready`.

Podle signatury se zvolí seznam odposlechů, který má být nakonfigurován, například pro protokol SMTP nebo POP3 jsou definice uvedeny v ukázce 20.

```
if self.protocol == "smtp":
   intercepts =
       ['email:háčky@čárky.cz','email:*@fit.vutbr.cz',
        'email:pachatel@*', 'email:gurpartap@patriots.in']
if self.protocol == "pop3":
   intercepts = ['email:*@fit.vutbr.cz', 'email:pachatel@*',
                 'pop3:pčr', 'pop3:pcr', 'pop3:sec6net.demo',
                 'pop3:\(limited_smtp,10.0.0.1\)',
                 'email:gurpartap@patriots.in']
```

Ukázka 20: Konfigurace odposlechů.

Následně se testovací skript pomocí příkazu SSH připojí na sondu a pomocí skriptu pause_remove_intercepts.sh pozastaví aktivní odposlechy na sondě. Připojení na sondu a předání parametru „pause“ skriptu pause_remove_intercepts.sh je zobrazeno v ukázce 21.

```
#pause active intercepts and add new
command = "ssh -o {rsa} -l {user} {host} -p {port} 'sh -s' <
          pause_remove_intercepts.sh pause"
os.system(command.format(**data))
```

Ukázka 21: Připojení na sondu a předání parametru.

Skript Pause_remove_intercepts.sh nejdříve zjistí všechna SID aktivních odposlechů pomocí sekvence příkazů z ukázky 22.

```
python3 /opt/slis/hi1.py all_intercepts
   | grep -Eo '\"sid\": [0-9]+, \"state\": \"active\",'
   | grep -Eo "[0-9]+"
```

Ukázka 22: Získání všech SID odposlechů.

A zjištěná SID využije k pozastavení odposlechů pomocí fragmentu kódu z ukázky 23.

```
if [ "$1" == "pause" ]; then
    for intercept in $targets
    do
        python3 /opt/slis/hi1.py pause $intercept
    done
fi
```

Ukázka 23: Pozastavení odposlechů.

Po znovupřipojení se na sondu nahrají nové odposlechy podle ukázky 24.

```
for intercept in intercepts:
    command = "ssh -o {rsa} -l {user} {host} -p {port}
              -t \"python3 /opt/slis/hi1.py insert" +
              str(i) + " " + intercept + " 1.1.1970 never\""
    os.system(command.format(**data))
    i = i+1
```

Ukázka 24: Vložení nových odposlechů.

Po úspěšném uložení odposlechů proběhne navázání spojení se sondou a jakmile se sonda dostane do definovaného stavu, spustí se testování. Metoda `wait_until_ready` je zde také využita k čekání, dokud nebudou všechny odposlechy řádně nakonfigurovány na sondu. Konfigurace každého odposlechu zabere několik sekund, a proto je nutno se zahájením testovaní nějakým způsobem vyčkat, dokud nebudou všechny odposlechy řádně přidány. V opačném případě by testy byly spuštěny ještě než by byly odposlechy nastaveny, a tak by některé testy skončily neúspěchem.

Odposlech obsahující identifikátor, na který je nastavena signatura pro metodu `wait_until_ready,` se přidává jako poslední ze všech konfigurovaných odposlechů. Jakmile je sonda schopna v odpovědi exportovat data s tímto identifikátorem, je jasné, že i ostatní, dříve konfigurované odposlechy, jsou již řádně zahájeny. Po ukončení testů je zavolána metoda `remove_intercepts` pro odstranění přidaných odposlechů, což je znázorněno v ukázce 25.

```
def remove_intercepts(self):
        command = "ssh -o {rsa} -l {user} {host} -p {port}
                  'sh -s' < pause_remove_intercepts.sh remove"
        os.system(command.format(**data))
```

Ukázka 25: Odstranění odposlechů.

Sonda je nyní připravena ke spuštění další sady testů dalších protokolů, jejichž proces spuštění a konfigurování sondy bude podobný jako výše popsaný způsob spuštění testů SMTP.

6.4.3 Parametrizovatelné testy

Aby bylo možné během testování efektivněji ověřit funkčnost síťových identifikátorů obsahujících diakritiku, bylo nutno upravit testy tak, aby se vybraným testům pomocí parametru dal předat seznam NIDů, pro které má být test spuštěn. Původní varianta totiž počítala pouze s jedním pevně daným NID vepsaným v testu. Za tímto účelem byla vytvořena třída ParametrizableTCMeta(), která umožňuje vzniknout z jednoho označeného testu více různých testů, z nichž každý bude spuštěn pro jiný NID ze seznamu nastavených NIDů.

Například v ukázce 26 je uveden seznam požadovaných NIDů k otestování.

```
test_nids = [
    ("punycode", ("háčky@xn--rky-dla3t.cz", )),
    ("diakritika", ("háčky@čárky.cz", )),
    ("email", ("email@seznam.cz", )),
]
```

Ukázka 26: Nastavení testovaných NIDů.

Test, který má být spuštěn se všemi výše uvedenými NIDy, označíme dekorátorem, jak je znázorněno v ukázce 27.

```
@ptc.parametrizableTest(test_nids)
def test_diacritics(self, nid):
        pckt = create_packet(sip=self.sip, dip=self.dip,
                             sport=self.sport,
                             dport=self.dport,
                             app="mail from:<"+nid+">\r\n")
self.assertTrue(self.s.send_and_check(self.probing_packet,
                pckt))
```

Ukázka 27: Dekorátor testu.

Výsledkem bude zánik původního testu `test_diacritics` a vznik dvou nových `test_punycode`, s NIDem "háčky@xn—rky-dla3t.cz", a `test_diakritika`, s NIDem „háčky@čárky.cz“. Tímto způsobem odpadá nutnost psát další testy pro NIDy s diakritikou, které by svým obsahem byly stejné jako ty testy, které jsou aktuálně implementované.

7 Testování

Po dokončení implementace podpory diakritiky v systému SProbe bylo nutno všechny jeho součásti důkladně dohromady otestovat. Systém jsem nejprve testovala pomocí různých PCAP souborů obsahujících síťovou komunikaci s identifikátory obsahujícími diakritiku a kontrolovala jsem, že sonda exportuje zachycená data. Takto jsem vyzkoušela PCAPy obsahující e-mailovou komunikaci, komunikaci POP3/IMAP klientů a serverů, a i PCAPy používající protokoly SIP a FTP. Pro otestování každého protokolu bylo využito nejméně pět různých PCAP souborů zachycujících komunikaci daného protokolu. Po úspěšném exportu všech těchto dat a celkovém odladění systému jsem přistoupila k rozsáhlejšímu testování.

Ve druhé fázi testování jsem do původních testovacích skriptů přidala testy obsahující NIDy s diakritikou. Pro každý podporovaný protokol (POP3, IMAP, SMTP, FTP a SIP) jsem vytvořila dle složitosti vytvářených regulárních výrazů jeden nebo více testů. Tam, kde kvůli diakritice vznikalo z jednoho původního odposlechu více odposlechů zapsaných jiným způsobem, bylo třeba pro každý takovýto odposlech vytvořit test, aby bylo ověřeno, že se generují všechny požadované řetězce a že PaSt dokáže exportovat PCAPy obsahující tyto výrazy.

Druhá testovací fáze ukázala, že způsob testování popsaný výše je značně neefektivní, a to ze dvou důvodů. Zaprvé je uživatel nucen před každým spuštěním testu ručně nakonfigurovat potřebné odposlechy, například skrze GUI, a jelikož nakonfigurování každého odposlechu zabere řádově několik sekund, než uživatel zadá všechny parametry odposlechu, je to velmi uživatelsky nepřívětivé. A zadruhé, jak již bylo několikrát v práci zmíněno, na sondu lze nahrát pouze omezený počet odposlechů. V testech se již před přidáním diakritiky vyskytovalo mnoho odposlechů, které bylo třeba pro úspěšný běh testů nakonfigurovat, a po přidání všech řetězců vzniklých z NIDu obsahující diakritiku by byla paměť pro uložení odposlechů vyčerpána a ani by se tam všechny požadované odposlechy nevešly. Díky naimplementované úpravě testovacího prostředí, jak bylo popsáno v podkapitolách 6.4.2 a 6.4.3, jsem mohla diakritiku úspěšně otestovat pomocí automatické konfigurace odposlechů pro každý spouštěný test pro všechny klíčové testy dříve implementované pro NIDy obsahující jen ASCII znaky. Počet takto spouštěných testů je variabilní a uživatel může snadno určit, kolik a které testy budou spouštěny, jak pro klasické NIDy bez diakritiky, tak pro NIDy obsahující diakritiku.

Automatická konfigurace testů celý průběh testování urychlila a především usnadnila, avšak stále je testování poměrně časově náročnou operací. V závislosti na počtu přidávaných odposlechů samotná konfigurace odposlechů na sondu trvá řádově několik jednotek nebo desítek sekund. Během testování byla doba konfigurace jednoho odposlechu průměrně 2 až 3 sekundy. Časově nezanedbatelnou je také doba než se se naváže spojení se sondou a než sonda přejde do definovaného stavu. Tato doba se velmi lišila. Někdy celý tento proces proběhl za několik sekund někdy to trvalo

i téměř půl minuty. A nakonec doba provedení samotných testů je také několik sekund v závislosti na délce testovacího skriptu.

8 Závěr

V rámci diplomové práce byla implementována podpora národních znaků v síťových identifikátorech. Zaměřila jsem se na implementaci podpory v protokolech SMTP, POP3, IMAP, FTP a SIP. Nejprve byla vypracována analýza současného stavu implementace vybraných síťových aplikací dle zkoumaných protokolů. Tento stav byl porovnán s podporou národních abeced zakotvenou v příslušných RFC. Dle zjištěného a otestovaného stavu současného softwarového vybavení projektu SProbe byly navrženy metody, jak tento software rozšířit o podporu diakritiky při zachytávání síťových identifikátorů.

Na základě získaných poznatků byla navržena řešení, jak v jednotlivých modulech systému zavést podporu UTF-8 znaků. Bylo nutno implementovat podporu diakritiky do GUI a do systému SLIS. Především však bylo nutné navrhnout pravidla a způsob, jak generovat nové, potencionálně možné síťové identifikátory z identifikátorů s diakritikou. Po úspěšnému dokončení práce následovalo detailní testování celého systému při práci s identifikátory obsahující diakritiku a ověření správného záchytu a exportu dat. Pro možné budoucí opakování těchto testů byly vytvořeny automatické testy, které mohou sloužit k průběžnému otestování, že systém i po přidání dalších metod a během vývoje software neztratil schopnost podpory národních abeced. Automatické testy nyní dokáží nakonfigurovat potřebné odposlechy a následně je po proběhnutí testů zase smazat. Tímto způsobem je šetřena paměť a lze provést větší množství testů.

Další vývoj řešeného problému spatřuji především v pokračujícím sledování vývoje podpory diakritiky v jednotlivých síťových aplikacích a protokolech. Je pravděpodobné, že software bude nutno dále upravovat, dle nových změn. Potencionálně zajímavé by také mohlo být věnovat se jednotlivým kódováním a například do GUI přidat možnost zachytávání odposlechu s určitým kódováním.

Poznatky získané během řešení této diplomové práce byly prezentovány na studentské konferenci Excel@FIT 2018 a zároveň byly představeny i během prezentací v předmětu NSB (Návrh, správa a bezpečnost).

Literatura

[1] Bazala David: Telekomunikace a VoIP telefonie 1.díl, BEN - technická literatura, 2006, ISBN 80-7300-201-9.

[2] Matoušek Petr: Síťové služby a jejich architektura. VUTIUM Brno, 2014, ISBN 978-80-214-3766-1.

[3] SIP: Session Initiation Protocol. *IETF Tools* [online]. [cit. 2017-12-11].

[4] Roy, Radhika Ranjan. Handbook on Session Initiation Protocol: Networked Multimedia Communications for IP Telephony. ISBN 978-149-8747-707.

[5] H.323 Uniform Resource Locator (URL) Scheme Registration. IETF Tools [online]. [cit. 2017-12-11]. Dostupné z: https://tools.ietf.org/html/rfc3508.

[6] FILE TRANSFER PROTOCOL (FTP). IETF Tools [online]. [cit. 2017-11-11]. Dostupné z: https://tools.ietf.org/html/rfc959.

[7] Kurose J. F., Ross K.W.:Computer Networking, A Top-Down Approach Featuring the Internet. Addison-Wesley, 2013, ISBN 978-0133594140.

[8] Post Office Protocol - Version 3. IETF Tools [online]. [cit. 2017-11-11]. Dostupné z: https://tools.ietf.org/html/rfc1939.

[9] Internet Message Access Protocol – version 4rev1. IETF Tools [online]. [cit. 2017-11-11]. Dostupné z: https://tools.ietf.org/html/rfc3501.

[10] Simple Mail Transfer Protocol. IETF Tools [online]. [cit. 2017-11-11]. Dostupné z: https://tools.ietf.org/html/rfc5321.

[11] Internationalized Domain Names. ICANN [online]. [cit. 2017-12-11]. Dostupné z: https://www.icann.org/resources/pages/idn-2012-02-25-en.

[12] Overview and Framework for Internationalized Email. IETF Tools [online]. [cit. 2017-11-11]. Dostupné z: https://tools.ietf.org/html/rfc6530.

[13] Internationalization of the File Transfer Protocol. IETF Tools [online]. [cit. 2017-11-11]. Dostupné z: https://tools.ietf.org/html/rfc2640.

[14] SMTP Extension for Internationalized Email. IETF Tools [online]. [cit. 2017-11-11]. Dostupné z: https://tools.ietf.org/html/rfc6531.

[15] Punycode: A Bootstring Encoding of Unicode for Internationalized Domain Names in Applications (IDNA). IETF Tools [online]. [cit. 2017-11-11]. Dostupné z: https://tools.ietf.org/html/rfc3492.

[16] Post Office Protocol Version 3 (POP3) Support for UTF-8. IETF Tools [online]. [cit. 2017-11-11]. Dostupné z: https://tools.ietf.org/html/rfc6856.

[17] IMAP Support for UTF-8. IETF Tools [online]. [cit. 2017-11-11]. Dostupné z: https://tools.ietf.org/html/rfc6855.

[18] Council Resolution of 17 January 1995 on the Lawful Interception of Telecommunications (96/C 329/01). 1996.

[19] Polčák Libor, Martínek Tomáš, Hranický Radek, Bárta Stanislav, Holkovič Martin, Franková Barbora and Kramoliš Petr. Zákonné odposlechy v moderních sítích-Shrnutí výsledků skupiny pro zákonné odposlechy projektu Moderní prostředky pro boj s kybernetickou kriminalitou na Internetu nové generace. FIT-TR-2014-07, Brno: Fakulta Informačních Technologií VUT, 2014.

[20] E-Detective. Decision Group Inc. [online]. [cit. 2017-12-11]. Dostupné z: http://www.edecision4u.com/E-DETECTIVE.html.

[21] Intercept Communications — Lawfully. Utimaco [online]. [cit. 2017-12-11]. Dostupné z: https://lims.utimaco.com/products/lawful-interception-management-system/.

[22] Unicode HOWTO. *Python* [online]. [cit. 2018-04-29]. Dostupné z: https://docs.python.org/2/howto/unicode.html.

Seznam příloh

Příloha 1 Testování podpory diakritiky v různých SIP Serverech

Switchvox

Verze	6.1.2
Test 1	V administrátorském portálu založení účtu uživatele s diakritikou.
Výsledek	Jméno a příjmení uživatele lze zadat s diakritikou, avšak jeho SIP číslo (extension) mohou být pouze číslice. Počet těchto číslic a další nastavení lze zvolit. V uživatelském portálu se lze pod tímto číslem přihlásit.
Test 2	Vyhledávání v seznamu kontaktů obsahujících diakritiku.
Výsledek	Vyhledávání kontaktů s diakritikou probíhá bez problémů, stejně jako zobrazování jmen.
Práce s diakritikou?	Ne, pouze zobrazování a vyhledávání ve jménech uživatelů systému.

Asterisk

Verze	14.5.0
Test 1	Na Asterisku vytvoření účtu (SIP adresy) uživatele s diakritikou.
Výsledek	Účet „talašová@10.0.2.15“ byl úspěšně vytvořen.
Test 2	Přihlášení se pomocí klienta Zoiper na účet „talašová@10.0.2.15“.
Výsledek	Klient se bez problému přihlásil.
Test 3	Volání z účtu „talašová@10.0.2.15“ na účet bez diakritiky.
Výsledek	Lze bez problému volat na nakonfigurovaného uživatele 100 na Asterisku.
Test 4	Volání z účtu „odesílatel@10.0.2.15“ na účet „talašová@10.0.2.15“
Výsledek	Volání proběhlo v pořádku (Asterisk hovor přijal), přenášely se UTF-8 znaky.
Test 5	Volání z účtu „odesílatel@10.0.2.15“ na účet „talašová“, do klienta zadaný jako „tala%c5%a1ov%c3%a1“.
Výsledek	Volání proběhlo v pořádku (Asterisk hovor přijal), diakritika se přenášela v hexadecimální podobě.
Test 6	Volání z účtu „odesílatel@10.0.2.15“ na účet „tala%c5%a1ová“.
Výsledek	Asterisk volání přijal, hovor proběhl v pořádku.
Práce s diakritikou?	Ano

Příloha 2 Testování podpory diakritiky v různých SIP klientech

Zoiper[18]

Verze	3.3
Test 1	Přihlášení se do aplikace pomocí účtu „talašová@10.0.2.15“ od Asterisku.
Výsledek	Přihlášení proběhlo v pořádku.
Testování diakritiky v uživatelské části adresy	Zmíněno v příloze 1, tabulka Asterisk.
Testování diakritiky v doménové části adresy	
Test 2	Volání na adresu „talas@talaš.net“
Výsledek	Byl zaslán DNS dotaz na „tala\305\241“ - přenáší se UTF-8 znak
Test 3	Volání na adresu „talas@xn--tala-j6a.net“
Výsledek	Byl zaslán DNS dotaz na „xn--tala-j6a.net“ - přenáší se Punycode
Test 4	Volání na adresu „talas@tala%c5%a1.net“
Výsledek	Byl zaslán DNS dotaz na adresu „tala%c5%a1,net“ - přenáší se hexadecimální podoba.
Test 5	Volání na adresu „talas@tala%c5%a1ová“.
Výsledek	Byl zaslán DNS dotaz na adresu „tala%c5%a1ov\303\241“.
Další testy	
Test 6	Vytváření kontaktů (jmen) s diakritikou a vyhledávání v adresáři.
Výsledek	Lze vytvářet kontakty s diakritikou ve jménech i následné vyhledávání v adresáři funguje bez problémů.
Práce s diakritikou?	Ano

Ekiga[19]

Verze	4.0
Test 1	Přihlášení se do aplikace pomocí účtu „talašová@10.0.2.15“ od Asterisku.
Výsledek	Přihlášení se nezdařilo, požadavek se neustále zpracovává bez pokroku.
Testování diakritiky v uživatelské části adresy	
Test 2	Volání na adresu „talašová“.
Výsledek	Došlo k překódování do hexadecimální podoby.
Test 3	Volání na adresu „tala%c5%a1“.
Výsledek	Došlo k překódování do hexadecimální podoby.
Test 4	Volání na adresu „tala%c5%a1ová“.

18 URL https://www.zoiper.com/en

19 URL http://www.ekiga.org/

Výsledek	Došlo k překódování do hexadecimální podoby.
Testování diakritiky v doménové části adresy	
Test 5	Volání na adresu „talašová“.
Výsledek	Došlo k překódování do UTF-8.
Test 6	Volání na adresu tala%“c5%a1“.
Výsledek	Došlo k překódování do UTF-8.
Test 7	Volání na adresu „xn--tala-j6a.net“.
Výsledek	Punycode zůstal.
Další testy	
Test 8	Vytváření kontaktů (jmen) s diakritikou a vyhledávání v adresáři.
Výsledek	Lze vytvářet kontakty s diakritikou ve jménech. Následné vyhledávání v adresáři chybí.
Práce s diakritikou?	Částečně

Linphone[20]

Verze	4.0
Test 1	Přihlášení se do aplikace pomocí účtu „talašová@10.0.2.15“ od Asterisku.
Výsledek	Přihlášení se nezdařilo, "Vaše SIP identita není platná".
Test 2	Volání na účet s diakritikou.
Výsledek	Volání na straně aplikace proběhlo v pořádku.
Test 3	Vytváření kontaktů (jmen) s diakritikou a vyhledávání v adresáři.
Výsledek	Lze vytvářet kontakty s diakritikou ve jménech. Následné vyhledávání v adresáři chybí.
Práce s diakritikou?	Částečně

SFLPhone[21]

Verze	1.3.0
Test 1	Přihlášení se do aplikace pomocí účtu „talašová@10.0.2.15“ od Asterisku.
Výsledek	Přihlášení se zdařilo.
Test 2	Volání na účet s diakritikou.
Výsledek	Volání na straně aplikace proběhlo v pořádku.
Test 3	Vytváření kontaktů (jmen) s diakritikou a vyhledávání v adresáři.
Výsledek	Lze vytvářet kontakty s diakritikou ve jménech. Následné vyhledávání v adresáři probíhá také bez problémů.
Práce s diakritikou?	Ano

20 URL http://www.linphone.org/

21 URL https://www.voip-info.org/wiki/view/SFLphone

Příloha 3 Testování podpory diakritiky u bezplatných SIP poskytovatelů

Onsip[22]

Test 1	Zvolení doménové části adresy "talašová.onsip.com".
Výsledek	Adresa nelze zadat, lze použít pouze malé znaky anglické abecedy a čísla.
Test 2	Zvolení uživatelské části adresy "talašová".
Výsledek	Adresa nelze zadat, pouze malé písmena anglické abecedy, číslice, pomlčky, tečky a podtržítka.
Práce s diakritikou?	NE

IPTel[23]

Poznámka	Doménová část nelze zvolit, pouze „iptel.org“.
Test 1	Zvolení uživatelské část adresy "talašová".
Výsledek	Adresa nelze zadat, lze použít pouze numerická adresa začínající "8" nebo alfanumerická adresa začínající znakem anglické abecedy.
Práce s diakritikou?	NE

Ekiga[24]

Poznámka	Doménová část nelze zvolit, pouze „ekiga.net“.
Test 1	Zvolení uživatelské části adresy "talašová".
Výsledek	Omezení pouze na numerickou adresu začínající "8" nebo alfanumerickou adresu začínající znakem anglické abecedy. Adresa "talašová@ekiga.net“ však zadat lze, avšak je špatně zakódována na „sip:talaÅ¡ovÃ¡@ekiga.net“.
Práce s diakritikou?	NE

SIP2SIP[25]

Poznámka	Doménová část nelze zvolit, pouze sip2sip.info.
Test 1	Zvolení uživatelské části adresy "talašová".
Výsledek	Adresa nelze zadat.
Práce s diakritikou?	NE

Antisip[26]

Poznámka	Doménová část nelze zvolit, pouze

22 URL https://www.onsip.com/

23 URL https://serweb.iptel.org/user/reg/index.php

24 URL https://ekiga.im/index.php?page=register

25 URL https://mdns.sipthor.net/register_sip_account.phtml

26 URL https://www.antisip.com/sip-antisip-com-register/

	„sip.antisip.com“.
Test 1	Zvolení uživatelské části adresy "talašová".
Výsledek	Adresa nelze zadat.
Práce s diakritikou?	NE

Příloha 4 Testování podpory diakritiky v různých FTP serverech

eFTP server

Verze	3.2.3.112
Licence	Free
Operační systém	Windows
Postup	Z klientské aplikace proběhl upload a následně download souboru "řčš.txt" a různých jiných souborů obsahujících diakritiku.
Test	Jako klientská aplikace byla zvolena aplikace eFTP klient.
Výsledek	Přenos souborů proběhl v pořádku, soubory byly ve správném kódování, žádný problém se nevyskytl.
Práce s diakritikou?	ANO
Lze zadat uživ. jméno a heslo s diakritikou?	ANO - byla jsem na server přihlášena jako uživatel "řč" s totožným heslem

Cerberus FTP server[27]

Verze	8
Licence	Free (nekomerční použití)
Operační systém	Windows
Postup	Z klientské aplikace proběhl upload a následně download souboru "řčš.txt" a různých jiných souborů obsahující diakritikou.
Test 1	Jako klientská aplikace byla zvolena aplikace Cerberus klient.
Výsledek 1	Přenos souborů proběhl v pořádku, soubory byly ve správném kódování, žádný problém se nevyskytl.
Test 2	Jako klientská aplikace byla zvolena aplikace FileZilla klient.
Výsledek 2	Přenos souborů proběhl v pořádku, soubory byly ve správném kódování, žádný problém se nevyskytl.
Práce s diakritikou?	ANO
Lze zadat uživ. jméno a heslo s diakritikou?	ANO

FileZilla server[28]

Verze	0.9.60.2
Licence	GNU GPL
Operační systém	Windows
Postup	Z klientské aplikace proběhl upload a následně download souboru

27 URL https://www.cerberusftp.com/

28 URL https://filezilla-project.org/

	"řčš.txt" a různých jiných souborů obsahující diakritiku.
Test 1	Jako klientská aplikace byla zvolena aplikace FileZilla klient.
Výsledek 1	Přenos souborů proběhl v pořádku, soubory byly ve správném kódování, žádný problém se nevyskytl.
Test 2	Jako klientská aplikace byla zvolena aplikace eFTP klient.
Výsledek 2	Přenos souborů proběhl v pořádku, ale soubory měly špatné kódování.
Práce s diakritikou?	ANO
Lze zadat uživ. jméno a heslo s diakritikou?	ANO

Wedos server[29]

Postup	Z klientské aplikace proběhl upload a následně download různých souborů obsahující diakritiku.
Test	Jako klientské aplikace byly zvoleny aplikace Dolphin, FileZilla,FireFTP, gFTp, Konqueror, Total Commander a další.
Výsledek	Přenos souborů proběhl v pořádku, soubory byly ve správném kódování, žádný problém se nevyskytl.
Práce s diakritikou?	ANO
Lze zadat uživ. jméno a heslo s diakritikou?	NE

29 URL https://webftp.wedos.net/

Příloha 5 Testování podpory diakritiky v různých FTP klientech

Dolphin[30]

Verze	4.14.3
Postup	Z aplikace Dolphin byl na server Wedos uploadován soubor "talašová.txt" a následně proběhl download souboru "řčš.txt".
Výsledek	Přenos souborů proběhl v pořádku, soubory byly ve správném kódování.
Práce s diakritikou?	ANO
Lze zadat uživ. jméno a heslo s diakritikou?	ANO

FileZilla klient[31]

Verze	3.26.2
Postup	Z aplikace FileZilla byl na server Wedos uploadován soubor "talašová.txt" a následně proběhl download souboru "řčš.txt".
Výsledek	Přenos souborů proběhl v pořádku, soubory byly ve správném kódování.
Práce s diakritikou?	ANO
Lze zadat uživ. jméno a heslo s diakritikou?	ANO

FireFTP[32]

Verze	2.0
Postup	Z aplikace fireFTP byl na server Wedos uploadován soubor "talašová.txt" a následně proběhl download souboru "talašová.txt".
Výsledek	Přenos souborů proběhl v pořádku, soubory byly ve správném kódování.
Práce s diakritikou?	ANO
Lze zadat uživ. jméno a heslo s diakritikou?	ANO

30 URL https://www.kde.org/applications/system/dolphin/

31 URL https://filezilla-project.org/

32 URL http://fireftp.net/

gFTP [33]

Verze	2.0.19
Postup	Z aplikace gFTP byl na server Wedos uploadován soubor "talašová.txt" a následně proběhl download souboru "talašová.txt".
Výsledek	Přenos souborů proběhl v pořádku, soubory byly ve správném kódování.
Práce s diakritikou?	ANO
Lze zadat uživ. jméno a heslo s diakritikou?	ANO

Konqueror[34]

Verze	4.14.3
Postup	Z aplikace Konqueror byl na server Wedos uploadován soubor "talašová.txt" a následně proběhl download souboru "talašová.txt".
Výsledek	Přenos souborů proběhl v pořádku, soubory byly ve správném kódování.
Práce s diakritikou?	ANO
Lze zadat uživ. jméno a heslo s diakritikou?	ANO

Total Commander[35]

Verze	9.0a
Postup	Z aplikace Total Commander byl na server Wedos uploadován soubor "talašová.txt" a následně proběhl download souboru "talašová.txt".
Výsledek	Přenos souborů proběhl v pořádku, soubory byly ve správném kódování.
Práce s diakritikou?	ANO
Lze zadat uživ. jméno a heslo s diakritikou?	ANO

33 URL https://www.gftp.org/

34 URL https://konqueror.org/

35 URL https://www.ghisler.com/

WS_FTP[36]

Verze	12.5
Postup	Z aplikace WS_FTP byl na server Wedos uploadován soubor "talašová.txt" a následně proběhl download souboru "talašová.txt".
Výsledek	Přenos souborů proběhl v pořádku, soubory byly ve správném kódování.
Práce s diakritikou?	ANO
Lze zadat uživ. jméno a heslo s diakritikou?	ANO

-

eFTP[37]

Verze	3.2.3.112
Postup	Z aplikace eFTP byl na server eFTP uploadován soubor "talašová.txt" a následně proběhl download souboru "řčš.txt".
Výsledek	Přenos souborů proběhl v pořádku, soubory byly ve správném kódování.
Práce s diakritikou?	ANO
Lze zadat uživ. jméno a heslo s diakritikou?	ANO

Cerberus[38]

Verze	8.0
Postup	Z aplikace Cerberus byl na server Cerberus uploadován soubor "talašová.txt" a následně proběhl download souboru "řčš.txt".
Výsledek	Přenos souborů proběhl v pořádku, soubory byly ve správném kódování.
Práce s diakritikou?	ANO
Lze zadat uživ. jméno a heslo s diakritikou?	ANO

36 URL http://www.wsftple.com/

37 URL https://en.wikipedia.org/wiki/EFTP

38 URL https://www.cerberusftp.com/

Příloha 6 Testování podpory diakritiky v různých SMTP serverech

Postfix[39]

Verze	3.1.0
Podpora EAI	ANO - s podporou SMTPUTF8
Podpora IDN	ANO - s podporou SMTPUTF8

Test 1	Test odeslání e-mailu s diakritikou s podporou UTF-8.
MAIL FROM:	talašová@talašová.cz.
RCPT TO:	talasova.irena@gmail.com.
Výsledek	Zpráva byla odeslána a doručena příjemci.
Test 2	Test odeslání e-mailu s diakritikou s podporou UTF-8.
MAIL FROM:	talasova@talašová.cz
RCPT TO:	student@localhost
Výsledek	Zpráva byla odeslána a doručena uživateli student.

Exim[40]

Verze	4.88
Podpora EAI	ANO - s podporou SMTPUTF8, avšak s problémy
Podpora IDN	ANO - s podporou SMTPUTF8, avšak s problémy

Test 1	Test odeslání e-mailu s diakritikou bez podpory UTF-8.
MAIL FROM:	talašová@talašová.cz
RCPT TO:	talasova.irena@gmail.com_
Výsledek	Nelze zadat takovouto adresu odesílatele.
Test 2	Test odeslání e-mailu s diakritikou s podporou UTF-8.
MAIL FROM:	talašová@talašová.cz
RCPT TO:	student@localhost
Výsledek	Zpráva byla odeslána a doručena příjemci.

Qmail[41]

Verze	1.06
Podpora EAI	NE
Podpora IDN	NE

Test 1	Test odeslání e-mailu s diakritikou.

39 URL http://www.postfix.com/

40 URL http://www.exim.org/

41 URL http://www.qmail.org/

MAIL FROM:	talasova@talasova.cz
RCPT TO:	jøran@blåbærsyltetøy.gulbrandsen.priv.no
Výsledek	Zprávu nelze odeslat.
Test 2	Test odeslání e-mailu s diakritikou.
MAIL FROM:	talasova@talasova.cz
RCPT TO:	testmail@háčkyčárky.cz
Výsledek	Zprávu nelze odeslat („sorry i couldn't find any host named h????ck???? rky.cz“).

Sendmail[42]

Verze	8.15.2
Podpora EAI	NE
Podpora IDN	NE

Test 1	Test odeslání e-mailu s diakritikou.
MAIL FROM:	talašová@talasova.cz
RCPT TO:	talasova.irena@gmail.com
Výsledek	Zprávu nelze odeslat („syntax error“).
Test 2	Test odeslání e-mailu s diakritikou.
MAIL FROM:	talasova@talasova.cz
RCPT TO:	jøran@blåbærsyltetøy.gulbrandsen.priv.no
Výsledek	Zprávu nelze odeslat („syntax error“).

42 URL http://www.sendmail.com/

Příloha 7 Testování podpory diakritiky ve freemailových službách

Seznam.cz

Test 1	Z postfixu byl odeslán e-mail s adresou („talašová@talasova.cz“) na adresu e-mailové schránky od „seznam.cz“.
Výsledek	E-mail nebyl doručen do schránky od „seznam.cz“.
Test 2	Pokud se snažíme odeslat e-mail na adresu „talašová@talasova.cz“ - nelze odeslat "Adresát je neplatný".
Test 3	Skrze webové GUI byla odeslaná zpráva s předmětem IDN na email „testmail@háčkyčárky.cz“.
Výsledek	Zpráva nejde odeslat.
Lze založit schránku s diakritikou?	NE
Podpora EAI	NE
Podpora IDN	NE

Centrum.cz, Atlas.cz, Volny.cz

Test 1	Z postfixu byl odeslán e-mail s adresou („talašová@talasova.cz“) na adresu e-mailové schránky od „centrum.cz“, „atlas.cz“ a „volny.cz“.
Výsledek	E-maily nebyly doručeny do schránek.
Test 2	Pokud se snažíme odeslat e-mail na adresu „talašová@talasova.cz“, e-mail nelze odeslat "Chybně zadaná adresa".
Test 3	Z mobilních verzí jednotlivých stránek také není možno odeslat e-mail na EAI adresu - "adresát je neplatný".
Test 4	Skrze webové GUI byla odeslaná zpráva s předmětem IDN na email „testmail@háčkyčárky.cz“.
Výsledek	Zpráva nebyla odeslána. Zobrazilo se chybové hlášení "Chybně zadaná adresa".
Lze založit schránku s diakritikou?	NE
Podpora EAI	NE
Podpora IDN	Ne

Mobilní verze Centrum.cz, Atlas.cz a Volny.cz

Test 1	Pokud se snažíme odeslat e-mail na adresu „talašová@talasova.cz“ - e-mail nelze odeslat "Chybně zadaná adresa".
Test 2	Skrze mobilní webové GUI byla odeslaná zpráva s předmětem IDN na

	email „testmail@háčkyčárky.cz“.
Výsledek	Zpráva byla úspěšně odeslána. Vrátila se i úspěšná potvrzující odpověď serveru "IDN OK". Adresa odesílatele se zobrazila jako „testmail@xn—hkyrky-ptac70bc.cz“.
Lze založit schránku s diakritikou?	NE
Podpora EAI	NE
Podpora IDN	ANO

Eposta.cz

Test 1	Z postfixu byl odeslán e-mail s adresou („talašová@talasova.cz“) na adresu e-mailové schránky od „eposta.cz“.
Výsledek	E-mail nebyl doručen do schránky.
Test 2	Pokud se snažíme odeslat e-mail na adresu talašová@talasova.cz – e-mail nelze odeslat "Zadejte příjemce".
Test 3	Skrze webové GUI byla odeslaná zpráva s předmětem IDN na email „testmail@háčkyčárky.cz“.
Výsledek	Zpráva byla úspěšně odeslána, avšak nevrací se potvrzující email "IDN Ok".
Lze založit schránku s diakritikou?	NE
Podpora EAI	NE
Podpora IDN	NE

Tiscali.cz

Test 1	Z postfixu byl odeslán e-mail s adresou („talašová@talasova.cz“) na adresu e-mailové schránky od „tiscali.cz“.
Výsledek	E-mail nebyl doručen do schránky.
Test 2	Pokud se snažíme odeslat e-mail na adresu „talašová@talasova.cz“ - e-mail nelze odeslat na adresu s diakritikou.
Test 3	Skrze webové GUI byla odeslaná zpráva s předmětem IDN na email „testmail@háčkyčárky.cz“.
Výsledek	Zpráva nelze odeslat.
Lze založit schránku s diakritikou?	NE
Podpora EAI	NE
Podpora IDN	NE

Yahoo.com

Test 1	Z postfixu byl odeslán e-mail s adresou („talašová@talasova.cz“) na adresu e-mailové schránky od „yahoo.com“.
Výsledek	E-mail nebyl doručen do schránky.

Test 2	Pokud se snažíme odeslat e-mail na adresu „talašová@talasova.cz“ - e-mail nelze odeslat - „zadejte příjemce“.
Test 3	Skrze webové GUI byla odeslaná zpráva s předmětem IDN na email „testmail@háčkyčárky.cz“.
Výsledek	Zpráva nejde odeslat.
Lze založit schránku s diakritikou?	NE
Podpora EAI	NE
Podpora IDN	NE

Gmail.com

Test 1	Z postfixu byl odeslán e-mail s adresou („talašová@talasova.cz“) na adresu e-mailové schránky od „gmail.com“.
Výsledek	E-mail byl úspěšně doručen do schránky.
Test 2	Lze odeslat e-mail na adresu s diakritikou. V domnění, že jde o jeden z dalších automatických odpovídačů, jsem odeslala mail na adresu "مثال@مثال.السعودية„. Odepsal mi Arnt Gulbrandsen s ochotou pomoci. Email tedy dorazil v pořádku.
Test 3	Skrze webové GUI byla odeslaná zpráva s předmětem IDN na email „testmail@háčkyčárky.cz“.
Výsledek	Zpráva byla úspěšně odeslána. Vrátila se úspěšná potvrzující odpověď "IDN OK", adresa odesílatele je ve správném tvaru s diakritikou.
Lze založit schránku s diakritikou?	NE
Podpora EAI	ANO
Podpora IDN	ANO

Hotmail.com

Test 1	Z postfixu byl odeslán e-mail s adresou („talašová@talasova.cz“) na adresu e-mailové schránky od „hotmail.com“.
Výsledek	E-mail nebyl doručen.
Test 2	Lze odeslat e-mail na adresu s diakritikou, zpráva se však nedoručí - „Outlook.com“ ji nepustí dál.
Test 3	Skrze webové GUI byla odeslaná zpráva s předmětem IDN na email „testmail@háčkyčárky.cz“.
Výsledek	Zpráva byla odeslána, ale nevrátila se úspěšná odpověď.
Lze založit schránku s diakritikou?	NE
Podpora EAI	NE
Podpora IDN	NE

Příloha 8 Testování podpory diakritiky v různých POP3/IMAP serverech

Dovecot[43]

Test 1	Vyhledání uložených zpráv obsahující diakritiku v lokální části.
Výsledek	Vyhledávání proběhlo v pořádku.
Test 2	Přihlášení se k serveru jako uživatel „talašová@talasova.cz“.
Výsledek	Server odmítl s chybovou hláškou "BAD 8bit data in atom".
Test 3	Vyhledání a práce se zprávami obsahujícími diakritiku v poli MAIL FROM i RCPT TO.
Výsledek	Vyhledávání proběhlo v pořádku.
Podpora EAI	ČÁSTEČNĚ (ano vyhledávání, ne přihlášení).
Podpora IDN	ANO

MS Exchange Server[44]

Test 1	Vyhledání a manipulace s uloženými zprávami obsahujícími diakritiku.
Výsledek	Vyhledávání a přesouvání proběhlo v pořádku.
Test 2	Vytvoření účtu s diakritikou.
Výsledek	Nelze, špatné znaky.
Podpora EAI	ČÁSTEČNĚ (ano vyhledávání, ne vytvoření účtu)
Podpora IDN	ANO

Cyrus[45]

Test 1	Vyhledání a manipulace s uloženými zprávami obsahujícími diakritiku.
Výsledek	Vyhledávání se nezdařilo.
Test 2	Vytvoření účtu s diakritikou.
Výsledek	Vytvoření účtu proběhlo v pořádku.
Test 3	Přihlášení se pomocí účtu s diakritikou.
Výsledek	Přihlášení se nezdařilo.
Podpora EAI	NE
Podpora IDN	NE

Courier[46]

Test 1	Vyhledání a manipulace s uloženými zprávami obsahujícími diakritiku.
Výsledek	Vyhledávání a přesouvání proběhlo v pořádku.
Test 2	Vytvoření účtu s diakritikou.

43 URL https://www.dovecot.org/

44 URL https://products.office.com/cs-cz/exchange/microsoft-exchange-server-2016

45 URL https://cyrusimap.org/

46 URL http://www.courier-mta.org/

Výsledek	Lze vytvořit takovýto účet.
Test 3	Přihlášení se pomocí účtu s diakritikou.
Výsledek	Přihlášení proběhlo úspěšně.
Podpora EAI	ANO
Podpora IDN	ANO

Příloha 9 Testování podpory diakritiky v různých POP3/IMAP klientech

Thunderbird[47]

Test 1	Odeslání zprávy na adresu „talašová@talasova.cz“.
Výsledek	E-mail nelze odeslat - „There are non-ASCII characters in the local part of the recipient address“.
Test 2	Odeslání zprávy na adresu bez diakritiky z adresy s diakritikou.
Výsledek	Odeslání proběhlo v pořádku.
Test 3	Odeslání zprávy na adresu „testmail@háčkyčárky.cz“.
Výsledek	Odeslání proběhlo v pořádku.
Podpora EAI	ANO v poli MAIL FROM, NE v poli RCPT TO
Podpora IDN	ANO

Roundcube Webmail[48]

Test 1	Odeslání zprávy na adresu „talašová@talasova.cz“.
Výsledek	Nelze odeslat.
Test 2	Odeslání zprávy na adresu bez diakritiky z adresy s diakritikou.
Výsledek	Nelze odeslat.
Test 3	Odeslání zprávy na adresu „testmail@háčkyčárky.cz“.
Výsledek	Odeslání proběhlo v pořádku.
Podpora EAI	NE
Podpora IDN	ANO

Pošta (Windows)

Test 1	Odeslání zprávy na adresu „talašová@talasova.cz“.
Výsledek	Nelze odeslat.
Test 2	Odeslání zprávy na adresu „testmail@háčkyčárky.cz“.
Výsledek	Odeslání proběhlo v pořádku.
Podpora EAI	NE
Podpora IDN	ANO

Outlook 2016[49]

Test 1	Odeslání zprávy na adresu „talašová@talasova.cz“.
Výsledek	Odeslání proběhlo v pořádku.
Test 2	Odeslání zprávy na adresu bez diakritiky z adresy s diakritikou.

47 URL https://www.mozilla.org/cs/thunderbird/

48 URL https://roundcube.net/

49 URL https://support.office.com/cs-cz/article/Nastaven%C3%AD-e-mailu-v-aplikaci-Po%C5%A1ta-pro-Windows-10-7ff79e8b-439b-4b47-8ff9-3f9a33166c60

Výsledek	Odeslání proběhlo v pořádku.
Test 3	Odeslání zprávy na adresu „testmail@háčkyčárky.cz“.
Výsledek	Odeslání proběhlo v pořádku.
Podpora EAI	ANO
Podpora IDN	ANO

Mutt[50]

Test 1	Odeslání zprávy na adresu „talašová@talasova.cz“.
Výsledek	Odeslání proběhlo v pořádku.
Test 2	Odeslání zprávy na adresu bez diakritiky z adresy s diakritikou.
Výsledek	Odeslání proběhlo v pořádku.
Test 3	Odeslání zprávy na adresu „testmail@háčkyčárky.cz“.
Výsledek	Odeslání proběhlo v pořádku.
Podpora EAI	ANO
Podpora IDN	ANO

50 URL http://www.mutt.org/

Příloha 10 Obsah paměťového média

Na paměťovém médiu přiloženém k této práci se nachází kompletní zdrojový kód včetně všech mnou provedených změn systému SLIS, GUI a testovacích souborů. Na paměťovém médiu se nachází soubory:

- *l7tester* – tento adresář obsahuje zdrojový kód testovacího prostředí,
- *slis* – zdrojový kód systému SLIS,
- *www* – adresář obsahující zdrojový kód GUI,