VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ

DEPARTMENT OF TELECOMMUNICATIONS

POKROČILÁ FILTRACE ELEKTRONICKÝCH DŮKAZŮ

ADVANCED FILTERING OF DIGITAL EVIDENCE

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE

AUTHOR

Ondřej Chudáček

VEDOUCÍ PRÁCE

SUPERVISOR

doc. Ing. Lukáš Malina, Ph.D.

BRNO 2022

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Informační bezpečnost**

Ústav telekomunikací

Student: Ondřej Chudáček ***ID:*** 221548

Ročník: 3 ***Akademický rok:*** 2021/22

NÁZEV TÉMATU:

Pokročilá filtrace elektronických důkazů

POKYNY PRO VYPRACOVÁNÍ:

Práce je zaměřena na pokročilou filtraci dat v podobě elektronických důkazů v open-source cloudovém úložišti Nextcloud. Cílem bakalářské práce je vhodný výběr parametrů pro vyhledávání a filtraci dat (min. podle vlastníka souboru, velikosti, typu souboru, data poslední úpravy a uživatele, který ji provedl) a vývoj funkčního rozhraní pro pokročilou, efektivní a bezpečnou filtraci. Toto rozšíření bude integrováno do webové aplikace Nextcloud a bude navrženo a realizováno tak, aby svým uživatelům významně ulehčilo orientaci a vyhledávání v uložených datech s ohledem na jejich formát a účel. Dílčím cílem je i zhodnocení vhodnosti nasazení homomorfního šifrování. Výsledné řešení bude rovněž otestováno v praktickém nasazení a podrobeno zátěžovým a uživatelským testům.

DOPORUČENÁ LITERATURA:

[1] MENEZES, Alfred, Paul C VAN OORSCHOT a Scott A VANSTONE. Handbook of applied cryptography. Boca Raton: CRC Press, c1997. Discrete mathematics and its applications. ISBN 0-8493-8523-7.

[2] HALEVI, Shai a Victor SHOUP, V. Design and implementation of HElib: a homomorphic encryption library. IACR Cryptol. ePrint Arch., 2020, s.1481

Termín zadání: 7.2.2022 ***Termín odevzdání:*** 31.5.2022

Vedoucí práce: doc. Ing. Lukáš Malina, Ph.D.

doc. Ing. Jan Hajný, Ph.D.
předseda rady studijního programu

ABSTRAKT

Práce se zaměřuje na možnosti implementace rozšířených možností filtrace elektronických důkazů uložených na open source cloudovém úložišti NextCloud. V práci je popsána právní analýza vývoje úložiště elektronických důkazů, architektura systému NextCloud, homomorfní šifrování a jeho možné použití, atributové vyhledávací šifrování a jeho využití a implementace jednoduché samostatně stojící filtrační klientské aplikace. V rámci tvorby práce byli osloveni odborníci na elektronické důkazy, aby zodpověděli otázky ohledně výběru filtračních kritérií. Stěžejní částí práce je popis implementace filtrace uvnitř systému NextCloud, které bylo rozšířeno o pět nových filtračních kritérií. Výsledek implementace filtrace uvnitř systému NextCloud byl podroben zátěžovým a funkčním testům.

KLÍČOVÁ SLOVA

Atributové vyhledávací šifrování, cloudové úložiště, elektronický důkaz, filtrace, homomorfní šifrování, NextCloud, WebDAV

ABSTRACT

This thesis focuses on the implementation of advanced filtering of digital evidence stored on an open source cloud storage NextCloud. The contents of the thesis are law analysis regarding the storage of digital evidence in NextCloud, design of NextCloud, homomorphic encryption and its possible usages, attribute based searchable encryption and its usages and implementation of a simple stand-alone filtration client. Experts on digital evidence were asked about the choice of filtration criteria as a part of the thesis. The key feature of the thesis is a description of implementation of filtration inside NextCloud which has been extended to contain five new filtration criteria. The result has been submitted to performance and functional tests.

KEYWORDS

Attribute Based Searchable Encryption, Cloud Storage, Digital Evidence, Filtration, Homomorphic Encryption, NextCloud, WebDAV

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

CHUDÁČEK, Ondřej. *Pokročilá filtrace elektronických důkazů*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2022, 68 s. Bakalářská práce. Vedoucí práce: doc. Ing. Lukáš Malina, PhD.

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	Ondřej Chudáček
VUT ID autora:	221548
Typ práce:	Bakalářská práce
Akademický rok:	2021/22
Téma závěrečné práce:	Pokročilá filtrace elektronických důkazů

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

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

Brno ....

podpis autora*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Chtěl bych poděkovat vedoucímu bakalářské práce panu doc. Ing. Lukáši Malinovi, PhD. za čas, který věnoval konzultacím a vedení bakalářské práce, Bc. Petru Muzikantovi za odbornou pomoc při řešení praktické části práce a JUDr. Mgr. Jakubu Haraštovi, Ph.D. z PrF MU, který asistoval u tvorby právnické části práce. Dále chci poděkovat členům Oddělení kybernetické kriminality KŘP JMK kpt. Ing. Bc. Ladislavu Vyskočilovi a mjr. Ing. Stanislavu Kovárníkovi za rozhovor, ve kterém bylo vyjasněno, jak postupovat při výběru filtračních kritérií a na co si dávat pozor při návrhu cloudového úložiště elektronických důkazů.

Obsah

Seznam obrázků

Seznam tabulek

Seznam výpisů

Úvod

Cílem práce je aktualizovat cloudové úložiště NextCloud tak, aby podporovalo nové možnosti filtrace použitelné i technicky méně vzdělanými uživateli z řad orgánů činných v trestním řízení.

Práce spadá do oblasti full stack vývoje webových aplikací. Konkrétně se v práci rozebírají témata spojená s ukládáním a zpracováním dat na open source cloudovém úložišti NextCloud. Dále práce popisuje různá šifrovací schémata – homomorfní šifrování, atributové vyhledávací šifrování a schémata použitá NextCloudem.

Hlavním tématem práce je prozkoumání systému NextCloud, zda a jak je možné jej rozšířit o nová filtrační kritéria, která by výrazně usnadnila práci příslušníkům orgánů činných v trestním řízení. NextCloud již obsahuje možnost filtrace obsahu, ale pouze podle názvu souboru. Tento původní systém vyhledávání souborů podle jejich názvu sloužil jako základ.

Právnická a kriminalistická část teoretické části práce zodpovídá na otázky, jak postupovat při tvorbě úložiště elektronických důkazů a jak chápat některé pojmy spojené s tématikou. Dále byla vybrána a nastíněna filtrační kritéria, která vzešla z rozhovoru s odborníky Policie ČR a odborníkem na elektronické důkazy z PrF MU. Bylo uvedeno úložiště NextCloud. V kapitole jsou dále popsána kryptografická schémata použitá při přesouvání dat mezi klientským a serverovým strojem. Protokol WebDAV se jevil jako možný způsob přidání nových filtračních kritérií, proto byl v teoretické části popsán. Dále bylo popsáno homomorfní šifrování a atributové vyhledávací šifrování. Ve stejné kapitole bylo popsáno a zhodnoceno možné využití obou konceptů pro implementaci do cloudového úložiště elektronických důkazů.

V práci byla implementována samostatně stojící klientská aplikace, která využívá WebDAV server nabízený v rámci sytému NextCloud. Její užití je omezené, ale může sloužit jako základ jiných implementací založených na protokolu WebDAV.

Stěžejní částí práce je implementace rozšířené filtrace uvnitř sytému NextCloud. Pro zjednodušení práce nebyl vytvořen nový systém filtrace, ale bylo navázáno na funkčnost „Unified Search“. Ta byla rozšířena o pět nových filtračních kritérií. Na frontendové části byl upraven filtrační formulář a na backendové části byl modifikován systém tvorby SQL dotazů. Frontend a backend spolu komunikují pomocí filtračního kritéria, které navazuje na dotazování v rámci funkčnosti „Unified Search“ a podporuje filtrační dotazování pomocí více jednoduchých kritérií naráz. Implementace byla otestována.

1 Teoretická část

Teoretická část se zabývá definicemi tématických pojmů a popisem některých algoritmů vyhledávání a software použitého v praktické části. Detailně jsou popsána kryptografická schémata systému NextCloud a byl definován protokol pro vzdálenou správu souborového systému WebDAV. Dále bylo popsáno homomorfní šifrování a atributové vyhledávací šifrování. V Tab. 1.1 a Tab. 1.2 se nachází pojmy použité v práci.

Tab. 1.1: Tabulka technických pojmů.

Technické pojmy [1] [2] [3] [4] [5] [6] [7]	
Šifra	Kryptografický systém sloužící k zajištění důvěrnosti přenášených zpráv.
Kryptografie	Kryptografie je věda zabývající se matematickými technikami spojenými s koncepty informační bezpečnosti jako jsou *důvěrnost, integrita, autentizace* a *neodmítnutí*.
Integrita	Zajištění nepozměnění informace neoprávněnými nebo neznámými způsoby.
Důvěrnost	Důvěrnost je kryptografická služba, která zajišťuje, že informace bude přístupná pouze autorizovaným entitám.
Autentizace	Autentizace je proces, při kterém entity prokazují, kým jsou.
Neodmítnutí	Neodmítnutí je kryptografická služba, která entitě znemožňuje odmítnout předchozí akce nebo závazky.
Proudová šifra	Proudová šifra je taková šifra, která šifruje zprávu po jednotlivých bitech.
Bloková šifra	Bloková šifra je kryptografické schéma, které rozdělí prostý text, aby byl odeslán pomocí bloků fixní délky t pod abecedou A. Každý blok je šifrován samostatně.
Symetrická šifra	Kryptografické schéma je symetrické, pokud pro každý asociovaný pár klíčů je jednoduché odvodit dešifrovací klíč z šifrovacího klíče nebo pokud jsou oba klíče stejné.
Asymetrická šifra	Kryptografický systém je asymetrický, pokud pro každý pár šifrovacího a dešifrovacího klíče existuje jeden veřejný a druhý soukromý klíč. Z veřejného klíče nemůžeme snadno získat soukromý klíč.
Hashovací funkce	Hashovací funkce je jednostranná výpočetně efektivní funkce zobrazující binární řetězce libovolné délky

	na binární řetězce fixní délky.
Hash (otisk)	Výsledek hashovací funkce.
Homomorfní šifrování	Šifra, která umožňuje práci nad zašifrovanými daty, aniž by znal vzdálený zpracovatel jejich obsah.
Atributové vyhl. šifrování	(= ABSE). Kryptografické primitivum, které kombinuje vlastnosti vyhledávacího a atributového šifrování.
Digitální podpis	Dodatečná data umožňující adresátovi ověřit autentičnost zprávy.
Cloudové úložiště	Cloudové úložiště je služba, která umožňuje ukládat data tak, že se přenesou přes internet nebo jinou síť do úložného systému mimo pracoviště. Tento systém je spravovaný třetí stranou.
WebDAV	WebDAV je rozšíření protokolu HTTP/1.1 umožňující práci nad vzdálenými souborovými systémy a systémy jim podobnými.
Filtrování	V rámci práce je filtrace chápána jako vyhledávání prvků datové struktury, které splňují zadanou podmínku.
Filtrační kritérium	V rámci práce se filtrační kritérium chápe jako formát podmínky, která se používá při filtrování.
Název fil. kritéria	Jednoznačné označení kritéria ve filtračním dotazu.
Argument fil. kritéria	Hodnoty vázající se k filtračnímu kritériu přenášené filtračním dotazem.
Vyhledávání	Zpřístupnění uložené informace podle zadané hodnoty.
Filtrační dotaz	Dotaz na backendovou třídu `FilesSearchProvider.php` zabalený uvnitř HTTP dotazu obsahující kombinaci názvů a argumentů filtračních kritérií. Formát je popsaný na výpisu 3.2.
Filtrační formulář	Grafické uživatelské rozhraní filtrace. Zobrazí se kliknutím na ikonu lupy vpravo nahoře.
Alias (atributu)	V práci je alias chápán jako označení řetězce, který identifikuje, SQL atribut v rámci práce s objekty typu `SearchComparison`.
NextCloud aplikace	Logicky oddělená funkcionalita systému NextCloud. Jedná se například o obsluhu úložiště, kalendář nebo nastavení. NextCloud je možné rozšiřovat o nové aplikace.

1.1 Elektronické důkazy

V následující kapitole se nachází tabulka Tab. 1.2, která obsahuje definice právnických a kriminalistických pojmů. Dále jsou v kapitole rozebrány rozdíly mezi úložištěm elektro-

nických důkazů, důkazních prostředků a stop. Podle rozhovoru byla navrhnuta a vybrána vhodná filtrační kritéria. Kapitola se opírá o neformální vyjádření Odboru kybernetické kriminality Krajského ředitelství policie Jihomoravského kraje (KŘP JMK) ohledně dotazu na podobu elektronických stop a možných filtračních kritérií.

Tab. 1.2: Tabulka právnických a kriminalistických pojmů.

Právnické a kriminalistické pojmy [8] [9]	
Trestní řád	Zákon č. 141/1961 Sb.
Důkaz	Vše, co může přispět k objasnění věci. Možné zdroje důkazů jsou vyjmenovány v § 89 odst. 2 trestního řádu.
Elektronický důkaz	Elektronický důkaz není definován českým trestním řádem. Dalo by se ovšem říci, že se jedná o důkazy s elektronickou povahou.
Důkazní prostředek	Zdroj z něhož orgán činný v trestném řízení důkazy čerpá.
El. důkazní prostředek	Není trestním řádem definovaný, ale můžeme říci, že se jedná o důkazní prostředek, k jehož převedení do srozumitelné podoby pro člověka je potřeba použít elektronické zařízení.
Stopa	Změny vyvolané událostí trestného činu.
OČTŘ	OČTŘ (= Orgány činné v trestním řízení) se podle §12 odst. 1 trestního řádu rozumí soud, státní zástupce a policejní orgány.
Policejní orgány	Policejní orgány jsou taxativně vypsány v § 12 odst. 2 trestního řádu. Objevuje se zde např. Policie ČR.

1.1.1 Vyjasnění pojmů v podkapitole 1.1

V Tab. 1.2 jsou popsány pojmy zde rozvíjené.

Stopy jsou změny vyvolané událostí trestného činu a nabývají podoby materiální a paměťové. Materiální stopou se rozumí stopa, která má odraz v materiálním prostředí. Paměťové stopy se odráží ve změně vědomí lidí. Ne všechny stopy mají vztah k trestnému činu a ty, které mají, nazýváme *stopami trestného činu* [9].

Stopy trestného činu, které objasňují věc a byly obstarány zákonným způsobem, se před soudem provádí jako důkazy. Podle § 89 odst. 2 trestního řádu jsou obě strany oprávněny „vyhledat, předložit nebo jeho provedení navrhnout“ důkazy, i pokud orgán OČTŘ takový důkaz před soud nepředvedl [8].

Obecně je elektronický důkazní prostředek „prostředek, k jehož převedení do srozumitelné podoby pro člověka je potřeba použít elektronické zařízení“. Důkazní prostředek jsou

v kontextu el. důkazů neanalyzovaná data a jsou (s dostatečným kontextem) nezávislá na původním nosiči [8].

1.1.2 Náležitosti cloudového úložiště podle interpretace právních pojmů

Při návrhu úložiště, které by bylo případně používané OČTŘ, je potřeba držet se pojmů. Pokud by se jednalo o *úložiště elektronických stop*, stačí podle Odboru kybernetické kriminality KŘP JMK povolit přístup pouze příslušnému policejnímu orgánu a státnímu zástupci.

V případě *úložiště elektronických důkazních prostředků* by měl mít k důkazním prostředkům, jakožto přílohám spisu, podle § 65 odst. 1 trestního řádu přístup i „obviněný, poškozený a zúčastněná osoba, jejich obhájci, zmocněnci“ a „opatrovník obviněného, poškozeného nebo zúčastněné osoby, jestliže tyto osoby nejsou plně svéprávné nebo je-li jejich svéprávnost omezena“. Podle § 65 odst. 1 a 6 trestního řádu je potřeba znemožnit přístup k osobním údajům svědků a údajům, ke kterým mohou nahlížet pouze OČTŘ a „úředníci Probační a mediační služby činní v dané věci“.

Úložiště elektronických důkazů lze chápat jako *úložiště elektronických důkazních prostředků*. Při užším výkladu podle definicí elektronického důkazu a důkazního prostředku v Tab. 1.2 by mělo být úložiště vybaveno možností interpretace el. důkazního prostředku například nějakým prohlížečem dokumentů a přehrávačem videa nebo zvukového záznamu.

1.1.3 Častá podoba stop

V rozhovoru bylo zmíněno, že se Policie ČR setkává s běžnými typy souborů jako jsou dokumenty, obrázky, videa nebo hudba například v rámci vyšetřování trestných činů proti autorským právům. Dále jsou častou podobou stop bitové kopie např. disků nebo celých strojů. Takové kopie mohou být značně objemné a dosahují velikosti desítek terabajtů. Jiné specifické typy stop, se kterými se policie setkává jsou záznamy telekomunikačního provozu a záznamy emailové nebo chatové konverzace.

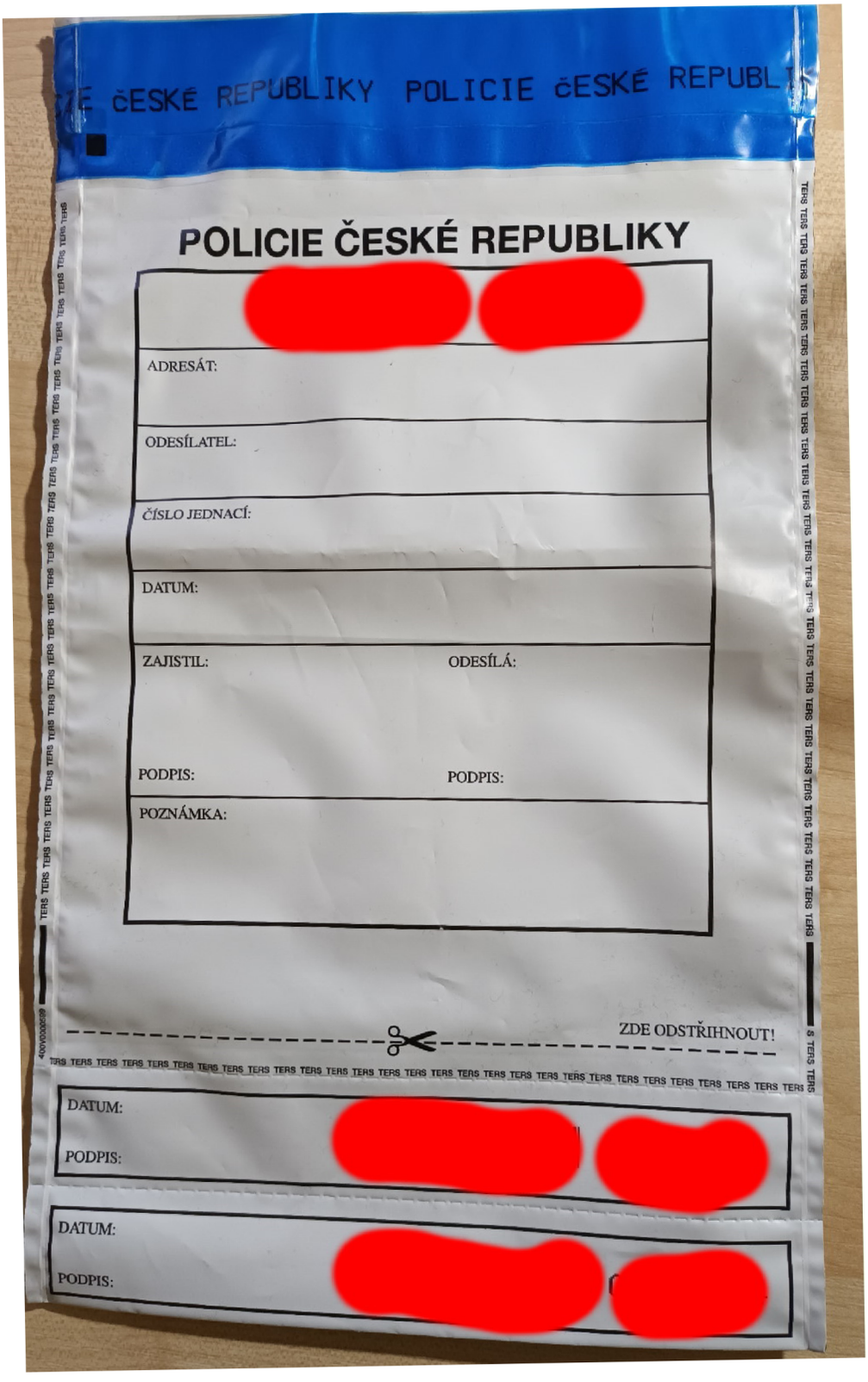
1.1.4 Filtrační kritéria z pohledu OČTŘ

Na základě neformálního vyjádření JUDr. Mgr. Jakuba Harašty, Ph.D. z Ústavu práva a technologií PrF MU byla navrhnuta kritéria *datum posledního otevření souboru*, *oprávnění pro přístup k souboru nebo složce* a *výstup hashovací funkce*[1].

Na Obr. 1.1 je vidět sáček, do kterého Policie ČR ukládá stopy po jejich zajištění. V případě elektronických stop by podle odborníků Policie ČR pravděpodobně každá z nich obsahovala např. v podobě štítku stejné informace jako na sáčku na obrázku. V seznamu níže je popis jednotlivých polí z Obr. 1.1. Ne všechny se vyplňují.

- **Odesílatel**: Vyplňuje se od koho a kde byla věc předána nebo odňata. Pole obsahuje jméno, příjmení a adresu.

[1] Kontrolní součet.



Obr. 1.1: Sáček pro zajištění stop používaný Policií ČR (upraveno).

- **Číslo jednací**: Alfanumerická identifikace případu. Seskupuje dohromady stopy v případu[2].
- **Datum zajištění**: Datum, kdy byla stopa zajištěna.
- **Zajistil**: Jméno příslušníka Policie ČR, který stopu zajistil.
- **Poznámka**: Do pole se píše číslo stopy a další poznámka ke stopě. Na druhé straně sáčku je pokračování pole „Poznámka“.

1.2 Filtrace dat na cloudovém úložišti

V rámci práce je filtrace chápána jako vyhledávání prvků datové struktury, které splňují zadanou podmínku. Rozdíl mezi filtrací a vyhledáváním je chápán tak, že při vyhledávání chce uživatel vidět konkrétní výsledek, o kterém neví, kde se nachází. Účelem filtrace je zorientovat se v prezentovaných datech.

Vyhledávání

Vyhledávání můžeme rozdělit podle místa uložených dat na *metody vnitřního vyhledávání* pro data načtená do operační paměti a *metody vnějšího vyhledávání* nad daty uloženými ve vnější paměti. Dále se dá vyhledávat buď *staticky* nad daty, které se nemění, a *dynamicky* nad měnícími se daty [3].

Zásadně se liší vyhledávání z hlediska seřazení dat. Nad neseřazenými daty musí být použity *sekvenční metody*, které pro pole při vyhledávání jednoho prvku mají časovou složitost $O(n)$. Sekvenční vyhledávání může být použito i při práci nad seřazeným polem s tím, že můžeme skončit, pokud je hledaný prvek ve vzestupně seřazeném poli menší než poslední porovnávaný prvek z pole. Notace omikron se v tomto případě nezmění [3].

Binární vyhledávání je typ nesekvenčního vyhledávání, který staví na principech půlení intervalů. Z pseudomediánu se udělá pomyslný kořen a porovná se, jestli je hledaný prvek menší než, větší než nebo rovný pseudomediánu. V případě, že je menší než pseudomedián, se z pseudomediánu stane pravý okraj pole a vytvoří se nový pseudomedián mezi levým a novým pravým okrajem. Obdobně pokud je hledaná hodnota větší než pseudomedián, stane se z něj nový levý okraj a vytvoří se nový pseudomedián. Pokud se hodnoty pseudomediánu a hledaného prvku rovnají, byl výsledek nalezen. Výše zmíněné kroky se rekurzivně opakují [3].

Další možností je využití jiných datových struktur než pole. Mezi dvě struktury vhodné pro vyhledávání patří *binární vyhledávací strom* (BVS) a *tabulka s rozptýlenými položkami* (hashovací tabulky). BVS v rámci vyhledávání funguje podobně jako *binární vyhledávání*, ale zároveň hodnoty udržuje vždy seřazené. Mapovací funkce hešovací tabulky vytvoří z dat klíč a na místo klíče v poli data vloží data. Na data pak při vyhledávání přistupujeme přímo [3].

[2]Lépe řečeno: „Seskupuje dohromady stopy ve věci.“ Jedná se o právnický pojem.

1.3 NextCloud

NextCloud je open source cloudové úložiště vyvíjené převážně německou společností NextCloud GmbH. NextCloud umožňuje instalaci aplikace na lokálním serveru nebo hosting přes partnery. Vývojová společnost také nabízí službu Nextcloud Enterprise, která zájemcům nabízí profesionální instalaci produktu NextCloud na server a otestování jeho chodu [10].

Výhodou úložiště z pohledu bezpečnosti je možnost využití prvků jako je ochrana proti útokům hrubou silou nebo šifrování dat [10]. Vývojový tým si je natolik sebejistý, že vypsal odměnu v hodnotě $10 000 za nalezení vzdáleně proveditelných bezpečnostních zranitelností [11]. NextCloud splňuje bezpečnostní standard *dodatek 14 ISO/IE 27001-2013* [12], který například vývojáře zavazuje, že bude používat pouze profesionálně vytvořené kryptografické knihovny nebo že jsou kryptografické standardy v projektu neprolomené [13].

NextCloud obsahuje standardní řízení přístupu k souborům (FAC). Adminitrátor může odepřít přístup k datům například podle skupiny, ve které se uživatel nachází, nebo podle jeho předpokládané lokace. Nabízená automatizace úkolů se dá použít pro usnadnění práce s velkým množstvím dat nebo při potřebě úpravy po splnění nějaké podmínky (například nahrání nového souboru na úložiště) [14].

1.3.1 Výhody systému NextCloud

Pro administrátory systému může být výhodou automatické vyhledávání chyb v instalaci. Administrátor má také kontrolu nad zásadami přijatelných hesel a může nastavovat dobu platnosti hesla. Při zadání nového hesla je zkontrolována jeho náchylnost vůči slovníkovým útokům. Kontrola probíhá přes databázi uniklých hesel na stránce *haveibeenpwned.com* [15][16].

NextCloud využívá umělou inteligenci na bázi neuronové sítě pro rozpoznání možných neoprávněných přístupů na cloud. Prvních 60 dní chodu systému[3] umělá inteligence trénuje svůj model na platných přihlašovacích pokusech. Po této době začne označovat pokusy o přihlášení, které se neuronové síti zdají být podezřelé. V záložce *Security* můžeme nalézt informace o jejich historii [17].

Význam pro práci má *NextCloud App Store*, který slouží jako repozitář pro rozšíření funkcionalit úložiště. Vedle zařazení do GitHub repozitáře je toto jedna z možností, jak výsledek práce zveřejnit. Monitorování provozu na serveru může administrátor provádět přes grafické rozhraní NextCloudu nebo se může napojit na API a vytvořit si vlastní monitorovací infrastrukturu. Sledovat dění na serveru se dá i přes logy [18].

1.3.2 Možnosti šifrování v systému NextCloud

NextCloud klasifikuje svoje kryptografické funkce do tří kategorií: Zabezpečení během přenosu, zabezpečení úložiště a šifrování v klientovi (end-to-end) [19].

[3] Šedesát dní je výchozí hodnota.

Hierarchie digitálních podpisů

Navíc uživatelé platformy mohou použít hierarchickou strukturu digitálních podpisů. Administrátor systému definuje autority a kdo jaké autoritě musí jaký typ dokumentu podat pro schválení. Autoritou v tomto případě může být například manažer. NextCloud sám o sobě neobsahuje žádný modul pro podepisování, ale administrátor může z *NextCloud App Store* stáhnout podle svých požadavků nějaký podpisový modul. Požadavky mohou být například zkušenosti s použitými kryptografickými knihovnami nebo organizační či právní důvody [14].

Šifrování na transportní vrstvě

Pro přenos dat používá NextCloud zabezpečení na transportní vrstvě (TLS). TLS není záležitostí samotné aplikace, takže si administrátor musí zajistit patřičný certifikát sám. Pokud na webovém serveru není zprovozněné TLS, bude administrátor na skutečnost upozorněn [20].

Standardní režim šifrování na úložišti

V následujícím seznamu je popsán postup dešifrování, pokud je využit standardní režim šifrování na úložišti. Pořadové číslo v seznamu odpovídá menším číslům na Obr. 1.2 [19].

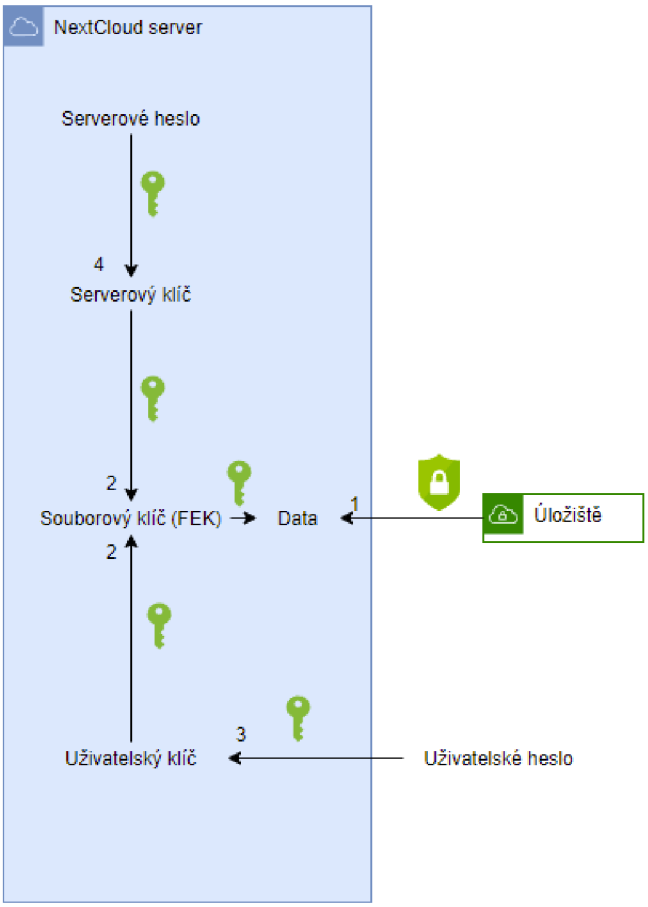
1. Šifrování dat na úložišti, respektive dešifrování, je prováděno na NextCloud serveru, který může být oddělen od úložiště.
2. FEK (file encryption key, česky souborový klíč), kterým je soubor zašifrován, je pro každý soubor unikátní. Správce si může vybrat dvě možnosti zašifrování FEK – buď serverovým klíčem nebo uživatelským klíčem.
3. Uživatelský klíč je šifrovaný uživatelským heslem, které uživatel zadává na straně klienta.
4. Uživatelé mají vlastní kopii serverových klíčů ve svém adresáři na serveru - ten je šifrovaný serverovým heslem, které je pro celý sever globální.

Postup dešifrování můžeme vidět na Obr. 1.2.

Pokud se NextCloud server i úložiště nacházejí na stejném stroji, hrozí možnost odcizení disku s daty i serverovým heslem zároveň. Proto je potřeba buď servery oddělit, používat šifrování uživatelským klíčem pro všechny soubory nebo zajistit utajení celodiskovým šifrováním - například pomocí aplikace *dm-crypt*. Celodiskové šifrování zamezí i úniku metadat, které jsou na serveru nešifrované. Jelikož šifrování a dešifrování dat probíhá na serveru, jeho kompromitace může vést k úniku dat [20].

End-to-end šifrování

Problém možného úniku dat ze severu řeší podpora end-to-end šifrování. Jedním koncem se myslí úložiště a druhým uživatelský klient – NextCloud server tak v tomto schématu žádné šifrování či dešifrování neřeší a k žádnému úniku dat na straně serveru ani nemůže

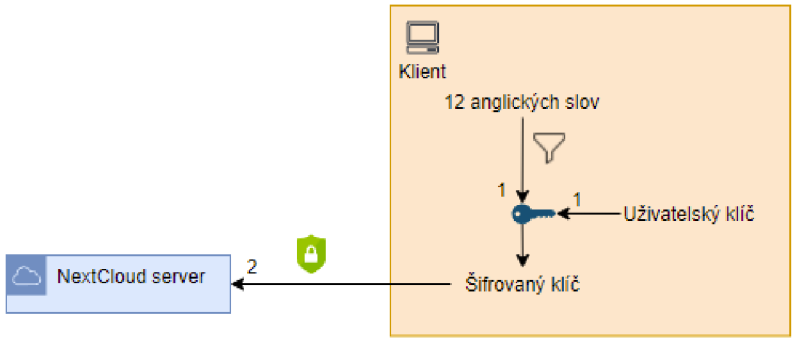


Obr. 1.2: Postup dešifrování dat na úložišti.

dojít. Teoreticky může být pomocí end-to-end klíče zašifrované celé úložiště, ale to by vzhledem ke ztrátám některých funkcí (například online editace, usnadněné sdílení nebo nemožnost náhledu souborového systému) bylo nepraktické. Proto si může uživatel vybrat pouze složky s potřebou vyššího stupně zabezpečení. Existuje i integrace s FAC, kde administrátor může požadovat, aby některé složky nebo soubory s nějakou stejnou charakteristikou (tag, přípona, ...) byly šifrovány end-to-end [20]. Před tvorbou uživatelského end-to-end klíče je ustanovena identita uživatele na klientovi. Identita se ověřuje přes X.509 certifikát podepsaný autoritou – NextCloud serverem. Uživatelský klíč si vytváří klient sám, ten pak posílá zašifrovaný na server:

1. Pro **zašifrování klíče** se používá náhodný „mnemotechnický prostředek“ sestavený z 12 anglických slov generovaných na straně klienta. Uživatelský klíč se šifruje 128bitovým otiskem prostředku pomocí standardu AES v režimu GCM. Ten se po zašifrování klíče zobrazí uživateli, aby si ho mohl někam poznamenat – používá se totiž pro obnovu klíče nebo pro sdílení dat s dalším zařízením nebo uživatelem.
2. Zašifrovaný se pak posílá na NextCloud server [21].

Postup tvorby uživatelského klíče a jeho zaslání na server je znázorněn na Obr. 1.3.

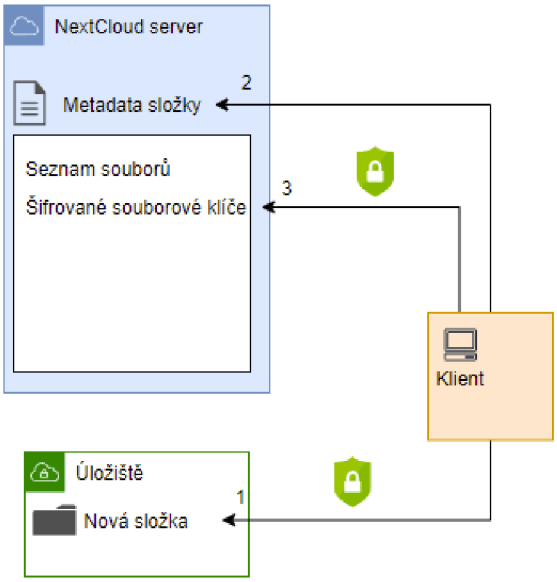


Obr. 1.3: Tvorba a posílání klíče na NextCloud server.

Následuje popis tvorby nové složky na serveru.

1. Až bude na serveru nahraný klíč a uživatel bude mít ověřený svůj certifikát, může **vytvořit v úložišti složku**. Po jejím vytvoření ji označí k šifrování – od tohoto momentu je nedostupná pro NextCloud server. Nelze šifrovat end-to-end složku, která již obsahuje data.
2. Ke každé složce náleží JSON soubor s metadaty – ten obsahuje seznam souborů a podsložek, informace o povolení přístupu a kopie klíčů určených pro šifrování samotných dat.
3. Tyto kopie jsou šifrovány uživatelskými klíči, jejichž majitelé mají přístup ke složce. [21].

Postup tvorby nové složky v úložišti je znázorněný na Obr. 1.4.

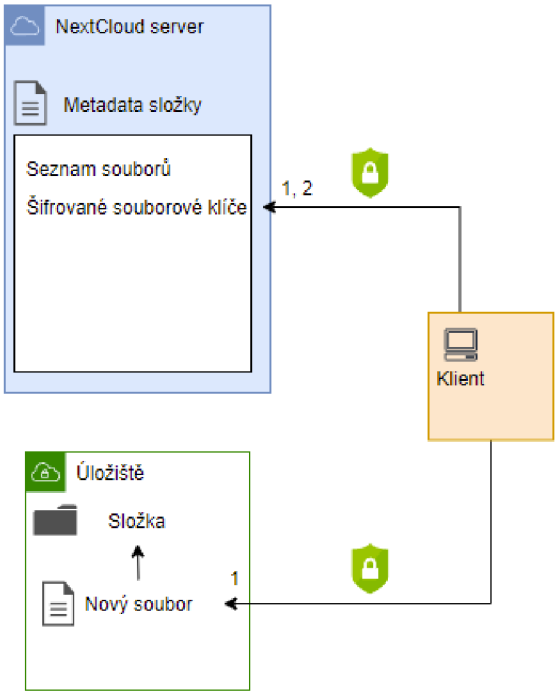


Obr. 1.4: Tvorba nové složky.

Nahrávání nového souboru nebo jeho aktualizace probíhají podobně.

1. Pro nahrávaná data se vytvoří nový tajný klíč, kterým se data na straně klienta zašifrují a odešlou na server.
2. Při aktualizaci dat se musí do metadat složky přidat nový souborový klíč zašifrovaný uživatelským klíčem. Při end-to-end šifrování prakticky neexistuje možnost data „aktualizovat“, pouze nahradit [21].

Postup nahrání a přepsání souboru ve složce je znázorněný na Obr. 1.5.



Obr. 1.5: Nahrávání souboru.

Následuje popis stahování dat ze serveru a jejich dešifrování.

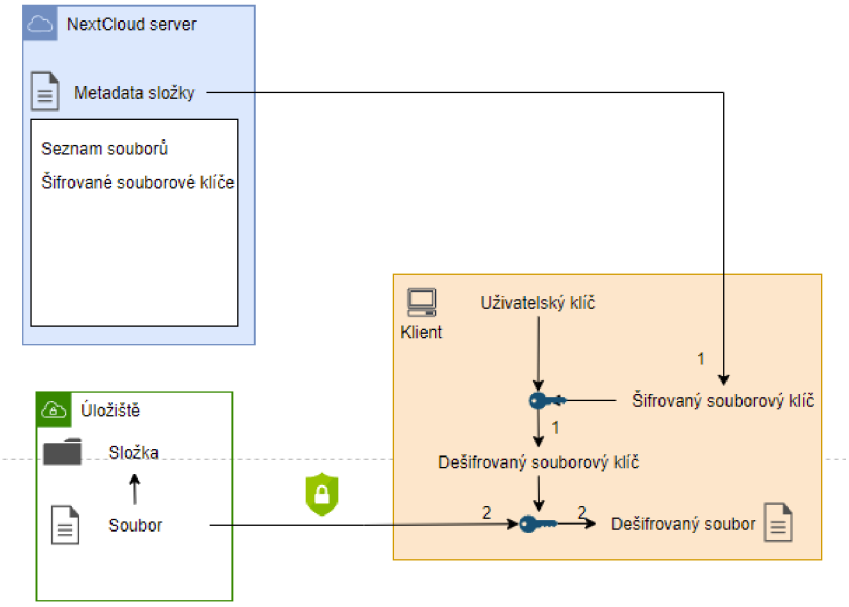
1. Pro **stáhnutí dat** si klient vyžádá metadata složky, ze kterých získá seznam souborů ve složce a jejich patřičné souborové klíče zašifrované uživatelským klíčem.
2. Následně soubor stáhne a u klienta dešifruje [21].

Postup stažení a dešifrování je znázorněn na Obr. 1.6.

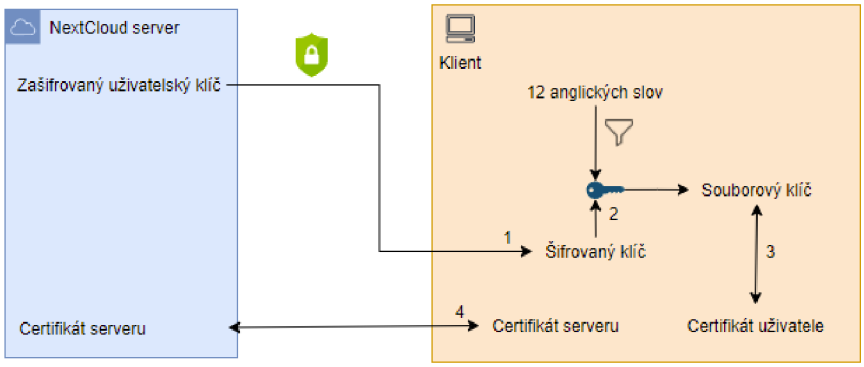
Následuje popis postupu propojení nového zařízení se stávajícím účtem.

1. Pokud chce uživatel užívat **nové zařízení**, musí si ze serveru stáhnout svůj zašifrovaný tajný klíč.
2. Ten pomocí zhešovaného „mnemotechnického prostředku“ (12 anglických slov) dešifrujeme.
3. Následně si klient ověří, zda patří klíč k dříve staženému certifikátu (viz tvorba certifikátu).
4. Také je ověřen certifikát serveru, který by měl být ideálně získaný z inicializačního klienta [21].

Postup přidružení nového zařízení k NextCloud účtu je znázorněn na Obr. 1.7.



Obr. 1.6: Stáhnutí dat ze serveru.



Obr. 1.7: Přidávání nového klienta ke složce.

1.4 WebDAV

WebDAV je rozšíření protokolu HTTP/1.1, které přes své metody dokáže pracovat se soubory a kolekcemi na serveru a přes `namespace` operace jim upravovat přístupové cesty. V RFC není vyloženě napsáno, že se WebDAV používá pouze pro souborové systémy. `Properties` v rámci protokolu umožňují uživateli vytvářet, mazat soubory a ze serveru o nich stahovat informace. Těla dotazů bývají ve formátu `XML` a v hlavičce se využívá sada nestandardních HTTP metod specifikujících operaci. Příklad těla požadavku je uveden na výpisu 2.3. Tvorba HTTP dotazu v jazyce Python se nachází na výpisu 2.2 [4].

Mezi HTTP metody využívané protokolem WebDAV patří `PROPFIND` pro získání metadat podporovaných WebDAV serverem podle zadané URL. `PROPFIND` na rozdíl od metody `SEARCH` přijímá pouze filtrační požadavky obsahující cestu k souboru/složce. Právě me-

toda `SEARCH` podporuje klíčová slova podobná těm v `SQL` a umožňuje přesunout filtraci dat z klienta na server [22] [4]. `PROPPATCH` se podobá systémem výběru souborů metodě `PROPFIND`, ale místo stažení se používá pro aktualizaci metadat (properties). `MKCOL` vytváří nový soubor uvnitř definované kolekce – v praktickém použití se jedná o tvorbu souboru ve složce. Při práci s požadavky se využívá i klasická čtveřice `HTTP` metod `GET, POST, PUT, DELETE`. Metody `GET` a `POST` ovšem nemají příliš význam a nejsou nijak standardizované. Jak již napovídá název, metoda `DELETE` maže buď kolekci nebo pouze vybraný soubor. `PUT` se podobá metodě `MKCOL`, ale oproti ní se vztahuje čistě na nahrávání nového souboru na server do již existující kolekce. `PUT` ovšem může mít vlastní definice pro použití nad kolekcí. `COPY` může být voláno nad kolekcí i souborem a jednoduše je duplikuje a přidá jim novou cestu. Analogicky operuje i `MOVE`. Na místě staré kolekce však nic nenechává. Poslední je dvojice `LOCK` a `UNLOCK`, které se starají o zamykání souborů a zabraňování souběhu na sdíleném zdroji. Každý zámek může odemčít nebo obnovit pouze majitel původního zámku [4].

1.5 Homomorfní šifrování

Homomorfní šifra je taková šifra, která umožňuje práci nad zašifrovanými daty, aniž by znal vzdálený zpracovatel jejich obsah. Díky vlastnostem homomorfních šifer je možné provádět nad zašifrovanými daty operace, které jsou ekvivalentní operacím nad nešifrovanými daty. Po dešifrování dat, nad kterými byla ekvivalentní operace provedena, dojdeme ke stejnému výsledku, jako by byla operace provedena nad nešifrovanými daty. Takovýto model nachází uplatnění, pokud je potřeba s daty pracovat na nedůvěryhodném vzdáleném bodě. Homomorfní šifry dělíme na částečně (PHE), „poněkud“ (SWHE) a plně homomorfní (FHE) [2].

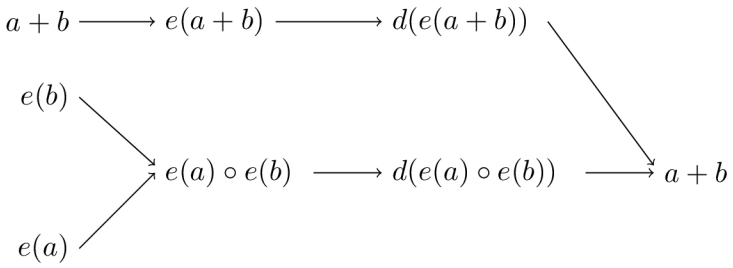
Částečně homomorfní šifry umožňují práci pouze s některými operacemi. Příkladem částečně homomorfní šifry je šifra RSA, která umožňuje provádět násobení nad zašifrovanými daty [2]. Jednoduchý důkaz o tvrzení nalezneme v bloku vzorců 1.1 [2]. První řádek znázorňuje zápis násobku dvou šifrovaných textů c_1 a c_2 do formátu dvou násobených výsledků šifrování textů m_1 a m_2 pomocí veřejného klíče pk algoritmu RSA. Druhý řádek odpovídá přepisu ze dvou instancí šifrování do jedné. Na třetím řádku vzorec umocníme soukromým klíčem sk a vzorec následně upravíme pro čtvrtý krok – tam dojde k dešifrování zprávy, protože soukromý klíč sk je inverzním prvkem veřejného klíče pk a vyjde nám součin zpráv m_1 a m_2. Tudíž platí $(c_1 \cdot c_2)^{pk} \mod n = m_1 \cdot m_2$ a šifra RSA je multiplikativně homomorfní s ekvivalentní operací (také) násobení.

Na Obr. 1.8 je vidět princip homomorfního šifrování ukázaný na aditivním částečně homomorfním modelu. Horní linka představuje zašifrování součtu zpráv a a b funkcí $e(m)$, která má na vstupu zprávu[4] m a podporuje aditivní homomorfní šifrování. V dolní lince jsou nejdříve zprávy a a b zašifrovány stejnou funkcí a následně je nad zašifrovanými zprávami provedena operace ekvivalentní k operaci sčítání nad již zašifrovaným součtem.

[4]Klíč je v tomto případě chápán jako konstanta.

$$
\begin{aligned}
c_1 \cdot c_2 &= \left(m_1^{pk} \mod n\right) \cdot \left(m_2^{pk} \mod n\right) \\
\left(m_1^{pk} \mod n\right) \cdot \left(m_2^{pk} \mod n\right) &= m_1^{pk} \cdot m_2^{pk} \mod n \\
\left(m_1^{pk} \cdot m_2^{pk}\right)^{sk} \mod n &= (m_1 \cdot m_2)^{pk \cdot sk} \mod n \;/^{sk} \\
(m_1 \cdot m_2)^{pk \cdot sk} \mod n &= m_1 \cdot m_2;\; sk \equiv pk^{-1} \mod \Phi(n) \\
\text{Výsledek} : (c_1 \cdot c_2)^{sk} \mod n &= m_1 \cdot m_2
\end{aligned}
\tag{1.1}
$$

Pokud oba šifrované texty dešifrujeme funkcí $d(c)$, vznikne v obou případech stejný výsledek. To znamená, že oba postupy jsou si ekvivalentní. Například v případě kryptosystému Pailler je k operaci násobení v kontextu aditivního homomorfního šifrování ekvivalentní operace násobení [2].



Obr. 1.8: Princip homomorfního šifrování ukázaný na aditivním PHE.

Poněkud homomorfní šifry dokáží nad zašifrovanými daty sčítat i násobit, ale za cenu zavedení tzv. *noise*. To je drobná chyba vznikající v zašifrovaných datech při provedení nějaké podporované operace. Chyba se postupem času kumuluje do té míry, že data nejde dešifrovat. Chyba se musí zavádět, aby nedošlo k odvození klíče. Poněkud homomorfní šifry proto mohou být nad daty používány jen do určité doby nebo musí být v systému implementovány korekční mechanismy jako jsou *bootstrapping* nebo *Learning With Errors*. Pokud na poněkud homomorfní šifru aplikujeme korekční mechanismus, vznikne plně homomorfní šifra [2].

Bootstrapping je založen na principu, že pokud jsou data dešifrována a následně opět zašifrována, dojde k redukci *noise* na nízkou hodnotu. Samotný *bootstrapping* využívá místo klasického dešifrování homomorfní dešifrování. To oproti klasickému dešifrování nepoužívá soukromý klíč, ale upravený veřejný klíč. Výsledkem dešifrování je zašifrovaný text s nízkou úrovní *noise* [2].

1.5.1 Dostupné možnosti implementace homomorfního šifrování

Rozšířenou možností implementace homomorfního šifrování je knihovna kryptografických funkcí `HElib`[5] napsaná v jazyce `C++`. Knihovna aktuálně podporuje schémata BGV a CKKS [23].

[5]Knihovna je stažitelná z repozitáře na adrese `https://github.com/homenc/HElib`.

Schéma BGV (Brakerski Gentry Vaikuntanathan) je plně homomorfní šifrovací schéma, které místo techniky *bootstrapping* umožňuje využívat kryptografického problému *LWE* (Learning with error) a *RLWE* (Ring learning with error). Existuje i možnost využít *bootstrapping* [24]. Samotné LWE je založeno na systému lineárních rovnic zkrácených o stejné modulo s malou chybou v konstantě na pravé straně rovnice. Pokud by se pro vyřešení soustavy použil nějaký standardní algoritmus, jako například Gaussova eliminační metoda, chyba by se nasčítala a výsledek algoritmu by se značně lišil od reálného výsledku soustavy. Tajemství vyjádřené vektorem přibližně správných řešení v přijatelném intervalu lze snadno ověřit jejich dosazením do soustavy rovnic. Úprava v podobě *RLWE* pak značně snižuje velikost klíče [25]. Schéma CKKS je založené na BGV a na problému *RLWE* [26].

1.5.2 Zhodnocení využití homomorfního šifrování

Pokud by se homomorfní šifrování použilo k úpravě samotných souborů, nedávalo by to vzhledem k povaze věci smysl. Pro úpravu elektronických stop/důkazů orgány činnými v trestním řízení neexistuje oprávněný důvod, a pokud k nějaké úpravě dojde, měl by se vypočítat nový otisk, zaznamenat čas přístupu a účet, který k datům přistupoval. Zároveň je stále potřeba uchovávat původní neupravenou verzi stopy.

Příležitost pro homomorfní šifrování nastává, pokud by k datům přistupoval i soudní znalec, který má za úkol s daty experimentovat a zjistit, zda mohla situace popsaná soudem nastat. V tomto případě je potřeba udělat kroky jako v odstavci výše.

Homomorfní šifrování by mohlo být implementováno v rámci módu šifrování na serveru NextCloud. Ten umožňuje vkládat vlastní šifrovací moduly s možností vlastního schématu zacházení s klíči. Stačí v rámci vlastní šifrovací aplikace implementovat interface `\OCP\Encryption\IEncryptionModule` [19]. Vzhledem k problémům s centralizací administrátorských práv a přidělování oprávnění při režimu šifrování na straně serveru je bezpečnější využívat end-to-end šifrování popsané v kapitole 1.3.2. Při využití end-to-end schématu se počítá s kompromitací NextCloud serveru například ze strany „zběhlého“ (rogue) administrátora. Zároveň nemá administrátor přístup k přidělení oprávnění ke složce, protože řízení přístupu je prováděno kryptografickými prostředky. Další možností, jak decentralizovaně řídit přístup k souborům, je atributové vyhledávací šifrování popsané níže v kapitole 1.6.

1.6 Atributové vyhledávací šifrování

Atributové vyhledávací šifrování (ABSE, anglicky attribute based searchable encryption) je kryptografické primitivum kombinující vlastnosti dvou dalších primitiv – vyhledávacího šifrování a ABE (attribute based encryption) [7]. Vyhledávací šifrování je takové šifrování, které umožňuje vyhledávat nad zašifrovanými daty, aniž by došlo k porušení důvěrnosti jak vyhledávaného klíčového slova, tak textu, nad kterým je vyhledáváno [27]. ABE pak k vyhledávacímu šifrování přidává funkcionalitu řízení přístupu založeného na kryptografii a možnost přiřazení jednoho nebo více klíčů k jednomu šifrovanému textu. Klíč i šifrovaný

text jsou v ABE schématu svázány s množinou atributů. K dešifrování doje právě tehdy, když se dané množství atributů klíče shoduje s atributy šifrovaného textu. Existují dva druhy ABE:

- ABE založené na pravidlech uložených uvnitř šifrovaného textu (CP-ABE)
- ABE založené na pravidlech uložených uvnitř klíče (KP-ABE)

Pokud jsou pravidla přístupu uložena v šifrovaném textu, jsou atributy uložené v klíči. Naopak pokud jsou pravidla uložena v klíči, jsou atributy uloženy v šifrovaném textu. CP-ABE má nevýhodu, že pravidla přístupu jsou nešifrovanou formou zaslány uživateli, který je může vidět [28].

ABSE tedy umožňuje dešifrovat informace pouze, pokud k nim má uživatel oprávnění a vyhledávat pomocí klíčových slov nad zašifrovaným textem. Příkladem ABSE schématu je A^2BSE, které k vlastnostem výše uvedeným přidává i možnost skrýt identitu uživatele [28]. ABSE je možné do NextCloud přidat podobným způsobem jako homomorfní šifrování v kapitole 1.5.2.

1.6.1 Zhodnocení využití ABSE

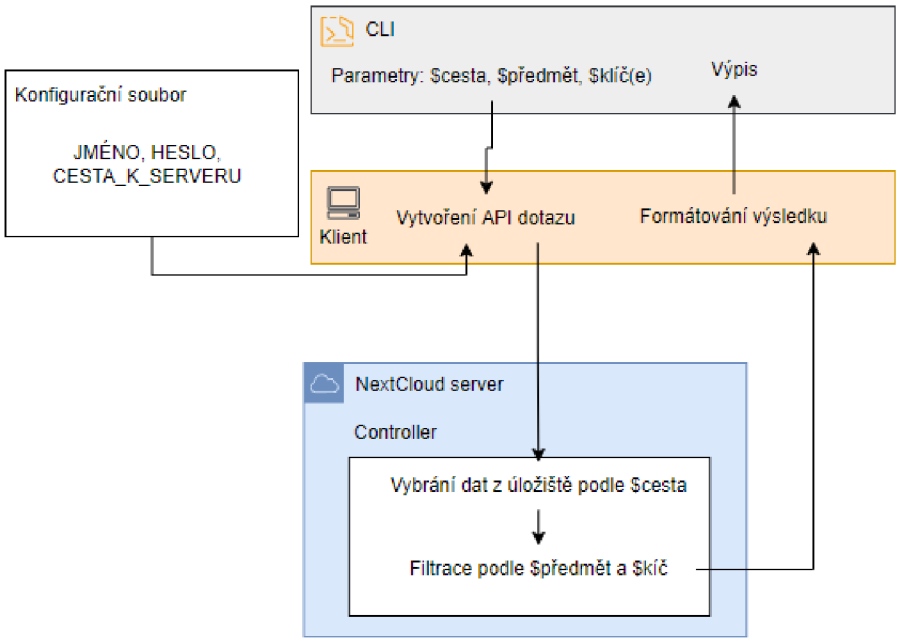
ABSE má využití pro implementaci v cloudovém úložišti, které je určeno na ukládání elektronických důkazů. Vyhledávání v textu je na cloudovém úložišti obecně výhodou a ABE umožňuje řízení přístupu pomocí kryptografie.

Uživatelé by v takovém modifikovaném sytému NextCloud disponovali klíčem s atributy, které by jim dovolily přístup k datům. Kategorie atributů mohou být například *Orgán činný v trestním řízení*, *Případ, na kterém příslušník pracuje*, *Stav bezpečnostní prověrky* nebo *Stát*, pokud by bylo úložiště důkazů použito mezinárodně.

2 Samostatně stojící filtrační aplikace

Účelem kapitoly je popsat implementaci samostatně stojící klientské aplikace, která má za účel vrátit data splňující podmínky filtrace z NextCloud serveru. Dalším účelem kapitoly je nastínit praktické využití protokolu WebDAV pro práci s NextCloudem.

Digram návrhu nalezneme na Obr. 2.1. Vybraný návrh se skládá ze zjednodušené klientské aplikace a XML souborů s těly API dotazů. Na straně serveru se používají již vytvořené API endpointy. Oproti implementaci uvnitř systému NextCloud v kapitole 3 není nijak upraven zdrojový kód samotného NextCloudu.



Obr. 2.1: Obecný návrh samostatně stojící klientské aplikace.

Princip filtrování tímto způsobem je založen na WebDAV API dotazech na NextCloud server. API endpoint naslouchá na HTTP požadavky s tělem ve formátu XML, které připomínají svou syntaxí i sémantikou SQL dotazy. Příklad XML těla je vidět na Obr. 2.3.

Pro účely filtrování byly vybrány tři kritéria: typ souboru, datum poslední úpravy a velikosti souboru. V případě kritéria „typ souboru“ je klíčem jeden argument, u zbytku se jedná o rozmezí dvou hodnot. Při používání aplikace si může uživatel vypsat nápovědu pomocí přepínače `-h` – jeho výstup se nachází ve výpisu 2.1 [29].

2.1 Implementace samostatně stojící klientské aplikace

Aplikace, která je jeden z výsledků práce, je vytvořená v programovacím jazyce Python ve verzi 3.10, nižší verze nejsou podporovány. Pro zpracování příchozích HTTP odpovědí se používá část knihovny `xml.etree.ElementTree`, komunikaci a zpracování HTTP požadavků provádí knihovna `requests`. Dále jsou použity méně důležité pomocné knihovny

Výpis 2.1: Výpis přepínače `-h`.

```
\$ python3.10 clientCustomRequest.py -h
SIMPLE PYTHON NEXTCLOUD WEBDAV CLIENT
-------------------------------------
USAGE:
 python3.10 clientCustomRequest.py [PATH or -h] [MODE] [MODE ARGUMENTS]

OPTIONS:
 [PATH] - Should begin with /. Path in your NextCloud storage from your root folder
 [-h] - Help information
 [MODE] - Choices:
     [-type] - Filter by type of file. For example 'image'
               or 'text'. List of file types is defined on NextCloud server.
         This option requires additional argument [SUBJECT]
     [-edit] - Filter by last date of editation of files in a folder.
         This option requires two additional arguments [FROM] [TO] of a type 'date'
         Use YYYY-MM-DD
     [-size] - Filter by the size of files in a folder.
         This option requires two additional arguments [FROM] [TO] of a type
         'data size'. No unit means the size is in bytes. You can also write
         the with a unit - eg. '10GB'.
         Bytes must be written without any unit!
```

`datetime` pro vytvoření standardizovaného času, `sys` pro čtení argumentů a nucené ukončení programu a `urllib` pro transformaci textu z HTTP URL kódování do UTF-8.

Na vstupu zadá uživatel podle návodu na výpisu 2.1 cestu, ze které chce získat filtrované soubory, kritérium filtrace a argumenty kritéria. Podle konfiguračního souboru `login.txt` si aplikace definuje konstanty znázorňující uživatelské jméno, heslo a cestu k WebDAV serveru.

Aplikace umožňuje filtraci podle typu souboru, data poslední editace a velikosti souboru. Při filtrování podle typu souboru se využívají tzv. top level *Media types* (popsané v kapitole 2.2).

Data posledních editací se vybírají ze zadaného rozsahu – tj. uživatel zadá dvě data a aplikace mu vrátí seznam souborů, které byly naposledy editovány v tomto rozmezí. Datum se zadává ve formátu `YYYY-MM-DD` dvakrát za sebou.

Poslední kritérium filtrace je filtrace podle velikosti souboru. Vesměs je implementace podobná předchozímu odstavci výše. Kraji intervalů jsou velikosti dat, kde uživatel může buď zadat hodnotu v bajtech bez veličiny[1] nebo k hodnotě přidat veličinu od `KB` do `TB`.

Dotazy ve formátu XML ve složce `files/` jsou využívány jako předlohy pro WebDAV API dotazy. Každé kritérium filtrace má vlastní předlohu, ve které jsou definovány proměnné – ty se doplňují o hodnoty v aplikaci při tvorbě dotazu. Pokud má HTTP odpověď návratovou hodnotu vyšší než 400, počítá se s tím, že na stala chyba a program se zastaví. Pro přihlašování se používá základní přihlašovací protokol HTTP. Více v kapitole 2.2.

Na výpisu 2.2 je vidět tvorba HTTP dotazu pomocí funkce `request` z knihovny

[1] Hodnota v bajtech musí být vždy bez veličiny.

`requests`. Funkce umožňuje tvorbu dotazů, které vyžadují nestandardní HTTP metody[2]. V tomto případě se jedná o metodu `SEARCH`. Dále do dotazu vkládáme cílovou `URL` adresu WebDAV serveru definovanou v konfiguračním souboru, XML tělo, hlavičku definující formát těla[3] a autentizační údaje.

Výpis 2.2: Tvorba HTTP dotazu v Python aplikaci.

```
1 r = requests.api.request("SEARCH",
2     URL,
3     data=xmlRequest,
4     headers=header,
5     auth=requests.auth.HTTPBasicAuth(USER, PASSWORD))
```

HTTP odpovědi ve tvaru XML jsou zpracovány knihovnou `xml` a výsledky strukturovaně prezentovány do CLI. NextCloud WebDAV server umožňuje vybrat více položek pro získaný soubor, ale těmi nejdůležitějšími, které jsou prezentovány uživateli, jsou v pořadí: *ID, velikost souboru, mime type, povolení přístupu, datum poslední modifikace* a *název souboru*. Navrácené hodnoty, které mohou obsahovat diakritiku, jsou převedeny ze zakódování používaném HTTP do UTF-8. Kritérium *velikost souboru* nevrací pro větší soubory stejnou hodnotu, jako ukazuje grafické rozhraní aplikace NextCloud, ale stále se jedná o orientační hodnotu.

2.2 Struktura API dotazů

NextCloud server přijímá WebDAV API požadavky přes nestandardní HTTP metody popsané v podkapitole 1.4. Pro požadavky implementace Python aplikace se používá metoda `SEARCH`. V hlavičce požadavku se dále nachází cesta k `WebDAV` serveru v rámci serveru NextCloud, přihlašovací údaje a typ formátu těla.

Autentizace probíhá přes již zmíněný protokol základního přihlašování HTTP, respektive `HTTPS`. V případě protokolu HTTP není přenos šifrovaný [30].

Tělo dotazu[4] metody `SEARCH` připomíná `SQL` dotaz – omezuje se ovšem kromě predikátů pouze na `SELECT`, `WHERE`, `FROM` a `ORDER BY`. Každý z těchto příkazů funguje jako samostatný oddíl dotazu. Klíčové slovo `WHERE` definuje, jaké atributy má odpověď vrátit – ty jsou pro všechny tři typy výběrů, popsaných níže, stejné. Výběr `WHERE` se pro každý typ výrazně liší a znázorňuje podmínku filtrace. Zde se dají využít klasické SQL operátory jako `LIKE` nebo >= v podobě `gte` [22].

`FROM` v rámci těla dotazu reprezentuje složku, respektive složky, ze kterých se vybírá a do jaké hloubky v souborovém systému v rámci vybrané složky chceme vstupovat. Základem cesty je vždy `/files/{USER}/`[5], ke které se následně přidávají další podsložky. Hloubka zanoření do souborového systému v rámci vybrané složky je nastavená

[2]Standardními metodami se myslí `GET, POST, PUT, DELETE`.
[3]Definováno výše v kódu.
[4]Těla dotazů jsou inspirována příklady z [29].
[5]USER je v tomto případě konstanta.

na `infinite` [22].

NextCloud WebDAV server podporuje velké množství možných kritérií, které se rozdělují na ty, co jsou definovány standardem RFC 5323, a ty, co jsou definovány přímo NextCloudem [22] [29]. WebDAV server ovšem nepodporuje stejná klíčová slova pro všechny atributy. Každý atribut lze vypsat, ale pouze podle některých lze vybírat.

Při volání dotazu pro filtrování podle typu souboru se ve výběrové části XML zajímáme o klíčové slovo `d:getcontenttype`, které znázorňuje media type. „Top level“ typ je definovaný jako proměnná a nižší úrovně jsou pak s pomocí operátoru `LIKE` vybírány libovolně (tj. jsou vráceny všechny typy nižších úrovní obsáhlé v typu nejvyšší úrovně).

Mime type (v dnešní době známý jako Media Type [31]) je způsob značkování obsahu na internetu. Značka Media/Mime type se skládá z „top level“ úrovně a podtypu skládaných do stromovité struktury. Typy jsou registrovány a spravovány organizací IANA, ale v rámci lokálního použití je možné definovat vlastní mime/media types v top level úrovni `x./`. Příklad mime/media type značky může být `text/markdown`, kde `text` je top level úroveň a `markdown` je podtyp [32].

Dotaz pro filtrování podle posledního data editace obsahuje v podmínce dvě spojené podpodmínky. První `d:gte` (větší než nebo rovno) má jako obsah typu `d:getlastmodified` a vrací hodnoty menší než datum v proměnné. Podobně `d:lte` vrací datum menší nebo rovné datu zadaného do proměnné. Oboje podmínky jsou spojené operací `AND`. Podobně funguje i dotaz na velikost souboru s tím rozdílem, že atributy `oc:size` jsou v řádech bajtů žádané hranice velikostí souborů. Příklad dotazu můžete vidět na výpisu 2.3.

Výpis 2.3: Příklad těla dotazu. Vrací ID a název textových souborů ve složce /test.

```
<?xml version="1.0" encoding="UTF-8"?>
<d:searchrequest xmlns:d="DAV:" xmlns:oc="http://owncloud.org/ns">
    <d:basicsearch>
        <d:select>
            <d:prop>
                <oc:fileid/>
                <d:displayname/>
            </d:prop>
        </d:select>
        <d:from>
            <d:scope>
                <d:href>/files/nextCloudUser/test</d:href>
                <d:depth>infinity</d:depth>
            </d:scope>
        </d:from>
        <d:where>
            <d:like>
                <d:prop>
                    <d:getcontenttype/>
                </d:prop>
                <d:literal>text/%</d:literal>
            </d:like>
        </d:where>
    </d:basicsearch>
</d:searchrequest>
```

2.3 Využití samostatně stojící filtrační aplikace

Využití filtrace formou popsanou v kapitole 2 je omezené, ale využití protokolu WebDAV pro komunikaci s NextCloud serverem existuje. Například je možné vyvinout skripty na manipulaci se soubory na úložišti. Příklad skriptu, který by mohl usnadnit práci s NextCloudem, může být automatická synchronizace s lokálním úložištěm. Pokud uživatel do dedikované složky vloží soubor, zavolá se WebDAV metoda `PUT` a soubor se nahraje na server. Naopak pokud uživatel soubor smaže, zavolá se metoda `DELETE`. Obecný návod k použití WebDAV dotazů se nachází v [33].

3 Filtrace uvnitř systému NextCloud

Tato kapitola obsahuje popis implementace filtrace pomocí úpravy zdrojového kódu NextCloudu. Jedná se o hlavní praktický výstup práce. Během vývoje bylo zasáhnuto přímo do kódu původní aplikace (v kontrastu s implementací pomocí samostatně stojící klientské aplikace popsané v kapitole 2). Upravena byla frontendová (webová) i backendová část systému NextCloud a byly využity jazyky PHP, JavaScript a možnosti frontendového aplikačního rámce Vue.js. V kapitole je uveden návrh systému filtrace, který sleduje tok dat při filtraci. Dále jsou popsány stěžejní části kódu i shrnutí testů implementace, návod na přidání nových filtračních kritérií a popis možností navázání na implementaci.

3.1 Prostředí

Vývoj rozšířených možností filtrace, popsaný v této kapitole, navázal na Github větev `stable-22` repozitáře `nextcloud/server`[1]. Tato větev byla vybrána, protože vývojáři NextCloudu předělávají frontendovou aplikaci do Vue.js [34]. Pokud by při vývoji byla použita nějaká živá větev (zejména `master`), mohlo by dojít ke značné ztrátě vytvořeného kódu. Oproti běžené instalaci je potřeba po zprovoznění provést kroky popsané v kapitole 3.1.1.

Vývoj probíhal ve virtualizačním prostředí Oracle Virtual Box verze 6.1.28 s klientským strojem fungujícím na operačním systému Kali Linux a serverovým strojem fungujícím na operačním systému Ubuntu 20.04.3 LTS. Klientský stroj byl použit pouze na přístup k HTTP serveru, který funguje na serverovém stroji.

Celý vývoj probíhal v IDE VS Code, který pomocí rozšíření *Visual Studio Code Remote - SSH*[2] dovoloval upravovat kód na serverovém stroji.

3.1.1 Instalace upraveného systému NextCloud

Instalace se opírá o oficiální instalační návod pro developery připravený týmem NextCloud [35]. V případě problémů či nejasností je doporučeno se podívat do dokumentace. Následující návod navazuje na virtuální stroj poskytnutý Bc. Petrem Muzikantem, ale je možné vytvořit podobné prostředí, pokud se budete řídit oficiálním návodem instalace na Linux [36]. Před implementací tohoto návodu je potřeba navázat na jednu z těchto možností.

1. Vytvořit novou databázi v rámci SQL serveru `nextcloud22plus` a předat veškerá práva k úpravě administrátorovi *ncadmin*[3].

```
1  $ sudo mysql
2  MariaDB [(none)]> CREATE DATABASE IF NOT EXISTS nextcloud22plus
3  CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;
4  MariaDB [(none)]> GRANT ALL PRIVILEGES ON nextcloud22plus.* TO
```

[1] Odkaz na repozitář: `https://github.com/nextcloud/server`

[2] Odkaz na stáhnutí rozšíření: `https://marketplace.visualstudio.com/items?itemName=ms-vscode-remote.remote-ssh`

[3] Krok byl poskytnut Bc. Petrem Muzikantem.

```
'ncadmin'@'localhost' WITH GRANT OPTION;
MariaDB [(none)]> FLUSH PRIVILEGES;
MariaDB [(none)]> QUIT;
```

2. Smazat aktuální obsah složky serveru.

```
sudo rm -r /var/www/nextcloud/*
```

3. Smazat skryté soubory, pokud nějaké zbyly.
4. Změnit práva přístupu pro jednodušší úpravu.

```
sudo chmod o+rw /var/www/nextcloud
```

5. Přejít do složky `/var/www/nextcloud`

```
cd /var/www/nextcloud
```

6. Stáhnout upravený NextCloud z GitHubu. Kód má přibližně 2,8 GB, proto stahování bude chvíli trvat[4]. Git je potřeba si v případě jeho absence doinstalovat.

```
git clone https://github.com/SPXcz/server-VUT .
```

7. Přepnout se do větve `Chudacek-22`.

```
git checkout Chudacek-22
```

8. Stáhnout dependencies.

```
git submodule update --init
```

9. Ve složce `/var/www/nextcloud` vytvořte složku `data`.

```
mkdir data
```

10. Předat serveru práva k vybraným složkám.

```
sudo chown -R www-data:www-data config data apps
```

11. Vymazat práva „ostatním“ uživatelům.

```
sudo chmod o-rw /var/www/nextcloud
```

12. Do prohlížeče na klientském stroji je potřeba zadat adresu `https://[IP ADRESA SERVERU]/nextcloud`
13. Po načtení stránky by měl být vidět formulář s registrací administrátorského účtu a spárování s databází.
14. Registrujte administrátora s libovolným jménem a heslem. Údaje si uložte. Složku s daty (`Data folder`) ponechejte na `/var/www/nextcloud/data`
15. Údaje k databázi jsou následující:
    - User – ncadmin
    - Password – ncadmin
    - Name – nextcloud22plus
    - Host – localhost

[4] Tečka na konci příkazu je důležitá.

16. Po registraci odmítněte všechny dodatečné aplikace. Jelikož jsou změny provedeny na verzi 22, mohly by se stáhnout nekompatibilní aplikace.
17. Nyní by měl uživatele vítat Dashboard.
18. Vlevo na liště klikněte na ikonu složky s popisem *Files*.
19. Pokud byla instalace úspěšná, měly by se uživateli po kliknutí na ikonu lupy vpravo nahoře objevit rozšířené filtrační možnosti.

3.2 Návrh upraveného systému NextCloud

Filtrace navazuje na princip „Unified Search“ [37], což je systém podpory vyhledávání v NextCloud aplikacích třetích stran v rámci jednoho vyhledávacího textového pole. Tento systém byl v rámci práce výrazně upraven, aby podporoval více filtračních kritérií. Pokud není nějaký termín zřejmý, je doporučeno se podívat do Tab. 1.1.

Grafický popis filtrace se nachází na Obr. 3.1. V diagramu nejsou zobrazeny některé méně významné třídy, proto není tok dat přesně reprezentován. Proces filtrace je rozdělen na čtyři kroky:

1. Vytváření filtračního dotazu uživatelem.
2. Zpracování filtračního dotazu.
3. Dotazování se SQL databáze.
4. Aplikace výsledků a zobrazení uživateli.

3.2.1 Vytváření filtračního dotazu

Nejdříve uživatel zadá údaje, podle kterých chce filtrovat do filtračního formuláře ukázaného na Obr. 3.2. Přesněji je filtrační formulář popsán v příloze A.1.5. Data z filtračního formuláře se přetaví ve filtrační dotaz.

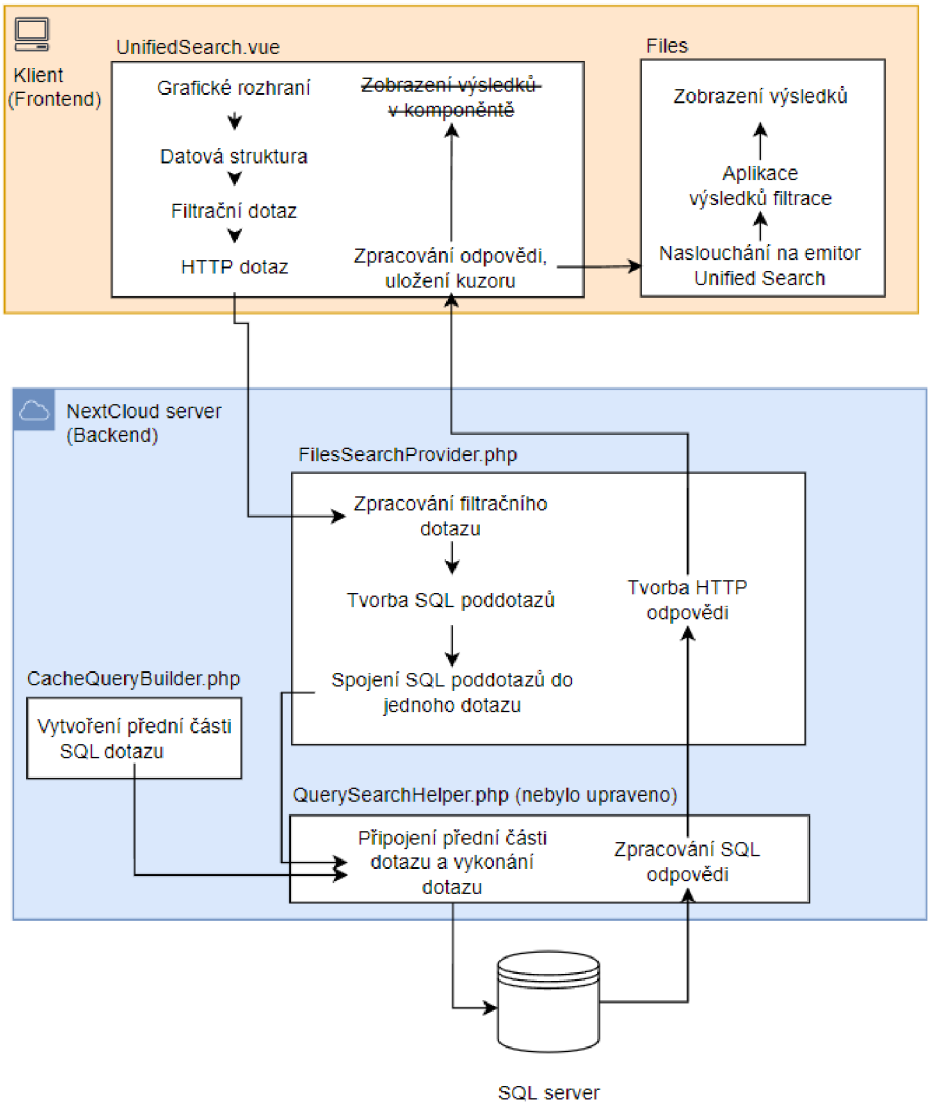
Filtrační dotaz se vytváří ve Vue.js komponentě `unified-search`, kde uživatel do filtračního formuláře zadá, podle čeho chce filtrovat. Během zadávání dat do formuláře se naplňuje speciální datová struktura, která filtrovaná data udržuje v přehledném stavu. Po stisknutí tlačítka *Submit* se přelije datová struktura do filtračního dotazu (FILTRAČNÍ DOTAZ ve výpisu 3.1) a spustí se metoda, která vytvoří a odešle HTTP dotaz na Search Providers ve standardním formátu ukázaném na výpisu 3.1 níže. Pro účely práce nebyl tento formát nijak upraven, pouze rozšířen. Rozšířený formát by měl být kompatibilní s ostatními NextCloud aplikacemi využívající Unified Search funkcionalitu.

Výpis 3.1: Formát HTTP dotazu používaný pro Unifed Search a filtrační dotazy.

```
1  /ocs/v2.php/search/providers/[CÍLOVÁ NEXTCLOUD APLIKACE]/search?
2  term=[FILTRAČNÍ DOTAZ]&from=[ZDROJOVÁ APLIKACE]?dir=[SLOŽKA]
3  &fileid=[ID SLOŽKY]
```

Vysvětlení pojmů v hranatých závorkách na výpisu 3.1 se nachází v následujícím seznamu.

1. CÍLOVÁ NEXTCLOUD APLIKACE – NextCloud aplikace, na kterou je dotaz namířen. V rámci NextCloud aplikací podporujících funkcionalitu Unified Search je

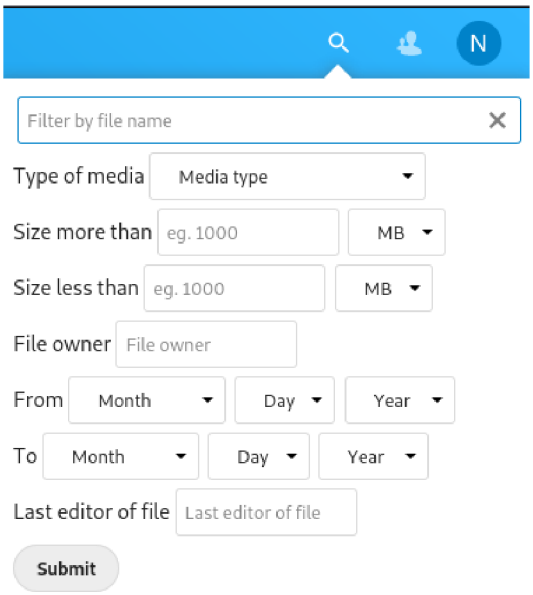


Obr. 3.1: Tok dat během filtrace v rámci systému NextCloud.

vždy registrovaná třída typu „Search Provider“, která obsluhuje Unified Search dotazy. Jak se taková třída tvoří je popsáno v dokumentaci [37].

2. FILTRAČNÍ DOTAZ – Dotaz, který se poskládá z dat zadaných do filtračního formuláře. Jeho formát je popsaný na výpisu 3.2.
3. ZDROJOVÁ APLIKACE – Aplikace, ve které se uživatel aktuálně nachází, když provádí filtraci. Pro účely práce by se vždy mělo jednat o aplikaci Files.
4. SLOŽKA – Složka, ve které se uživatel v rámci aplikace Files aktuálně nachází. Kořenová složka, která se uživateli zobrazí při prvním načtení aplikace Files je označena jako „/“.
5. ID SLOŽKY – File ID složky, ve které se uživatel v rámci aplikace Files aktuálně nachází.

Pro účely práce byl formát dotazu rozšířen, aby umožňoval filtrovat podle více kritérií a mohl být dynamicky prodlužován podle počtu vyplněných polí ve filtračním formuláři



Obr. 3.2: Filtrační formulář.

uživatelem. Jednotlivá kritéria s jejich argumenty jsou oddělena řetězcem „___“[5] a jednotlivé argumenty jsou odděleny Formát filtračního dotazu (FILTRAČNÍ DOTAZ) je popsán na výpisu 3.2 níže.

Výpis 3.2: Formát filtračního dotazu

```
1    [FILE NAME]___[KRITÉRIUM 1]::[ARGUMENT 1]::[ARGUMENT 2]
2    ___[KRITÉRIUM 2]::[ARGUMENT 1]
```

Vysvětlení pojmů v hranatých závorkách:

1. FILE NAME – Část názvu (podřetězec) souboru, který chce uživatel vyhledat. Toto kritérium má vlastní pevnou pozici na začátku každého dotazu, kvůli kompatibilitě s dalšími NextCloud aplikacemi. Pokud žádný název souboru uživatel nezadal, chová se pole jako prázdný řetězec ("").
2. KRITÉRIUM n – Název kritéria, které bylo uživatelem zvoleno k filtraci. Na místě je napsáno například *last_updater* nebo *owner*. Každé takové pole následuje m argumentů (hodnot) podle toho, kolik jich dané kritérium vyžaduje pro svou funkci. Kritéria společně s jejich argumenty jsou rozděleny řetězcem obsahujícím dvě následující podtržítka „___“.
3. ARGUMENT m – Vyjadřuje, co je obsahem filtračního kritéria pro danou filtraci. Některá kritéria vyžadují více argumentů, ty se společně s názvem kritéria (KRITÉRIUM) spojují řetězcem o dvou dvojtečkách „::“.

Níže na výpisu 3.4 je zobrazen případ filtrace, pokud by uživatel chtěl ve složce `/Folder1` zobrazit soubory vlastněné uživatelem, jehož uživatelské jméno obsahuje řetězec „ncad“, a byly naposledy upraveny po 21. květnu 2017.

[5] Dva znaky podtržítek za sebou.

Výpis 3.3: Příklad HTTP dotazu.

```
1  /ocs/v2.php/search/providers/contacts/search?term=___owner
2  ::ncad___date_to::May::7::2017&from=
3  /nextcloud/apps/files/?dir=/Folder1&fileid=65
```

Celý popsaný proces je definován v souboru `UnifiedSearch.vue`.

3.2.2 Zpracování filtračního dotazu

Systém Unified Search [37] předává třídě `FileSearchProvider` pouze filtrační dotaz a některé další informace, protože třída implementuje rozhraní `IProvider` a je registrovaná jako Search Provider.

Ve třídě `FileSearchProvider` je filtrační dotaz rozdělen, z jeho částí vytvořeny SQL poddotazy a následně jsou SQL poddotazy svázány operátorem `AND`. Přesný popis uvedeného postupu se nachází v následujícím seznamu.

1. Filtrační dotaz se rozdělí na jednotlivé filtrační poddotazy tím, že se rozdělí podle řetězce „___“.
2. Kvůli své specifičnosti se samostatně vytvoří SQL poddotaz pro kritérium *Název souboru*.
3. Vytvoří se složený SQL poddotaz s podmínkou, aby byla filtrace svázaná pouze se složkou, ve které se uživatel aktuálně nachází.
4. Iteruje se přes pole s filtračními poddotazy a vytváří se z nich SQL poddotazy podle definovaných pravidel pro jednotlivá filtrační kritéria. Jaké kritérium se v cyklu iterace bude zpracovávat, je určeno pomocí `switch` statement a množina podporovaných kritérií je tak omezená na ručně definovaná kritéria (každý `case` je systémový název kritéria). Uživatel proto nemůže filtrovat podle nepovolených kritérií – například pokud by ručně upravil/a HTTP dotaz. Samotná specifika jednotlivých filtračních kritérií jsou popsána v podkapitole 3.2.5.
5. Spojením všech SQL poddotazů logickým operátorem `AND` vznikne jeden dlouhý SQL dotaz. Stále se ještě nejedná o celý SQL dotaz, ale pouze jeho část za klauzulí `WHERE`.
6. Definuje se limit návratových hodnot. Ten je nastaven na 999 výsledků.
7. Dotaz se pošle třídám zodpovědným za vykonání dotazu.

To, co je v této kapitole chápáno jako *SQL poddotaz*, je část SQL dotazu obsahující zpravidla systémový název kritéria, jeden argument kritéria a to celé je svázané jedním logickým operátorem. V některých případech se nemusí jednat o filtrační kritérium, ale o jiná data získaná v rámci funkcionality Unified Search. *Složený SQL poddotaz* je více takových SQL poddotazů svázaných jedním nebo více logickými operátory. Ve výpisu 3.4 je ukázán příklad SQL poddotazu, který je vytvořen z kritéria last_updater s hodnotou „user“ svázané operátorem `LIKE`.

Výpis 3.4: Příklad SQL poddotazu.

```
1  last_updater LIKE "user"
```

3.2.3 Dotazování se SQL databáze

Většina této kapitoly popisuje procesy, které jsou NextCloudem již využívány. Pro vytvoření celého dotazu se zavolá metoda `search` třídy `Folder`. Ta následně zavolá metodu `searchInCaches` třídy `QuerySearchHelper`. Zde se k SQL dotazu vytvořenému v minulém kroku připojí jeho přední část vytvořená v metodě `selectFile`
`Cache` ve třídě `CacheQueryBuilder`. Přední část SQL dotazu je ukázaná na výpisu 3.5. Tři tečky ve výpisu za klauzulí `SELECT` znázorňuje velké množství vybíraných atributů.

Výpis 3.5: Přední část SQL dotazu.

```
1  SELECT ... FROM filecache AS file LEFT JOIN filecache_extended
2  AS fe LEFT JOIN share AS sh
```

Dále se k tomu celému připojí zadní část SQL dotazu s „aplikačními filtry“ a identifikátorem úložiště. Aplikační filtry jsou filtry již obsáhlé v původní verzi NextCloud a zužují výběr položek na vybranou NextCloud aplikaci. Tyto filtry jsou pro uživatele v rámci této implementace nedostupné, jejich možné zpřístupnění je popsáno v kapitole 3.6.

Následně se provede dotaz a výsledky vyhledávání v SQL databázi se pošlou zpět na frontend ve formátu popsaném v dokumentaci [37].

3.2.4 Aplikace výsledků a zobrazení uživateli

V následujícím seznamu je popsán postup aplikace výsledků filtrace:

1. Po získání výsledků si Vue.js komponenta `UnifiedSearch` uloží výsledky pro zobrazení v sobě sama. Tato funkcionalita je v rámci práce vypnutá, ale je možné ji zapnout podle instrukcí v kapitole 3.6. Problém s touto funkcionalitou byl hlavně výkonnostní.
2. Výsledky se zašlou pomocí `emittoru` souboru `fileSearch.js`. Zde jsou výsledky převedeny do pole s názvy souborů, které splňují podmínky filtrace.
3. Pole s názvy souborů se zašle souboru `filelist.js`, kde se nejdříve schovají všechny soubory a následně se znovu objeví ty soubory, které se nachází v poli. Tohoto efektu je dosáhnuto přidáváním a odebíráním třídy `.hidden` HTML kontejnerům znázorňujícím jeden soubor v seznamu.

3.2.5 Podporovaná filtrační kritéria

Kritéria a jejich formát byla vybrána po konzultaci s odborníky Policie ČR. Bral se ohled na uživatelské pohodlí a na potřeby příslušníků OČTŘ. Kritéria se řetězí pomocí logického operátoru `AND`, proto se se zvyšujícím počtem uživatelem vyplněných polí (kritérií) zmenší množství souborů splňujících filtrační podmínky. Kritéria, která mají jako argument řetězec, se považují jako podřetězce a je pro ně využit SQL operátor `LIKE`. Tím je zvýšeno uživatelské pohodlí. Podporovanými filtračními kritérii, které jsou určeny i pro technicky méně vzdělané uživatele jsou:

- Jméno souboru
- Typ souboru (podle přípony)

- Velikost souboru (horní i spodní hranice)
- Vlastník souboru
- Datum poslední úpravy (horní i spodní hranice)
- Uživatel, který jako poslední upravil soubor (poslední editor)

V následujícím seznamu jsou kritéria rozepsána. Jedná se o popis stavu při zadávání do filtračního formuláře. Popis odpovídá filtračnímu formuláři na Obr. 3.2.

- *Jméno souboru*: Podřetězec názvu souboru. Do názvu souboru se počítá i přípona, ale ne cesta k souboru.
- *Typ souboru*: Typ souboru je svázán s definovaným Media type. Media type je pak spojen s příponou souboru. Ve filtračním formuláři se jedná o menu typu „Drop down menu“. Formáty podporované kritériem *Typ souboru* jsou popsané v podkapitole „Formáty kritéria *Typ souboru*“.
- *Velikost souboru*: Toto kritérium má ve formuláři dva argumenty – Číslo velikosti a jednotku s číslem svázanou. Pole ve formuláři pro vkládání čísla velikosti je omezeno na znaky číslic. Jednotky jsou prezentovány pomocí drop down menu v rozmezí B až TB. Kritérium existuje ve verzi „velikost menší než“ a „velikost větší než“.
- *Vlastník souboru*: Vlastníkem souboru je uživatel, který nechal soubor sdílet. Filtruje se podle tzv. „NextCloud ID“. Je to jméno, se kterým se uživatel přihlašuje do systému a je každému unikátní. Při využití tohoto kritéria se omezí filtrace pouze na soubory, které jsou označeny, že jsou někým sdílené.
- *Datum poslední úpravy*: Má tři argumenty – měsíc, den, rok. Měsíc je formátu řetězce, zbytek je ve formátu integer. Kritérium existuje ve verzi „naposledy upravené před zadaným datem“ a „naposledy upravené po zadaném datu“.
- *Poslední editor*: Poslední editor je uživatel, který upravil soubor, nechal soubor sdílet nebo změnil stav sdílení. Filtruje se podle tzv. „NextCloud ID“. Je to jméno, se kterým se uživatel přihlašuje do systému a je každému unikátní. Zbylá kritéria výše popsaná byla NextCloudem nějakým způsobem již zaznamenávána do databáze, toto kritérium bylo navrhnuto a implementováno kompletně od začátku pro účely práce. Kritérii se věnuje kapitola 3.3.4.

Formáty kritéria *Typ souboru*

Toto kritérium původně mělo svazovat top level media type s typem souboru (např. *Text* s `text/%`). Problém nastal u typů souborů, které nemají vlastní top level media type nebo se některé přípony svazují s jiným top level media type (typicky s `application/%`). Další problém nastal u typů souborů, které mohou mít více definic. Svázání typů souborů s media types se nachází v následujícím seznamu.

- *Text*: Základem je top level media type `text/%` a další konkrétní media types, které vytváří podporu pro soubory s příponami `.doc`, `.docx`, `.odt`, `.pdf` a `.epub`. V top level media type `test/%` jsou zahrnuty například přípony `.txt` nebo zdrojové kódy [38] [31].
- *Images*, *Video*, *Audio*: Všechny tři typy souborů jsou definovány pouze svými top

level media types. Konkrétně se jedná o top level media types `image/%`, `video/%`, `audio/%`.

- *Disk and Computer image*: Jedná se o pravděpodobně nejsložitější typ souboru na definici. V zájmu co nejširšího záběru definice obrazu disku a počítače bylo rozhodnuto, že budou podporovány přípony `.iso`, `.vdi`, `.bin`, `.rpm`, `.adi`, `.dim` a `.ova`. Na práci by mohlo být navázáno rozhovorem s příslušníkem OČTŘ nebo soudním znalcem a získat od nich informaci, jaké přípony generuje jimi používaný software na tvorbu obrazů disků.
- *Compressed archive*: Tento typ souboru podporuje přípony `.gz`, `.zip`, `.7z`, `.rar` a `.tar`.

Definice media types byly svázány s jejich příponami v souboru `mimetypemapping.json` podle oficiálního návodu [39].

3.3 Realizace upraveného systému NextCloud

Cesty k frontendovým souborům se nacházejí v Tab. 3.1 a k backendovým souborům v Tab. 3.3. Cesty jsou v obou případech relativní ke kořenové složce v GitHub repozitáři[6] a ke složce `/var/www/nextcloud` ve virtuálním stroji Ubuntu NextCloud serveru. Obě dvě možnosti jsou součástí příloh. Upravené části kódu jsou shrnuty v Tab. 3.2 a Tab. 3.4. Důležité a obsáhlé soubory mají vlastní podkapitoly. Kromě souboru `mimetypemapping.json` nebyl žádný jiný soubor vytvořen, všechny ostatní soubory byly pouze upraveny.

Tab. 3.1: Tabulka s cestami k souborům použitých k tvorbě frontendu.

Název souboru	Cesta k souboru
UnifiedSearch.vue	core/src/views/UnifiedSearch.vue
filelist.js	apps/files/js/filelist.js
fileSearch.js	apps/files/src/legacy/fileSearch.js
filesummary.js	apps/files/js/filesummary.js
HeaderMenu.vue	core/src/components/HeaderMenu.vue
additionalScripts.js	apps/files_sharing/js/dist/additionalScripts.js

3.3.1 UnifiedSearch.vue

Soubor `UnifiedSearch.vue` obsahuje (model, view, controller) definici komponenty `unified-search`, který obsahuje filtrační formulář. Dále se v souboru vytváří filtrační dotaz, filtrační dotaz je zapouzdřen do HTTP dotazu a výsledky filtrace (HTTP odpověď) se přeposílají na aplikaci Files – přesněji na soubor `fileSearch.js`. Předmětem změny kódu nebyla stylová část souboru. Provedené změny kódu jsou uvedeny v následujícím seznamu.

- Model formuláře

[6] GitHub repozitář: `https://github.com/SPXcz/server-VUT/tree/Chudacek-22`

Tab. 3.2: Tabulka změn souborů použitých k tvorbě frontendu.

Název	**Funkce souboru pro práci**	**Nové funkce a změny**
`UnifiedSearch.vue`	Filtrační formulář a tvorba dotazů.	Viz podkapitola „UnifiedSearch.vue“.
`filelist.js`	Zobrazení seznamu souborů a aplikace filtrace.	Viz podkapitola „filelist.js“.
`fileSearch.js`	Komunikace mezi `UnifiedSearch.vue` a `filelist.js`.	Převádění JSON objektu na pole názvů souborů.
`filesummary.js`	Statistiky o aktuálním seznamu souborů.	Převedení výsledku filtrace z řetězce na pole. Vypnutí zobrazování statistik.
`HeaderMenu.vue`	Část vzhledu filtračního formuláře.	V definici kontejneru `__content` změna řádku `width=380px`.

- Úprava funkcionality aplikačních filtrů a zobrazování výsledků.
- Přidání atributu třídy `queryObject` a úprava kódu, aby uměl s atributem pracovat.
- Přidání metod vracejících hodnoty pro tvorbu filtračního formuláře.
- Přidání metody pro tvorbu filtračního dotazu `stringifyQuery`.
- Úprava metody `onInput`, která je zavolána po stisknutí tlačítka *Submit*.

Model formuláře se nachází na samém vrcholu souboru. U každého prvku formuláře byla odebrána funkčnost `debounce`, která spouštěla filtraci po dopsání nějakého podřetězce a krátké pauze. Docházelo k předčasné filtraci, což u složek s vyšším počtem souborů zpomalovalo chod filtrace. Proto se nyní spouští filtrace až po stisknutí tlačítka *Submit*. Definice tlačítka *Submit* je na výpisu 3.6.

Výpis 3.6: Model tlačítka *Submit*.

```
1  <!--Submit button-->
2  <div>
3    <button v-on:click="onInput()">Submit</button>
4  </div>
```

První textové pole bylo již v komponentě obsažené, další pole pro kritéria bylo potřeba dopsat. Každé kritérium má vyhrazen vlastní kontejner a každý argument kritéria má uvnitř další vlastní kontejner. V následujícím seznamu jsou svázána filtrační kritéria s jejich typy vstupů do formuláře a Vue.js tagem. Popis odpovídá filtračnímu formuláři na Obr. 3.2.

- *Jméno souboru*: Textové pole, převzato z Unified Search (`input`)
- *Typ souboru*: Scroll down menu (`select`)
- *Velikost souboru*: Textové pole a scroll down menu pro každou ze dvou položek (`input`, `select`)
- *Vlastník souboru*: Textové pole (`input`)

Tab. 3.3: Tabulka s cestami k souborům použitých k tvorbě backendu.

Název souboru	Cesta k souboru
FileSearchProvider.php	apps/files/Search/FileSearchProvider.php
SearchBuilder.php	lib/private/Files/Cache/SearchBuilder.php
CacheQueryBuilder.php	lib/private/Files/Cache/CacheQueryBuilder.php
Propagator.php	lib/private/Files/Cache/Propagator.php
FederatedShareProvider.php	apps/federatedfilesharing/lib/FederatedShare-Provider.php
DefaultShareProvider.php	lib/private/Share20/DefaultShareProvider.php
Version13000Date2017-0718121200.php	core/Migrations/Version13000Date201707-18121200.php
mimetypemapping.json	config/mimetypemapping.json
QuerySearchHelper.php	lib/private/Files/Cache/QuerySearchHelper.php
QuerySearchHelper.php	lib/private/Files/Cache/CacheQueryBuilder.php
Manager.php	apps/file_sharing/lib/External/Manager.php

- *Datum poslední úpravy*: Třikrát scroll down menu pro každou ze dvou položek (`select`)
- *Poslední editor*: Textové pole (`input`)

Příklad položky ve filtračním formuláři pro kritérium *typ souboru* je na výpisu 3.7. Obsah `div` kontejnerů byl inspirován původním `input` polem Unified Search. Níže v souboru `UnifiedSearch.vue` se nachází definice modelu aplikačních filtrů a zobrazování výsledků. Obě funkčnosti jsou zakomentované.

Výpis 3.7: Příklad kontejneru definujícího položku pro vybírání *typu souboru*.

```
1  <!-- Media type selector -->
2  <div class="unified-search__form-mimetype">
3    <label for="morethan">Type of media</label>
4    <select
5      v-model="queryObject.mimetype"
6      class="unified-search__form-mimetype-selector"
7      id="mimetype"
8      @keypress.enter.prevent.stop="onInputEnter">
9      <option disabled value="">Media type</option>
10     <option value="text">Text</option>
11     <option value="image">Images</option>
12     <option value="video">Video</option>
13     <option value="audio">Audio</option>
14     <option value="disk_image">Disk and Computer image</option>
15     <option value="archive">Compressed archive</option>
16     <option value="">None</option>
17   </select>
18 </div>
```

Atribut komponenty `unified-search` (definované v souboru `UnifiedSearch.vue`) JavaScript `Object queryObject` v sobě dynamicky nese argumenty kritérií během toho, co uživatel vyplňuje filtrační formulář. Například na výpisu 3.7 na řádku 5 je vidět, že se

Tab. 3.4: Tabulka změn souborů použitých k tvorbě backendu.

Název	**Funkce třídy**	**Nové funkce a změny**
`FileSearch-Provider.php`	Zpřístupňuje filtraci na backendu.	Viz podkapitola „FileSearchProvider.php“.
`SearchBuilder.php`	Pravidla filtrace.	Přidání podpory SQL dotazů pro SQL atributy *majitel souboru*, *rodičovská složka* a *poslední editor*.
`CachQuery-Builder.php`	Základ SQL dotazu.	Přidání svázání tabulek `oc_filecache` a `oc_share`.
`Propagator.php`	Změna záznamu v databázi při změně souboru.	Aktualizace atributu `last_updater` při změně souboru.
`FederatedShare-Provider.php`	Přidává záznam o sdílení souboru do databáze.	Atribut `last_updater` se vkládá do DB po sdílení souboru s jiným uživatelem.
`DefaultShare-Provider.php`	Přidává záznam o sdílení souboru do databáze.	Atribut `last_updater` se vkládá do DB po sdílení souboru s jiným uživatelem. Atribut `last_updater` se aktualizuje při změně stavu sdílení souboru.
`Version13000-Date20170718-121200.php`	Definice tabulky `oc_share`.	Přidání atributu `last_updater` do tabulky `oc_share` během inicializace databáze.
`mimetype-mapping.json`	Správcovská definice media types.	Upraveny media types podle definic z [38].

vkládají data z filtračního formuláře pro argument kritéria *Typ souboru* do proměnné `queryObject.mimetype`. Původně byly hodnoty z formuláře vedeny v atributu typu řetězec, proto byl kód změněn na několika dalších místech, aby byl podporován atribut typu `Object`. Po stisknutí tlačítka *Submit* se z dat v `queryObject` vytvoří filtrační dotaz. Ten se vytvoří v metodě `stringifyQuery`.

Filtrační formulář si pro svou tvorbu volá skupinu nových metod. Jedná se o metody `getDayArray`, `getMonthsArray` a `getYearArray`.

Filtrační dotaz je tvořen v metodě `stringifyQuery`. Zde se převádí `queryObject` na řetězec reprezentující filtrační dotaz. Aby bylo kritérium přidáno do filtračního dotazu, musí uživatel vyplnit patřičný prvek ve filtračním formuláři – tj. filtrační dotaz neobsahuje prázdné argumenty kritérií. Dále se u některých kritérií musí splnit další podmínky, aby nebyl filtrační dotaz narušen. Ze stejného důvodu doplní escape characters pro kritické podřetězce „___“, „::“ a „in“.

Metoda `onInput` se spustí po stisknutí tlačítka *Submit*. Postup nejdůležitějších operací, jak za sebou jdou v metodě `onInput`, je vyjmenován v následujícím seznamu.

1. Vytvoří se filtrační dotaz zavoláním metody `stringifyQuery`. Viz výpis 3.8.

Výpis 3.8: Vytvoření filtračního dotazu.

```
1    let query = this.stringifyQuery()
```

2. První argument se z důvodu kompatibility zašle aplikacím, co jsou součástí Unified Search. Viz výpis 3.9.

Výpis 3.9: Kompatibilita s Unified Search.

```
1    emit('nextcloud:unified-search.search', { query: this.queryObject.name })
```

3. Filtrační dotaz se zabalí do HTTP dotazu a HTTP dotaz je vykonán[7]. Viz výpis 3.10.

Výpis 3.10: Tvorba HTTP dotazu a uložení odpovědi.

```
1  // Init cancellable request
2  const { request, cancel } = search({ type, query })
3  this.requests.push(cancel)
4  // Fetch results
5  const { data } = await request()
```

4. Zaslání HTTP odpovědi NextCloud aplikaci Files. Odpověď je v rámci NextCloud aplikace Files zpracována souborem `fileSearch.js`. Viz výpis 3.11.

Výpis 3.11: Kompatibilita s Unified Search.

```
1    emit('nextcloud:unified-search.searchFiles', { query: this.orderedResults})
```

3.3.2 filelist.js

Soubor `filelist.js` se stará o zobrazení seznamu souborů a o aplikaci filtrace. V souboru byla změněna pouze metoda `setFilter` a atribut `_filter` byl převeden z řetězce na pole řetězců.

Metoda `setFilter` je *zavolána* uvnitř souboru `fileSearch.js`, kde jí je předán argument s výsledky filtrace. Hlavni změnou je způsob aplikace filtrace ve spodní části metody. Aplikace výsledků filtrace je zobrazena na výpisu 3.12. Nejdříve se vykoná část v dolní části výpisu, kde se každému souboru v seznamu souborů ve složce přidá třída `.hidden` (řádek 12). Na každý soubor v seznamu se aplikuje výsledek filtru zavoláním funkce `filterRows` (řádek 14). Ve funkci `filterRows` se iteruje přes všechny názvy souborů, které prošly filtrací (řádek 3). Pokud se název souboru v seznamu souborů shoduje s některým názvem souboru, který prošel filtrací, odebere se třída `.hidden` (řádek 4 a 6).

Výpis 3.12: Část kódu, která řeší aplikaci výsledků filtrace.

```
1    function filterRows(tr) {
2        var $e = $(tr);
```

[7]Popis toho, co je již v původní implementaci NextCloudu.

```
3           for(var individualFilter of filter){
4             if ($e.data('file').toString().indexOf(individualFilter) !== -1) {
5               visibleCount++;
6               $e.removeClass('hidden');
7             }
8           }
9         }
10
11    var $trs = this.$fileList.find('tr');
12    $($trs).addClass('hidden');
13    do {
14      _.each($trs, filterRows);
15      if (visibleCount < total) {
16        $trs = this._nextPage(false);
17      }
18    } while (visibleCount < total && $trs.length > 0);
```

3.3.3 FileSearchProvider.php

Třída `FileSearchProvider` definována v `FileSearchProvider.php` obsluhuje HTTP dotazy přicházející z Vue.js komponenty `unified-search` – viz výpis 3.10. Dále tvoří SQL dotazy a předává je třídám, které je vykonají. Výsledky se vrátí do třídy `FileSearchProvider`, kde jsou výsledky zpracovány a odeslány zpět `UnifiedSearch`. Třída je registrovaný Search Provider a podporuje funkcionality Unified Search [37]. Postup, co se děje v pro práci relevantní metodě `search`, je v podkapitole 3.2.2.

Důležitou strukturou, která se používá pro tvorbu SQL příkazů, je `SearchComparison`. Ta ukládá SQL poddotazy do takové formy, aby mohly být pospojovány do plnohodnotného SQL dotazu. Formát a příklad volání konstruktoru `SearchComparison` je na výpisu 3.13.

Výpis 3.13: Formát a příklad `SearchComparison`.

```
1     //FORMAT
2     new SearchComparison(ISearchComparison::[OPERACE], [ALIAS ATRIBUTU], [ARGUMENT])
3     //PRIKLAD
4     new SearchComparison(ISearchComparison::COMPARE_EQUAL, 'mimetype',
5     'application/gzip')
```

Vysvětlení pojmů z výpisu 3.13 je v následujícím seznamu.

- `OPERACE`: Jedna z konstatnt představující SQL operátor. Jejich seznam je vidět na řádku 40 v souboru `SearchBuilder.php`.
- `ALIAS ATRIBUTU`: Váže se k SQL atributu v podporované tabulce (`oc_filecache`, `oc_share`). SQL atribut a alias můžou být ten samý řetězec. Seznam aliasů a jejich datových typů je vidět na výpisu 3.14. Výpis byl vybrán ze souboru `SearchBuilder.php`.
- `ARGUMENT`: Argument, podle kterého se má vyhledávat v SQL databázi.

Výpis 3.14: Aliasy a jejich datové typy. Převzato ze souboru souboru `SearchBuilder.php`.

```
1   'mimetype' => 'string',
2   'mtime' => 'integer',
3   'name' => 'string',
4   'path' => 'string',
```

```
5     'size' => 'integer',
6     'tagname' => 'string',
7     'favorite' => 'boolean',
8     'fileid' => 'integer',
9     'storage' => 'integer',
10    'owner' => 'string',
11    'parent' => 'integer',
12    'last_updater' => 'string',
13    'file_target' => 'string',
```

Další pro vyhledávání v SQL databázi důležitou datovou strukturou je `SearchBinary Operator`. Ta spojuje SQL poddotazy logickým operátorem. Struktura a popis volání poddotazu jsou vidět na výpisu 3.15.

Výpis 3.15: Struktura a popis `SearchBinaryOperator`.

```
1     //FORMAT
2     new SearchBinaryOperator(ISearchBinaryOperator::[LOGICKY OPERATOR],
3     [POLE S PODDOTAZY])
4     //PRIKLAD
5     new SearchBinaryOperator(ISearchBinaryOperator::OPERATOR_AND, $provisionalQuery)
```

Vysvětlení výše posaných pojmů je v následujícím seznamu.

- `LOGICKY OPERATOR`: Konstanta svázaná s logickým operátorem SQL.
- `POLE S PODDOTAZY`: Pole s poddotazy typu `SearchComparison`.

3.3.4 Nový atribut v databázi `last_updater`

Aby mohla být možná filtrace podle posledního editora souboru, bylo potřeba do databáze přidat nový atribut `last_updater`, který by byl svázaný 1:1 s každým sdíleným souborem. Bylo možné tento atribut přidat buď do tabulky `oc_filecahce` nebo `oc_share`. První možnost má tu výhodu, že by se ke každému souboru napsal pouze jeden poslední editor. Výhodou druhé možnosti je přehlednost, protože filtrace podle posledního editora má logickou vazbu na sdílení souboru. Pro přidání posledního editora byly upraveny tři soubory vypsané v následujícím seznamu.

- `Version13000Date20170718121200.php`: Zde byla vložena definice atributu `last_updater` pro automatickou inicializaci databáze. Zobrazeno na výpisu 3.16.
- `DeafultShareProvider.php`: Zde byla definována inicializace hodnoty atributu `last_updater`, pokud je soubor sdílený. Dále je zde nastavena aktualizace hodnoty, pokud je změněn status sdílení. Změny jsou vidět na výpisu 3.17. Podobně byl upraven i soubor `FederatedShareProvider.php`, který má podobnou funkci.
- `Propagator.php`: V tomto souboru je definována změna hodnoty `last_updater` při změně souboru. Pokud je soubor změněn, zavolá se metoda `propagateChange`, ve které se nejdříve z databáze vyberou `fileid`s souborů, které se nacházejí v dané složce na daném úložišti. Následně se změní *všechny* záznamy sdílení (v tabulce `oc_share`), které odkazují na soubor s vybraným `fileid`. Změna se nachází pod řádkem 96.

Výpis 3.16: Přidání atributu do databáze.

```
$table->addColumn('last_updater', 'string', [
      'notnull' => true,
      'length' => 255,
    ]);
```

Výpis 3.17: Úpravy v `DeafaultShareProvider.php` na řádcích 203 a 266.

```
//Radek 203
$qb->setValue('last_updater',
$qb->createNamedParameter($share->getShareOwner()));
//Radek 267
->set('last_updater', $qb->createNamedParameter(\OC_User::getUser()))
```

3.4 Otestování upraveného systému NextCloud

Implementace byla otestována třemi typy testů: Výkonnostními testy nad zvyšujícím se množstvím souborů, testy přesnosti filtrace a testy, při kterých se testovaly nestandardní vstupy filtračního formuláře. Výsledky testů byly zpracovány a vyhodnoceny v excelové příloze. Grafy se nacházejí v této kapitole. Soubory byly vygenerovány pomocí skriptu napsaného v jazyce Python.

3.4.1 Generátor souborů

K otestování aplikace bylo potřeba obstarat velké množství souborů, které připomínají běžný souborový systém nebo souborový systém uživatele pracujícího s elektronickým důkazním materiálem. Možnosti obstarání testovací množiny dat jsou:

- Kontaktovat příslušníka OČTŘ.
- Kontaktovat někoho, kdo pracuje s velkým množstvím dat.
- Použít osobní souborový systém (např. studijní materiály).
- Vytvořit náhodný generátor souborů.

Nakonec byla zvolena poslední varianta díky flexibilitě tvorby souborů.

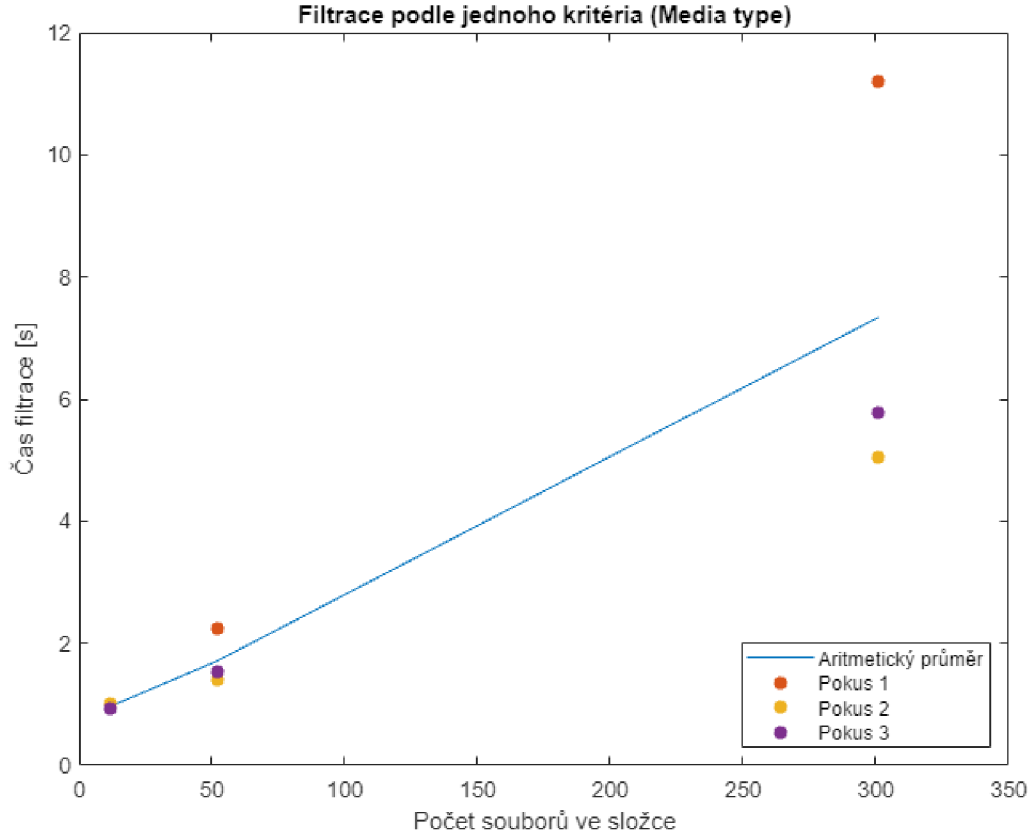
Generátor je vlastnoručně naprogramovaný v jazyce Python. Při tvorbě byly použity knihovny `xml`, `uuid`, `random`, `time`, `math` a `datetime`. Dále byl jako inicializační seznam možných přípon použit seznam od MDN Web Docs. Tento seznam svazuje vybrané přípony s jejich MIME Type [38]. Seznam je použitý i při tvorbě seznamu Media Types pro instanci NextCloud.

Aplikace se spouští v terminálu distribuce OS Linux příkazem

```
python3 fileCreator.py
```

a následně je uživatel vyzván, aby zadal počet vygenerovaných souborů. Celkem se vygeneruje $n+1$ souborů – n náhodně generovaných pojmenovaných pořadovým číslem s příponou a soubor `summary.txt`, který obsahuje záznamy o vygenerovaných souborech. Každý záznam obsahuje jméno, velikost, časové razítko a MIME Type asociovaný s příponou. Vygenerovaná množina souborů je uložena ve složce pojmenované náhodným `UUID`.

Každý soubor má náhodně generovány a zaznamenány relevantní statistiky pro otestování filtrace. Jedná se o příponu, media type, velikost souboru a mtime (časové razítko poslední změny). Media/MIME Type jsou získány z HTML stránky [38] se seznamem přípon a MIME Types, která byla stažena pomocí programu `wget` a zpracována jako XML soubor. Velikost souboru se vytváří tak, že se do něj vloží náhodné množství bajtů (max. 1 MB). Po vytvoření souboru je „zfalšován“ mtime, aby to nevypadalo, že byl každý soubor naposledy upraven v čas generování, a byla zajitěna dostatečná variace mtimes.



Obr. 3.3: Test s jednoduchým kritériem (Test 1).

3.4.2 Výkonnostní testy

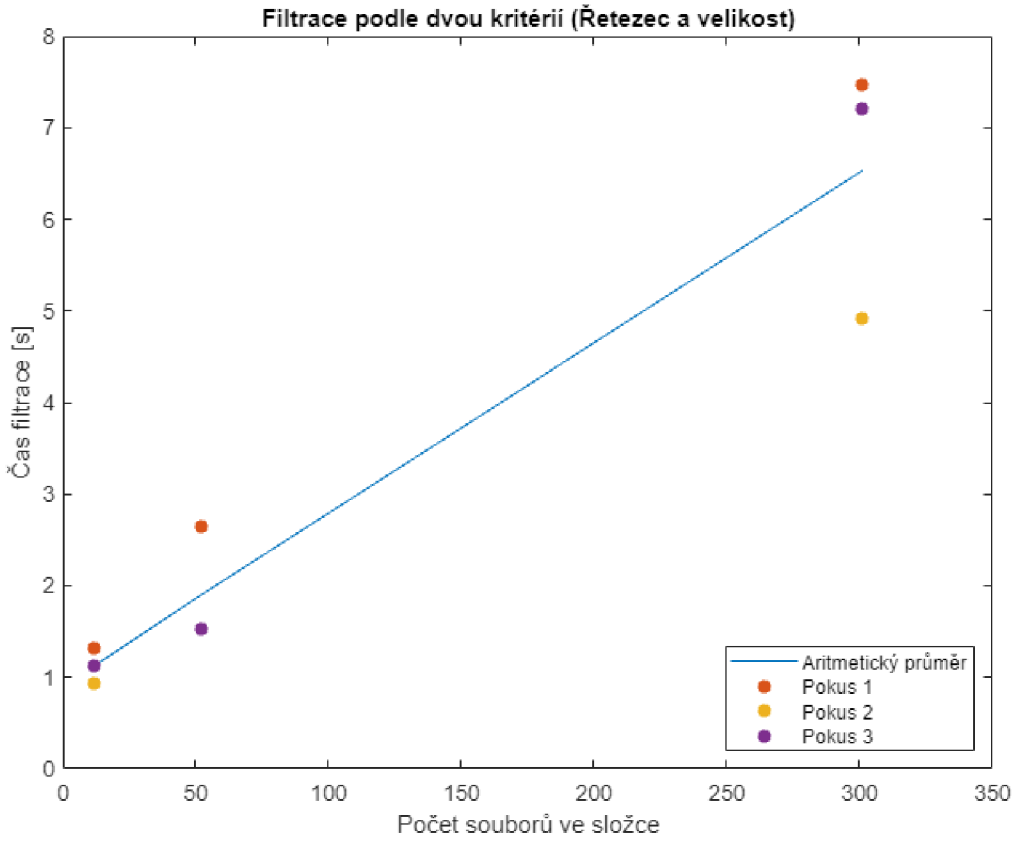
Pro otestování výkonnosti byly vytvořeny tři složky s 12, 52 a 301 soubory. Následně bylo vybráno jedno jednoduché a dvě složená filtrační kritéria – výběr souborů typu *Video* (Test 1), výběr názvu souboru s řetězcem *mp* a naposledy upravené po datu *6-11-2008* (Test 2) a výběr souborů o velikosti mezi *12 B* a *4 KB*[8] (Test 3). Všechna tři kritéria byla aplikována na každou ze tří složek třikrát za sebou.

U prvních dvou výše zmíněných testů (Test 1 a Test 2) se čas počítal tak, že člověk spustil časomíru při stisknutí tlačítka *Submit* ve filtračním formuláři a pozastavil časomíru při načtení seznamu vyfiltrovaných souborů. Tyto testy jsou zaměřené primárně na výkonnostní stránku implementace filtrace. Poslední test (Test 3) probíhal podobně s tím

[8]Pro složku s 301 soubory bylo použito rozmezí 12 KB až 1 MB.

rozdílem, že časomíra se začala odpočítávat s otevřeným oknem s filtračním formulářem těsně před jeho vyplněním. Tento test je zaměřený primárně na změření uživatelského pohodlí.

Všechny tři testy byly zaneseny do grafů a číselné hodnoty do excelové přílohy. Graf Testu 1 je na Obr. 3.3, Testu 2 na Obr. 3.4 a Testu 3 na Obr. 3.5. V grafech je vidět spojnice aritmetických průměrů pro jednotlivé počty souborů a hodnoty, ze kterých byl aritmetický průměr vypočítán.



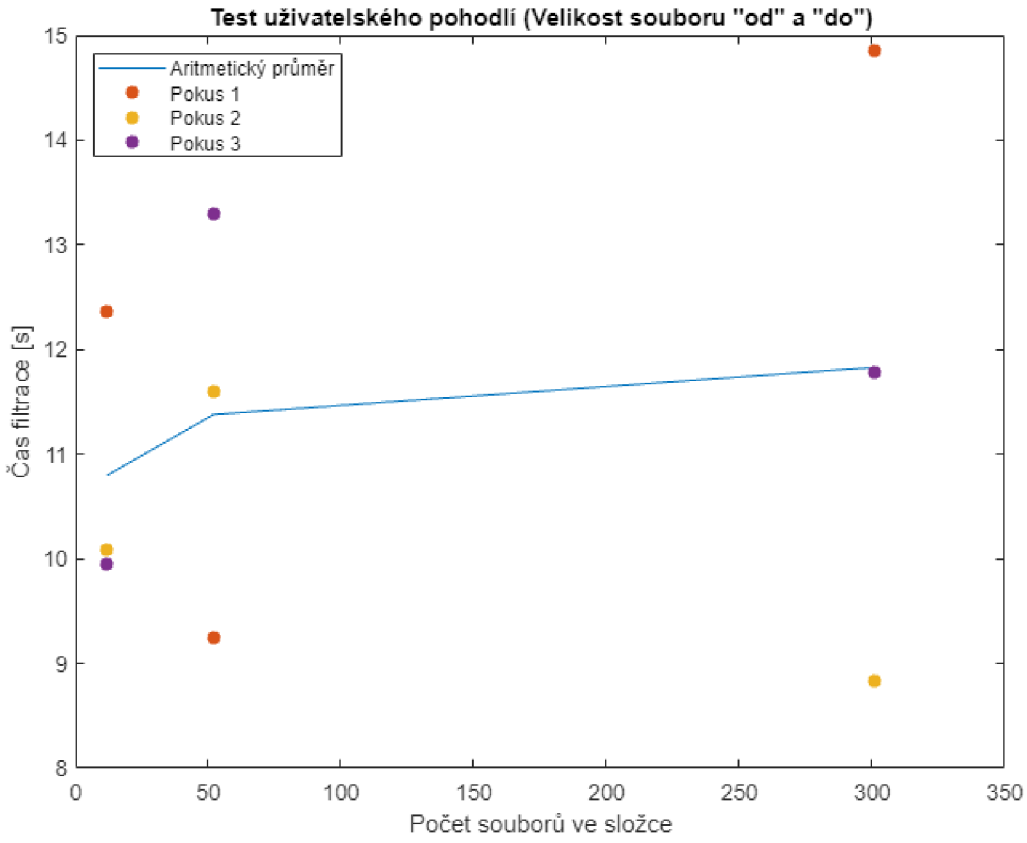
Obr. 3.4: Test se složeným kritériem (Test 2).

Z výsledků vyplývá, že rozdíly hodnot mezi hodnotami naměřenými u jednoduchého a složeného kritéria jsou zanedbatelné. Dále je možné vidět, že se v prvních dvou testech zvyšuje čas filtrace v závislosti na množství filtrovaných souborů. Například u Testu 1 trvala filtrace pro 12 souborů průměrně $0,97\ s$, pro 301 souborů to bylo již $7,34\ s$. Z Testu 3 (Obr. 3.5) vyplývá, že uživatelské pohodlí není při zvyšujícím se počtu souborů narušeno.

Testy byly prováděny na hardwarově omezených strojích, proto je potřeba se dívat na pohyby v datech místo na konkrétní časové údaje. Pro porovnání, spuštění NextCloud aplikace Files (kliknutí na tlačítko s ikonou složky v liště vlevo nahoře) trvalo ze tří hodnot v průměru $5,88\ s$. Klientskému virtuálnímu počítači s distribucí Kali Linux bylo přiřazeno 4096 MB operační paměti a dvě jádra procesoru. Serveru operujícímu nad distribucí Ubuntu 20.04.3 LTS bylo přiřazeno 2048 MB operační paměti a jedno jádro procesoru.

Dalším faktorem může být použití vývojové verze systému NextCloud místo produkční verze.

Uživatelské pohodlí není možné ohodnotit pouze kvantifikační metrikou, je potřeba zohlednit i subjektivní pocity z použití filtrace. Dalším subjektivním faktorem je lidská chyba. U Testu 1 a Testu 2 je nežádoucí odchylka způsobená zapínáním a pozastavováním časomíry člověkem. U Testu 3 taková odchylka není problém, protože cílem bylo otestovat použití filtrace člověkem.



Obr. 3.5: Test se složeným kritériem zaměřeným na uživatelské pohodlí (Test 3).

3.4.3 Testy přesnosti

Tento typ testu se zaměřoval na to, jestli filtrace proběhla podle očekávání a uživateli se zobrazil seznam souborů, který odpovídá zadaným filtračním kritériím.

Pro test byly vytvořeny pro účet *ncadmin* tři složky, které obsahují 57 souborů – 50 náhodně generovaných, `summary.txt`, tři soubory s obrazy disků a tři soubory sdílené účtem *testUser*. Pět souborů z každé ze tří složek bylo sdíleno s uživatelem *testUser*, který nově sdílené soubory přesunul do jedné složky. Tato složka je využita pro pátý test přesnosti. V Tab. 3.6 je vidět, kým byl jaký soubor v jaké složce sdílený. Červeně jsou označeny soubory, které mají příznak *posledního editora* nastavený na opačného uživatele,

než kým byl soubor sdílený. Všechny soubory v tabulce označené jako *Sdílený uživatelem ncadmin* byly použity v pátém testu přesnosti.

Dále byla vybrána čtyři složená filtrační kritéria, která byla aplikována na každou složku. Před zahájením filtrace byly vybrány až tři soubory, které by uživatel očekával, že budou výsledkem filtrace[9] a tři soubory, které by uživatel neočekával jako výsledek filtrace. Poslední pátý test probíhal ve složce u uživatele *testUser* a oproti ostatním testům se otestovaly všechny hodnoty místo tří vybraných. Tento test se soustředí na ověření nově přidané funkčnosti označující uživatele, který jako poslední upravil soubor (*Poslední editor*). Výsledek i očekávané výsledky (jsou stejné) pátého testu jsou vyobrazeny v tabulce 3.5. Tabulka s očekávanými výsledky všech testů přesnosti se nachází v excelové příloze.

Tab. 3.5: Příklad: Očekávané výsledky filtrování pro zadané kritérium.

Vlasntník a poslední editor	OBSAHUJE soubory			NEOBSAHUJE soubory		
Shared by ncadmin	8	31	45	1	19	22
	11	14.mpkg	32	27	14.azw	33
	43	summary	testVdi			
Obsah kritéria	Owner			„ncad“		
	Last editor			„ncad“		

Tab. 3.6: Sdílené soubory ve svých složkách.

Set 1	Sdílený uživatelem ncadmin	1	8.xml	19	31	45
	Sdílený uživatelem testUser	1	2	3		
Set 2	Sdílený uživatelem ncadmin	11	14	22	27	32
	Sdílený uživatelem testUser	4	5	6		
Set 3	Sdílený uživatelem ncadmin	14	33	43	summary	testVdi
	Sdílený uživatelem testUser	7	8	9		

3.4.4 Bezpečnostní a edge case testy

Cílem testu bylo otestovat reakci filtračního formuláře na nestandardní vstupy. Především se jedná o řetězce, které by mohly vést k incidentům, pokud by je NextCloud server špatně zpracoval.

Testy probíhaly u všech kritérií, která obsahují pole pro zadávání řetězců. Jedná se o *Filter by name*, *Size less than*, *Size more than*, *File owner* a *Last editor*. Pole *Size less than* a *Size more than* berou jako vstupy pouze číslice 0-9 a jsou jim vyhrazeny poslední dva řetězce v tabulce Tab. 3.7. Zbylá textová pole byla otestována na zbylých vstupech v Tab. 3.7. Žádný z níže zmíněných testů nevedl k narušení filtrace.

[9] V některých případech výsledkem filtrace bylo méně než tři soubory, proto je v tabulce v excelové příloze méně výsledků.

Formulář byl otestován na útok SQL injection, jehož úspěšné provedení by mohlo zobrazit všechny soubory a složky nacházející se na serveru, včetně systémových souborů a souborů cizích uživatelů. Další potenciální slabinou by mohl být escape character („\“) jako poslední znak řetězce, protože by mohl zneplatnit rozdělující znak, na který navazuje další poddotaz – situace je ukázána na výpisu 3.18, kde je odebrán jeden ze dvou rozdělujících znaků „___“. Protože byl takto jeden znak podtržítka odebrán, byl celý filtrační poddotaz zneplatněn. Ukázka možného poškození filtračního dotazu dosazením escape character na konec řetězce je na výpisu 3.18.

Výpis 3.18: Ukázka možného problému s escape character.

```
1 /search?term=test\__mimetype::text -> /search?term=test_mimetype::text
```

Byly otestovány i řetězce, které jsou využity k logickému rozdělení filtračního dotazu. Jedná se o řetězce „::“ a „___“. Bez žádné úpravy by došlo k rozdělení filtračního dotazu ve špatném místě a k zneplatnění dotazu. Poslední kategorií je test numerických polí, které reagují pouze na numerické vstupy. Všechny otestované řetězce se nachází v Tab. 3.7.

Tab. 3.7: Seznam testovaných řetězců.

Typ vstupu	Řetězec
SQL injection	’ OR 1=1; /*
	’ OR 1=1 –
	"OR 1=1 /*
	"OR 1=1 –
Escape character	\
Rozdělující znaky	___
	::
Kompatibilita	in
Testy numerických polí	test
	132t

3.4.5 Test responzivnosti grafického rozhraní

Ukázka reakce grafického rozhraní na změnu velikosti prohlížeče je vidět na videu `screenSizeTest.ogv` v rámci přílohy A.1.8. Ve zkratce není uživatel omezen změnou velikosti rozlišení zařízení.

3.5 Návod na přidání nových filtračních kritérií

Tato podkapitola se zabývá standardizovaným způsobem přidávání nových kritérií podporovaných NextCloudem. Nutnou podmínkou pro to, aby mohlo být zvolené kritérium standardizovaně přidáno mezi podporovaná kritéria, je, aby splnilo tyto podmínky:

1. Musí být obsažené v databázi.

2. Musí být obsaženo v tabulkách `filecache` nebo `share`.
3. Musí mít unikátní alias, který nekoliduje s žádným kritériem, které je definováno v rámci NextCloudu (včetně kritérií definovaných interně). Seznam již použitých aliasů s jejich datovým typem je vidět od řádku 195 dolů v souboru `SearchBuilder.php`.

Pokud se atribut nachází v jiné tabulce než `filecache` nebo `share`, je potřeba najít relaci mezi tabulkami a doplnit přední část SQL dotazu v souboru `CahceQueryBuilder.php` v metodě `selectFileCache`.

Postup přidávání nových filtračních kritérií je popsán v následujícím seznamu. Alias atributu, název atributu v tabulce a název kritéria v rámci filtračního dotazu můžou být ten samý řetězec. Alias atributu je řetězec definovaný v souboru `SearchBuilder.php` svázaný s SQL atributem v databázi NextCloudu.

1. V souboru `SearchBuilder.php` v metodě `validateComparison` se do pole `$types` napíše alias atributu a jeho datový typ.
2. Ve stejné metodě se o něco níže v poli `$comparisons` sváže alias s podporovanými operacemi.
3. V metodě `getOperatorFieldAndValue` postupujte podle komentáře a přidejte vlastní `elseif`, ve kterém se sváže alias atributu s reálným názvem atributu v databázi. Pro tabulku `oc_filecache` použijte SQL alias `file` a pro tabulku `oc_share` SQL alias `sh`. Příklad svázání aliasu `last_updator` s názvem SQL atributu v tabulce `oc_share` je vidět na výpisu 3.19.
4. V souboru `FilesSearchProvider.php` se v rovině komentáře `//TUTORIAL` vepíše nový `case` pro nové filtrační kritérium. Název filtračního kritéria se musí použít i v dalším bodě. Uvnitř `case` se mohou definovat složitější podmínky pro dané kritérium, ale jednoduchý příklad je vidět na výpisu 3.20. `NÁZEV KRITÉRIA` musí být stejný jako v dalším kroku. `OPERACE` je jedna z konstant definovaná od řádku 40 dolů v souboru `SearchBuilder.php` a `N` je relativní umístění argumentu kritéria ve filtračním dotazu.
5. Do souboru `UnifiedSearch.vue` se do definice objektu `queryObject` okolo řádků 358 a 581 vloží nové filtrační kritérium se svými argumenty.
6. V kontejneru se třídou `unified-search__input-wrapper` se vloží kontejner obsahující komponentu, která bude zobrazena ve filtračním formuláři. Zde se inspirujte již vytvořenými částmi formuláře. Hlavní je výslednou hodnotu z formuláře vložit do objektu `queryObject` pomocí vložení Vue.js atributu `v-model= "queryObject.[ARGUMENT KRITÉRIA]"` do definice komponenty.

Výpis 3.19: Příklad svázání aliasu.

```
1  elseif($filed === 'file_target'){
2    $filed = 'sh.file_target';
3  }
```

Výpis 3.20: Příklad nového `case`.

```
case "[NÁZEV KRITÉRIA]":
array_push($queryArray, new SearchComparison(ISearchComparison::[OPERACE],
'[ALIAS]', '%' . $exploded[N] . '%'));
break;
```

3.6 Možnosti navázaní na práci

Čtyři nedostatky implementace, které neomezují funkčnost filtrace, ale můžou mít vliv na uživatelské pohodlí jsou:

- Vypnutí vestavěné funkčnosti filtrace podle NextCloud aplikace.
- Paginace výsledků
- Neoznačení posledního editora u souboru
- Provedení implementace na nižší verzi NextCloud

Kvůli nedostatečnému otestování funkcionality Unified Search je pro uživatele nedostupná možnost filtrace podle NextCloud aplikace mezi různými NextCloud aplikacemi. Povolená je pouze filtrace v NextCloud aplikaci Files. Při navrhování „protokolu“ předávání filtračních kritérií mezi frontendem a backendem byla zohledněna kompatibilita s ostatními NextCloud aplikacemi využívajícími funkcionalitu Unified Search. Na frontendu je kompatibilita docílena pomocí `emitoru`, který vysílá dotaz na aplikace, které jsou přihlášeny k odběru odpovědi podle dokumentace [37]. Emitor je zobrazen na výpisu 3.21 ze souboru `UnifiedSearch.vue`.

Výpis 3.21: `Emitor` obsluhující ostatní NextCloud aplikace.

```
emit('nextcloud:unified-search.search', { query: this.queryObject.name })
```

Pokud uživatel použije textové pole s popisem *Filter by name*, které bylo původně použito pro funkcionalitu Unified Search, vyplní se první pole filtračního dotazu bez popisu – viz `FILE NAME` ve výpisu 3.2. Tím, že se jedná o čistý Unified Search dotaz bez jakýchkoliv popisků (jak to bylo původně u funkcionality Unfied Search), může být využit dotaz pro filtraci i v jiných aplikacích než je Files a je dosaženo kompatibility i pro backend. Vhodné by bylo vypnout ve filtračním formuláři rozšířené filtrační možnosti, pokud se nachází uživatel v jiné NextCloud aplikaci, než je Files, aby nedošlo ke zmatení.

Dále je potřeba v souboru `UnifiedSearch.vue` odkomentovat kód zobrazující filtrační možnosti pro filtraci podle NextCloud aplikace mezi řádky 241 a 249. Stejně tak pro zobrazení seznamu výsledků v rámci komponenty `unified-search` je nutné odkomentovat kód mezi řádky 278 a 303. V obou dvou případech jsou kusy kódu označeny komentáři. Další problém mohou být funkce zpracovávající tento typ filtrace, protože nebyly předmětem implementace v rámci práce.

Paginace výsledků je v této implementaci využita, ale bylo by vhodné ji upravit. Je doporučeno zmenšit množství vrácených hodnot z 999 na nějaké nižší číslo a vytvořit někde v grafickém rozhraní tlačítko *Načíst více výsledků*, které by spustilo funkci `loadMore`

v souboru `UnifiedSearch.vue`. Číslo počtu vrácených vyhledaných hodnot je možné definovat na řádku 240 v souboru `FilesSearchProvier.php` (na výpisu 3.22) použitím vlastní definice (typu `integer`) nebo využitím návratové hodnoty metody `$query->getLimit()`.

Výpis 3.22: Místo v kódu, kde se definuje počet vrácených vyhledaných hodnot.

```
1    array_key_exists("fileid", $query->getRouteParameters()) ? 999 :
2    $query->getLimit(),
```

Každý sdílený soubor má v databázi v tabulce `oc_share` vyplněný atribut `last_updater`, který označuje posledního editora souboru, ale uživateli to nemusí být zřejmé na první pohled, protože soubory nejsou nijak označeny. Pro zvýšení úrovně uživatelského pohodlí by bylo vhodné doplnit implementaci o indikátor posledního editora podobně jako na Obr. 3.6. K tomu je nutné nejdříve získat informace o tom, jaký soubor byl

Obr. 3.6: Vizualizace doplnění implementace o indikátor posledního editora.

kým naposledy upraven, informace poslat na frontend a na frontendu informace zobrazit. K tomu je nutné upravit následující soubory.

- `additionalScripts.js`: Okolo řádku 299 se přidá podpora zobrazení posledního editora. Styly se převezmou od tříd `action`, `action-share`, `permanent` a `shared-style`. Okolo řádku 445 ve funkci `icon` je vidět, jak vygenerovat profilový obraz posledního editora.
- `Manager.php`: Přidat podporu zasílání jména na frontend. Atribut, který by měl být z SQL databáze získán, se jmenuje `last_updater`.

Seznam výše upřesňuje, kde začít změny dělat. Je pravděpodobné, že seznam souborů a míst v souborech není kompletní.

Vývoj probíhal ke dni 10. 5. 2022 na starší verzi NextCloudu (verzi 22), protože se postupně klientská webová aplikace přesouvá na Vue.js [34]. Kvůli odevzdávání bakalářské práce byla pro jistotu zvolena starší stabilní verze, aby nedošlo k náhlé změně části důležité pro práci. Do budoucna by se měla implementace převést na novější verzi NextCloudu.

Závěr

V práci byla shrnuta teorie okolo filtrace dat. Byly zodpovězeny některé právní a kriminalistické otázky okolo tvorby cloudového úložiště elektronických důkazů. Na základě rozhovoru s odborníky Policie ČR byla vybrána možná filtrační kritéria, která mohou být implementována do cloudového úložiště elektronických důkazů. Výsledkem rozhovoru je, že Policie ČR většinou pracuje se standardními typy souborů, ale mělo by být pomyšleno na implementaci filtrace forenzních kopií – v bakalářské práci nazvané jako *Disk and Computer image*.

Dále bylo popsáno open source cloudové úložiště NextCloud se změřením na zajištění důvěrnosti dat na úložišti. Bylo zjištěno, že NextCloud disponuje dvěma módy šifrování – kombinací šifrování pomocí TLS a šifrováním na serveru a šifrováním end-to-end. První možnost kombinace TLS a šifrování na serveru je užitečná pro méně důležité soubory, protože je uživatelsky přívětivější. Druhá možnost šifrování end-to-end se používá, pokud není důvěřováno NextCloud serveru. Nevýhodou end-to-end šifrování je nemožnost využít některé funkce NextCloudu.

Dále bylo popsáno a zhodnoceno využití homomorfního šifrování na úložišti elektronických důkazů. Výsledkem zhodnocení je, že homomorfní šifrování implementované do úložiště elektronických důkazů nemá využití. Případné využití homomorfního šifrování bylo chápáno tak, že by se pomocí něho upravovaly zašifrované soubory na cloudovém úložišti. Upravovat elektronické důkazy je ale nežádoucí. Homomorfní šifrování bylo porovnáno s atributovým vyhledávacím šifrováním, které by díky možnosti řízení přístupu využití mělo.

V rámci bakalářské práce se protokol WebDAV jevil jako možná cesta přidání podpory nových filtračních kritérií. V jazyce Python byla vytvořena jednoduchá samostatně stojící klientská aplikace, která podporuje tři nová filtrační kritéria. Nevýhodou takového přístupu je nutnost využívat samostatně stojící aplikaci, což by bylo uživatelsky nepřívětivé. Protokol WebDAV ovšem má využití například při synchronizaci lokáního a cloudového úložiště.

Hlavním výstupem práce je implementace filtrace uvnitř systému NextCloud. Byla využita a přepracována již původně implementována funkcionalita Unified Search, která umožňuje developerům externích NextCloud aplikací jednoduše podporovat vyhledávání podle řetězců. Unified Search používá i NextCloud aplikace starající se o zobrazení souborů na úložišti. Bylo upraveno webové rozhraní komponenty Unified Search, které bylo doplněno o pět nových filtračních kritérií. Komunikace mezi frontendem a backendem probíhá na infrastruktuře funkcionality Unified Search, ale do pole HTTP dotazu určeného pro dotaz Unified Search byl vložen tzv. filtrační dotaz. Ten přenáší data o (rozšířené) filtraci z frontendu na backend. Na backendu byla upravena třída, která přijímá dotaz Unfied Search, tak, aby uměla zpracovat příchozí filtrační dotaz a vytvořit z něj SQL dotaz. Výsledky filtrace jsou následně uživateli zobrazeny. Implementace byla řádně otestována a výsledky byly zaneseny do tabulek a grafů. Také bylo popsáno, jak je možné implementovat podporu nových filtračních kritérií.

V poslední části byly uvedeny některé možnosti navázání na práci a jejich řešení. Jedná se o zprovoznění funkcionality Unified Search, která byla deaktivována pro nerelevantní NextCloud aplikace, zlepšení paginace výsledků, přidání označení posledního editora každému sdílenému souboru a převedení výsledku práce na novou verzi NextCloudu. Další možností navázání je přidání filtračních kritérií, která se nachází na policejním sáčku na důkazy. Nutná je úprava podporovaných přípon typu souborů *Disk and Computer image*. Samotné přípony nebyly konzultovány s žádným odborníkem na elektronické důkazy a pro účely práce byla použita co možná nejširší definice obrazu disku a počítače. Zde může být využit návod implementace nových filtračních kritérií. Při tvorbě filtračního formuláře nebyly výrazně upraveny kaskádové styly a nebyl brán ohled na estetický vzhled formuláře. Bylo by vhodné formulář upravit tak, aby vypadal esteticky lépe.

Literatura

[1] MENEZES, Alfred J., Paul C.van OORSCHOT a Scott A. VANSTONE. *Handbook of applied cryptography.* Boca Raton: CRC Press, 1997. ISBN 0-8493-8523-7.

[2] DZURENDA, Petr. HAJNÝ, Jan. *Techniky homomorfního šifrování a jejich praktické využití.* Elektrorevue: Časopis pro elektrotechniku [online]. International Science and Engineering Society, o.s., 2014, 20.04.2014 [cit. 2021-10-5]. ISSN 1213-1539. Dostupné z: http://www.elektrorevue.cz/cz/clanky/informacni-technologie/0/techniky-homomorfniho-sifrovani-a-jejich-prakticke-vyuziti--techniques-of-homomorphic-encryption-and-their-practical-usage-/

[3] HONZÍK, Jan M., HRUŠKA, Tomáš a MÁČEL, Michal. *Vybrané kapitoly z programovacích technik: [Určeno pro kurs Programovací techniky].* Brno: VUT, 1991. ISBN 80-214-0345-4. Dostupné také z: https://ndk.cz/uuid/uuid:6619e0f0-6466-11e2-bc24-005056827e51

[4] DUSSEAULT, ED., L. HTTP Extensions for Web Distributed Authoring and Versioning (WebDAV). *IETF* [online]. červen 2007 [cit. 2021-11-30]. Dostupné z: https://www.ietf.org/rfc/rfc4918.txt

[5] *Termíny cloud computingu* [online]. Microsoft [cit. 2021-12-02]. Dostupné z: https://azure.microsoft.com/cs-cz/overview/cloud-computing-dictionary/

[6] BURDA, Karel. *Aplikovaná kryptografie.* Brno: VUTIUM, 2013, 255 s. : il. ; 23 cm. ISBN 978-80-214-4612-0.

[7] YIN, Hui, Jixin ZHANG, Yinqiao XIONG, Lu OU, Fangmin LI, Shaolin LIAO a Keqin LI. CP-ABSE: A Ciphertext-Policy Attribute-Based Searchable Encryption Scheme. *IEEE Access* [online]. 2019, 7, 5682-5694 [cit. 2022-02-20]. ISSN 2169-3536. Dostupné z: doi:10.1109/ACCESS.2018.2889754

[8] POLČÁK, Radim, František PÚRY a Jakub HARAŠTA. *Elektronické důkazy v trestním řízení.* Brno: Masarykova univerzita, 2015. ISBN 978-80-210-8073-7. Dostupné z: https://is.muni.cz/publication/1338828/Polcak_Elektronicke_dukazy.pdf

[9] PJEŠČAK, Ján. *Kriminalistika: učebnice pro právnické fakulty.* Praha: Naše vojsko, 1982. Dostupné také z: https://dnnt.mzk.cz/uuid/uuid:7733b7d0-d556-11e5-88b1-5ef3fc9ae867

[10] *About.* You should control your data [online]. NextCloud [16. 10. 2021]. Dostupné z: https://nextcloud.com/about/

[11] POORTVLIET, Jos. *Nextcloud Conference News: Nextcloud GmbH doubling HackerOne security bug bounties!.* [online]. NextCloud GmbH, 13. 9. 2019 [16. 10. 2021]. Dostupné z: https://nextcloud.com/blog/nextcloud-conference-news-nextcloud-gmbh-doubling-hackerone-security-bug-bounties/

[12] *Security and authentication.* [online]. NextCloud [16. 10. 2021]. Dostupné z: https://nextcloud.com/secure/

[13] RESCHKE, Lukas. *Nextcloud server.* [online]. CII Best Practices, 14. 6. 2016. Aktualizováno 13. 7. 2020. [17. 10. 2021]. Dostupné z: https://bestpractices.coreinfrastructure.org/en/projects/209#security

[14] *Automate repetitive tasks.* [online]. NextCloud [18. 10. 2021]. Dostupné z: https://nextcloud.com/workflow/

[15] *Encryption and hardening.* [online]. NextCloud [17. 10. 2021]. Dostupné z: https://nextcloud.com/encryption/

[16] POORTVLIET, Jos. *Nextcloud will check passwords against database of HaveIBeenPwned.* [online]. NextCloud GmbH, 26. 2. 2018 [17. 10. 2021]. Dostupné z: https://nextcloud.com/blog/nextcloud-will-check-passwords-against-database-of-haveibeenpwned/

[17] POORTVLIET, Jos. *Nextcloud 16 becomes smarter with Machine Learning for security and productivity.* [online]. NextCloud GmbH, 25. 4. 2019 [17. 10. 2021]. Dostupné z: https://nextcloud.com/blog/nextcloud-16-becomes-smarter-with-machine-learning-for-security-and-productivity/

[18] *Monitoring your Nextcloud server.* [online]. NextCloud [18. 10. 2021]. Dostupné z: https://nextcloud.com/monitoring/

[19] Server-side Encryption: Securing data at rest. *NextCloud* [online]. Stuttgart: NextCloud, 2018 [cit. 2022-02-20]. Dostupné z: https://nextcloud.com/whitepapers/

[20] POORTVLIET, Jos. *Encryption in Nextcloud.* [online]. NextCloud, 5. 2. 2018 [20. 10. 2021]. Dostupné z: https://nextcloud.com/blog/encryption-in-nextcloud/

[21] *End-to-End Encryption Design.* [online]. NextCloud, 20. 9. 2017 [23. 10. 2021]. Dostupné z: https://nextcloud.com/whitepapers/

[22] BABICH, A., J. DAVIS, MITRIX, S. REDDY, GREENBYTES a J. RESCKE. Web Distributed Authoring and Versioning (WebDAV) SEARCH. *IETF Datatracker* [online]. IBM, 2008, listopad 2008 [cit. 2021-11-29]. Dostupné z: https://datatracker.ietf.org/doc/html/rfc5323

[23] HALEVI, Shai a Victor SHOUP, V. *Design and implementation of HElib: a homomorphic encryption library* [online]. 2020 [cit. 2022-02-19], s.1481. Dostupné z: https://ia.cr/2020/1481. Cryptology ePrint Archive, Report 2020/1481.

[24] BRAKERSKI, Zvika, Craig GENTRY a Vinod VAIKUNTANATHAN. *Fully Homomorphic Encryption without Bootstrapping* [online]. 2011 [cit. 2022-02-19]. Dostupné z: https://ia.cr/2011/277. Cryptology ePrint Archive, Report 2011/277.

[25] REGEV, Oded. The Learning with Errors Problem (Invited Survey). In: *2010 IEEE 25th Annual Conference on Computational Complexity* [online]. IEEE, 2010, 2010, s. 191-204 [cit. 2022-02-19]. ISBN 978-1-4244-7214-7. Dostupné z: doi:10.1109/CCC.2010.26.

[26] CHEON, Jung Hee, Andrey KIM, Miran KIM a Yongsoo SONG. *Homomorphic Encryption for Arithmetic of Approximate Numbers* [online]. 2016 [cit. 2022-02-19]. Dostupné z: https://ia.cr/2016/421. Cryptology ePrint Archive, Report 2016/421.

[27] DAWN XIAODING SONG, D. WAGNER a A. PERRIG. Practical techniques for searches on encrypted data. In: *Proceeding 2000 IEEE Symposium on Security and Privacy. S&P 2000* [online]. IEEE Comput. Soc, 2000, s. 44-55 [cit. 2022-02-20]. ISBN 0-7695-0665-8. Dostupné z: doi:10.1109/SECPRI.2000.848445

[28] CHAUDHARI, Payal a Maniklal DAS. *Privacy-preserving Attribute Based Searchable Encryption* [online]. 2015 [cit. 2022-05-17]. Dostupné z: https://ia.cr/2015/899. Cryptology ePrint Archive, Report 2015/899.

[29] *Developer manual: Search* [online]. NextCloud [cit. 2021-11-28]. Dostupné z: https://docs.nextcloud.com/server/latest/developer_manual/client_apis/WebDAV/search.html

[30] *HTTP authentication* [online]. Mozilla and individual contributors, 2021 [cit. 2021-11-29]. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/HTTP/Authentication

[31] FREED, Ned, Alexey MELNIKOV a Murray KUCHERAWY. Media Types. *IANA* [online]. IANA, 7. 12. 2021 [cit. 2021-12-11]. Dostupné z: https://www.iana.org/assignments/media-types/media-types.xhtml

[32] FREED, Ned, J. KLENSIN a T. HANSEN. Media Type Specifications and Registration Procedures. *RFC Editor* [online]. Oracle, AT&T Laboratories, leden 2013 [cit. 2021-12-11]. Dostupné z: https://www.rfc-editor.org/rfc/rfc6838.html

[33] *Basic APIs.* [online]. NextCloud [12. 11. 2021]. Dostupné z: https://docs.nextcloud.com/server/latest/developer_manual/client_apis/WebDAV/basic.html

[34] *Nextcloud Server: Working with front-end code* [online]. NextCloud [cit. 2022-04-03]. Dostupné z: https://github.com/nextcloud/server#working-with-front-end-code-

[35] *Development environment* [online]. NextCloud [cit. 2022-04-17]. Dostupné z: https://docs.nextcloud.com/server/latest/developer_manual/getting_started/devenv.html

[36] *Installation on Linux* [online]. NextCloud [cit. 2022-05-19]. Dostupné z: https://docs.nextcloud.com/server/latest/admin_manual/installation/source_installation.html

[37] *Search* [online]. NextCloud [cit. 2021-12-07]. Dostupné z: https://docs.nextcloud.com/server/latest/developer_manual/digging_deeper/search.html

[38] *Common MIME types* [online]. MDN, 2022 [cit. 2022-04-25]. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/MIME_types/Common_types

[39] *Mimetypes management* [online]. NextCloud [cit. 2021-12-11]. Dostupné z: https://docs.nextcloud.com/server/latest/admin_manual/configuration_mimetypes/index.html

Seznam symbolů a zkratek

API	Application Programming Interface
ABE	Attribute based encryption (atributové šifrování)
ABSE	Attribute based searchable encryption (atributové vyhledávací šifrování)
AES	Advanced Encryption Standard
BGV	Brakerski Gentry Vaikuntanathan (kryptografické schéma)
BVS	Binární vyhledávací strom
CKKS	Cheon, Kim, Kim and Song (kryptografické schéma)
CLI	Command line interface
CP-ABE	Cipher Policy Attribute Based Encryption
DHCP	Dynamic Host Configuration Protocol
el.	Elektronický
FAC	File access control (Řízení přístupu)
FHE	Fully homomorphic encryption
FEK	File encryption key
fil.	Filtrační
GB	Gigabyte
GCM	Galois Counter Mode
HTML	Hypertext Markup Language
HTTP	Hypertext transfer protocol
ID	Identity (identita)
IDE	Integrated development editor
IP	Internet Protocol
JSON	Java Script Object Notation
KB	Kilobyte
KP-ABE	Key Policy Attribute Based Encryption
KŘP JMK	Krajské ředitelství Policie Jihomoravského kraje

LWE	Learning with errors
MB	Megabyte
MIME	Multipurpose Internet Mail Extensions
NAT	Network Address Translation
OS	Operační systém
OČTŘ	Orgán činný v trestním řízení
PHE	Partially homomorphic encryption
pk	Public Key (veřejný klíč)
Sb.	Sbírky zákonů
PrF MU	Právnická fakulta Masarykovy univerzity
RLWE	Ring learning with errors
RSA	Rivest Shamir Adleman
sk	Secret Key (soukromý klíč)
SQL	Structured Query Language
SSH	Secure Shell Protocol
SWHE	Somewhat homomorphic encryption
TB	Terabyte
TLS	Transport layer security
UTF-8	8 Bit Unicode Transformation Format
URL	Universal Resource Locator
VS	Visual Studio
VUT	Vysoké učení technické v Brně
vyhl.	Vyhledávací
WebDAV	Web Distributed Authoring and Versioning
XML	Extensible Markup Language
$O(f(n))$	Značení časové složitosti *Omikron*
$\Phi(n)$	Eulerova funkce

Seznam příloh

A Obsah příloh

Některé přílohy jsou odevzdány pouze elektronicky a některé jsou i vloženy do tištěné verze práce.

A.1 Elektronické přílohy

Zde se nachází seznam elektronických příloh. Elektronické přílohy byly odevzdány do systému VUT jako jeden komprimovaný adresář ve formátu `zip`.

A.1.1 Python klientská aplikace

Klientskou aplikaci si stáhnete pomocí příkazu v souboru `python-client-github.txt`. Pro spuštění aplikace je potřeba mít nainstalovanou aplikaci `Python3.10`. Dále je potřeba ověřit, zda jsou nainstalované balíčky `xml`, `requests`, `datetime`, `sys` a `urllib`. Pro zobrazení dalších možností použití klientské aplikace napište do terminálu příkaz `$ python3.10 clientCustomRequest.py -h`. Výpis příkazu najdete na výpisu 2.1.

A.1.2 Zdrojový kód upraveného systému NextCloud

Odkaz na zdrojový kód upraveného systému NextCloud se nachází v souboru `zdrojovy-kod-nc.txt`. Odkaz vede na větev forknutého GitHub repozitáře se zdrojovým kódem NextCloudu. Tato příloha je určena hlavně na čtení kódu, pokud chcete upravený NextCloud nainstalovat, řiďte se návodem v podkapitole 3.1.1.

A.1.3 Virtuální stroj s NextCloud serverem

Soubor `NC test - new device.ova` nacházející se na odkazu v souboru `stahnuti-VM.txt` obsahuje virtuální stroj s NextCloud serverem fungujícím na OS Linux distribuce Ubuntu 20.04.3 LTS a webovém serveru Apache 2.4.41. Pro stáhnutí virtuálního stroje musíte být přihlášeni pod VUT Google účtem. Data na serveru se nachází ve složce `/var/www/nextcloud` a shodují se s kódem v GitHub repozitáři z přílohy A.1.2. Základ tvořil stroj poskytnutý Bc. Petrem Muzikantem, na nějž byla aplikovaná instalace z podkapitoly 3.1.1 (pouze body, co jsou specifiky vyjmenované v kapitole). Stroj byl importován do virtualizačního prostředí Oracle Virtual Box verze 6.1.28 a do sítě za NAT serverem v rozsahu `10.0.2.0/24` s DHCP serverem. V síti by měl přístroj mít nastavenou adresu `10.0.2.6`, ale raději se o tom přesvědčete pomocí příkazu `ifconfig`.

Na virtuálním stroji se nachází všechny testovací množiny ve svých složkách. Složky a k nim vázající se testy se nachází v tabulce s výsledky testů v příloze A.1.7. Přistoupit k webovému grafickému rozhraní nebo SSH serveru můžete, pokud si vytvoříte klientský Virtual Box stroj, který se nachází v sítí za NAT v rozsahu `10.0.2.0/24`. Přihlašovací jméno do Ubuntu i heslo jsou `nextcloudadmin`. Přihlašovací údaje k administrátorskému účtu NextCloud jsou `ncadmin`, `ncadmin`. Přihlašovací údaje k běžnému

účtu NextCloud jsou `testUser`, `testUser`. NextCloud se nachází na adrese `https://[IP ADRESA]/nextcloud`.

A.1.4 Zdrojový kód generátoru náhodných souborů

Python skript na vytváření náhodných testovacích souborů se nachází ve složce `fileCreator`. Ve složce se nachází skript na tvorbu souborů `fileCreator.py` spustitelný příkazem `python3.10 fileCreator.py`. Dále se zde nachází skript na tvorbu JSON databáze media types `htmlToJson.py` a složka s několika výsledky tvorby souborů a JSON databází.

A.1.5 Uživatelský manuál

Uživatelský manuál uvnitř souboru `uživatelský-návod.pdf` je doporučeno si přečíst a brát jako doplněk bakalářské práce. Nachází se zde návod k otevření filtračního formuláře, popis formátu vstupů filtračního formuláře a řešení možných chyb a nedorozumění.

A.1.6 Instalační manuál

Instalační manuál `README-install.txt` se shoduje s instalačním návodem v podkapitole 3.1.1.

A.1.7 Výsledky testů

Jedná se o excelový soubor `DataTestNextcloud.xls`, který obsahuje tabulky se všemi výsledky testů filtrace. Vlevo se nachází tabulky s výsledky výkonnostních testů a vpravo tabulky s výsledky testů přesnosti. Nachází se zde i další informace o testovacích množinách.

A.1.8 Ukázková videa

Soubor `odkazy-videa.txt` obsahuje tři krátká videa ukazující funkčnost pokročilé filtrace obsažené v systému NextCloud. Pro zobrazení videí musíte být přihlášeni pod VUT Google účtem.

- `filterReplacement.ogv` ukazuje funkčnost nového atributu v databázi `last_updater`.
- `filterTest.ogv` ukazuje test přesnosti ukázaný na Tab. 3.5 a dodatečný test filtrace obrazových a textových souborů.
- `screenSizeTest.ogv` ukazuje test responzivity grafického rozhraní.

A.2 Přílohy vložené v tištěné verzi

Přílohy vložené do tištěné verze bakalářské práce.

A.2.1 Tabulka s výsledky testů

Tabulka vložená do tištěné verze práce se shoduje s tabulkou v elektronické příloze A.1.7.