Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

BAKALÁŘSKÁ PRÁCE

Webová aplikace pro vizualizaci a práci se sítí důvěry veřejných klíčů rodiny PGP

2024

Petr Gajdošík

Vedoucí práce:
Mgr. Radek Janoštík, Ph.D.

Studijní program: Informatika,
Specializace: Programování a vývoj software

Bibliografické údaje

Autor:	Petr Gajdošík
Název práce:	Webová aplikace pro vizualizaci a práci se sítí důvěry veřejných klíčů rodiny PGP
Typ práce:	bakalářská práce
Pracoviště:	Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci
Rok obhajoby:	2024
Studijní program:	Informatika, Specializace: Programování a vývoj software
Vedoucí práce:	Mgr. Radek Janoštík, Ph.D.
Počet stran:	39
Přílohy:	elektronická data v úložišti katedry informatiky
Jazyk práce:	český

Bibliographic info

Author:	Petr Gajdošík
Title:	A web application for visualization and management of the PGP-family's public keys web of trust
Thesis type:	bachelor thesis
Department:	Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense:	2024
Study program:	Computer Science, Specialization: Programming and Software Development
Supervisor:	Mgr. Radek Janoštík, Ph.D.
Page count:	39
Supplements:	electronic data in the storage of department of computer science
Thesis language:	Czech

Anotace

Svět digitálního komunikačního prostředí vyžaduje zabezpečení dat a ověření identity. Koncept síťové důvěry v rámci PGP (Pretty Good Privacy) přináší možnost ověřování identity pomocí digitálních podpisů a důvěry přímo mezi uživateli. Existující nástroje, které implementují PGP, však tento koncept často nezahrnují. Součásti této práce je tedy webové aplikace, která umožní uživatelům vizualizovat síť důvěry a spravovat ji. Práce dále analyzuje teoretické základy asymetrické kryptografie, principy sítě důvěry a problémy spojené s centralizovanými řešeními ověřování identity.

Synopsis

The digital communication environment requires data security and identity verification. The concept of web of trust within PGP (Pretty Good Privacy) brings the possibility of identity verification using digital signatures and trust directly between users. Existing tools that implement PGP, however, often do not include this concept. Part of this work is therefore a web application that allows users to visualize and manage the web of trust. The work further analyzes the theoretical foundations of asymmetric cryptography, the principles of the web of trust, and the problems associated with centralized identity verification solutions.

Klíčová slova: webová aplikace; síť důvěry; vizualizace sítě; PGP, OpenPGP, GNU Privacy Guard, Ruby on Rails, Neo4j

Keywords: web application; web of trust; network visualization; PGP, OpenPGP, GNU Privacy Guard, Ruby on Rails, Neo4j

Rád bych vyjádřil upřímné poděkování svému vedoucímu bakalářské práce Mgr. Radku Janoštíkovi, Ph.D. za jeho cenné rady a trpělivost během celého procesu vzniku bakalářské práce. Také bych rád poděkoval své rodině za jejich neustálou podporu. Bez nich by tato práce nevznikla.

Odevzdáním tohoto textu jeho autor/ka místopřísežně prohlašuje, že celou práci včetně příloh vypracoval/a samostatně a za použití pouze zdrojů citovaných v textu práce a uvedených v seznamu literatury.

Obsah

Seznam obrázků

Seznam zdrojových kódů

1 Úvod

V moderním světě závislém na informacích je zajištění jejich bezpečné výměny klíčovou prioritou. Je potřeba zajistit, aby data zůstala neporušena a aby nedošlo k přístupu neoprávněnou osobou k citlivým informacím. Díky kryptografickým nástrojům jsme schopni dosáhnout spolehlivého koncového šifrování. Jak ale můžeme vědět, že osoba, se kterou komunikujeme, je opravdu ta, kterou jsme chtěli kontaktovat.

Jedním z možných řešení je koncept *sítě důvěry* (angl. *web of trust*). V kontextu Pretty Good Privacy (PGP) je to decentralizovaný model, kde uživatelé mohou ověřit identitu jiných uživatelů pomocí digitálních podpisů. Tento model je založen na důvěře mezi uživateli a vzájemném ověřování jejich identit. [1] Pro úspěšné fungování sítě důvěry je klíčové, aby uživatelé měli důvěru v ostatní členy a aby byli ochotni aktivně ověřovat jejich identitu.

1.1 Motivace

V současné době existuje mnoho nástrojů, které umožňují uživatelům pracovat s PGP klíči. Tyto nástroje jsou ale často složité a často vůbec neimplementují síť důvěry. Pro běžného uživatele je tak obtížné pracovat s tímto konceptem a mnoho uživatelů se ho tak vůbec nedotkne. Jmenovitě se jedná např. o nástroje:

- GNU Privacy Guard (GPG) – nejznámější implementace PGP; je ale příkazový nástroj a pro běžného uživatele obtížně ovladatelný; implementuje ale síť důvěry [2],
- Keybase – webová aplikace spojující koncové šifrování s vlastnostmi sociální sítě; umožňuje profil provazovat s externími identitami; neimplementuje síť důvěry [3],
- ProtonMail, Enigmail a další – emailové programy s koncovým šifrováním; provazuje klíč pouze s emailovou identitou; umožňují uživatelům vytvářet a ověřovat klíče; neumožňují spravovat síť důvěry [4] [5].

Vytvoření nástroje, který by umožnil s tímto konceptem pracovat, zatímco uživatelé nejsou nuceni opustit prostředí pro správu klíčů, na které jsou zvyklí, by mohlo zvýšit přístupnost sítě důvěry.

1.2 Cíle práce

Cílem této práce je vytvořit webovou aplikaci, která umožní pracovat se sítí důvěry veřejných klíčů rodiny PGP. Aplikace dovolí vizualizovat síť, vyhledávat v ní a nalézt nejkratší cestu důvěry k jiným uživatelům. Po ověření, zda je uživatel držitelem privátního klíče, mu bude umožněno upravovat vlastní profil v síti.

2 Teoretická část

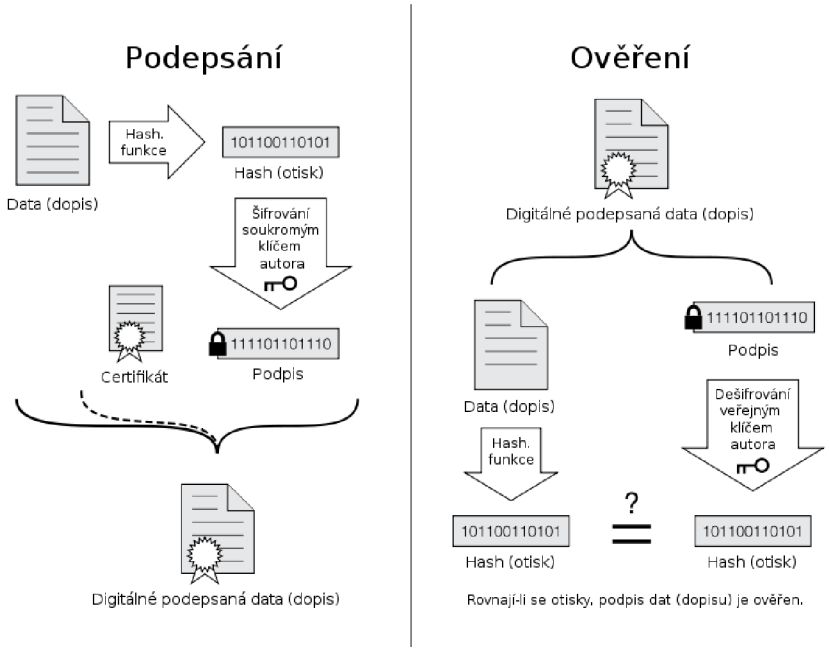
V této části se zaměříme na teoretické základy, které jsou nezbytné pro pochopení problematiky sítí důvěry a práce s PGP klíči. V neposlední řadě se zaměříme na problémy centralizovaných řešení a možnosti decentralizace digitálních identit.

2.1 Principy asymetrické kryptografie

Asymetrická kryptografie je metoda šifrování, která používá dva klíče. Pomocí veřejného klíče jsou data zašifrována, pomocí soukromého klíče dešifrována. Veřejný klíč se používá pro šifrování informací zasílaných majiteli soukromého klíče. Zprávy, které jsou zašifrovány uživatelovým veřejným klíčem, lze dešifrovat jen jeho soukromým klíčem.

Odpadá potřeba přenosu tajného klíče nějakým bezpečným kanálem jako u symetrického šifrování, veřejný klíč může být přenášen bez omezení, soukromý klíč se nikam nepřenáší. Majitel soukromého klíče může tento klíč použít pro svou **jednoznačnou identifikaci**, tzn. že odesílanou zprávu zašifruje svým soukromým klíčem a příjemce ji dešifruje jeho veřejným klíčem, jehož pravost potvrzuje *digitální certifikát*. [6]

Pro potřeby této práce se zaměříme na použití asymetrické kryptografie v kontextu digitálních podpisů. Jde o proces, kdy uživatel vytvoří digitální podpis zprávy pomocí svého soukromého klíče. Tento podpis je následně ověřitelný pomocí veřejného klíče uživatele. Můžeme tak kryptograficky ověřit, že zpráva byla skutečně odeslána uživatelem, který tvrdí, že ji odeslal. Podpisy taky mimo jiné dovolují ověřit integritu zprávy, tzn. že zpráva nebyla po odeslání změněna.



Obrázek 1: Princip podpisů v asymetrické kryptografii [7]

Proces vytvoření digitálního podpisu je znázorněn na obrázku 1. V principu jde o vytvoření otisku zprávy (pomocí hashovací funkce) a následné zašifrování tohoto otisku soukromým klíčem. Ověření podpisu probíhá v opačném směru dešifrováním otisku veřejným klíčem a porovnáním s otiskem zprávy.

2.2 Rodina klíčů PGP

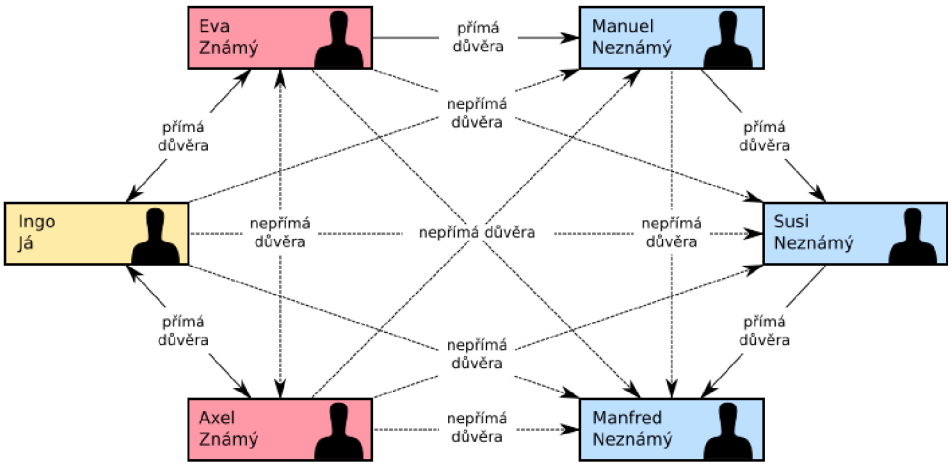
Pretty Good Privacy (PGP) je šifrovací program založený na asymetrické kryptografii. Odesílatel zašifruje zprávu veřejným klíčem adresáta, zpráva je pak dešifrovatelná jen soukromým klíčem adresáta. Může se používat samostatně pro zabezpečení souborů před jejich přenosem nebo v kombinaci s elektronickou poštou pro ochranu zasílaných dopisů před odposlechem. PGP rovněž nabízí funkci digitálního podpisu. [8]

I když je PGP komerčním produktem od PGP Inc., verze tohoto softwaru s použitím výhradně otevřených algoritmů byla představena jako Request for Comments (RFC) pod číslem RFC4880. Tato verze je známá jako *OpenPGP* a je implementována v mnoha dalších programech. [9]

Jednou z nejznámějších implementací je GNU Privacy Guard (GPG). Jedná se o svobodný software, který je kompatibilní mimo jiné i s OpenPGP. Je dostupný pro většinu operačních systémů a je často používán jako náhrada za komerční PGP. [10]

2.3 Síť důvěry

Síť důvěry je koncept, který umožňuje ověřit identitu jiných uživatelů pomocí digitálních podpisů. Uživatelé mohou vytvářet *vztahy důvěry* vzájemně a tím vytvářet síť. Síť důvěry je součástí PGP a umožňuje uživatelům ověřit, že klíč patří skutečné osobě. [11]



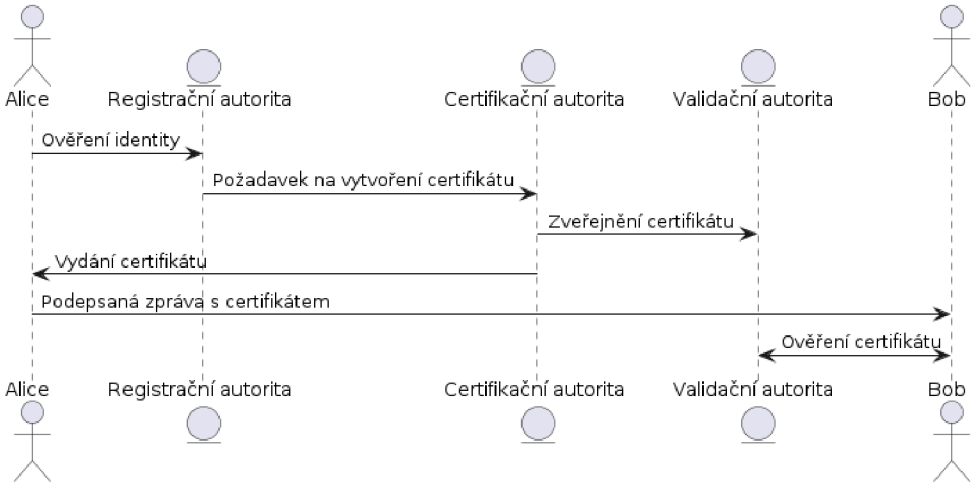
Obrázek 2: Diagram sítě důvěry [12]

Na obrázku 2 můžeme vidět příklad sítě důvěry. Lze pozorovat, že ačkoliv uživatel Ingo nemá přímý vztah s uživatelem Manuel, může mu nepřímo důvěřovat díky vztahu s uživatelem Eva. Díky konceptu, který lze laicky popsat jako *přátelé mých přátel jsou mými přáteli*, můžeme získat důvěru v identity, které nejsou přímo ověřeny.

Ačkoliv se tento koncept může zdát jako neškálovatelný, *teorie šesti stupňů separace*[13] nám říká, že každý člověk na světě je spojen s každým jiným prostřednictvím nejvýše šesti známých. V teoretickém případě, kdy by existovala jedna síť důvěry pro celou planetu, bychom mohli ověřit identitu každého člověka na světě za pomocí nejvýše šesti kroků.

2.4 Problémy centralizovaných řešení

Za centralizované řešení ověření identity se rozumí Public Key Infrastructure (PKI) – infrastruktura, která zajišťuje vytváření, distribuci a správu digitálních certifikátů. [14] Funguje na principu existence centrální autority, která vydává certifikáty a ověřuje identity. Uživatel, který důvěřuje centrální autoritě, může důvěřovat i certifikátům, které tato autorita vydala. Z toho vyplývá, že pokud důvěřujeme autoritě, můžeme důvěřovat i identitám, které tato autorita ověřila.



Obrázek 3: Diagram vytvoření a ověření certifikátu

Na obrázku 3 můžeme vidět, jak Bob přijal zprávu od Alice, která obsahuje certifikát. Tento certifikát Alici vydala Certifikační autorita (CA) na základě ověření její identity skrze Registrační autorita (RA). Bob může ověřit pravost certifikátu pomocí Validační autorita (VA), která ověří, zda certifikát byl vydán autoritou, které důvěřuje.

Fáze ověření identity je přirozeně velmi citlivá. A v závislosti na tom, kdo centrální autoritu provozuje, může být zneužita ke sledování uživatelů ať už státními organizacemi, nebo soukromými firmami. V případě, že autorita ztratí důvěru, ztrácí ji i všechny certifikáty, které vydala. [15]

V neposlední řadě je zde problém s dostupností centrální autority. Pokud by autorita přestala fungovat, ztratili bychom možnost ověřit identitu uživatele, který má certifikát vydán touto autoritou. [16]

3 Analýza a návrh

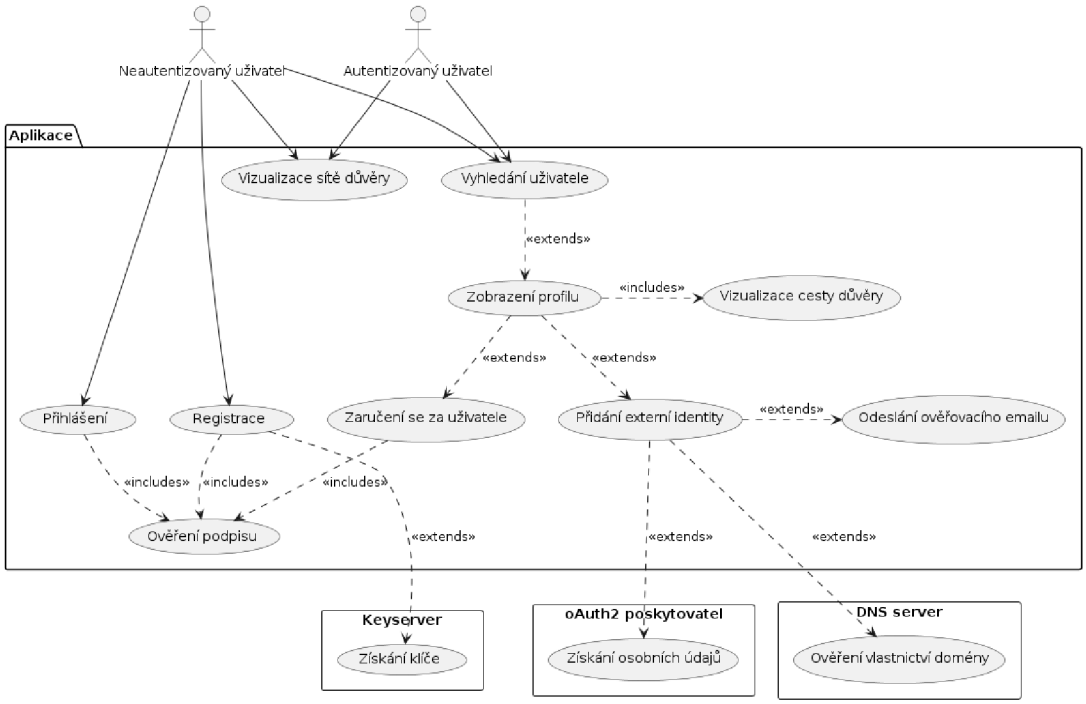
Z důvodů výše zmíněných problémů, ale zároveň s cílem zvýšit popularitu a využitelnost sítě důvěry, jsem se rozhodl vytvořit podpůrnou webovou aplikaci pro její správu.

Takové řešení je ze své podstaty centralizované, a proto je nutné ji navrhnout tak, aby:

1. bylo co nejvíce transparentní,
2. aby bylo možné samostatně ověřovat platnost certifikátů,
3. aby si uživatel mohl vést vlastní síť důvěry nezávisle na aplikaci,
4. aby uložiště klíčů bylo standardizované a umožňovalo administrátorům snadný přenos klíčů do jiných aplikací.

3.1 Analýza potřeb uživatelů

Byla provedena neformální analýza požadavků na funkcionalitu vznesených na různých platformách a fórech. Tito uživatelé byli vesměs technicky znalí a v častých případech již byly uživateli programů implementujícíc OpenPGP. Požadavky jsou shrnuty v následujícím diagramu případů užití na obrázku 4.



Obrázek 4: Diagram případů užití aplikace

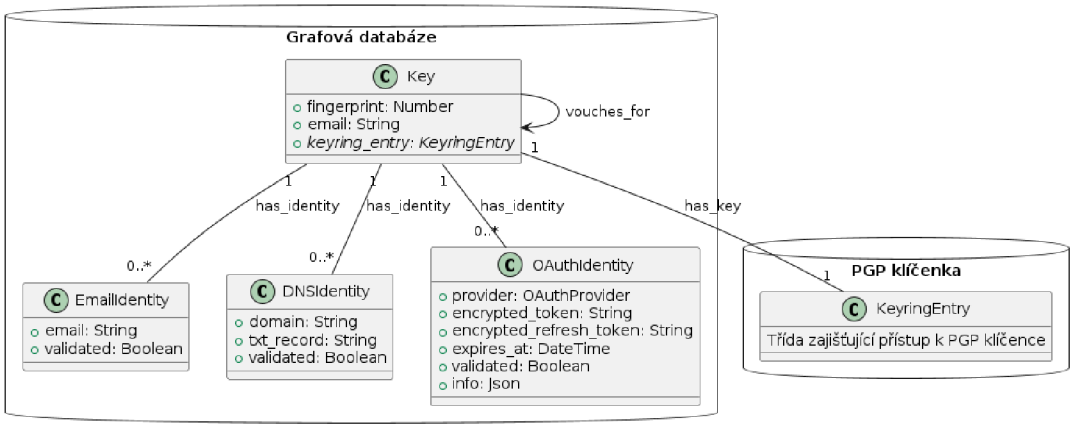
3.2 Architektura aplikace

Architektura je rozdělena na čtyři hlavní komponenty:

1. klientská část – zajišťuje vizualizaci sítě a komunikaci se serverem,
2. serverová část – zajišťuje komunikaci s databází a PGP klíčenkou,
3. grafová databáze – uchovává data o externích identitách a vztahy mezi klíči,
4. PGP klíčenka – uchovává veřejné klíče uživatelů a umožňuje jejich ověření, je jednoduše exportovatelná.

3.3 Databázová struktura

Databázová struktura, jak je znázorněna na obrázku 5, uchovává informace o klíčích v třídě `Key`. Dále uchovává informace o externích identitách v třídách `EmailIdentity`, `DNSIdentity` a `OAuthIdentity`. Vytváříme taky třídu `KeyringEntry` pro práci s klíči v PGP klíčence.

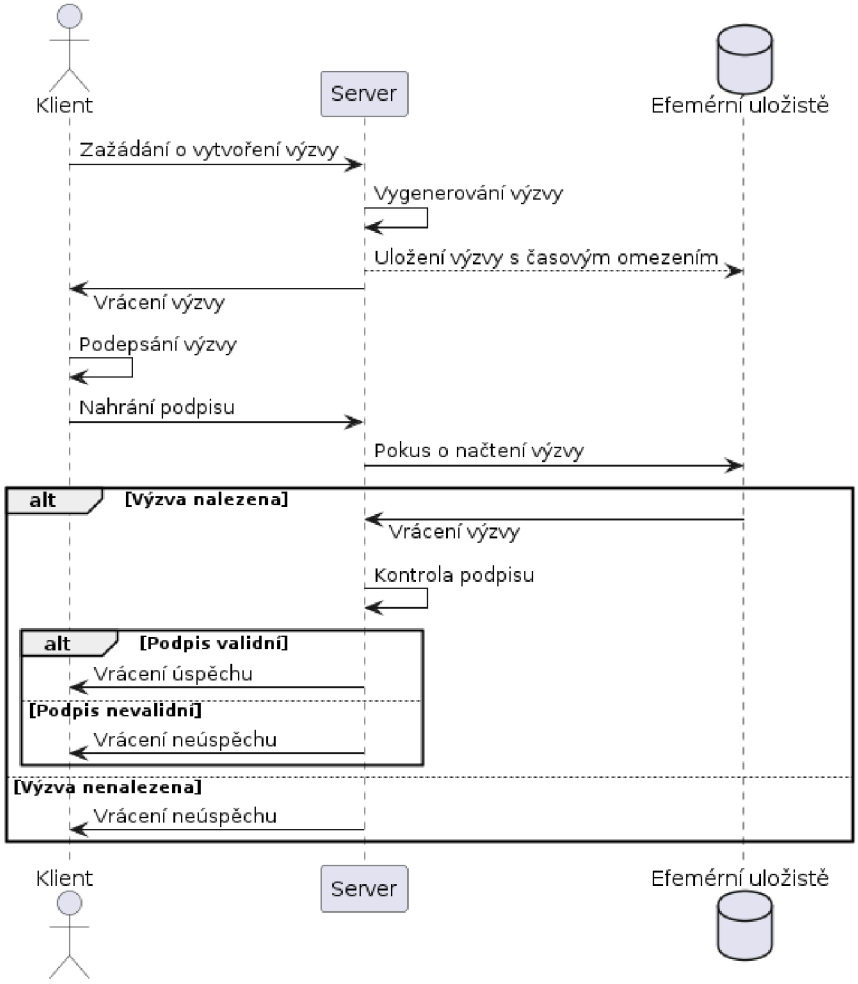


Obrázek 5: Databázová struktura

3.4 Návrh autentizace uživatele

V kontextu, kdy uživatel používá PGP klíče, vzniká myšlenka, že by se mohl držitel tohoto asymetrického páru ověřit pomocí soukromého klíče. K tomuto využijeme schopnosti PGP – kryptografickému podepisování.

Pří *podpisové výzvě* je uživateli prezentován náhodný řetězec určený k podepsání. Tento řetězec má omezenou dobu, po kterou je jeho podpis považován za validní. Při úspěšné validaci na serveru jsme s jistotou schopni říci, zda uživatel je, nebo není držitelem daného soukromého klíče, a můžeme jej tedy náležitě autentizovat. Díky náhodnosti a dočasnosti řetězců se zároveň bráníme možnému útoku typu *replay attack*. Postup ověření podpisovou výzvou je znázorněn na obrázku 6.



Obrázek 6: Princip autentizace uživatele pomocí podpisové výzvy

3.5 Koncept ověřených externích identit

Jedním z požadavků uživatelů je možnost přiřadit k PGP klíči externí identitu. Taková identita by mimo informační charakter měla nést i vlastnost ověřitelnosti. To znamená, že by mělo být možné zjistit, zda daná skutečně patří k danému klíči.

Pro aktuální potřeby aplikace bude implementováno přiřazení následujících typů identit:

1. identita emailové adresy (pomocí emailového ověření),
2. identita webové stránky (pomocí DNS ověření),
3. GitHub/Twitter/Discord identita (pomocí OAuth2 ověření).

3.5.1 Emailové ověření

Pod tímto typem ověření si lze představit klasický proces ověření emailové adresy. Uživatel dostane email s tajným kódem, který musí následně zadat do aplikace. Tímto způsobem můžeme s jistotou říci, že uživatel je vlastníkem dané emailové adresy.

3.5.2 DNS ověření

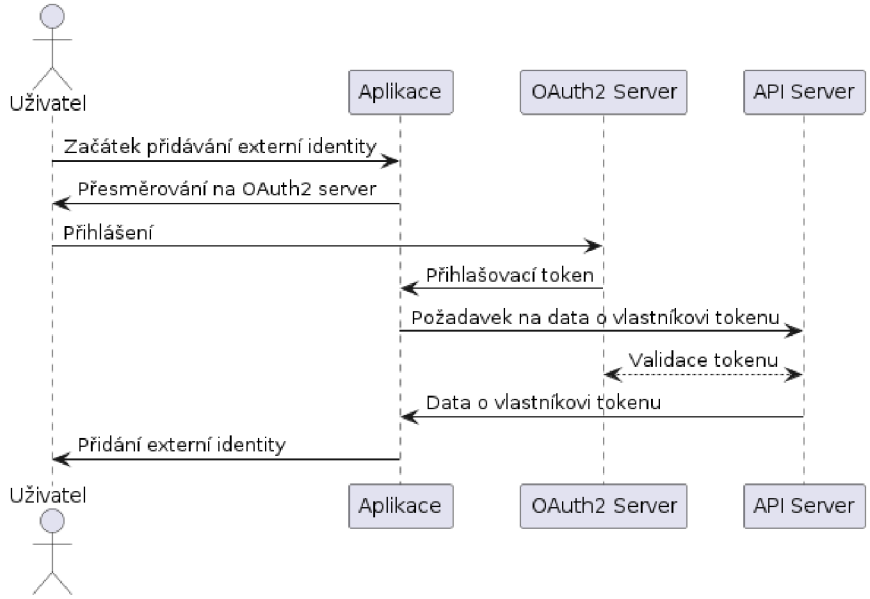
Ověření vlastnictví domény je založeno na principu, kdy uživatel musí do Domain Name System (DNS) záznamu domény přidat speciální TXT záznam. Následně (po propagaci záznamu napříč DNS servery) je možné ověřit, že uživatel je vlastníkem domény a tím i webové stránky.

Doména	Typ	Hodnota
_trustroute-challenge	TXT	`"1e3a6c651fe956eeb19a...`

3.5.3 OAuth2 ověření

OAuth2 je protokol, který umožňuje aplikacím získat přístup k uživatelským účtům na jiných webových službách. Uživatel při tom má dohled nad rozsahem přístupu k osobním datům, které aplikace získává. [17]

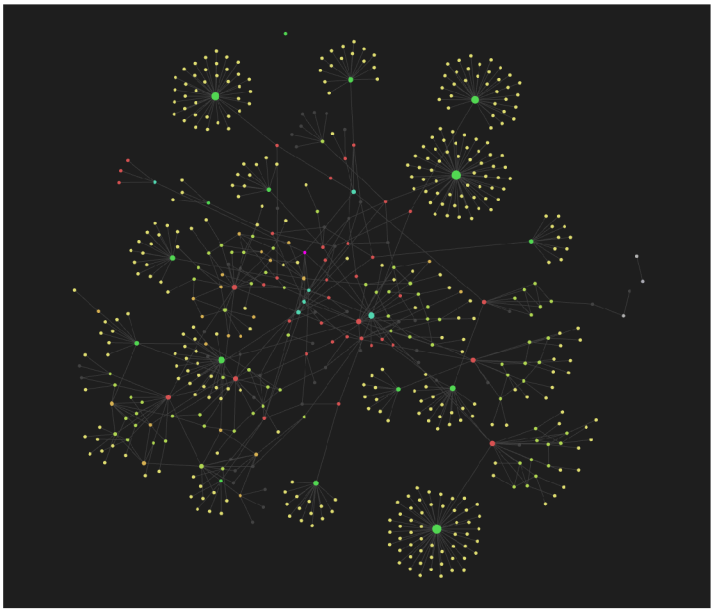
Toto ověření je založeno na principu, kdy se uživatel přihlásí do svého účtu na dané platformě a následně podle OAuth2 protokolu udělí aplikaci přístup k určitým informacím, např. jméno, email a jiné. Tímto způsobem můžeme s jistotou říci, že uživatel je vlastníkem daného účtu a tím i dané identity.



Obrázek 7: Princip využití OAuth2 pro ověření identity

3.6 Návrh uživatelského rozhraní

Primárním zdrojen inspirace pro návrh uživatelského rozhraní byl grafový pohled v aplikaci *Obsidian* [18]. Tento pohled je založen na principu *force simulation*, kdy se uzly a hrany grafu pohybují tak, aby dosáhly stabilního stavu. Poskytuje uživateli přehled o celé síti a umožňuje mu interaktivně vyhledávat a prozkoumávat jednotlivé uzly.



Obrázek 8: Grafový pohled v aplikaci Obsidian

Design uživatelského rozhraní byl vytvořen v nástroji *Figma*[19] a je dostupný v příloze (`docs/UI-design.pdf`) A.2. Dle požadavků uživatelů jsem určil, že aplikace bude postavena jako webová aplikace, která bude dostupná primárně na desktopových zařízeních. Zajištění responzivity pro mobilní zařízení je tedy považováno za sekundární.

4 Implementace

Tato část ze zaměřuje na implementaci jednotlivých komponent aplikace. Začneme klientskou částí, která je zodpovědná za vizualizaci sítě a komunikaci se serverem. Následně se zaměříme na serverovou část, která zajišťuje spojení s databází a PGP klíčenkou.

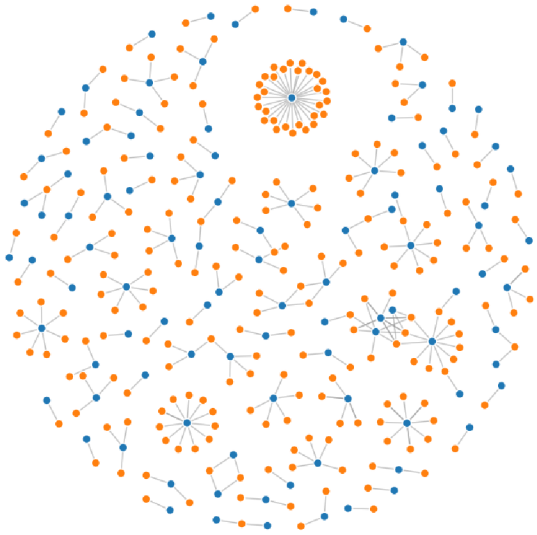
4.1 Klientská část

Klientská část je zpracována za pomocí slim šablon přímo v Ruby on Rails 4.2.1, protože se jedná o jednoduchou aplikaci, která nepotřebuje klientskou reaktivitu. Kromě pár jednoduchých javascriptových funkcí a D3.js knihovny je klientská část převážně statická.

4.1.1 D3.js

D3.js je javascriptová knihovna pro vizualizaci dat. Umožňuje vytvářet interaktivní grafy, mapy a další vizualizace. Poskytuje rozhraní pro nahrávání dat a manipulaci s nimi pomocí *DOM* a *SVG*. [20]

Pro naše potřeby použijeme D3.js pro obrazovku zbrazení sítě důvěry. V zájmu zvýšení přehlednosti transformujeme graf pomocí simulace síly (angl. *force simulation*). Ukázka simulace síly v knihovně D3.js je zobrazena na obrázku 9.



Obrázek 9: Ukázka simulace síly v D3.js [20]

4.2 Serverová část

Serverová část je zodpovědná za komunikaci s databází a PGP klíčenkou. Externě komunikuje také s PGP keyservery, OAuth2 providery a DNS servery. Pro tyto účely byl zvolen framework Ruby on Rails.

4.2.1 Ruby on Rails

Ruby on Rails je open-source webový framework s následujícími vlastnostmi:

- používá syntax jazyka Ruby,
- základá se na architektuře Model-View-Controller (MVC),
- je známý pro svou ideu *konvence před konfigurací*, což umožňuje akcelerovaný a iterovaný vývoj,
- těží z velkého ekosystému a mnoha dostupných knihoven zvaných *gems*.

Rails sice nepatří mezi nejrychlejší frameworky, ale díky své jednoduchosti a poskytovanému vývojařskému komfortu je vhodný pro malé projekty, které ještě nenabyly jasné podoby a zároveň nevyžadují vysoký výkon. [21]

4.2.2 Neo4j

Neo4j je grafová databáze napsaná v Javě. Je to open-source databáze, která je optimalizována pro ukládání a dotazování grafových dat. Z principu je grafová databáze vhodná pro ukládání dat sítě důvěry. Díky doplňkové knihovně Awesome Procedures on Cypher (APOC) je možné provádět složité operace nad grafovými daty přímo v databázi pomocí jazyka *Cypher*. [22]

Cypher je deklarativní jazyk pro dotazování grafových databází. Je navržen tak, aby byl co nejvíce přirozený pro práci s grafovými daty. Je inspirován jazykem SQL, díky čemuž je přenositelný i do světa ORM knihoven (např. v Rails *ActiveRecord* rozšířený o *ActiveGraph*).

4.2.3 GPGME

GnuPG Made Easy (GPGME) je knihovna poskytující vysokoúrovňové rozhraní pro práci s PGP klíči a PGP klíčenkou. Umožňuje jednoduché podepisování a šifrování zpráv, stejně jako ověřování klíčů a další operace. Dá se také vnímat jako API pro GPG [23]

4.2.4 Elasticsearch

Elasticsearch je full-textový vyhledávací engine postavený na Apache Lucene. Je distribuovaný, což znamená, že je možné ho rozdělit na více uzlů a tím zvýšit jeho výkon. Je to open-source software, který je široce používán pro vyhledávání a analýzu strukturovaných i nestrukturovaných dat. [24]

V našem případě bude Elasticsearch použit pro vyhledávání nad externími identitami uživatelů a metadaty klíčů. Vzhledem k rozsahu aplikace a požadavkům na výkon není nutné Elasticsearch distribuovat. Využijeme knihovnu *elasticsearch-ruby*, která poskytuje Ruby API pro komunikaci s Elasticsearch. Definujeme si indexovatelný JSON nad modelem klíče, který při změně v databázi automaticky indexuje do Elasticsearch.

Zdrojový kód 1: Ukázka indexovatelného JSON dokumentu klíče

```
{
  "name"=>"Beryl Schinner",
  "sha"=>"61f78f02",
  "long_sha"=>"bee9993a61f78f02",
  "keyid"=>"232a967199ce6bb00ad902eabee9993a61f78f02",
  "indexable_oauth_identities"=>[
    {
      provider => "github",
      uid => "bee9993a",
      name => "beryl"
    }
  ],
  "indexable_email_identities"=>[
    {email => "beryl@schneider.com"}
  ],
  "indexable_dns_identities"=>[
    {"domain"=>"ziemann.name"}
  ],
}
```

4.2.5 Docker

Docker je open-source platforma pro vývoj, nasazení a provoz aplikací. Umožňuje oddělení aplikace od infrastruktury a zajišťuje, aby aplikace běžela stejně ve všech prostředích. Docker používá kontejnerizaci, což znamená, že aplikace je spuštěna v izolovaném prostředí, které sdílí operační systém s hostitelským systémem. [25]

V našem případě budeme používat Docker pro sestavení *Docker image* s aplikací a všemi potřebnými závislostmi. Tento image bude následně nasazen na server spolu s databází a Elasticsearch za pomocí *Docker Compose*.

4.3 Vývojové prostředí

Zajímavostí na závěr může být použití *Nix Shell* pro vytvoření předdefinovaného vývojového prostředí. Ve své podstatě funguje podobně jako Docker, ale z pohledu vývojáře je mnohem jednodušší. Stačí definovat `shell.nix` A.1 soubor, který popisuje všechny závislosti aplikace, a následně spustit `nix-shell`. Uživatel je pak přenesen do prostředí, kde má k dispozici všechny potřebné závislosti.

Takový shell má přístup ke všem programům a knihovnám, které jsou globálně v systému, ale zároveň přidává všechny závislosti aplikace. To znamená, že vývojář nemusí řešit instalaci a správu závislostí a může se plně soustředit na vývoj aplikace. Pokud vývojář aktivně pracuje na více projektech, je možné mít pro každý projekt vlastní nix shell a tím pádem nevznikne problém s konflikty závislostí, nebo s různými verzemi knihoven.

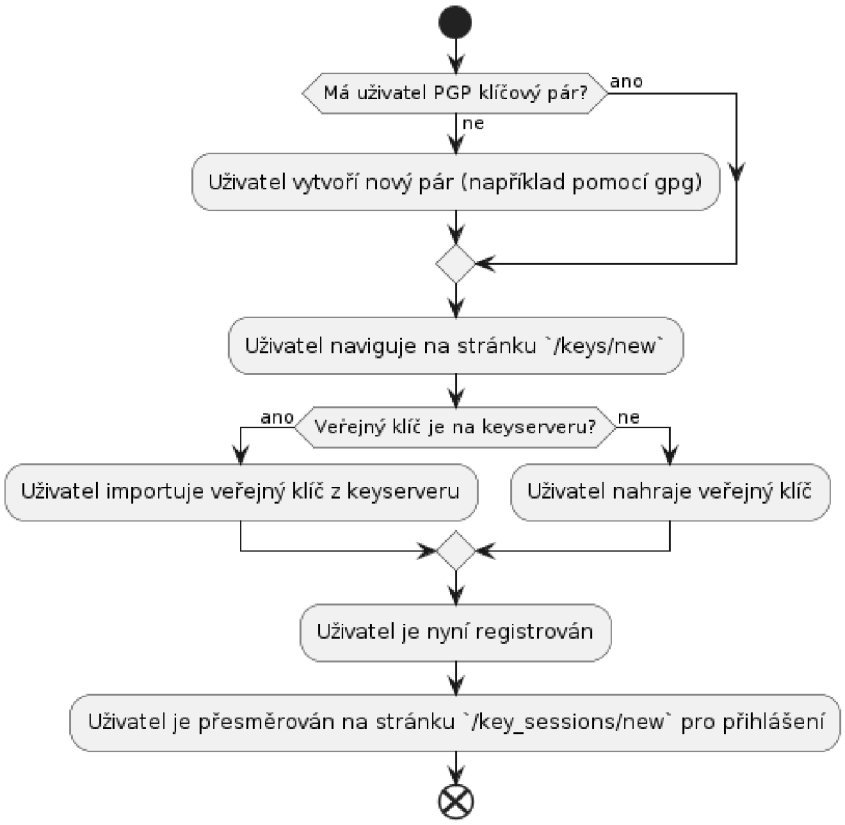
Nix shell je součástí *Nix ekosystému*, který je známý pro svou deklarativní a funkcionální povahu. Nix je balíčkovací systém, který je schopen spravovat závislosti a vytvářet izolované prostředí pro aplikace. [26]

5 Uživatelská příručka

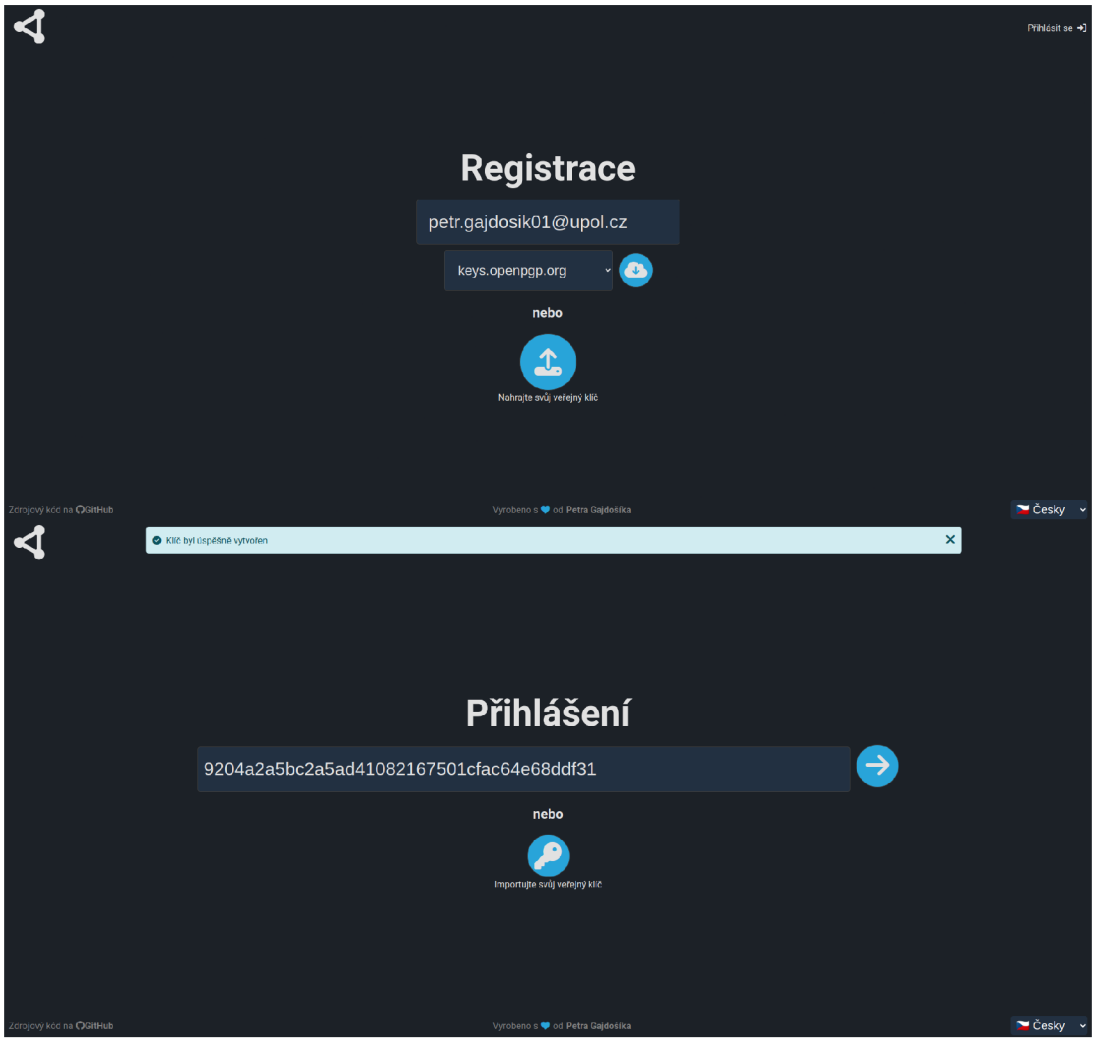
V této kapitole se seznámíme s možnými scénáři užití aplikace. Jednotlivé interakce jsou popsány jako páry uživatelských vývojových diagramů (angl. *user flow diagrams*) a snímků obrazovky pořízených v aplikaci. Diagramy popisují možné cesty, kterými může uživatel projít při interakci s aplikací. Snímky obrazovky jsou prezentovány v chronologickém pořadí a zobrazují jednotlivé kroky uživatele.

5.1 Registrace

V prvním scénáři užití se uživatel zaregistruje do aplikace. Podle diagramu 10 uživatel, buďto importuje veřejný klíč z keyserveru, nebo ho nahraje přímo do aplikace. Snímky obrazovky 11 zobrazují zdárný průběh registrace.



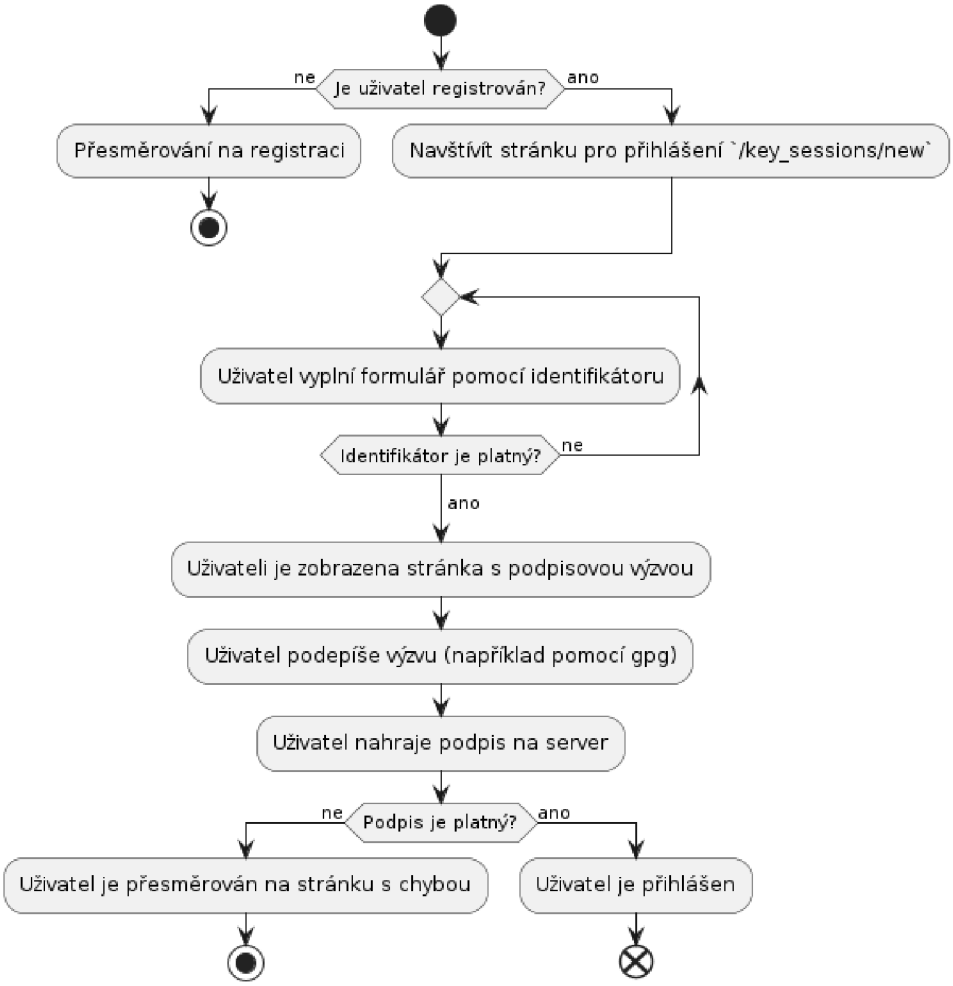
Obrázek 10: User flow diagram registrace



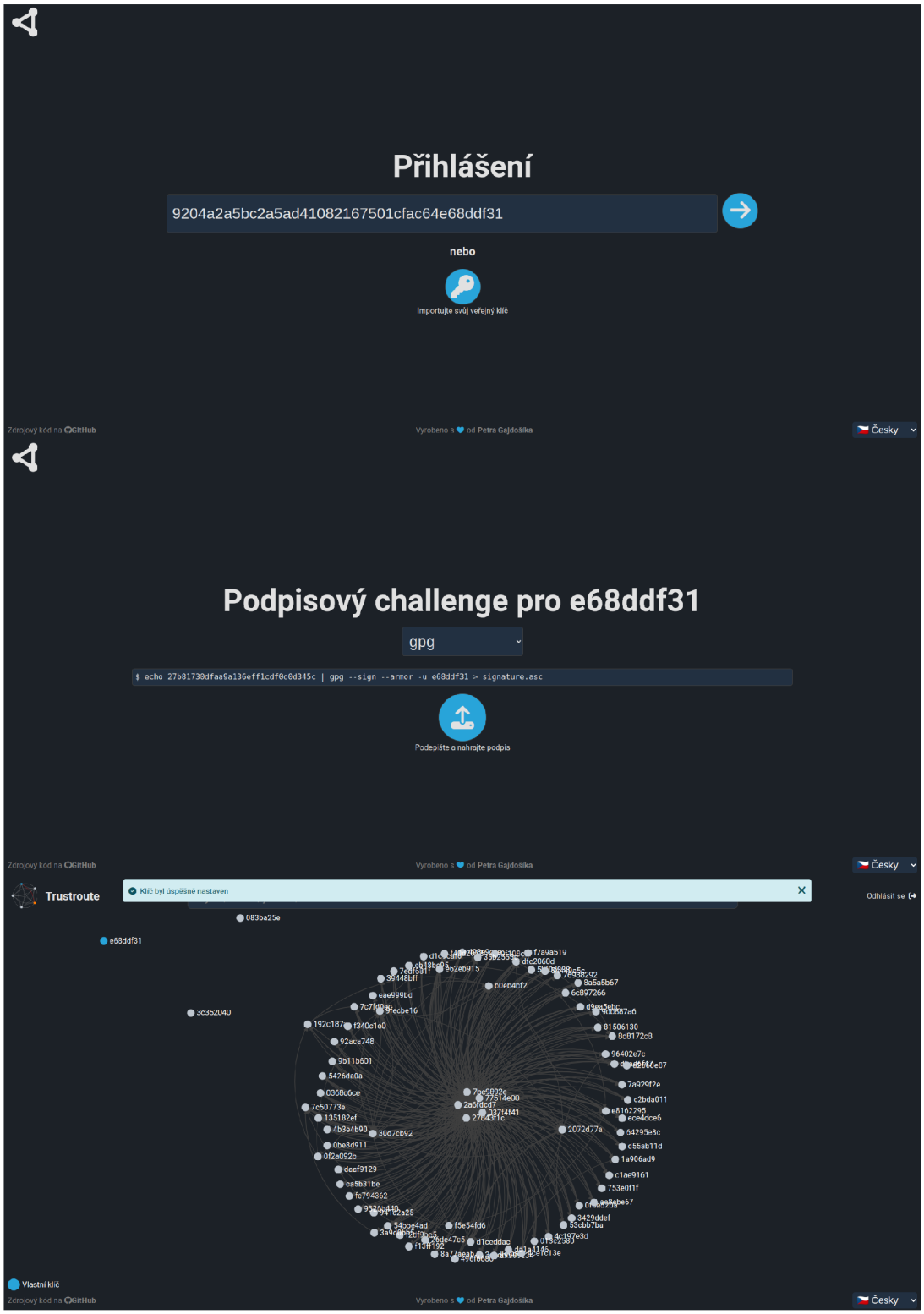
Obrázek 11: Snímky obrazovky zobrazující import veřejného klíče při registraci

5.2 Přihlášení

Další scénář užití se týká přihlášení pomocí podpisové výzvy 3.4. Uživatel zadá svůj PGP identifikátor a aplikace mu vygeneruje podpisovou výzvu, na kterou očekává odpověď. Po úspěšném ověření je uživatel přihlášen, v opačném případě mu je zobrazena chybová hláška. Tento postup můžeme vidět na diagramu 12. Úspěšný průchod scénářem je pak na snímcích obrazovky 13.



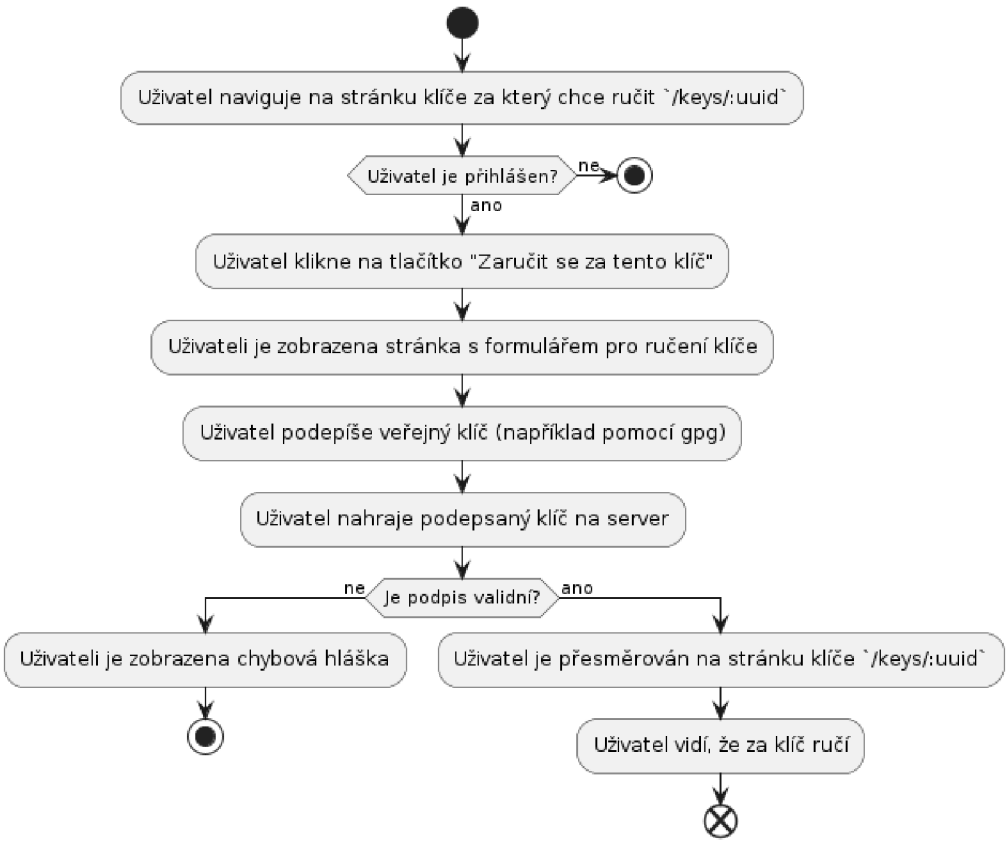
Obrázek 12: User flow diagram přihlášení



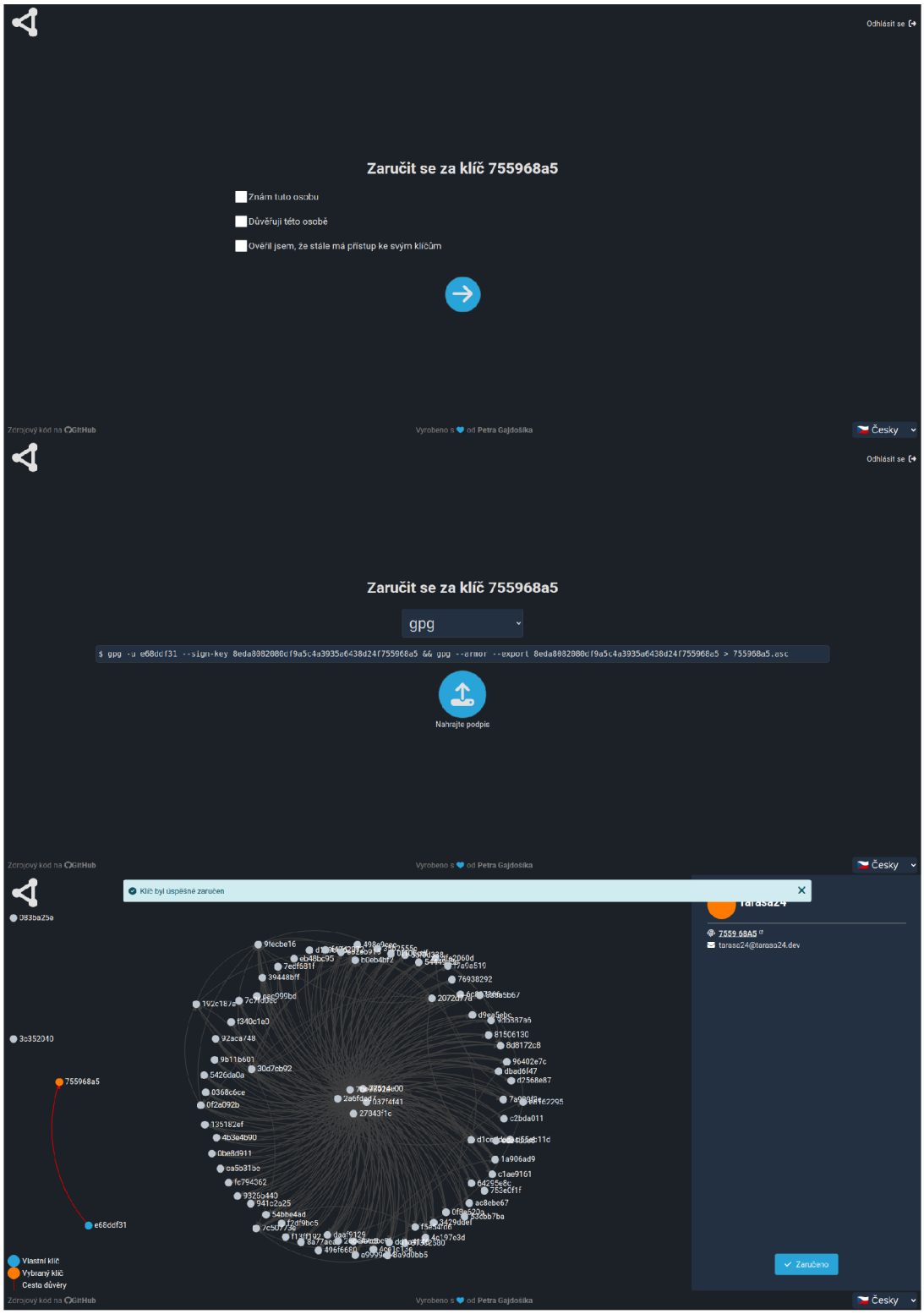
Obrázek 13: Snímky obrazovky zobrazující proces přihlášení

5.3 Zaručení se za klíč

Scénář zaručení se za klíč je základním prvkem pro fungování sítě důvěry. Uživatel se může zaručit za klíč jiného uživatele, čímž vytváří vztah důvěry. Proces zaručení probíhá pomocí podpisu veřejného klíče, jak můžeme vidět na diagramu 14. Jednotlivé kroky potřebné k zaručení se za klíč v kontextu aplikace jsou zobrazeny na snímcích obrazovky 15.



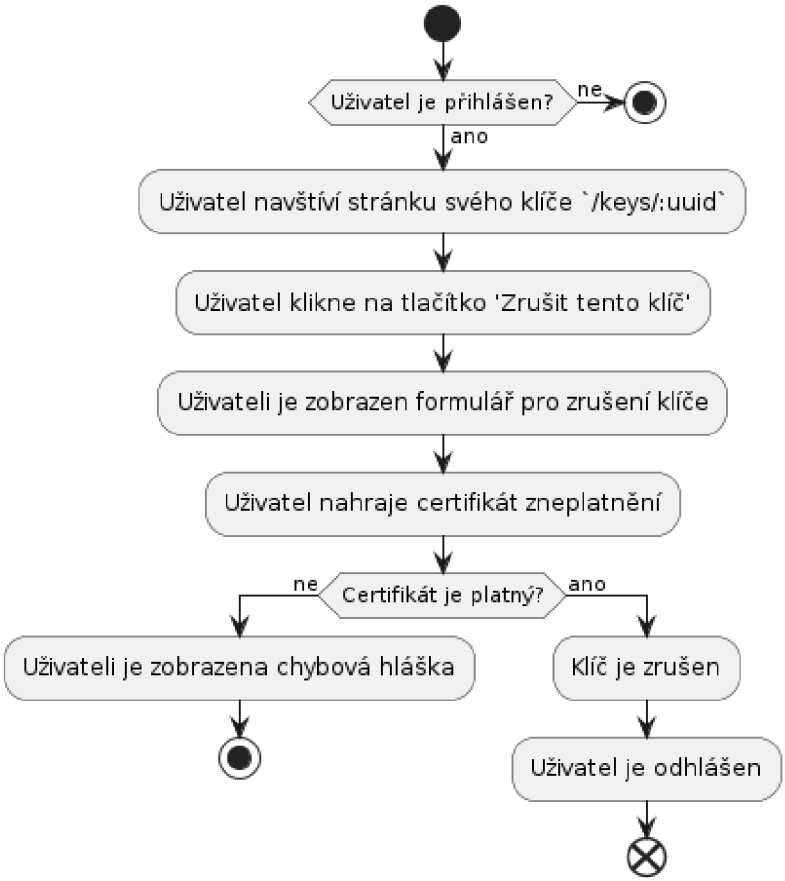
Obrázek 14: User flow diagram zaručení se za klíč



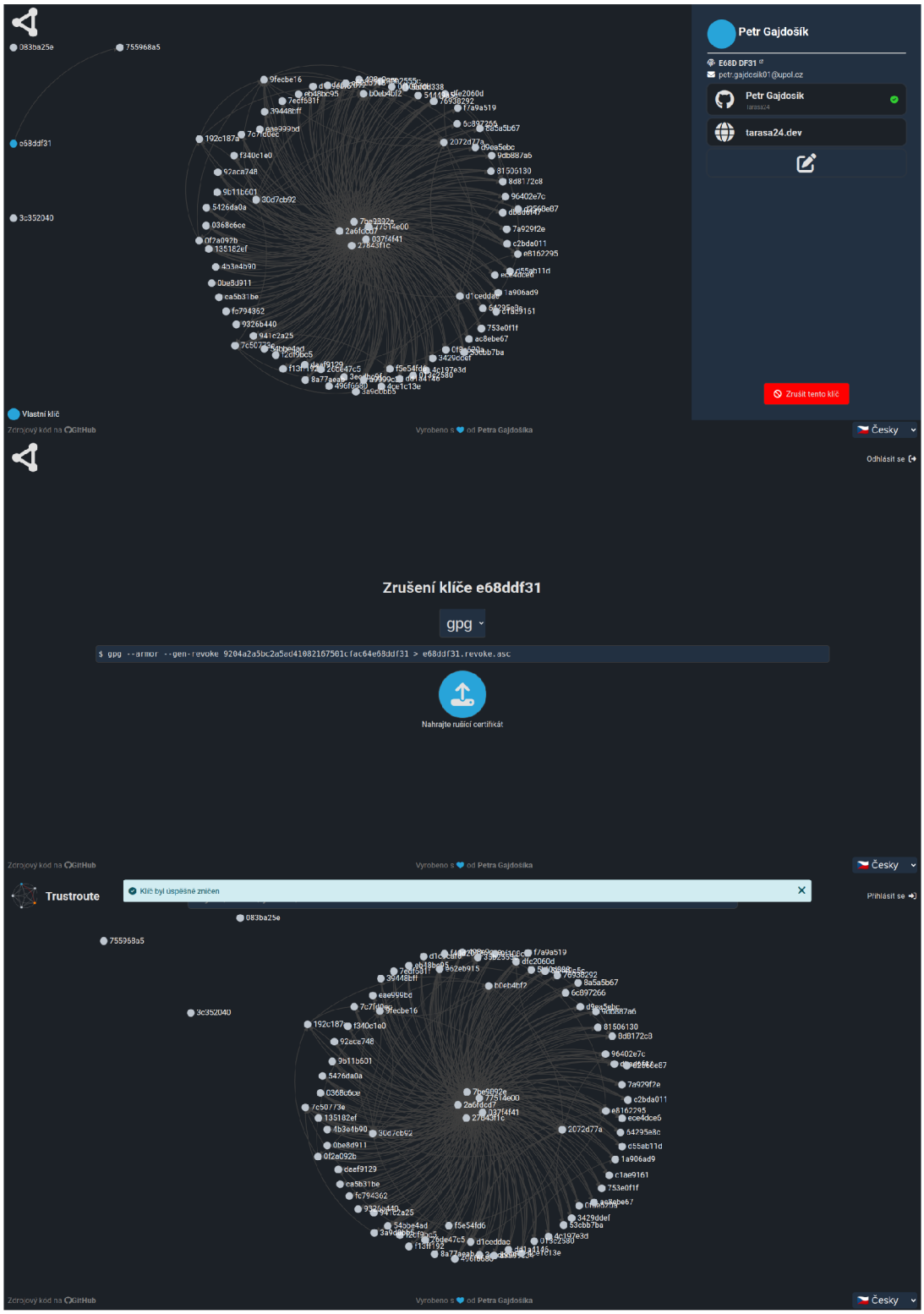
Obrázek 15: Snímky obrazovky zobrazující proces zaručení se za klíč

5.4 Zrušení klíče

Uživatel může chtít svůj klíč ze sítě důvěry odebrat. Tento proces je znázorněn na diagramu 16. K takovému úkonu je potřeba *certifikát zneplatnění*, což je speciální typ certifikátu přímo definovaný v OpenPGP standardu. Server ověří, zda je certifikát validní, nasledně klíč odstraní a uživatele odhlásí. Snímky obrazovky 17 znázorňují průběh úspěšného zrušení klíče.



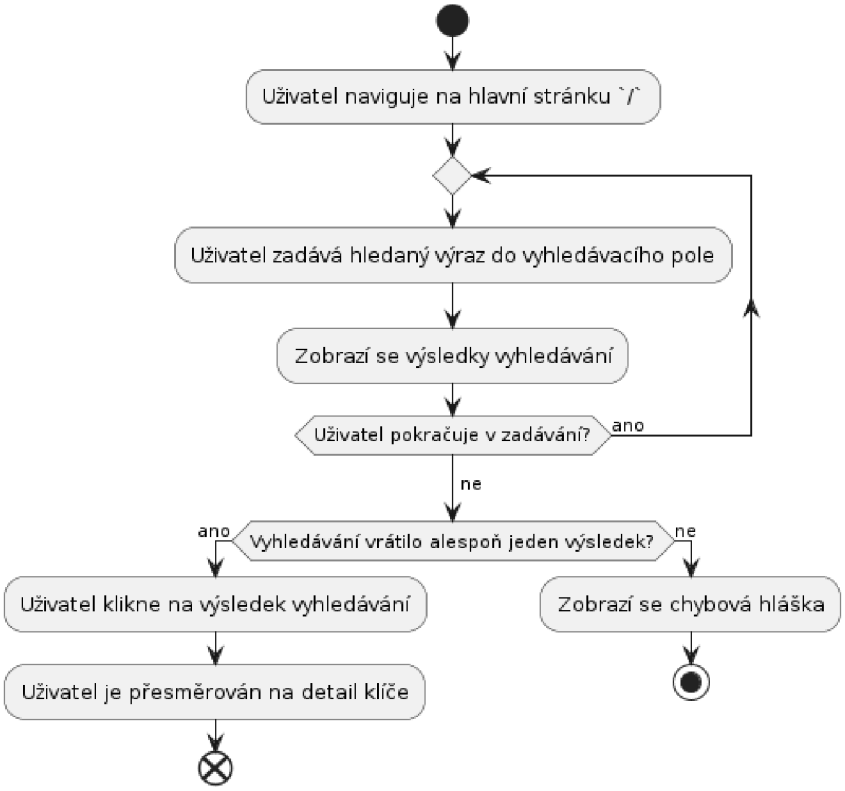
Obrázek 16: User flow diagram zrušení klíče



Obrázek 17: Snímky obrazovky zobrazující proces zrušení klíče

5.5 Vyhledání klíče

Jedním ze základních požadavků aplikace je možnost vyhledání klíče. Uživatel vyplní vyhledávací pole na hlavní stránce a v průběhu zadávání se mu zobrazují výsledky. Pokud je klíč nalezen, uživatel může navigovat na jeho detail, jinak je mu zobrazena chybová hláška – viz diagram 18. Snímky obrazovky naznačují situaci, kdy byl klíč nalazen 19 a situaci kdy klíč nalezen nebyl 20 .



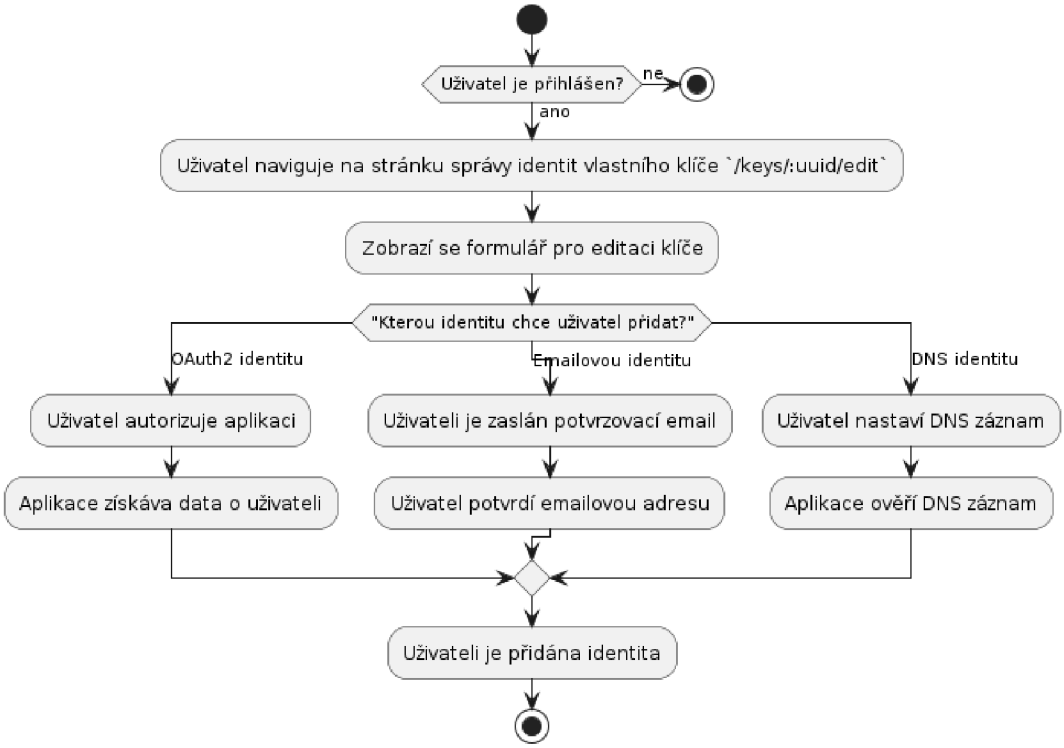
Obrázek 18: User flow diagram vyhledání klíče

Obrázek 19: Snímek obrazovky zobrazující vyhledávací pole s výsledky

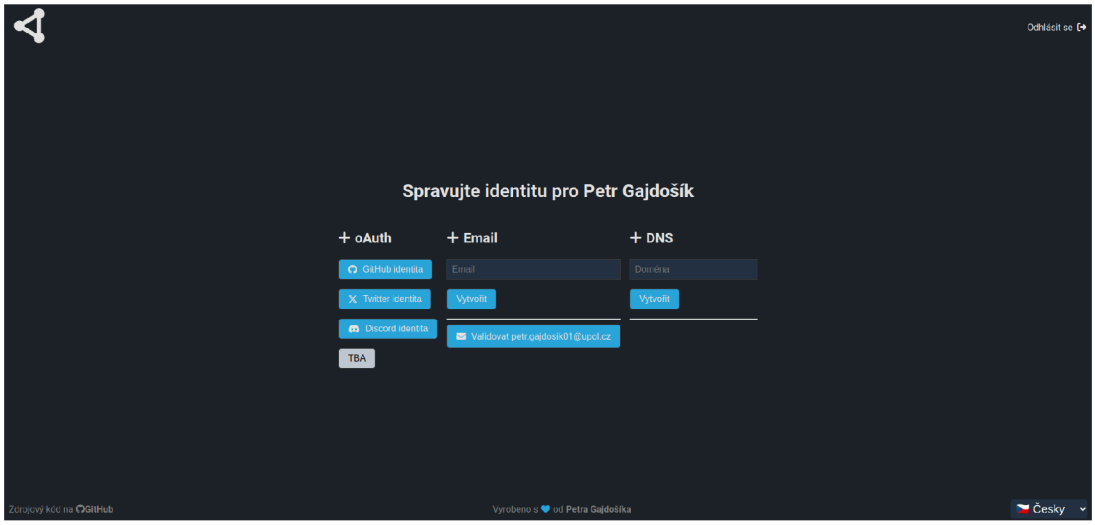
Obrázek 20: Snímek obrazovky zobrazující vyhledávací pole s chybovou hláškou

5.6 Vytvoření externí identity

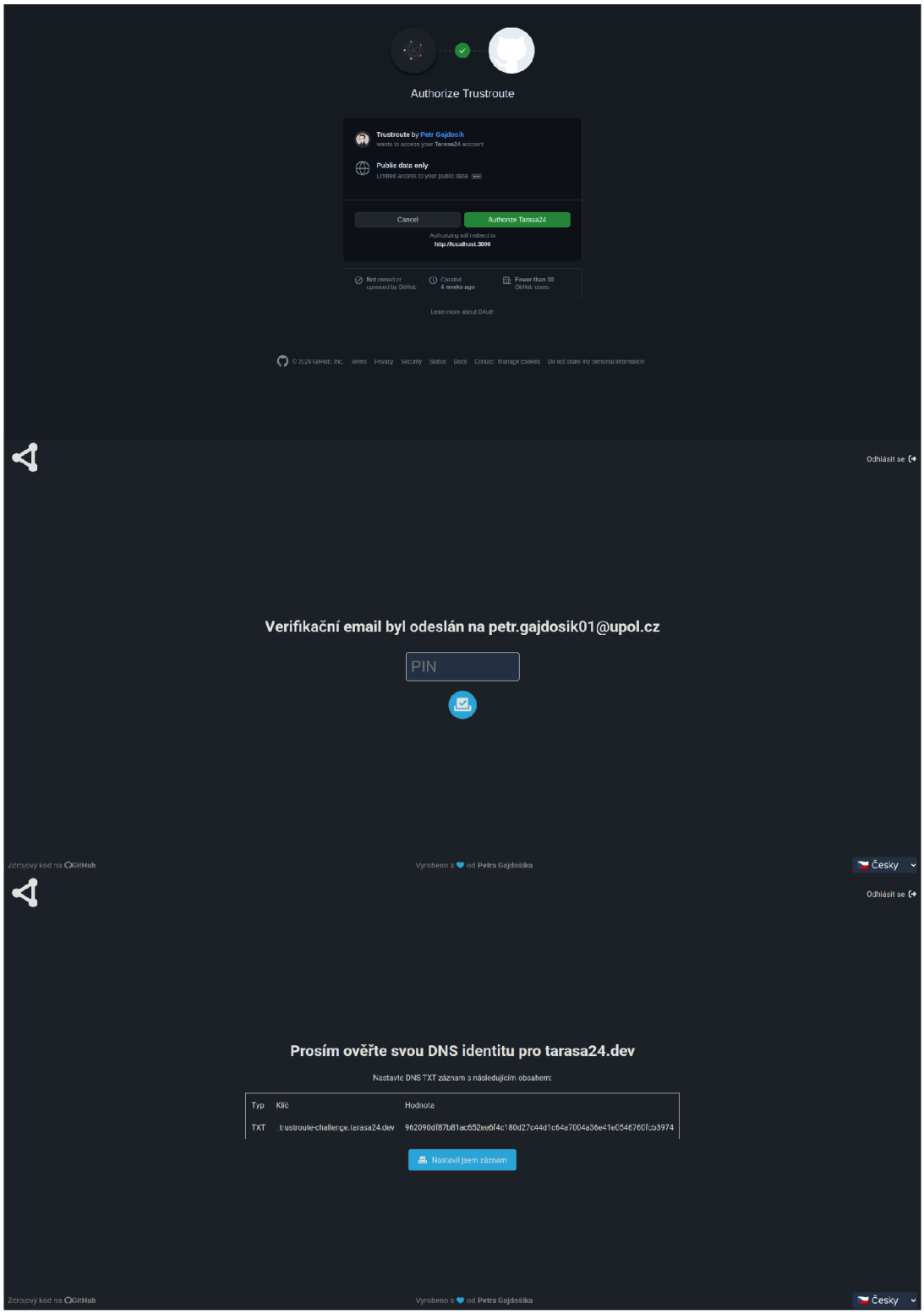
V posledním scénáři užití se uživatel pokusí vytvořit novou externí identitu. Pro tento účel má k dispozici tři možnosti – OAuth2, email a DNS. Každá z těchto možností má své specifické požadavky, které jsou zobrazeny na diagramu 21. Výběr typu identity je prezentován na samostatném pohledu, jehož snímek obrazovky je zobrazen na obrázku 22. Snímky obrazovky 23 pak zachycují pohledy vykreslené při zvolení jednotlivých typů identit.



Obrázek 21: User flow diagram vytvoření externí identity



Obrázek 22: Snímek obrazovky zobrazující výběr typu identity při vytváření nové externí identity



Obrázek 23: Snímky obrazovky zobrazující pohledy při zvolení jednotlivých typů identit (v pořadí OAuth2, email, DNS)

Závěr

V rámci mé bakalářské práce byla navržena a implementována webová aplikace pro správu sítě důvěry. Aplikace umožňuje uživatelům spravovat své PGP klíče a přiřazovat k nim ověřené externí identity. Uživatelé mohou vytvářet vztahy důvěry s ostatními uživateli a tím vytvářet síť důvěry. Implementace byla provedena tak, aby byla rozšiřitelná, např. o další PGP Keyservery, jiné poskytovatele identit, atd.

Návrhem pro další vývoj by mohla být implementace plného znění OpenPGP standardu ohledně sítě důvěry, obvzlášť v oblasti vyhledávání cesty mezi klíči. Dalším možným rozšířením by mohlo být zavedení serverové federace, která by umožnila propojit instance aplikace a tím decentralizovat ze své podstaty centralizovanou službu.

Bylo by vhodné pokračovat v rozvoji aplikace s cílem prozkoumat výše zmíněné možnosti a případně je implementovat. V neposlední řadě je zamýšlena i optimalizace pro mobilní zařízení, včetně celkového zlepšení uživatelského rozhraní.

Conclusions

In the scope of my bachelor thesis, a web application for managing a web of trust was designed and implemented. The application allows users to manage their PGP keys and assign verified external identities to them. Users can create trust relationships with other users and thus create a web of trust. The implementation was done in such a way that it is extensible, e.g. by adding other PGP Key servers, other identity providers, etc.

A possible extension could be the implementation of the full OpenPGP standard regarding the web of trust, especially in the area of finding a path of trust between keys. Another possible extension could be the introduction of server federation, which would allow to connect instances of the application and thus decentralize a centralized service.

It would be appropriate to continue the development of the application with the aim of exploring the above-mentioned possibilities and possibly implementing them. Last but not least, optimization for mobile devices is planned, including overall improvement of the user interface.

A Obsah přiloženého média

A.1 Adresářová struktura

- `app/` – definice aplikace
  - `controllers/` – kontrolery
  - `models/` – modely
  - `views/` – pohledy
  - `services/` – služby
  - `jobs/` – úlohy zpracovávané na pozadí
  - `mailers/` – emailové šablony
- `config/` – konfigurační soubory
- `db/` – databázová konfigurace
  - `neo4j/` – migrace pro Neo4j databázi
  - `seeds.rb` – seedovací data
- `public/` – veřejné soubory
- `docs/` – dokumentace A.2
- `docker/` – docker soubory pro produkční nasazení
- `Gemfile` – seznam Ruby gemů
- `shell.nix` – Nix shell pro vývoj
- `docker-compose.yml` – definice služeb pro vývoj

A.2 Uživatelská dokumentace

- `docs/UI-design.pdf` – design uživatelského rozhraní
- `README.md` – návod k instalaci, spuštění aplikace v *angličtině*
- `README.cz.md` – návod k instalaci, spuštění aplikace v *češtině*
- `docs/thesis.pdf` – text této práce

Seznam zkratek

API Application Programming Interface

APOC Awesome Procedures on Cypher

CA Certifikační autorita

DNS Domain Name System

GPG GNU Privacy Guard

GPGME GnuPG Made Easy

JSON JavaScript Object Notation

MVC Model-View-Controller

ORM Object-Relational Mapping

PGP Pretty Good Privacy

PKI Public Key Infrastructure

RA Registrační autorita

RFC Request for Comments

SQL Structured Query Language

VA Validační autorita

Literatura

[1] Ulrich, Alexander; Holz, Ralph; Hauck, Peter; Carle, Georg. *Computer Security – ESORICS 2011*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. ISBN 978-3-642-23822-2.

[2] Project, GnuPG. *GnuPG*. 2020. Official website of GnuPG. Dostupný z: ⟨https://gnupg.org/⟩.

[3] Keybase, Inc. *Keybase*. 2024. Official website of Keybase. Dostupný z: ⟨https://keybase.io/⟩.

[4] AG, Proton Technologies. *ProtonMail*. 2024. Official website of ProtonMail. Dostupný z: ⟨https://protonmail.com/⟩.

[5] Brunschwig, Patrick. *Enigmail*. 2024. Official website of Enigmail. Dostupný z: ⟨https://www.enigmail.net/⟩.

[6] Sklenák, Vilém. *KTD: Česká terminologická databáze knihovnictví a informační vědy (TDKIV)*. Praha: Národní knihovna ČR, 2003. Dostupný z: ⟨https://aleph.nkp.cz/F/?func=direct&doc_number=000000668&local_base=KTD⟩.

[7] ToOb. *Digital Signature diagram cs.svg*. 2009. Dostupný z: ⟨https://commons.wikimedia.org/wiki/File:Digital_Signature_diagram_cs.svg⟩.

[8] Sklenák, Vilém. *KTD: Česká terminologická databáze knihovnictví a informační vědy (TDKIV)* [online]. 2003 [cit. 2024-4-17]. Dostupný z: ⟨https://aleph.nkp.cz/F/?func=direct&doc_number=000000636&local_base=KTD⟩.

[9] Finney, Hal; Donnerhacke, Lutz; Callas, Jon; Thayer, Rodney L.; Shaw, Daphne. *OpenPGP Message Format*. 2007. 90 s. Request for Comments. RFC 4880⟨https://www.rfc-editor.org/info/rfc4880⟩.

[10] Project, GNU. *GNU Privacy Guard*. 2020. Official website of the GNU Privacy Guard. Dostupný z: ⟨https://www.gnupg.org/⟩.

[11] Stallings, William. The PGP Web of Trust: Managing public keys with the PGP (Pretty Good Privacy) web of trust. 1995, roč. 20, č. 2, s. 161–?? Dostupný také z: ⟨https://web.archive.org/web/20080704132005/http://www.byte.com/art/9502/sec13/art4.htm⟩. ISSN 0360-5280 (print), 1082-7838 (electronic). ISSN 0360-5280 (print), 1082-7838 (electronic).

[12] Kku. *Český překlad Web of Trust-en.svg*. 2019. Dostupný z: ⟨https://commons.wikimedia.org/wiki/File:Web_of_Trust-en.svg⟩.

[13] Milgram, Stanley. The Small-World Problem. *Psychology Today*. 1967, roč. 1, č. 1, s. 61–67.

[14] Ford, Dr. Warwick S.; Chokhani, Dr. Santosh; Wu, Stephen S.; Sabett, Randy V.; Merrill, Charles (Chas) R. *Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework*. 2003. 94 s. Request for Comments. RFC 3647⟨https://www.rfc-editor.org/info/rfc3647⟩.

[15] Mozilla. *CA/WoSign Issues*. 2021. Information on WoSign's issues and Mozilla's response. Dostupný z: ⟨https://wiki.mozilla.org/CA/WoSign_Issues⟩.

[16] Ogden, Cody. *Killed by Google*. 2024. List of all Google products killed by Google. Dostupný z: ⟨https://killedbygoogle.com/⟩.

[17] Hardt, Dick. *The OAuth 2.0 Authorization Framework*. 2012. 76 s. Request for Comments. RFC 6749⟨https://www.rfc-editor.org/info/rfc6749⟩.

[18] Obsidian. *Obsidian is the private and flexible writing app that adapts to the way you think.* 2024. Official website of Obsidian. Dostupný z: ⟨https://obsidian.md/⟩.

[19] Figma, Inc. *Figma*. 2024. Dostupný z: ⟨https://www.figma.com/⟩.

[20] Bostock, Mike; Observable, Inc. *D3.js*. 2024. Official website of D3.js. Dostupný z: ⟨https://d3js.org/⟩.

[21] Hansson, David Heinemeier; contributors. *Ruby on Rails*. 2024. Official website of Ruby on Rails. Dostupný z: ⟨https://rubyonrails.org/⟩.

[22] Neo4j, Inc. *Neo4j*. 2024. Official website of Neo4j. Dostupný z: ⟨https://neo4j.com/⟩.

[23] Project, GnuPG. *GPGME - GnuPG Made Easy*. 2020. Official website of GPGME. Dostupný z: ⟨https://gnupg.org/software/gpgme/index.html⟩.

[24] Elasticsearch. *Elasticsearch*. 2024. Official website of Elasticsearch. Dostupný z: ⟨https://www.elastic.co/elasticsearch/⟩.

[25] Docker, Inc. *Docker - Build, Share, and Run Any App, Anywhere*. 2024. Official website of Docker. Dostupný z: ⟨https://www.docker.com/⟩.

[26] Foundation, NixOS. *NixOS*. 2024. Official website of NixOS. Dostupný z: ⟨https://nix.dev/⟩.