VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

OCHRANA FIREMNÍCH DAT POMOCÍ ROZŠÍŘENÍ DO WEBOVÉHO PROHLÍŽEČE

WEB BROWSER EXTENSION PROVIDING PROTECTION OF COMPANY DATA

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. JAN DVOŘÁK**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. LIBOR POLČÁK, Ph.D.**
SUPERVISOR

BRNO 2021

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav informačních systémů (UIFS) Akademický rok 2021/2022

Zadání diplomové práce

Student: **Dvořák Jan, Bc.**
Program: Informační technologie a umělá inteligence
Specializace: Počítačové sítě
Název: **Ochrana firemních dat pomocí rozšíření do webového prohlížeče**
Web Browser Extension Providing Protection of Company Data
Kategorie: Web

Zadání:

1. Nastudujte si problematiku ochrany důležitých firemních dat a existující řešení, zaměřte se na služby typu SaaS.
2. Nastudujte si technologie pro tvorbu webových rozšíření a požadavky pro zveřejnění rozšíření do Chrome Web Store. Zaměřte se na možnosti ochrany zobrazených dat (např. omezení možnosti tisku, kopírování apod.).
3. Navrhněte konfigurovatelné rozšíření webového prohlížeče, které umožní definování a aplikování bezpečnostních politik podle požadavků konkrétní firmy.
4. Implementujte navržené řešení.
5. Otestujte implementované řešení na praktických scénářích a demonstrujte tak užitečnost vašeho řešení.
6. Vyhodnoťte práci a dopad na bezpečnost firemních dat. Identifikujte slabá místa řešení.

Literatura:

- ČSN EN ISO/IEC 27001 Informační technologie - Bezpečnostní techniky - Systémy řízení bezpečnosti informací - Požadavky. 2017.
- Hill, David G. Data protection: governance, risk management, and compliance. CRC Press, ISBN 978-1439806920. 2009.
- Niu, Dang-dang, Lei Liu, Xin Zhang, Shuai LÜ a Zhuang LI. Security Analysis Model, System Architecture and Relational Model of Enterprise Cloud Services. International Journal of Automation and Computing, vol. 13, no. 6, pp. 574-584, 2016.

Při obhajobě semestrální části projektu je požadováno:

- Body 1 až 3 zadání.

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

Vedoucí práce: **Polčák Libor, Ing., Ph.D.**
Vedoucí ústavu: Kolář Dušan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2021
Datum odevzdání: 18. května 2022
Datum schválení: 15. března 2022

Abstrakt

Tato práce se věnuje poskytnutí ochrany firemních dat uložených v úložištích typu SaaS. Oproti datům uchovávaným on-premise jsou tyto data vystavena dodatečným rizikům jak zevnitř tak i z vnějšku organizace. Hlavním rizikem, jehož dopady se budeme v této práci snažit zmírňovat, je neúmyslné zveřejnění jedním ze zaměstnanců organizace. Často v případech neúmyslného ale i úmyslného zveřejnění informací figuruje webový prohlížeč, pomocí něhož jsou data vynesena mimo organizaci. K řešení tohoto problému je navrženo a implementováno rozšíření webového prohlížeče Google Chrome. V rámci této práce se také zabýváme komplikacemi při vývoji rozšíření způsobených přechodem na novou verzi manifestu. Práce je zakončena provedením experimentů simulujících praktické scénáře a vyhodnocením vytvořeného řešení a rozhodnutím o vhodnosti využití webového rozšíření pro vytvoření nástroje poskytujícího ochranu firemních dat.

Abstract

This thesis focuses on securing company data stored in SaaS storage. In contrast to data stored on-premise, data stored in the cloud face several additional risks, be it from the outside as well as from the inside of the organization. The main risk, whose impacts we try to mitigate in this work, is an accidental data leak by one of the organization's employees. An extension for the Google Chrome web browser was designed and implemented as a solution to this issue. Implementation complications rising from transitioning to a new version of the manifest are also examined in part of this work. The thesis is concluded with the execution of experiments simulating practical data loss scenarios and a verdict on whether a web extension is an appropriate means for creating a tool providing data loss prevention capabilities.

Klíčová slova

Ochrana proti ztrátě dat, DLP, Rozšíření webového prohlížeče, Rozšíření Google Chrome, Bezpečnost informací

Keywords

Data Loss Prevention, DLP, Web browser extension, Google Chrome extension, Information security

Citace

DVOŘÁK, Jan. *Ochrana firemních dat pomocí rozšíření do webového prohlížeče*. Brno, 2021. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Libor Polčák, Ph.D.

Ochrana firemních dat pomocí rozšíření do webového prohlížeče

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením pana Ing. Libora Polčáka Ph.D. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Jan Dvořák
17. května 2022

Poděkování

Zde bych rád poděkoval vedoucímu práce Ing. Liboru Polčákovi Ph.D. za věcné konzultace a rady při vypracovávání diplomové práce.

Obsah

Kapitola 1

Úvod

Problémem řešeným v rámci této práce je bezpečnost firemních dat uložených v cloudových úložištích typu SaaS, ke kterým je typicky přistupováno pomocí webového prohlížeče. Při práci s webovým prohlížečem může dojít ke ztrátě dat různými způsoby. V rámci této práce se budeme věnovat ochraně proti nebezpečnému nahrávání souborů pomocí Google Chrome do úložišť, které nejsou v organizaci využívány k uchovávání dat, ochranou proti kopírování dat dostupných pomocí Google Chrome a ochranou proti pořizování snímků okna Google Chrome.

Data uložená v cloudových úložištích jsou vystavena stejným rizikům jako data uchovávaná on-premise, ve stejné či větší míře. Kromě známých rizik on-premise úložišť existují také rizika specifická pro cloudové služby a cloudová úložiště. Pro přístup ke cloudovým úložištím a cloudovým službám obecně uživatel nejčastěji využívá webový prohlížeč. S pomocí webového prohlížeče může uživatel vynést data důležitá pro chod či konkurenceschopnost firmy mimo bezpečnostní perimetr organizace. K vynášení může docházet záměrně v případě úmyslné sabotáže či omylem způsobeným např. nepozorností zaměstnance, neznalostí bezpečného zacházení s daty atd.

Cílem této práce je vytvoření návrhu a implementaci rozšíření pro webový prohlížeč Google Chrome, které poskytuje ochranu pro ztrátě firemních dat. Rozšíření dokáže ovlivnit pouze chování uživatele, ke kterému dochází v rámci webového prohlížeče. Existuje pouze malé množství řešení poskytujících ochranu firemních dat formou rozšíření webového prohlížeče, a to nejznámější Microsoft Compliance Extension je obvykle pro malé až středně velké firmy finančně těžko dostupné.

V kapitole 2 jsou popsány stavy, ve kterých se mohou data nacházet, co je to bezpečnost informací a co jsou hlavní výzvy jejího dosažení pro data v jednotlivých stavech, dále je popsán systém řízení bezpečnosti dat podle specifikace normy ISO27001. Kapitola 2 ještě obsahuje popis cloudových služeb a data loss prevention řešení a na závěr kapitoly jsou uvedena existující řešení poskytující ochranu firemních dat formou rozšíření webového prohlížeče. Kapitola 3 obsahuje popis architektury rozšíření pro prohlížeč Google Chrome, dále jsou zde uvedeny informace související s vývojem rozšíření a nakonec také popis zveřejnění rozšíření do Chrome Web Store. Kapitola 4 obsahuje návrh řešení pro implementaci data loss prevention v rámci rozšíření webového prohlížeče. V kapitole 5 jsou uvedeny technologie použité k implementaci řešení, komplikace, které byly objeveny při implementování a nakonec popis samotné implementace jednotlivých částí řešení. Kapitola 6 obsahuje popis experimentů pro ověření funkcionality na praktických scénářích. Na závěr v kapitole 7 jsou sepsány závěry, které vycházejí ze zpracování tématu této práce a poznatků získaných při implementování navrženého řešení.

Kapitola 2

Ochrana firemních dat

Ochrana firemních dat je jedním z nejzávažnějších problémů každé firmy a jeho řešení by mělo být mezi hlavními prioritami. Cílem ochrany firemních dat je udržet informace, bez kterých společnost nemůže efektivně, ne-li vůbec, fungovat v bezpečí. Elektronicky uchovávané informace jsou dnes jedním z hlavních důvodů konkurenceschopnosti na trhu. Ztráta těchto důvěrných informací (např. údaje o zákaznících) nebo zveřejnění tajných informací (např. krádež zdrojových souborů, výkresů, know-how atd.) by mohla mít kritický dopad na fungování a konkurenceschopnost dané firmy [13]. Kromě toho, že ztráta dat může mít dopad na samotnou činnost firmy, musí společnosti často splňovat různé regulace a dodržovat zákony o ochraně soukromých osobních údajů, příklady těchto regulací a zákonů jsou GDPR (General Data Protection Regulation), HIPAA (Health Insurance Portability and Accountibility Act), GLBA (Gramm-Leach-Bliley Act) atd.

Sekce 2.1 obsahuje informace ohledně toho, v jakých stavech se data ve firmě vyskytují a jaké nebezpečí jim hrozí. V sekci 2.2 je definováno, co je to bezpečnost informací a jak jí dosáhnout. Sekce 2.3 obsahuje informace o tom, co je to systém řízení bezpečnosti informací. V sekci 2.4 je vysvětleno, co je to cloud computing a jaké výhody a nevýhody přináší jeho využití. Dále v sekci 2.5 je popsána problematika data loss prevention a nakonec v sekci 2.6 jsou popsána existující řešení, která se zabývají DLP v rámci webového prohlížeče pomocí rozšíření.

2.1 Dělení dat

Firemní data lze obecně rozdělit do třech kategorií, podle toho jakým způsobem jsou využívány a kde jsou uchovávány. Obdobně by se dalo říci, že se data nachází v jednom z následujících třech stavů.

data at rest
: Označují data, která jsou uložena v distribuovaných souborových systémech, velkých centralizovaných datových skladech, databázích atd.

data in use
: Někdy také označovány jako *data at the endpoint*, jsou data která zaměstnanci běžně využívají k práci a nejčastěji se nachází na koncových zařízeních jako jsou počítače, notebooky, mobilní zařízení, USB zařízení a externí disky atd.

data in motion
: Označují data, která jsou odesílána mimo lokální síť pomocí emailu, „instant messa-

ging“ aplikací (Microsoft Teams, Zoom, Messenger, WhatsApp atd.), FTP (File Transfer Protocol) nebo jinými komunikačními mechanismy.

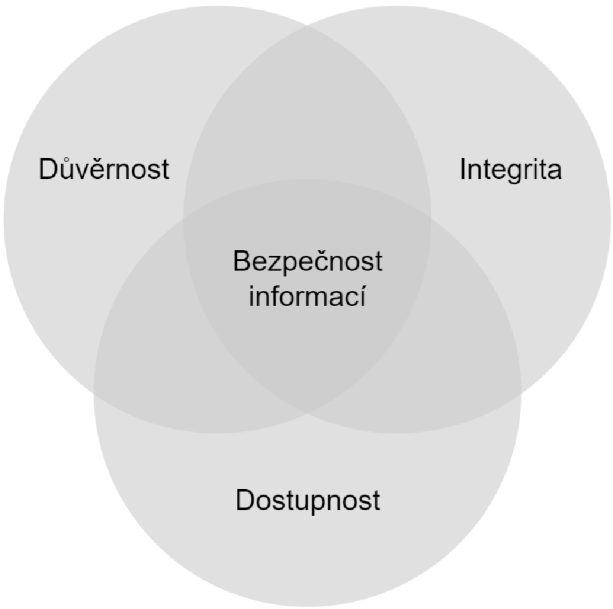
Data v každém stavu vyžadují jiné techniky pro jejich efektivní ochranu, např. monitorování síťového provozu může být efektivní pro data in motion, ale v případě data at rest, která jsou uložena stále na stejném místě, nám monitorování síťové aktivity nijak zvlášť nepomůže [16]. Při navrhování systému pro ochranu firemních dat je tedy nezbytné brát v úvahu to, v jakém stavu se data nacházejí.

2.2 Bezpečnost informací

V této sekci se budeme zabývat bezpečností informací. Pojmy *data* a *informace* jsou v rámci této kapitoly využívány jako synonyma, nicméně v kontextu informační vědy každý termín nabývá jiného významu. V roce 2007 vyšel článek [25], jehož účelem bylo prozkoumat základní koncepty informační vědy, jimiž jsou právě zmíněná data, informace a dále také *znalosti*. Článek obsahuje definice těchto tří termínů formulované odborníky z oboru informačních věd.

Data jsou všechny zaznamenané nezpracované údaje o pozorovaném jevu a stávají se informacemi až ve chvíli, kdy jsou zasazeny do kontextu. Termín znalost je poněkud obtížnější na uchopení ve smyslu, že znalosti můžeme dále dělit na různé typy a nelze je jednoduše definovat. Také v [25] se definice více či méně rozcházejí. Termín znalost není pro tuto práci stěžejní, proto není potřebné ho zde explicitně uvádět.

Informace/data pro firmu představují aktivum (zdroj), které je pro firmu stejně nebo i více důležité jak ostatní hmotná či nehmotná aktiva a je tedy potřeba ho adekvátně chránit. Bezpečnosti informací je dosaženo, pokud se nám podaří zajistit následující tři vlastnosti: *důvěrnost*, *dostupnost* a *integritu* informací[6]. Na obrázku 2.1 je ilustrován vztah mezi bezpečností informací a důvěrností, integritou a dostupností informací. Těmto vlastnostem se často říká „CIA triad“ a jsou dále popsány v podsekcích 2.2.1, 2.2.2 a 2.2.3.



Obrázek 2.1: Znázornění bezpečnosti informací

2.2.1 Důvěrnost

V [6] je důvěrnost definována jako vlastnost, že informace není dostupná nebo není zpřístupněna neoprávněným jednotlivcům, entitám nebo procesům. Jinými slovy dochází k porušení pokud se k informacím dostane člověk, který nemá pro přístup k těmto datům oprávnění. Mezi běžné prostředky vynucení důvěrnosti patří šifrování, ACL (Access Control Lists) nebo unixová přístupová práva souborů.

Důvěrnost dat je potřeba vynucovat zejména u dat, která obsahují citlivé informace jako jsou např. zdravotní záznamy, rodná čísla, čísla bankovních účtů, hesla a další. Pokud se činnost organizace mimo jiné skládá ze sbírání a zveřejňování dat, ať už jsou to zdravotní informace nebo výsledky různých průzkumů, musí daná organizace zajistit, že není možné spojit výsledky či odpovědi z konkrétní osobou. Statistické metody zajištění důvěrnosti se používají ke zveřejnění takovýchto dat. Tyto metody jsou založeny na vynechání konkrétních informací, pomocí kterých by bylo možné identifikovat konkrétní osobu.

2.2.2 Integrita

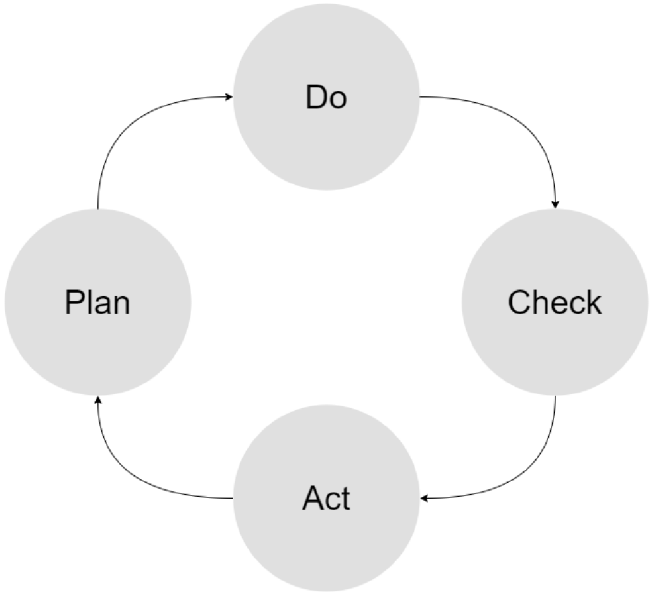
Integritou dat je rozuměno zajištění přesnosti a úplnosti [6]. V článku [21] je uvedeno 5 různých definic integrity dat. Pro případ bezpečnosti informací je nejlépe aplikovatelná definice založená na možnosti modifikace dat, která zní následovně: „Integrita je ovlivněna nevhodnou modifikací informací. Modifikace informací může znamenat vkládání nových informací a mazání nebo změnu existujících informací.“

2.2.3 Dostupnost

Podle [6] je dostupnost definována jako vlastnost vyjadřující přístupnost a použitelnost na žádost oprávněné entity. V aktuální době jsou DoS (Denial of Service) útoky běžnou hrozbou. DoS útoky mají za cíl znemožnit přístup k informacím a porušit tím bezpečnost informací [23].

2.3 Systém řízení bezpečnosti informací

Systém řízení je normou ISO27000 [6] definován jako soubor vzájemně souvisejících nebo vzájemně na sebe působících prvků organizace k ustavení politik, cílů a procesů k dosažení těchto cílů. Systém řízení bezpečnosti informací (angl. Information Security Management System, ISMS) popsán v normě ISO27001 [7] je tvořen politikami, postupy, směrnicemi a příslušnými zdroji a činnostmi, které řídí organizace, za účelem zajištění ochrany informačních aktiv. ISMS představuje systematický přístup k ustavení, implementování, provozování, monitorování, přezkoumávání, udržování a zlepšování bezpečnosti informací organizace tak, aby byly dosaženy její cíle. Tento systém je obvykle aplikovatelný pro všechny typy organizací, ať už soukromé nebo veřejné. ISMS založený na normě ISO27001 představuje cyklický model známý jako „Plan-Do-Check-Act“ (PDCA, viz obrázek 2.2), který zavádí opakované plánování, implementaci, monitorování a vyhodnocování aktuálního systému. Na základě vyhodnocení aktuálního stavu jsou naplánovány vylepšení, pomocí kterých je dosaženo lepší bezpečnosti informací.



Obrázek 2.2: PDCA model ISO27001 systému řízeni bezpečnosti informací

Následuje popis jednotlivých fází PDCA cyklu, ve kterých je uvedeno, na co se musí daná organizace v jednotlivých fázích zaměřit, aby její systém řízení bezpečnosti informací byl v souladu s normou ISO27001[1].

2.3.1 Plánování

Ve fázi plánování ISMS musí organizace určit příležitosti a rizika s ohledem na interní a externí aspekty a na požadavky zainteresovaných stran, které jsou relevantní k bezpečnosti informací. Jako aspekt lze brát cokoli, co nějakým způsobem ovlivňuje výkon organizace, např. koncoví zákazníci, dodavatelé, prostory, internetové připojení, normy a legislativa nebo přírodní katastrofy [17]. Určení rizik a příležitostí je nezbytné pro zajištění, že ISMS může dosáhnout zamýšleného výstupu a pro minimalizování výskytu nežádoucích vedlejších následků. Organizace musí plánovat opatření zaměřená na tato rizika a příležitosti a přijít na způsob, kterým je bude integrovat a implementovat do procesů ISMS a následně vyhodnocovat jejich účinnost.

Jak bylo zmíněno výše, organizace musí určit rizika bezpečnosti informací. S tím se pojí také povinnost definovat a aplikovat proces posuzování těchto rizik. Tento proces stanoví a udržuje kritéria pro posuzování a akceptaci rizik bezpečnosti informací. Dále zajišťuje, že opakovaná posouzení rizik produkují konzistentní, opodstatněné a porovnatelné výsledky. Součástí posuzovacího procesu musí být také identifikace rizik spojených se ztrátou důvěrnosti, integrity nebo dostupnosti, analýza potenciálních následků, pravděpodobnost výskytu a úroveň rizik a vyhodnocení priority rizik bezpečnosti informací.

Kromě procesu pro posuzování rizik musí být definovány další procesy, mimo jiné pro ošetření rizik, pro výběr vhodných variant ošetření rizik, určení všech nezbytných opatření pro ošetření rizik a formulaci plánu ošetření rizik bezpečnosti informací. Dále je potřeba definovat cíle bezpečnosti informací relevantní jednotlivým funkcím a úrovním řízení. Cíle

[1] Tento popis je zjednodušený a není určený k tomu, aby sloužil jako návod pro návrh ISMS, pro podrobný popis je doporučené řídit se samotnou normou ISO27001.

musí být konzistentní s politikou bezpečnosti informací, musí být měřitelné, komunikovány a dle potřeby aktualizovány. Při plánování dosažení cílů bezpečnosti informací musí být předem určeno, co bude vykonáno, jaké zdroje budou k provedení potřeba, kdo bude za provedení odpovědný, kdy to bude dokončeno a jak budou výsledky vyhodnoceny [7].

2.3.2 Provozování

Provozováním je myšlena fáze „Do“ v PDCA cyklu. V této fázi musí organizace plánovat, implementovat a řídit procesy potřebné ke splnění požadavků bezpečnosti informací, implementovat opatření a plány k dosažení cílů definované v předchozí fázi. Tato fáze také obsahuje posuzování rizik v pravidelných intervalech (také když je to nezbytné) a implementaci plánu ošetření rizik bezpečnosti informací. Je nutné dokumentovat provádění všech procesů v nezbytném rozsahu, aby bylo zpětně ověřitelné, že byly prováděny korektně a tak, jak byly plánovány [7].

2.3.3 Hodnocení výkonnosti

Organizace musí pravidelně vyhodnocovat výkonnost bezpečnosti informací a efektivnost ISMS. Za tímto účelem musí být dáno, co je potřeba monitorovat a měřit. K tomu je potřeba určit použitelné metody monitorování, měření, analýzy a hodnocení k zajištění platných výsledků. Metody jsou považovány za platné pokud dávají reprodukovatelné a porovnatelné výsledky. Dále by mělo být dáno, kdo a kdy bude provádět monitorování a měření a kdo a kdy je bude analyzovat a vyhodnocovat.

Firma by také měla v plánovaných intervalech provádět interní audity k získání informací o tom, zda systém řízení bezpečnosti informací vyhovuje požadavkům organizace a požadavkům normy ISO27001 a zda je efektivně implementován a udržován. Audit je v rámci této normy definován jako systematický, nezávislý a dokumentovaný proces k získání ověřitelných informací relevantních pro kritéria auditu a jejich objektivního hodnocení, aby se určil rozsah v jakém jsou kritéria auditu splněna. Před každým auditem by měla být definována kritéria a rozsah auditu. Interní audit je audit prováděný samotnou organizací nebo externí stranou, které ho provádí ve jménu organizace. Na základě výsledků auditů a monitorování by mělo vedení organizace nebo zodpovědné osoby vyhodnotit, zda systém řízení bezpečnosti dat funguje podle požadavků, a v případě, že nejsou některé požadavky splněny, určit oblasti pro zlepšení [7].

2.3.4 Zlepšování

Poslední fází cyklu zavádění ISMS je udržování a zlepšování. V případě výskytu nesplnění požadavku (neshody) bezpečnosti informací, musí organizace na tuto událost reagovat a pokud je to možné, přijmout opatření k řízení a nápravě těchto nedostatků. Při výskytu neshody je dále potřeba identifikovat příčiny neshody a určit, zda existují nebo by se mohly potenciálně vyskytnout podobné neshody. Poté je potřeba implementovat potřebná opatření k nápravě neshody a preventivní opatření pro odstranění potenciálních neshod. Organizace by měla uchovávat dokumentované informace o nalezených neshodách a přijatých opatřeních a jejich výsledcích. Tento cyklus musí být neustále opakován, aby docházelo k celkovému zlepšování systému řízení bezpečnosti informací [7].

2.4 Cloud computing

Cloud computing je model pro poskytování on-demand přístupu přes internet ke sdíleným konfigurovatelným výpočetním zdrojům (např. síťový prvkům, serverům, úložištím, aplikacím nebo službám). Tyto zdroje mohou být ihned po zažádání k dispozici nebo opuštěny a odevzdány zpět poskytovateli s minimálním úsilím nebo interakcí s poskytovatelem těchto služeb [9].

Model cloudových služeb může být rozdělen na tři úrovně, podle míry abstrakce nad poskytovanými zdroji. Jedná se o služby typu Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS) a Infrastructure-as-a-Service (IaaS) [3].

Software-as-a-Service (SaaS)
: Poskytovaná služba je software, ke kterému lze přistoupit pomocí internetového prohlížeče.

Platform-as-a-Service (Paas)
: Poskytuje prostředí pro vývoj nových aplikací s vzdáleně nasazenými a konfigurovanými technologiemi a API.

Infrastructure-as-a-Service (IaaS)
: Poskytuje výpočetní zdroje (procesory, úložiště, síťové prvky) jako službu přes internet.

Cloudové služby mohou být nasazeny v různých režimech. Tyto režimy se odvíjejí od toho, odkud k nim lze přistupovat. Jedná se o public cloud, private cloud a hybrid cloud [3].

Public cloud
: Cloudové služby nasazené v tomto režimu jsou dostupné veřejně všem uživatelům na internetu. Zdroje jsou poskytované na bázi pay-as-you-use.

Private cloud
: Cloudové služby jsou nasazené v tomto režimu konkrétní organizací pro vlastní účely. Přístup k nim má pouze omezená skupina uživatelů.

Hybrid cloud
: Kombinace předchozích dvou režimů. Tohoto režimu je využíváno pokud, implementace private cloud vyžaduje služby nebo zdroje dostupné v public cloud.

2.4.1 Výhody a nevýhody cloudových služeb

V následujícím seznamu jsou uvedeny výhody cloudových řešení oproti *on-premise* („v prostorách organizace“) přístupu [2].

- Cloud computing je pravděpodobně nejefektivnější metoda, co se týče výše nákladů používání, udržování a upgradování. Hardware pro budování infrastruktury vyžaduje vysoké pořizovací náklady, personál na udržování a aktualizace, rychle zastarává a náklady na upgradování jsou opět vysoké. V případě cloud computing odpadávají náklady na pořizování, údržbu i upgrade, což výrazně snižuje celkové náklady. Cloudové služby poskytují spoustu škálovatelných platebních plánů.

- Navýšení kapacity cloudového úložiště je snazší a levnější než v případě on-premise úložiště

- Protože většina poskytovatelů nabízí obnovení informací jako součást služby, je zálohování a obnova dat mnohem jednodušší než ukládání záloh na fyzických zařízeních.
- Informace uložené v cloudu nebo aplikace běžící v cloudu jsou dostupné téměř odkudkoliv, kde je přístup k internetu.
- Snadné škálování pro velké korporace a rychle rostoucí firmy.

Následuje seznam nevýhod cloudových řešení [2].

- Technické problémy mohou zapříčinit výpadek u poskytovatele cloudových služeb. To má za důsledek ztrátu přístupu k datům či aplikacím a ztrátu zisků jako pro poskytovatele tak pro jeho zákazníky.
- Bezpečnost je společně s výpadky největším problémem. Odběratel cloudových služeb odevzdává svá důvěrná data třetí straně. Toto představuje velký risk, co se jejich bezpečnosti týče, viz podsekce 2.5.5. V případě některých malých firem, naopak může docházet ke zlepšení zabezpečení, protože malé a středně velké firmy nemusí mít dostatek zdrojů (peněz či expertů) pro vybudování dostatečné bezpečnostní infrastruktury.
- Velké množství uchovávaných dat činí z poskytovatelů cloudových služeb lákavé cíle externích útoků a tím roste riziko plynoucí z jednotného zabezpečení.
- Firma ztrácí flexibilitu, co se týče využívaných aplikací a technologií. Když se organizace rozhodne pro konkrétního poskytovatele, je často nucena využívat jeho proprietární aplikace a formáty.
- Uživatel SaaS služeb nemá k dispozici ani samotný spustitelný soubor, který provádí danou činnost a nemůže tedy určit, co ve skutečnosti daná služba vykonává za operace a ani její činnost nijak ovlivnit [22].

2.5 Data Loss Prevention

Data Loss Prevention (DLP), někdy také Data Leak Prevention, systém je řešení navržené pro včasné detekování potenciálních a případné zabránění incidentům, které by mohly mít za následek ztrátu dat. Jednotlivé incidenty jsou detekovány pomocí monitorování dat, ať už to jsou *data in motion*, *data in use* nebo *data at rest*, viz sekce 2.1. DLP pomáhá zabraňovat úniku dat mimo bezpečnostní perimetr organizace [24].

V rámci návrhu systému řízení bezpečnosti informací (ISMS, viz sekce 2.3) organizace definuje procesy popsané v [7]. DLP řešení slouží jako nástroj k implementaci a automatizaci některých částí navrženého ISMS. Nelze však tvrdit, že pomocí DLP řešení je ISMS kompletně implementován. Ke správnému fungování DLP řešení je potřeba také lidský zásah, který definuje, jak se má konkrétní řešení chovat a provádí vyhodnocování a zlepšování nasazeného DLP řešení [16].

Efektivní DLP řešení musí fungovat, aniž by docházelo ke znatelnému ovlivnění výkonu zařízení, na kterém běží a mělo by disponovat následujícími čtyřmi vlastnostmi.

Správa

Umožňuje definovat politiky pro zacházení s daty, hlásit incidenty a nastavit korektivní akce, které mají být provedeny v případě detekování incidentu.

Objevování
Umožňuje definovat citlivost dat, lokalizovat citlivá data a uchovávat jejich pozice. Včetně dat ve stavu „at rest“ a „in use“.

Monitorování
Umožňuje monitorovat používání citlivých dat, mapovat způsob jakým je s daty zacházeno a získat přehled o datech v organizaci. Toto může zahrnovat inspekci síťové komunikace nebo monitorování souborů na koncových stanicích.

Ochrana
Umožňuje vynucovat politiky za účelem proaktivního zabezpečení dat a zabránit tomu, aby opustili vnitřní prostor organizace. Umožňuje omezit možnosti tisku, ukládání, kopírování, přístupu, přesouvání, stahování a nahrávání citlivých firemních dat.

2.5.1 Nástroje s obdobnou funkcionalitou

Kromě DLP řešení existují jiné ochranné technologie, jejichž funkcionalita se více či méně překrývá s DLP systémy. Primární účel těchto řešení je ale ochrana dat před hrozbami zvenčí. Jako příklady těchto nástrojů jsou zde uvedeny host-based firewall a IDS/IPS systémy, které jsou obecně popsány v rámci této podsekce.

Host-based firewall

Firewally jsou systémy sloužící jako bezpečnostní bariéra mezi lokální sítí a veřejným internetem. Účelem firewallů je chránit vnitřní síť před síťovými hrozbami a poskytovat jedno úzké místo, kde lze vynucovat bezpečnostní politiky a auditování [19]. Host-based firewall je nainstalován na koncovém zařízení a monitoruje příchozí a odchozí síťovou komunikaci z daného zařízení. Síťový provoz může být blokován, pokud dojde k porušení nastavených pravidel [10]. V tomto se host-based firewall principem podobá DLP systému, které je obvykle nainstalované také na koncových zařízeních, ale kromě monitorování síťové komunikace se zabývá i dalšími kanály, kterými lze vynést data.

IDS/IPS systémy

Intrusion Detection System (IDS) je aplikace využívaná k monitorování síťového provozu za účelem detekce anomálií a potenciálních DoS (Denial of Service) útoků. DoS útoky cílí na omezení dostupnosti informací zahlcením systému do takové míry, že běžný uživatel nemůže k těmto informacím přistoupit. Tím je porušena bezpečnost informací (viz sekce 2.2), protože se nelze k daným informacím dostat. Jakmile IDS detekuje nějakou anomálii, odešle upozornění administrátorovi. Intrusion Prevention System (IPS) je rozšířením nad IDS, kde IPS také monitoruje síťovou aktivitu, ale v momentě, kdy objeví nějakou abnormalitu v provozu, začne sbírat a logovat informace o útoku, reportuje administrátorovi, že dochází k útoku a snaží se zablokovat útočníkův provoz. IPS také pozoruje události během útoku a generuje zprávy o aktivitě. Na základě těchto událostí se snaží identifikovat typ útoku a nebezpečné pakety a ty následně blokovat [14].

2.5.2 Vektory ztráty dat

Ke ztrátě dat obvykle dochází po prolomení nastavených bezpečnostních opatření. Toto prolomení nemusí přicházet pouze zvenčí organizace. Podle reportu RiskBased Security

za rok 2020 se v 23 % případů datových úniků jednalo o úniky dat způsobených osobou, která se nacházela uvnitř firmy [1]. Podle toho odkud byl útok iniciován, rozlišujeme interní a externí datové úniky. Interní úniky mohou být způsobeny neúmyslně některým ze zaměstnanců nebo úmyslně jako součást sabotáže zaměstnancem či jinou osobou, více informací je uvedeno níže [5].

Neúmyslné datové úniky

Neúmyslné datové úniky jsou způsobeny neúmyslnými zveřejněními informací zaměstnancem nebo firemním partnerem, např. přiložením citlivých informací v emailu, zveřejněním na webu atd. Nejčastějšími příčinami neúmyslných incidentů jsou nedostatek motivace učit se a využívat bezpečnostní praktiky, nedostatek povědomí o bezpečnostních metodách (jak vypadá silné heslo, známky phishingu a sociálního inženýrství atd.), nesprávné návyky (zaměstnanci věří, že se chovají v souladu s bezpečnostní politikou), riskantní chování zaměstnanců, neadekvátní využití technologie (sebelepší technologie neuspěje v řešení bezpečnostních problému bez kontinuální lidské kooperace a efektivního využití dané technologie) [18]. Podle [1] je až v 77 % případů interních úniků dat na vinně chyba zaměstnance.

Interní úniky

Interní úniky jsou zapříčiněny buď osobou, která se do prostor organizace dostala neoprávněným způsobem nebo některým zaměstnancem nejčastěji z důvodu nespokojenosti se zaměstnavatelem či peněžním ohodnocením, v některých případech také z důvodu peněžního zisku od třetí strany, která má o daná data zájem [5].

Externí úniky

Externí datové úniky jsou obvykle zapříčiněny osobou zvenčí, která se dostane do systému nepovoleným způsobem, např. pomocí malware, viru, sociálního inženýrství nebo využitím chyby v konfiguraci bezpečnostních politik. V rámci této práce se budeme věnovat převážně interním hrozbám a proto jsou externí datové úniky popsány pouze okrajově a uvedeny pro úplnost.

2.5.3 Techniky detekce dat

Hlavním úkolem DLP systémů je identifikovat, monitorovat a ochránit důvěrné informace před ztrátou důvěrnosti. Identifikace je obvykle prováděna na základě obsahu informací nebo obklopujícího kontextu monitorovaných dat. Techniky detekce jsou tedy rozděleny na dva přístupy: *content-based* přístup, který je založen na inspekci obsahu souborů a *context-based* přístup, který je založen na analýze metadat a kontextu, ve kterém je k datům přistupováno [5]. Oba přístupy jsou popsány v rámci této podsekce.

Content-based přístup

Jak již bylo zmíněno výše, content-based přístup je založen na analýze obsahu konkrétních souborů. Tento přístup lze využít pro data ve všech stavech, nicméně tento přístup je efektivní pouze proti neúmyslnému zveřejnění či odeslání souborů neoprávněným osobám. V případě záměrné exfiltrace dat lze tento způsob detekce obejít pomocí zamlžení (změna formátu, nahrazení některých znaků atp.) nebo šifrování dat.

Content-based techniky hledají citlivá data s předem známým formátem, pomocí metod jako jsou fingerprinting, lexikální analýza obsahu souborů (např. regulární výrazy) nebo statistické metody. V případě fingerprintingu jsou potenciální úniky dat odhaleny porovnáváním monitorovaných dat se signaturami známých souborů s důvěrnými daty. Signatura může být hash hodnota souboru. Lexikální analýza je využívána k nalezení citlivých informací, které mají pevně daný formát, jako jsou např. rodná čísla, čísla bankovních účtů, termíny ze zdravotních záznamů atd.

Jak již bylo zmíněno výše, analýza obsahu souborů není příliš efektivní v případech, kdy se nejedná o neúmyslný únik dat. Dále může docházet ke zpomalení systému, pokud bychom museli soubor skenovat při každé operaci, obzvlášť pokud by se jednalo o velké soubory [5].

Context-based přístup

Context-based přístup k detekci potenciálních incidentů je založen na modelování chování zaměstnanců při práci s daty. Místo detekování důvěrných informací uvnitř souborů se tento přístup snaží odhalit abnormality v chování zaměstnanců vůči vytvořeným modelům.

Mnoho context-based přístupů je založeno na dolování informací z dat nebo strojovém učení. Výhodou metod založených na strojovém učení je fakt, že není potřeba poskytovat přesné charakteristiky anomálního chování. Největší překážkou při využívání těchto metod pro detekci anomálií je nedostatek trénovacích dat.

Dalším využívaným context-based přístupem je watermarking. Watermarking je využíván k detekci a prevenci ztráty dat, pomocí značkování relevantních dat při vstupu a odchodu ze sítě. Přítomnost značky v odchozím souboru indikuje potenciální únik dat. Watermarking může být také využívám k forenzní analýze v případě incidentu, např. k identifikaci osoby zodpovědné za únik [5].

2.5.4 Audit

Auditování je proces sledování a zaznamenávání (logování) zajímaých událostí, ke kterým došlo za běhu systému. Kolekce informací může sloužit k ověření správné funkcionality a efektivity zavedených bezpečnostních politik a také v případě vyšetřování incidentů pro účely vyšetřování. K implementaci auditu je využíváno logování odolné vůči manipulaci pro případ, že by někdo chtěl smazat nebo upravit záznamy o nějaké události v pozorovaném systému [11]. Auditování se skládá ze dvou částí:

1. ze sbírání a uspořádání dat a
2. z analýzy sesbíraných dat.

Při kolekci dat často dochází k nahromadění velkého množství dat, toto je potřeba brát v potaz při návrhu a zavádění auditního systému. Nasbíraná data se týkají konkrétních operací provedených konkrétními uživateli na konkrétních datech, jedná se tedy o data s velmi nízkou úrovní abstrakce. K analýze obvykle dochází, pouze pokud existuje podezření, že došlo k porušení bezpečnostních nařízení. V takovém případě se analyzují pouze data, která souvisí s konkrétním incidentem [20].

2.5.5 DLP a cloudová úložiště

Cloudová úložiště jako jsou SharePoint, Box, Google Drive atd. jsou stále široce využívané služby napříč organizacemi. Zmíněné služby lze používat také pomocí webového prohlížeče,

tím poskytují snadný přístup ke sdílení souborů a spolupráci, snižují náklady na fyzické datové úložiště a udržují vysokou spolehlivost dat. Nicméně citlivé informace mohou být odcizeny nebo zveřejněny kvůli bezpečnostním nedostatkům v cloudových službách [12]. Toto je externí hrozba, která vzniká na straně poskytovatele dané cloudové služby. Co se interních hrozeb týče, tak kromě rozlišování neúmyslných a záměrných datových úniku je potřeba odlišit také úniky způsobené na straně poskytovatele cloudových služeb a na straně organizace, která danou službu využívá.

Pokud je osoba, snažící se získat data, zaměstnancem poskytovatele cloudových služeb, jedná se o nejhorší možný scénář. Tento zaměstnanec může být např. systémovým administrátorem a využít svá přístupová práva pro přístup k citlivým datům. Následky mohou mít podobu zveřejnění nebo krádeže citlivých dat nebo vážného poškození napadených systémů a dat. Efektivní obrana proti tomuto typu vnitřní hrozby vyžaduje velké množství opatření, jak na straně poskytovatele, tak na straně zákazníka. Na straně zákazníka je komplikované i v případě služeb typu IaaS, kde má zákazník největší přístup k samotnému hardwaru, detekovat neoprávněný přístup. Pro zajištění důvěrnosti a integrity dat lze využít šifrování. Šifrování je však užitečné převážně u statických dat. Šifrování a dešifrování dat při každé operaci je náročné na výkon, navíc dešifrovací klíč musí být také uložen na stejném stroji. Existují řešení, které pracují přímo se zašifrovanými daty, ale tyto přístupy jsou nepraktické kvůli vysoké režii operací. Opatřením pro ochranu dostupnosti dat je využití několika datových center, které se nacházejí v různých částech světa. Mezi opatření na straně poskytovatele patří rozdělení odpovědnosti mezi zaměstnanci, logování aktivity, právní závazek atd., viz [15]. Zaměstnanec poskytovatele cloudových služeb může být také donucen data vydat na základě cizí legislativy [4].

V případě, kdy se snaží data získat zaměstnanec firmy, která využívá cloudové služby, jedná se o případ interního úniku, který je popsán v podsekci 2.5.2 s několika odlišnostmi týkajících se obtížnosti získání dat a vyhnutí se podezření. Získat data tímto způsobem z cloudového úložiště je snazší než v případě on-premise úložiště, už jenom z toho důvodu, že k datům v cloudovém úložišti má přístup více lidí a tedy využití cloudových úložišť představuje větší bezpečnostní hrozbu[15].

Existují řešení, které poskytují data loss prevention pro cloudová úložiště. Buď ve formě rozšíření webového prohlížeče, jako součást funkcionality cloudového úložiště nebo jako součást robustnějšího DLP řešení. Vybraná existující řešení jsou popsána v sekci 2.6. Kromě DLP řešení se k zabezpečení cloudových služeb používá *Cloud Access Security Broker (CASB)*. Princip CASB je založen na proxy serveru, který běží mezi cloudovými aplikacemi a uživateli. Tato proxy zachycuje, detekuje a šifruje citlivá data. CASB se musí přizpůsobit protokolům konkrétních cloudových aplikací, což nelze jednoduše škálovat pro různé poskytovatele cloudových služeb [12].

2.6 Existující řešení

V této sekci se budeme věnovat existujícím řešením, které se snaží implementovat DLP funkcionalitu pomocí rozšíření webového prohlížeče. Existují robustní DLP řešení, které využívají rozšíření webového prohlížeče pouze jako svou komponentu. Účel rozšíření v takovémto řešení je poskytnout jednodušší přístup k událostem a prováděným operacím v prohlížeči. Tento způsob zapojení doplňku webového prohlížeče využívají mimo jiné DLP produkty od firem McAfee a Symantec.

2.6.1 Microsoft Compliance Extension

Microsoft Compliance Extension je rozšíření pro Google Chrome, které poskytuje možnost konfigurace bezpečnostních politik pro práci v prohlížeči. Toto rozšíření je, jak název napovídá, vyvíjeno společností Microsoft, která ho nabízí jakou součást předplatného Microsoft 365. Microsoft 365 je balíček cloudových služeb, který je poskytován na bázi předplatného a obsahuje nástroje od sady kancelářských, přes nástroje pro plánování práce a zvýšení produktivity až po bezpečnostní software, kam patří právě i toto rozšíření[2]. Microsoft Compliance Extension je k dispozici s předplatným Microsoft 365 E5 nebo Microsoft 365 A5. Předplatné E5 je předplatné Enterprise úrovně a vychází na 35USD měsíčně na jednoho uživatele[3]. Předplatné A5 je určené pro studenty a měsíční poplatek činí 6USD.

Pro nasazení Microsoft Compliance Extension na koncovou stanici, musí být tato stanice zavedena do systému Microsoft 365 Endpoint data loss prevention, což je technologie monitorující operace uživatelů s daty, které byly označeny za citlivé. Další prerekvizity jsou uvedeny na stránkách dokumentace k Microsoft 365[4].

Rozsah funkcionality je uveden v tabulce 2.1.

Operace	Popis operace	Režim restrikce
Upload	Detekce pokusů o nahrání citlivého obsahu na zakázanou doménu přes Google Chrome	Zaznamenat, Blokovat
Tisk	Detekce pokusů o tisk citlivého obsahu otevřeného v Google Chrome na lokální nebo síťové tiskárně	Zaznamenat, Blokovat s možností override, Blokovat
Kopírování do schránky	Detekce kopírování citlivého obsahu, který je zobrazen v Google Chrome a následně kopírován do jiné aplikace	Zaznamenat, Blokovat s možností override, Blokovat
Kopírování na odjímatelné úložiště	Detekce pokusů o kopírování citlivého obsahu zobrazeného v Google Chrome na odjímatelné úložiště	Zaznamenat, Blokovat s možností override, Blokovat
Kopírování na síťové úložiště	Detekce pokusů o kopírování citlivého obsahu zobrazeného v Google Chrome na síťové úložiště	Zaznamenat, Blokovat s možností override, Blokovat

Tabulka 2.1: Sledované operace a podporované korekční akce

Microsoft Compliance Extension monitoruje aktivitu uživatelů, která je dostupná v Microsoft 365 compliance center. Záznamy o aktivitě uživatelů jsou vizualizovány ve formě grafů, které jsou filtrovatelné podle data výskytu, typu aktivity (vytvoření souboru, kopírování do schránky, upload, kopírování na síťové úložiště atd.), dále pak podle uživatele nebo podle lokace, kde se aktivita udála (koncová stanice, email exchange server, SharePoint, OneDrive).

[2]https://www.microsoft.com/en-ww/microsoft-365/products-apps-services
[3]https://www.microsoft.com/en-us/microsoft-365/enterprise/office-365-e5
[4]https://docs.microsoft.com/en-us/microsoft-365/compliance/dlp-chrome-get-started

Samotné rozšíření ale pouze monitoruje aktivitu uživatelů při práci s prohlížečem, jako jsou navštívené stránky a stahované soubory. Rozšíření samo o sobě však žádné bezpečnostní politiky nevynucuje, to zařizuje infrastruktura Microsoft 365 Endpoint data loss prevention a rozšíření tak slouží pouze jako monitorovací komponenta rozsáhlejšího řešení jako v případě webových rozšíření společností Symantec, McAfee a dalších.

Toto řešení je vhodné pro velké organizace, které si mohou dovolit měsíčně platit za předplatné v plném rozsahu Microsoft 365. Nasazení vyžaduje množství systémových požadavků a nasazení Endpoint data loss prevention systému. Bezpečnostní politiky jsou však nasazovány jednotně jak pro Endpoint data loss prevention tak pro Microsoft Compliance Extension a monitorovaná data jsou viditelná také na jednom místě, což výrazně zjednodušuje práci osoby zodpovědné za bezpečnost informací.

Kapitola 3

Vývoj webových rozšíření pro Google Chrome

Rozšíření (někdy označovány jako doplňky) prohlížečů jsou malé softwarové celky založené na událostech (angl. event-based), které modifikují uživatelskou zkušenost na míru danému uživateli. Každé rozšíření poskytuje určitý typ funkcionality jako např. nástroje pro zvýšení produktivity, agregace informací (počasí, aktuality, ...), přístupnost webu pro handicapované uživatele a další. Každé rozšíření by mělo mít jeden snadno definovatelný účel[1], aby nebylo v rozporu se „single-purpose“ politikou, kterou Google vydal v lednu roku 2013. Tato politika tvrdí, že by rozšíření měla mít pouze jeden, úzce zaměřený účel, aby se předcházelo zahlcení UI a zpomalování prohlížeče rozšířeními, které mají příliš široký záběr.

Webová rozšíření jsou vyvíjena s využitím typických webových technologií jako jsou HTML, JavaScript a CSS. V případě prohlížeče Google Chrome mají rozšíření danou obecnou architekturu a strukturu zdrojových souborů, tyto jsou podrobněji popsány v sekci 3.1. V rámci této struktury musí existovat soubor `manifest.json`, ze kterého prohlížeč získává informace o konkrétních rozšířeních jako třeba název a verze rozšíření, které soubory má prohlížeč použít pro zobrazení ikony a také verze manifest souboru. Formát manifest souboru je popsán v podsekci 3.1.1, od verze Google Chrome 88 je k dispozici manifest verze 3. Důvody k vydání nové verze a změny, jež obsahuje, jsou popsány v sekci 3.2. Při všech zmínkách manifestu v této kapitole je, pokud není explicitně uvedeno jinak, myšlena verze 3.

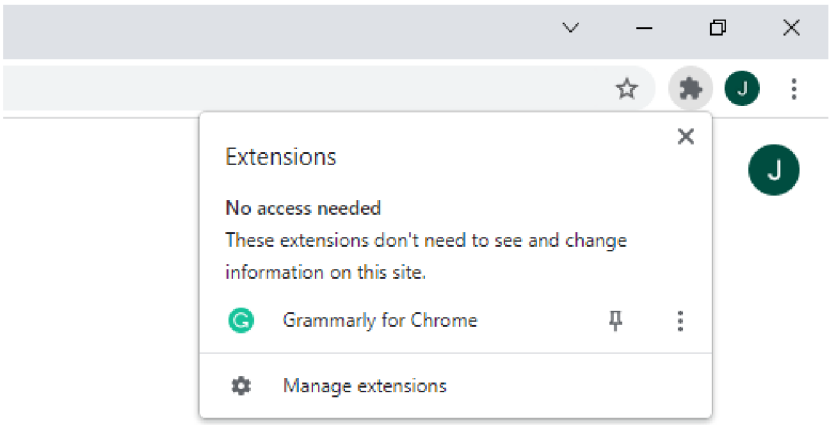
Všechny rozšíření pro Google Chrome musí být distribuovány z Chrome Web Store nebo pomocí alternativního mechanismu pro distribuci, který je popsán v podsekci 3.3.3. Oba dva způsoby distribuce však vyžadují, aby bylo dané rozšíření publikováno v Chrome Web Store. Co je to Chrome Web Store a jak do něho publikovat vytvořené rozšíření je detailně popsáno v sekci 3.3.

3.1 Architektura rozšíření prohlížeče Google Chrome

Rozšíření jako takové se skládá z HTML, CSS, JavaScriptu, ikon a dalších souborů, které jsou zabaleny do komprimovaného souboru s koncovkou zip. Každé rozšíření se může skládat z různých souborů, které mohou mít různý účel, každé ale musí obsahovat soubor manifest. Dále je vhodné přidat ikonu, která bude zobrazena v prohlížeči v panelu nástrojů pro

[1] https://developer.chrome.com/docs/extensions/mv3/single_purpose/#one

rozšíření. Panel nástrojů pro rozšíření (angl. extensions toolbar) se nachází v pravém horním rohu prohlížeče vedle adresního řádku, viz obrázek 3.1.



Obrázek 3.1: Panel nástrojů pro rozšíření

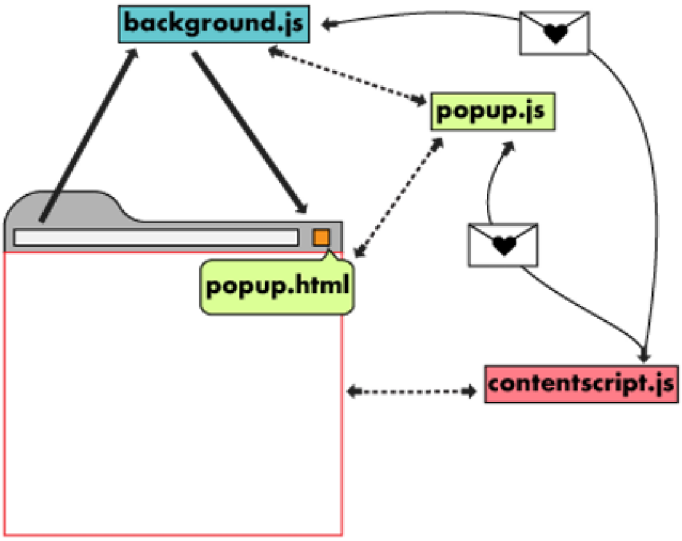
Rozšíření, která ke správnému fungování vyžadují přístup k informacím a zdrojům poskytovaných webovým prohlížečem, musí tyto požadavky uvést předem. Tento model poskytuje uživatelům podrobnější kontrolu nad tím, jak rozšíření využívá jejich zdroje a ke kterým informacím má přístup. Další výhodou je, že v případě napadení rozšíření malwarem má útočník pouze omezený přístup k informacím a zdrojům napadeného počítače[2]. Jak definovat požadovaná oprávnění je popsáno v podsekci 3.1.1 v části Permissions.

Jak bylo zmíněno výše, rozšíření se skládá z různého počtu a různých typů souborů. Architektura takových řešení se bude převážně odvíjet od funkcionality konkrétního rozšíření. Avšak většina robustních rozšíření by měla obsahovat pět základních komponent, které jsou uvedeny v následujícím seznamu.

- Manifest
- Background script
- Content scripts
- Elementy uživatelského rozhraní
- Stránka s nastavením

Každá z těchto komponent má svůj vlastní účel a může komunikovat s ostatními částmi rozšíření pomocí metod poskytovaných `chrome.extension` API. Všechny komponenty mají zároveň přístup k rozhraní pro zasílání zpráv a `storage` API, které umožňuje přístup k uloženým hodnotám. Komponenty zmíněné výše jsou popsány v následujících pěti podsekcích. Na obrázku 3.2 je znázorněna obecná architektura rozšíření se všemi částmi. Slabé plné šipky představují komunikaci pomocí rozhraní pro zasílání zpráv. Tučná plná šipka představuje interakci mezi stránkou a background scriptem, kdy background script vyhodnocuje podmínky definované autorem rozšíření a po jejich splnění zpřístupní ikonu rozšíření pro

[2]https://developer.chrome.com/docs/extensions/mv3/declare_permissions/



Obrázek 3.2: Znázornění částí rozšíření a komunikace mezi nimi [8]

zobrazení popup stránky rozšíření. Čárkované šipky představují společný kontext, ve kterém scripty běží.

3.1.1 Soubor manifest

Manifest rozšíření je sepsán v souboru `manifest.json`, který, jak již bylo zmíněno výše, předává prohlížeči informace o daném rozšíření. Prohlížeč tyto informace využívá k zobrazení komponent rozšíření, identifikaci zdrojových souborů a rozlišení jejich účelů, dále k identifikaci rozšíření, lokalizaci uživatelského rozhraní, přidělení oprávnění a dalších.

Povinné položky

V následujícím seznamu jsou popsány položky, které musí být uvedeny v každém manifestu.

`manifest_version`
: Číselná hodnota uvádějící verzi manifestu, aktuální verze je verze 3, od ledna 2022 již nelze zveřejňovat nová rozšíření s manifestem verze 2 a od ledna 2023 taková rozšíření nelze ani aktualizovat[3]. Více informací o verzi manifestu je uvedeno v sekci 3.2.

`name`
: Název rozšíření použitý v prohlížeči a v Chrome Web Store.

`version`
: Řetězec jednoho až čtyř čísel oddělených tečkou, uvádí verzi rozšíření. Před zveřejněním každé aktualizace do Chrome Web Store musí být hodnota této položky inkrementována. Více informací je uvedeno v podsekci 3.3.2.

[3] https://developer.chrome.com/docs/extensions/mv3/mv2-sunset/

Doporučené položky

Výše uvedené položky jsou povinné pro každý manifest, v následujícím seznamu jsou položky, které jsou při tvorbě rozšíření doporučené.

`action`
: Umožňuje využití API `chrome.action`, pomocí kterého lze kontrolovat tlačítko rozšíření v panelu nástrojů pro rozšíření.

`default_locale`
: Specifikuje výchozí lokalizaci rozšíření. Toto pole je doporučeno, ale v případě, že rozšíření neobsahuje složku `_locales` a tedy nemá k dispozici žádné lokalizace, tak manifest tuto položku obsahovat nesmí. V opačném případě, když rozšíření obsahuje složku `_locales`, tak manifest tuto položku obsahovat musí.

`description`
: Textový řetězec (maximální délky 132 znaků) popisující rozšíření.

`icons`
: Slovník obsahující cesty k obrázkům konkrétních velikostí. Klíč jednotlivých položek, by měl být textový řetězec udávající velikost hrany obrázku v pixelech. Rozšíření by mělo poskytnout ikony o velikost 128×128 a 48×48 pixelů ve formátu PNG.

Uvedeny jsou pouze základní položky, které jsou povinné nebo doporučené pro každé rozšíření. Kompletní seznam[4] položek podporovaných v manifestu lze najít na stránkách s dokumentací.

Permissions

Ke specifikování oprávnění jsou k dispozici následující 3 položky manifestu.

`permissions`
: Obsahuje seznam známých řetězců. Tyto řetězce jsou uvedeny u každého API, které vyžaduje některé oprávnění a kompletní seznam je dostupný na stránkách dokumentace[5].

`optional_permissions`
: Obsahuje seznam stejných řetězců jako `permissions`, ale tato oprávnění jsou udělena až za běhu, na rozdíl od `permissions`, která jsou udělována při instalaci rozšíření. Udělování oprávnění za běhu by mělo uživateli poskytnout kontext k rozhodnutí o udělení přístupu.

`host_permissions`
: Obsahuje seznam vzorů[6] stránek, ke kterým bude mít rozšíření přístup.

3.1.2 Background script

V úvodu této kapitoly bylo zmíněno, že rozšíření jsou programy založené na událostech, background script je komponenta, která tyto události zpracovává. Background script je

[4] https://developer.chrome.com/docs/extensions/mv3/manifest/
[5] https://developer.chrome.com/docs/extensions/mv3/declare_permissions/
[6] https://developer.chrome.com/docs/extensions/mv3/match_patterns/

neaktivní dokud se neobjeví událost, která je pro dané rozšíření podstatná, v ten moment je aktivován a provede požadované operace[7].

Background script může být tvořen různými technologiemi. K manifestu verze 2 se pojí background page, což je HTML soubor, který může obsahovat JavaScript kód. Hlavní výhodou background page je, že může být perzistentní a tedy běžet po celou dobu běhu prohlížeče. Manifest verze 2 může dále specifikovat množinu JavaScript souborů jako background scripty, avšak manifest může obsahovat buď pouze background page nebo pouze background scripty. Tyto scripty čekají na události a následně je zpracovávají. V případě manifestu verze 3 je background script tvořen service workerem, který popsán níže v rámci této podsekce.

Události v počítačovém programu jsou akce, ke kterým dochází uvnitř systému a o kterých nás systém informuje a my na ně následně můžeme reagovat. Například když ve webovém prohlížeči otevřeme novou záložku, klikneme na tlačítko myši nebo na klávesu klávesnice, odešleme formulář nebo je dokončeno načítání dokumentu, je vytvořena příslušná událost, na kterou lze reagovat. Reakci na událost můžeme definovat pomocí *obslužných rutin událostí* (angl. *event handler*), tvořených blokem kódu, v Javascriptu typicky funkcí[8].

V rámci rozšíření Google Chrome se background script registruje v `manifest.json` souboru položkou `background`. Položka `background` je JSON objekt, který obsahuje následující dvě položky:

`service worker`
: Tato položka obsahuje řetězec s názvem souboru, který obsahuje kód background scriptu (musí být umístěn v kořenovém adresáři). Pokud je položka `background` v manifestu použita, je povinné `service worker` specifikovat.

`type`
: Volitelná položka, které lze přidělit hodnotu `"module"`. Pokud je tato položka uvedena, service worker je využit jako ES modul, což nám dovoluje staticky importovat kód z jiných zdrojových souborů.

Od manifestu verze 3 přechází Google od přístupu *background pages* k technologii zvané *service worker*. Background page je webová stránka, která běží ve vlastním procesu, není tedy závislá na žádné aktuálně otevřené záložce a má k dispozici globální proměnnou `window`. Background page také může, na rozdíl od service workeru, běžet po celou dobu běhu prohlížeče, což je užitečné pokud potřebujeme dlouhodobě uchovávat aktuální stav rozšíření[9].

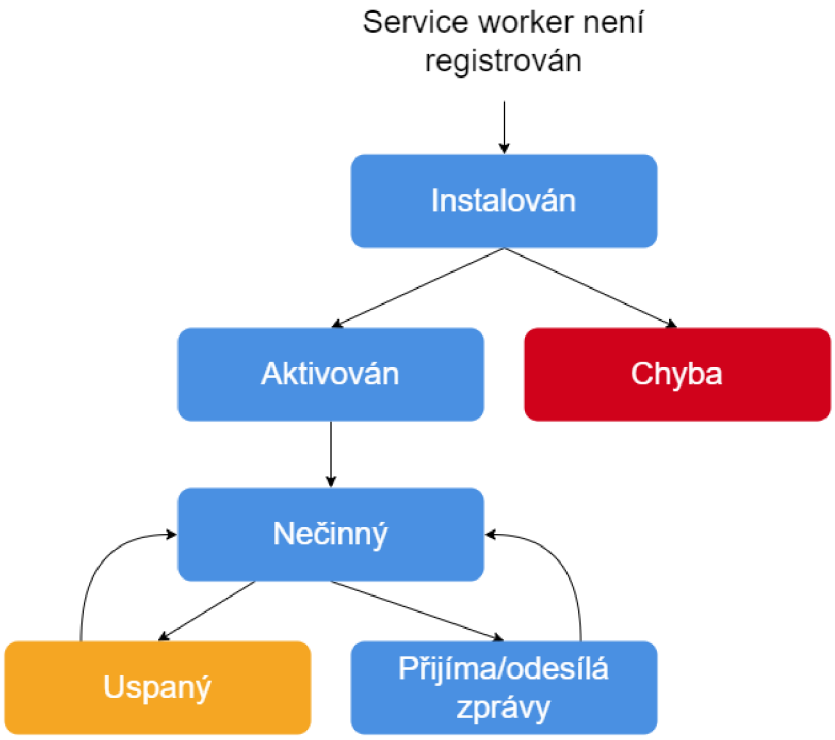
Service Worker

Service worker je koncept založený na Web Worker API a je řízený událostmi. Web Worker API představuje úlohu běžící na pozadí, která může být vytvořena pomocí samostatného zdrojového souboru, který obsahuje kód workeru. V rámci tohoto kódu lze komunikovat se stránkou, která ho vytvořila a s dalšími workery, které si může worker sám dále vytvářet (nový worker ale musí mít stejný *origin*[10] jako rodičovský worker). Web Worker API poskytuje jedno-vláknovému Javascriptu paralelní zpracování.

[7] https://developer.chrome.com/docs/extensions/mv3/architecture-overview/#background_script
[8] https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Building_blocks/Events
[9] https://developer.mozilla.org/en-US/docs/Mozilla/Add-ons/WebExtensions/Anatomy_of_a_WebExtension#background_scripts
[10] https://developer.mozilla.org/en-US/docs/Web/Security/Same-origin_policy

Jelikož service worker běží na pozadí v samostatném procesu, nemůže tak přímo přistupovat k DOM (Document Object Model[11]) webové stránky otevřené v aktivní záložce prohlížeče, ale pomocí rozhraní pro zasílání zpráv se nejdříve musí spojit se skriptem, který přístup má a ten následně může DOM číst a případně modifikovat.

Než začneme service worker využívat musíme ho zaregistrovat. V případě webové aplikace je k dispozici rozhraní pro registraci service workeru, v případě webových rozšíření předáme informaci o service workeru uvnitř manifest souboru, jak bylo popsáno výše a prohlížeč se o registraci workeru postará. Pokud je registrace úspěšná je spuštěna instalační fáze, během které se typicky stahují a cachují statické zdroje. Pokud jsou všechny zdroje úspěšně staženy a uloženy do cache, je instalační fáze dokončena a nastává aktivační fáze, ve které se zase typicky cache spravuje, např. odstranění nepotřebných a zastaralých záznamů. Po aktivaci se service worker dostane do stavu, ve kterém je buď uspán, aby šetřil zdroje, nebo pracuje a nebo přijímá/odesílá zprávy. Zjednodušený model service workeru je znázorněn na obrázku 3.3. Když je potřeba service worker aktualizovat, tak v případě



Obrázek 3.3: Zjednodušený životní cyklus service workeru[12]

využití v rámci webové aplikaci dochází k situaci, kdy má uživatel stažený a nainstalovaný jak starý tak nový service worker, ale starý je stále aktivní a nový čeká na aktivaci[13]. V případě webového rozšíření lze service worker aktualizovat pouze tak, že aktualizujeme jeho zdrojový soubor a novou verzi nahrajeme do Chrome Web Store (operacím v rámci Chrome Web Store je věnována sekce 3.3). Rozšíření je aktualizováno při dalším restartu webového prohlížeče.

[11] https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Introduction
[12] https://developers.google.com/web/fundamentals/primers/service-workers
[13] https://developers.google.com/web/fundamentals/primers/service-workers

3.1.3 Content scripts

Content script je soubor obsahující Javascript, který je proveden v kontextu webové stránky, která byla v prohlížeči načtena. Má tedy přístup k DOM otevřených stránek. Content script také může komunikovat se zbytkem rozšíření pomocí zasílání zpráv. Content script existuje v soukromém izolovaném prostředí, které mu dovoluje měnit vlastnosti svého prostředí, aniž by to ovlivňovalo načtenou stránku nebo ostatní content scripty rozšíření a naopak proměnné content scriptu nejsou viditelné načtenou stránkou ani dalšími content scripty rozšíření.

Content scripty mohou být deklarovány staticky v rámci manifestu, nebo programově načteny za běhu. Staticky deklarované scripty jsou definovány v manifestu pomocí položky `content_scripts`, která obsahuje pole objektů skládajících se z následujících čtyř položek:

`matches`
: Povinné pole řetězců udávajících URL stránek, do kterých mají být dané scripty načteny. URL mohou být definovány pomocí vzorů a masek[14].

`css`
: Volitelné pole řetězců udávajících cesty k CSS souborům, které mají být načteny do stránky ještě před konstrukcí DOM.

`js`
: Volitelné pole řetězců udávajících cesty k Javascript souborům, které mají být vloženy do stránky.

`match_about_blank`
: Volitelná boolean hodnota, která specifikuje, zda by měli být scripty načteny i do `about:blank` rámce, pokud rodič nebo otevírající rámec odpovídá některému vzoru z `matches`. Výchozí hodnota je `false`.

Programové načítání scriptů se využívá v případech, kdy je potřeba spustit content script jako reakci na nějakou událost a nebo pouze ve specifických případech. Pro dynamické načítání scriptů je potřeba, aby rozšíření mělo přidělená oprávnění přístupu ke stránkám, do kterých se snaží scripty načíst. Tyto oprávnění se přidělují buď pomocí `host_permissions` v manifestu nebo dočasně pomocí `activeTab`. Položka manifestu `host_permissions` obsahuje pole URL, na která se stahujíc stejná pravidla jako na položku `matches` při statické deklaraci. Tyto URL jsou pak rozšíření přístupné pomocí `chrome.permissions` API. Oprávnění `activeTab` dává rozšíření možnost přistoupit k aktivní záložce, když je rozšíření nějakým způsobem vyvoláno. Toho lze dosáhnout pomocí kliknutí na ikonu rozšíření, na položku v kontextovém menu, stisknutím klávesové zkratky definované pro dané rozšíření nebo přijetím návrhu od `omnibox` API[15].

3.1.4 Elementy uživatelského rozhraní

Uživatelské rozhraní (UI) rozšíření by mělo být jednoduché a minimalistické a nemělo by uživatele rozptylovat od běžného používaní webového prohlížeče. Hlavním prvkem grafického uživatelského rozhraní rozšíření jsou tzv. akce. Akce jsou zobrazeny v panelu nástrojů pro rozšíření (viz obrázek 3.1). Ikona daného rozšíření tedy funguje nejen jako indikátor nainstalovaného rozšíření, ale také jako UI pro definované akce. Pokud je ikona zobrazena barevně, tak má rozšíření definovanou akci a na ikonu lze kliknout, v opačném případě je

[14] https://developer.chrome.com/docs/extensions/mv3/match_patterns/
[15] https://developer.chrome.com/docs/extensions/mv3/content_scripts/

ikona zobrazena v odstínech šedi. Mezi další prvky rozhraní popup, který je zobrazen při aktivaci definované akce a může obsahovat samostatný HTML soubor. K dispozici jsou další prvky, které zde nejsou zmíněny, více informací lze nalézt v dokumentaci[16].

3.1.5 Stránka s nastavením

Stránka s nastavením poskytuje možnost přizpůsobení funkcionality rozšíření na míru každému uživateli. Nastavení lze v Google Chrome najít na stránce `chrome://extensions` a u konkrétního rozšíření pod tlačítkem `Details`. Existují dva typy stránek s nastavením tzv. *full page* a *embedded*, typ nastavení je dán způsobem, kterým je definována v manifestu. Full page nastavení se zobrazuje v nové záložce prohlížeče a v manifestu je definováno položkou `options_page`, která obsahuje cestu k HTML souboru s nastavením. Embedded nastavení je definováno pomocí JSON objektu `options_ui`, který obsahuje dvě položky `page` a `open_in_tab`, kde `page` má stejnou funkci jako `options_page` v předchozím případě a `open_in_tab` je boolean hodnota, která udává, zda-li se má nastavení otevřít v nové záložce, což musí být tedy nastaveno na hodnotu `false`[17].

3.2 Manifest verze 3

Prvním podstatným krokem ve vývoji platformy pro rozšíření prohlížečů byl přechod na tzv. *webby* model, který spočíval ve využití webových technologií a webového bezpečnostního modelu. Později Google představil v rámci Chrome *permissions* model, který uživatelům poskytl granulárnější kontrolu ohledně informací a zdrojů dostupných rozšířením. Kromě toho také oddělil rozšíření do samostatných procesů, za účelem poskytnutí dodatečné bezpečnosti. Google chce nadále posouvat vývoj rozšíření Google Chrome ve směru, který poskytuje „lepší ochranu soukromí, bezpečnost a výkon se zachováním nebo rozšířením možností a *webby* přístupu k vývoji rozšíření“[18].

K dosažení těchto cílů Google vydal a má v plánu vydat několik nových možností, které budou poskytovány při vývoji rozšíření. Tyto novinky ale vznášejí nové požadavky na funkcionalitu, které má za úkol řešit manifest verze 3. Mezi hlavní změny patří následující.

- Přechod z background pages na service worker, rozdíl mezi těmito přístupy byl popsán výše v podsekci 3.1.2.
- Pro modifikaci a blokování síťové komunikace je nyní dostupné nové `declarativeNetRequest` API[19]. Základem tohoto API je předání odpovědnosti za modifikaci síťových požadavků samotnému prohlížeči místo toho, aby to rozšíření dělalo samo. Rozšíření deklaruje pravidla a vzory, které určují požadavky, jež mají být modifikovány nebo blokovány a prohlížeč to obstará způsobem, který je efektivnější a zachovává soukromí.
- Vzdáleně dostupný kód již nelze v rámci rozšíření využívat. Veškerý zdrojový kód, který rozšíření využívá musí být součástí balíčku rozšíření.
- `Promise` je objekt, který reprezentuje eventuální dokončení asynchronní operace a její výslednou hodnotu[20]. S manifestem verze 3 přichází podpora `Promise` API. Tato

[16] https://developer.chrome.com/docs/extensions/mv3/user_interface/
[17] https://developer.chrome.com/docs/extensions/mv3/options/
[18] https://developer.chrome.com/docs/extensions/mv3/intro/platform-vision/
[19] https://developer.chrome.com/docs/extensions/reference/declarativeNetRequest/
[20] https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Promise

změna by měla být časem dostupná u všech podstatných API, pokud nějaké API dosud nepodporuje využití Promise, je možné využít callback funkce. Metody API vracejí Promise pouze v případech, kdy není callback parametr specifikován.

Další méně významné novinky jsou uvedeny v dokumentaci pro vývoj rozšíření[21].

Jak bylo zmíněno výše Google chce posouvat vývoj rozšíření ve směru, který poskytuje „lepší ochranu soukromí, bezpečnost a výkon se zachováním nebo rozšířením možností a *webby* přístupu k vývoji rozšíření“. Podstatná část vývojářů webových rozšíření však poukazuje na to, že manifest verze 3 tyto cíle nenaplňuje, naopak omezuje možnosti vývojářů rozšíření, které poskytují zmíněné kvality a také spousty dalších[22] [23].

3.3 Zveřejnění do Chrome Web Store

V této sekci je popsán postup publikace rozšíření do Chrome Web Store (CWS), možnosti publikace a požadavky potřebné, které musí být splněny, aby bylo možné odeslat rozšíření do schvalovacího procesu.

Doplňky jsou zveřejněny do Chrome Web Store pomocí Developer Dashboard portálu[24]. Pro přístup do tohoto portálu je nutné vytvořit si Google účet pro vývojáře a zaplatit jednorázový poplatek. Po zaplacení poplatku je možné přihlásit se do Developer Dashboard, ale abychom mohli přidat nový příspěvek (neboli rozšíření či motivy) je nutné nejdříve potvrdit emailovou adresu, tím by mělo být vytváření účtu kompletní.

Každé rozšíření v Chrome Web Store má své ID, které je tvořeno řetězcem složeného z malých písmen o délce 32 znaků. ID rozšíření lze nalézt na stránce `chrome://extensions` a každé rozšíření má zobrazeno ID na své kartě, tímto způsobem však lze nalézt pouze ID nainstalovaných rozšíření. Pro získání ID jakéhokoli rozšíření dostupného v Chrome Web Store, stačí vyhledat dané rozšíření v obchodě a kliknout na jeho stránku, z adresní řádky pak lze vyčíst ID daného rozšíření. URL rozšíření je ve tvaru `chrome.google.com/webstore/detail/ext-name/aaaabbbbccccddddeeeeffffggggghhhhiiii`, kde řetězec `ext-name` je název rozšíření a řetězec znaků za následující za lomítkem je ID daného rozšíření v tomto případě tedy `aaaabbbbccccddddeeeeffffggggghhhhiiii`. Posledním způsobem jak získat ID svého nezveřejněného rozšíření je přes Developer Dashboard, po nahrání souborů je rozšíření zobrazeno v pohledu `Položky` po kliknutí na dané rozšíření je jeho ID zobrazeno v levém horním rohu pod názvem rozšíření. Pokud potřebujeme využít ID rozšíření v kódu rozšíření, je nutné nahrát neúplné rozšíření do Developer Dashboard, vyhledat ID a po dokončení kódu jej aktualizovat. Jak rozšíření nahrát do konzole Developer Dashboard a zveřejnit ho zveřejnit ho do CWS je popsáno v následující podsekci.

3.3.1 Nahrání a zveřejnění balíčku

Když je kód rozšíření dokončen (nebo je potřeba ID rozšíření ve fázi vývoje, viz úvod sekce 3.3), tak je potřeba rozšíření nahrát a zveřejnit do Chrome Web Store ve speciálním `.zip` souboru s příponou `.crx`. Soubor je převeden do formátu `.crx` automaticky během zveřejňování do CWS.

Nejdříve je tedy potřeba zabalit veškeré potřebné zdroje jako jsou zdrojové soubory, multimediální soubory, manifest, lokalizační soubory atd. do `.zip` souboru. Tento balíček

[21] https://developer.chrome.com/docs/extensions/mv3/intro/mv3-overview/
[22] https://www.eff.org/deeplinks/2021/12/googles-manifest-v3-still-hurts-privacy-security-innovation
[23] https://www.ghostery.com/blog/manifest-v3-the-ghostery-perspective
[24] https://chrome.google.com/webstore/devconsole

musí obsahovat validní manifest soubor (viz podsekce 3.1.1) v kořenovém adresáři. Pokud je v manifestu definován background script, tak soubor uvedený v položce `service worker` musí být také v kořenovém adresáři, viz podsekce 3.1.2. V momentě kdy balíček toto splňuje můžeme ho nahrát do Developer Dashboard pomocí tlačítka „Nová položka“ (angl. „Add new item“). Po vybrání vytvořeného balíčku proběhne kontrola manifestu, pokud je nalezena nějaké chyba, tak balíček nebude nahrán dokud tato chyba nebude opravena.

V momentě kdy je balíček nahrán, je daná položka zobrazena v pohledu Položky. Dalším krokem ke zveřejnění rozšíření do CWS je odeslání ke kontrole. Balíček lze odeslat ke kontrole, když vyplníme všechny požadované informace v pohledech „Záznam v obchodu“, „Postupy v oblasti ochrany soukromí“ a „Platby a distribuce“ (angl. „Listing“, „Privacy“, „Payment and Distribution“), ke kterým se dostaneme po kliknutí na danou položku. Mezi požadované položky patří popis, ikona, kategorie, snímky obrazovky, odůvodnění požadovaných oprávnění a také viditelnost rozšíření, pro tuto položku jsou na výběr tři možnosti:

Veřejné
: Položka je viditelná všemi uživateli ve vybraných zemích.

Neveřejná
: Položka je viditelná pouze pro uživatele, kteří mají odkaz.

Soukromá
: Položku vidí pouze uživatelé uvedení jako důvěryhodní testeři.

Po vyplnění těchto informací můžeme odeslat balíček ke kontrole, pomocí tlačítka „Odeslat ke kontrole“, pokud jsou některá políčka nevyplněna nebo neobsahují validní data (např. špatná velikost ikony nebo snímků obrazovek), nelze požadavek odeslat.

Když je požadavek úspěšně odeslán, přichází na řadu manuální kontrola zdrojových souborů. Po kliknutí na tlačítko „Odeslat ke kontrole“ je zobrazen dialog, obsahující mimo jiné zaškrtávací políčko, které udává, zda má být položka po úspěšném dokončení kontroly automaticky zveřejněna. Zveřejnění lze odložit i zpětně po odeslání ke kontrole se zaškrtnutým políček o automatickém zveřejnění. Odložení může trvat maximálně 30 dní, po vypršení této doby bude položka vrácena do fáze návrhu a celý proces bude muset být opakován[25]. Délka kontroly se může pro každou položku lišit, v roce 2021 bylo 90 % položek zkontrolováno během tří dnů od odeslání[26].

3.3.2 Aktualizace rozšíření

K aktualizaci zveřejněné položky lze opět využít Developer Dashboard a nahrát všechny změněné i nezměněné způsoby jako bylo popsáno v podsekci 3.3.1 a aktualizovat všechny informace, které je zapotřebí vyplnit, aby bylo možné odeslat položku ke kontrole. Aktualizovaná položka bude zveřejněna se stejnou viditelností jako aktuální verze.

Pokud byl změněn kód rozšíření, manifest nebo ostatní soubory zabalené s rozšířením, je nutné nejdříve inkrementovat položku `version` v manifestu a následně vytvořit nový balíček a ten nahrát do Developer Dashboard. Pokud je potřeba aktualizovat pouze informace o položce, které jsme vyplňovali v pohledech „Záznam v obchodu“, „Postupy v oblasti ochrany soukromí“ a „Platby a distribuce“, tak stačí provést potřebné změny právě na těchto kartách a není potřeba inkrementovat verzi a nahrávat nový balíček.

[25] https://developer.chrome.com/docs/webstore/publish/
[26] https://developer.chrome.com/docs/webstore/review-process/#review-time

Pokud je aktualizace odeslána ke kontrole, tak aktuální verze není ovlivněna do té doby, dokud není nová verze zveřejněna. Aktuální uživatelé tedy stále mohou používat aktuální verzi a noví uživatelé si ji nainstalovat. V momentě, kdy je zveřejněna nová verze, to prohlížeč zaznamená a při dalším spuštění je rozšíření automaticky aktualizováno[27].

3.3.3 Distribuce rozšíření

Všechny rozšíření jsou uživatelům distribuovány z Chrome Web Store, instalace rozšíření distribuovaných ze zdrojů třetích stran není možná. Toto pravidlo má však tři výjimky:

- Rozšíření instalovaná pomocí *enterprise policy*[28]
- Nezabalená rozšíření přidána v rámci vývojářského režimu
- Rozšíření instalovaná na stroje s operačním systémem Linux

Existuje také alternativní systém distribuce pro případy, kdy nechceme instalovat rozšíření manuálně z Chrome Web Store. Následující dvě situace, jsou typickými příklady, kdy tento alternativní systém chceme využít:

- Rozšíření je součástí nějakého dalšího softwaru a chceme ho nainstalovat pokaždé, kdy instalujeme tento software.
- Síťový administrátor chce nainstalovat stejné rozšíření napříč spravovanou organizací.

Aby bylo možné tento způsob využít musí být dané rozšíření zveřejněno v CWS (v případě instalace na systém Linux toto neplatí, ale musíme mít k dispozici `.crx` soubor rozšíření). Dále je potřeba znát ID rozšíření a aktuální verzi. K instalaci na operačních systémech Mac OS a Linux se využívá JSON souboru `preferences.json` a na Windows strojích je instalace provedena pomocí Windows registrů. Detailní popis postupu pro Linux a Mac OS[29] i pro Windows[30] je k dispozici v dokumentaci rozšíření pro Google Chrome.

[27] https://developer.chrome.com/docs/webstore/update/
[28] https://support.google.com/chrome/a/answer/7666985
[29] https://developer.chrome.com/docs/extensions/mv3/external_extensions/#preferences
[30] https://developer.chrome.com/docs/extensions/mv3/external_extensions/#registry

Kapitola 4

Návrh architektury a funkcionality webového rozšíření

Tato kapitola obsahuje návrh implementovaného rozšíření pro prohlížeč Chrome společnosti Google. Rozšíření implementuje data loss prevention uvnitř prohlížeče. Rozšíření poskytuje možnost konfigurace bezpečnostních opatření pro podporované operace, viz podsekce 4.1.1. Návrh celkového řešení je rozdělen do čtyř sekcí. Sekce 4.1 je věnována návrhu funkcionality rozšíření, nejdříve je popsán návrh DLP funkcionality a následně návrh přístupu auditu incidentů. V sekci 4.2 je popsán návrh rozdělení funkcionality mezi komponenty rozšíření, které jsou uvedeny v sekci 3.1. Dále v sekci 4.3 je popsán návrh způsobu konfigurace rozšíření síťovým administrátorem. Nakonec v sekci 4.4 je uveden návrh instalace na koncová zařízení.

4.1 Funkcionalita rozšíření

V této sekci je popsán návrh funkcionality rozšíření. Popis navrhované funkcionality je rozdělen na dvě části. V podsekci 4.1.1 Data Loss Prevention je uveden přístup k vynucování bezpečnostních politik na jednotlivých kanálech, kterými mohou data uniknout. Podsekce 4.1.2 Audit incidentů popisuje návrh přístupu ke kolekci a uchovávání informací o bezpečnostních incidentech detekovaných v rámci prohlížeče.

4.1.1 Data Loss Prevention

Jak bylo uvedeno v sekci 2.5, hlavním účelem DLP řešení je zajistit to, aby se důležitá data nedostala do nesprávných rukou. DLP se snaží chránit před hrozbami zevnitř organizace, zároveň ale nesmí omezovat zaměstnance ve výkonu své práce. Rozšíření vyvíjené v rámci této práce by mělo poskytovat ochranu před neúmyslným zveřejněním firemních dat, ke kterému může dojít různými způsoby v rámci webového prohlížeče.

Největší riziko mezi způsoby zveřejnění představuje nahrávání souborů jako přílohu k emailu či do cloudových úložišť. Nahrávání bude tedy jedním hlavních aspektů, na který se v rámci této práci zaměřím, dále pak také na kopírování obsahu souborů otevřených v prohlížeči do schránky a metody snímání obrazovky.

Předpokládáme tedy, že organizace používá cloudové úložiště, do kterého ukládá své důvěrné informace a přístup k nim mají pouze oprávnění zaměstnanci. Hlavní část funkcionality by měla spočívat v detekování a případnému zastavení akcí provedených v prohlížeči, které by ohrozily bezpečí těchto informací. Sledované operace a podporované režimy restrikce jsou uvedeny v následující tabulce 4.1.

Operace	Popis operace	Režim restrikce
Upload	Detekce nahrávaného souboru pomocí prohlížeče Chrome mimo bezpečné úložiště	Zaznamenat, Notifikovat, Blokovat
Kopírování do schránky	Detekce kopírování citlivého obsahu nebo souboru do schránky v prohlížeči Chrome	Zaznamenat, Notifikovat, Blokovat
Snímek obrazovky	Detekce snímku aktivního okna Google Chrome	Zaznamenat, Notifikovat, Blokovat

Tabulka 4.1: Sledované operace a podporované korekční akce

Režimy restrikce v předchozí tabulce představují způsob, jakým lze reagovat na detekovanou operaci. Režim *Zaznamenat* představuje pouze vytvoření záznamu o operaci a žádné další opatření nejsou provedeny. Režim *Notifikovat* znamená, že kromě vytvoření záznamu je uživateli zobrazena notifikace, která ho upozorňuje na to, že prováděná operace je riziková. Nakonec režim *Blokovat* vytvoří záznam o incidentu, vytvoří notifikaci pro uživatele, obsahující informace o zablokované operaci a operaci zablokuje.

4.1.2 Audit incidentů

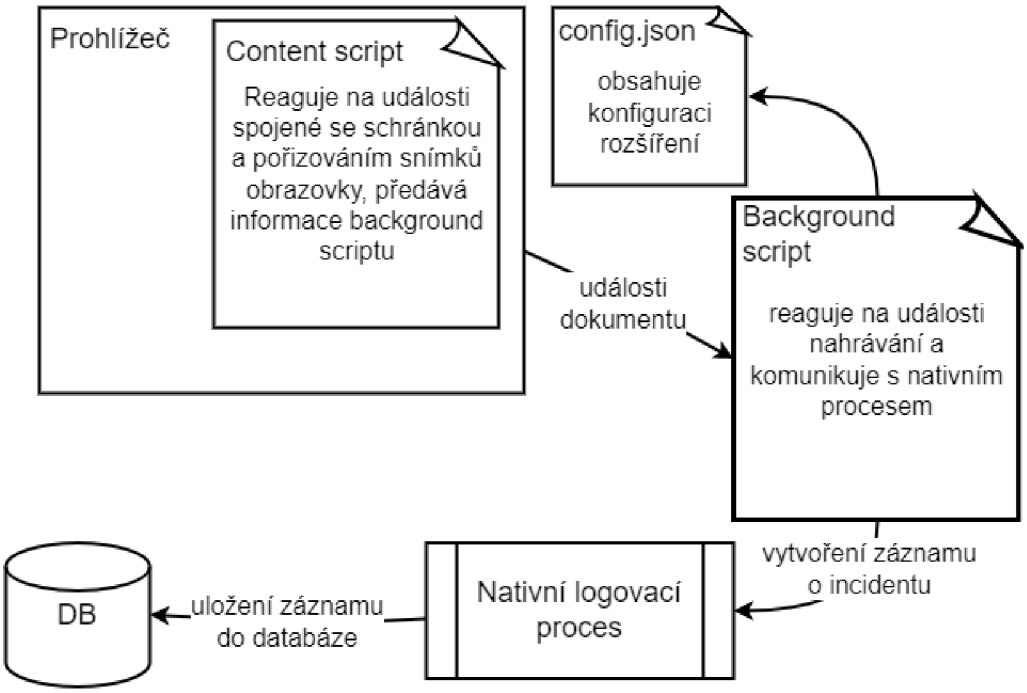
Rozšíření nainstalovaná na počítačích organizace odesílají informace o všech incidentech, ke kterým došlo na dané stanici, do centrální databáze. Databáze by měla být na stanici, která je dostupná z ostatních stanic. Rozšíření pro Google Chrome nemají k dispozici žádné API, pomocí kterého by bylo možné komunikovat s databází na přímo. Ke komunikaci s databází je možné využít dva způsoby.

Prvním z nich je využití nativního procesu, se kterým rozšíření již komunikovat může a tento proces zase může komunikovat s danou databází. Rozšíření tedy po detekování incidentu zašle zprávu danému procesu. Pokud proces neběží, je spuštěn a následně obdrženou zprávu zpracuje. Zpracování spočívá v transformaci přijaté zprávy na dotaz pro přidání záznamu do databáze. Nevýhodou tohoto způsobu je potřeba distribuce tohoto nativního procesu na všechny stroje společně s rozšířením.

Druhým způsobem, jak dostat informace z rozšíření do databáze, je přeposílání informací přes webovou aplikaci, která slouží jako proxy mezi rozšířením a databází. V rámci manifestu lze definovat URL adresu, na které běží zmíněná aplikace a se kterou může rozšíření komunikovat pomocí `fetch` API a HTTP protokolu. Tato webová aplikace pak provádí obdobnou činnost jako nativní proces v prvním příkladu. V tomto případě nám odpadá nutnost distribuovat soubor na všechny stanice. Nicméně webová aplikace musí někde běžet, což vyžaduje další zdroje. Navíc s touto aplikací budou komunikovat všechny instance rozšíření, mohlo by tedy docházet k zahlcení stroje, na kterém aplikace poběží. Dále tím vzniká *single point of failure*, tedy pokud aplikace nepoběží nebudou do databáze chodit žádné záznamy ze žádných stanic, kdežto pokud selže jeden proces, nebudou chodit záznamy pouze z jedné stanice. V rámci této práce budeme využívat řešení s nativním procesem, protože existuje alternativní způsob distribuce (viz podsekce 3.3.3) rozšíření společně s dalším softwarem.

4.2 Komponenty řešení

V sekci 3.1 jsou uvedené komponenty rozšíření a doplňující komponenty, mezi které je rozdělena jeho funkcionalita. Rozdělení funkcionality již bylo částečně popsáno v podsekci 4.1.1. V této sekci bude popsáno, které komponenty budou využity a jakou funkcionalitu budou obsahovat. Zjednodušený model celkového řešení je uveden na obrázku 4.1.



Obrázek 4.1: Zjednodušený model řešení

Zmíněné cloudové úložiště v podsekci 4.1.1 je identifikováno URL adresou a je považováno za bezpečné. Typický scénář pro práci rozšíření začíná v momentě, kdy zaměstnanec zapne Google Chrome.

4.2.1 Background script

Při zapnutí prohlížeče je vytvořen proces, ve kterém běží background script. Background script si jako první věc načte konfiguraci zadanou administrátorem, která je sdílená mezi rozšířeními na všech počítačích v rámci organizace. Po načtení konfigurace naslouchá background script událostem prohlížeče a reaguje pouze na ty, které indikují jednu z operací uvedených v tabulce 4.1. V případě zachycení jedné ze zmíněných událostí, na ni background script reaguje na základě načtené konfigurace. Pokud je operace vyhodnocena jako nebezpečná, dochází k incidentu. Po detekci incidentu jsou aplikovány bezpečnostní politiky podle zvoleného režimu. Následně je vytvořen záznam o incidentu a ten je odeslán do centrální databáze.

4.2.2 Content script

Content script je načten do aktuální stránky při jejím načítání. Tento script bude načítán do webových stránek, na kterých budou zakázány operace kopírování do schránky a pořizování snímků obrazovky. Jeho úkolem je naslouchat událostem `copy` a `paste` poskytovaných

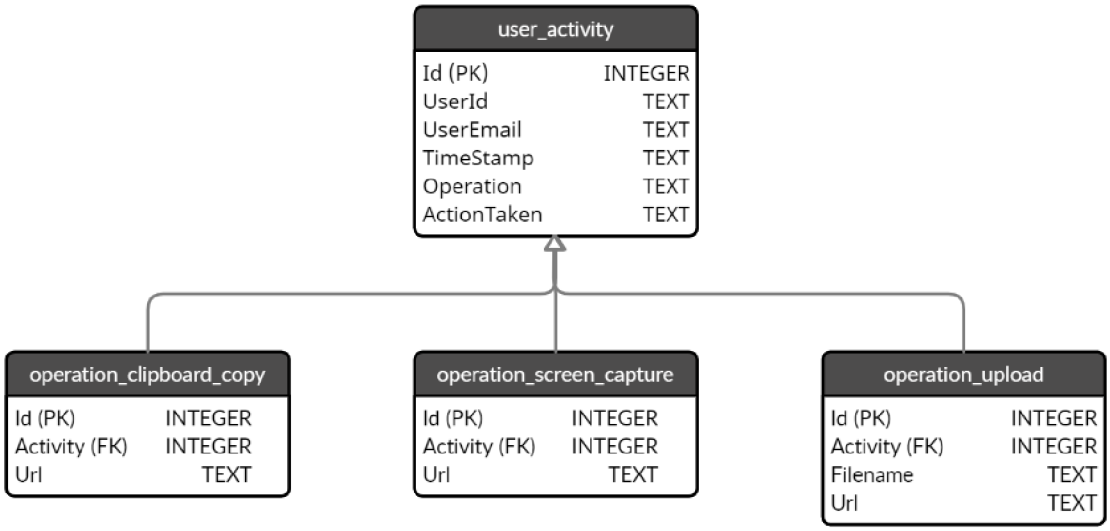
rozhraním `navigator.clipboard` a v momentě detekce některé z těchto událostí zaslat relevantní informace background scriptu, který je následně zašle nativnímu procesu pro uložení do databáze.

4.2.3 Nativní logovací proces

Tento proces implementuje zasílání záznamů o incidentech do centrální databáze. Možnosti provedení ukládání záznamů do databáze jsou uvedeny v sekci 4.1.2. Zvolený přístup spočívá ve využití nativního procesu, který přijímá zprávy od rozšíření, které s ním může komunikovat pomocí rozhraní pro zasílání zpráv. Nativní proces zpracuje přijaté zprávy ve formátu JSON a vytvoří dotaz, který vloží data do příslušné tabulky databáze.

4.2.4 Databáze incidentů

Databáze incidentů obsahuje záznamy ze všech stanic, na kterých je rozšíření nainstalováno. Na obrázku 4.2 je uvedeno schéma databáze.



Obrázek 4.2: ER diagram databáze pro ukládání logů

Schéma databáze je jednoduché a skládá se pouze ze čtyř tabulek. Tabulka `user_activity` slouží pro uchovávání základních informacích o všech incidentech, tabulky `operation_upload`, `operation_clipboard_copy` a `operation_screen_capture` obsahují informace specifické pro každou z operací. Tabulky pro konkrétní operace obsahují sloupec s klíčem záznamu v tabulce `user_activity`, který slouží pro modelování dědičnosti mezi tabulkami, aby nedocházelo k duplikaci sloupců v jednotlivých tabulkách.

Součástí databáze je také pohled, který agreguje záznamy ze všech tabulek konkrétních operací a slouží pro zobrazení všech incidentů, ke kterým došlo napříč organizací.

4.3 Možnosti konfigurace rozšíření

Rozšíření je konfigurováno administrátorem pomocí souboru ve formátu `json`. Tento soubor má dané schéma, kterému musí odpovídat. Tento soubor je přístupný pomocí API `storage`, které umožňuje rozšíření ukládat, získávat a pozorovat změny uživatelských dat v lokální

databázi. Toto API má tři různé oblasti. Jednou z nich je `managed` oblast, která je pro rozšíření pouze *read only* a hodnoty jsou nastaveny administrátorem pomocí *enterprise policy*. Všechny instance rozšíření v organizaci tedy mohou přistupovat k centrálně vytvořené konfiguraci a žádná z nich ho nemůže měnit.

4.4 Instalace na koncové stanice

Instalace na koncové stanice vyžaduje, aby bylo rozšíření zveřejněné v Chrome Web Store jako veřejné či neveřejné. Instalace je provedena pomocí instalačních scriptů, které vytvoří registry pro instalaci rozšíření do prohlížeče, viz podsekce 3.3.3. Pro instalaci rozšíření je využit registrový klíč `HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Google\Chrome\ExtensionInstallForcelist`, tento klíč je využíván, když chce síťový administrátor vynutit instalaci rozšíření pomocí *group policy*. Tento způsob instalace má za následek, že uživatel nemůže rozšíření odinstalovat běžným způsobem, ale musí smazat příslušný registr. Tato ochrana proti odinstalování rozšíření není stoprocentně efektivní, ale měla by být dostačující pro běžného uživatele. Ochrana proti zápisu či mazání registrů je mimo rozsah této práce. Po zapsání registru pro vynucení instalace je zapsán registr, který obsahuje cestu k manifestu nativního procesu.

Kapitola 5

Implementace

V této kapitole jsou uvedeny použité technologie a jazyky využité k implementace jednotlivých částí řešení, viz sekce 5.1. Dále jsou v této kapitole, konkrétně v sekci 5.2, popsány komplikace s manifestem verze 3 a API poskytovanými Google pro vývoj rozšíření, které se objevily během implementace. Dále následuje sekce 5.3, kde je popsána samotná implementace jednotlivých částí řešení. V poslední sekci 5.4 této kapitoly je popsán způsob, kterým je prováděna instalace celého řešení.

5.1 Použité technologie

K implementaci rozšíření byl použit jazyk TypeScript v kombinaci s nástrojem webpack, který je používán ke kompilaci TypeScript souborů a k vytvoření balíčku z vygenerovaných JavaScript souborů.

Nativní proces zajišťující ukládání záznamů o incidentech do databáze je implementován v jazyce C#. Ke zpracování zpráv od rozšíření, které jsou zasílány ve formátu JSON, je využit balíček `System.Text.Json` a pro práci s databází balíček `Microsoft.Data.Sqlite`.

Databáze je implementována pomocí SQLite, protože SQLite databáze je snadno použitelná a multiplatformní. Pro tento případ, kdy databáze nemá složité schéma, ale obsahuje pouze malý počet tabulek a jeden pohled, je tedy SQLite ideální.

5.2 Komplikace při implementaci

Během implementace navrženého řešení se ukázalo několik problémů, se kterými se během návrhu nepočítalo buď z důvodu neúplné dokumentace, pouze částečné podpory prohlížeče některých funkcí, omezené funkcionality některých API či přílišného omezení možností pro vývojáře v rámci manifestu verze 3 oproti verzi 2. Toto jsou důvody, kvůli kterým nebylo možné implementovat navržené řešení v plném rozsahu.

V následujících podsekcích jsou podrobněji popsány, pro tuto práci, nejzávažnější z výše zmíněných problémů. Konkrétně se jedná o omezení manifestu verze 3, kvůli kterým bylo nutné přepsat již existující částečnou implementaci do manifestu verze 2, tyto omezení jsou popsány v podsekci 5.2.1. V podsekci 5.2.2 je popsán bug ve `webRequest` API, který v některých případech znemožňuje čtení těla HTTP dotazů a nakonec v podsekci 5.2.3 je popsán problém s nepřístupným API pro manipulaci schránky.

5.2.1 Manifest verze 3

Jak je uvedeno v podsekci 3.1.2, background script rozšíření využívajícího manifest verze 3 je tvořen service workerem, který je po krátké době inaktivity uspán. Tato vlastnost je nevhodná v kombinaci s `webRequest` API, které poskytuje možnost zachytit události související s jednotlivými HTTP dotazy. Uspání a probuzení service workeru mezi každým dotazem by výrazně zatěžovalo CPU stanice, na které rozšíření běží. Pokud by service worker nebyl uspáván a probuzen mezi každým dotazem, mohlo by naopak dojít k situaci, kdy bude service worker neustále zpracovávat události bez uspání, v takovém případě ho Chrome po pěti minutách ukončí. Service worker tedy není perzistentní na rozdíl od background page, která je k dispozici v manifestu verze 2.

Manifest verze 3 poskytuje `declarativeNetRequest` jako náhradu `webRequest` API. Toto nové API není pro tuto práci dostačující, protože poskytuje pouze deklarativní vytváření pravidel, která aplikují danou operaci (povolení, blokování, přesměrování, úprava hlaviček nebo úprava schématu) na dotazy, které vyhoví podmínce definované v daném pravidle. Podmínky se mohou vztahovat mimo jiné na zdrojovou či cílovou doménu nebo použitou metodu v dotazu, kompletní seznam všech možností je dostupný v dokumentaci[1]. Toto by bylo dostačující pokud bychom nepotřebovali získat data z těla dotazu, který při nahrávání souboru na cloud nebo do služeb pro sdílení souborů a dalšího obsahu tento soubor přenáší. K těmto datům je potřebný přístup pro navazující analýzu dat, která rozhoduje, zda soubor obsahuje data, která by se neměla dostat mimo bezpečnostní perimetr firmy.

Verze `webRequest` API dostupná v manifestu 3 je pouze v neblokující verzi. Nelze tedy blokovat dotazy v rámci obslužné rutiny události vygenerované před jejich odesláním. Dle návodu pro migraci[2] z manifestu verze 2 na manifest verze 3 lze použít blokující `webRequest`, ale pouze v případech, kdy je rozšíření instalováno pomocí `ExtensionInstallForcelist` položky v *Group Policy* šabloně pro Google Chrome. Group Policy poskytuje správu a konfiguraci operačních systémů aplikací a uživatelských nastavení v rámci Windows Active Directory. Tímto způsobem by tedy šlo využít `webRequest` API zamýšleným způsobem s manifestem verze 3 (pořád by platilo omezení uvedené v prvním odstavci této podsekce), ale tímto nastává jiný problém. Během vývoje by muselo být rozšíření po každé úpravě či přidané funkcionalitě zveřejněno do Chrome Web Store. Tento proces by byl značně zdlouhavý, protože každá aktualizace balíčku by musela projít schvalovacím procesem, viz sekce 3.3.

Další komplikací při využití service workeru pro navrhované řešení je absence dokumentového objektového modelu. DOM je v rámci řešení využíván pro čtení a zápis do schránky, protože v rámci service workeru není přístupné `clipboard` API. Tento problém je dále vysvětlen v následující podsekci.

5.2.2 Bug v `chrome.webRequest` API

Výše zmíněné `webRequest` API obsahuje událost `onBeforeRequest`[3], ke které lze zaregistrovat obslužnou rutinu. Tato obslužná rutina přijímá dva povinné argumenty `callback`, `filter` a volitelný parametr `extraInfoSpec`. Argument `callback` je funkce, která je vo-

[1] https://developer.chrome.com/docs/extensions/reference/declarativeNetRequest/#type-RuleCondition

[2] https://developer.chrome.com/docs/extensions/mv3/intro/mv3-migration/#when-use-blocking-webrequest

[3] https://developer.chrome.com/docs/extensions/reference/webRequest/#event-onBeforeRequest

lána v případě výskytu této události. Při volání dostane jako vstupní parametr objekt, který by měl obsahovat mimo jiné také položku `requestBody`, ta je přítomna pouze pokud parametr `extraInfoSpec` obsahuje řetězec `'requestBody'`. Položka `requestBody` by měla obsahovat:

- chybovou hlášku nebo
- objekt obsahující hodnoty ve formátu klíč-hodnota v případě, že se jedná o HTTP dotaz s metodou POST a data jsou zakódována ve formě `multipart/form-data` nebo `application/x-www-form-urlencoded` a nebo
- surová data v případě, že nedošlo k chybě, dotaz obsahoval metodu PUT či POST a data nebyla zpracována předchozím způsobem.

K položce `requestBody` existuje tiket na Chromium Bugs[4], který říká, že v případě nahrávání souborů položka `requestBody` neobsahuje data nahrávaného souboru, ale podle dokumentace je očekávaná situace opačná a dotaz by měl zmíněná data obsahovat. Závěrem zmíněného tiketu je zjištění, že Chrome zcela nepodporuje tuto funkcionalitu a vznikl návrh na upravení dokumentace tak, aby bylo uvedeno, že položka `requestBody` není zcela implementována. Nicméně tiket byl 10. července 2020 archivován z důvodu inaktivity a popsané chování stále přetrvává i v době psaní této práce a dokumentace tento fakt dosud také nijak nezmiňuje.

Toto je poměrně zásadní komplikace, jelikož to znemožňuje analýzu nahrávaných souborů a tím znemožňuje využití a případnou implementaci content-based přístupu (viz podsekce 2.5.3) k detekci citlivých dat. Na základě toho vznikla pouze omezená implementace ochrany nahrávání souborů, která je popsána v podsekci 5.3.1.

5.2.3 Nepřístupné `clipboard` API

Pro práci se schránkou je v rámci prohlížeče doporučené API `clipboard`[5], které je poskytováno rozhraním `navigator`. Tímto API je nahrazeno již zastaralé API `document.execCommand`, které funguje analogicky ke kopírování obsahu z webové stránky do schránky či vkládání dat ze schránky do webové stránky pomocí klávesových zkratek, kdežto `clipboard` API čte či zapisuje libovolná data přímo do schránky.

Nicméně implementace `navigator` rozhraní v service workeru neobsahuje `clipboard` API a není ho tedy možné využít s manifestem verze 3. Toto je problém pro větší množství vývojářů a je pro něj vytvořen tiket Chromium Bugs[6], který je v době psaní stále otevřen a nejsou dostupné informace, zda bude tento problém adresován.

Protože webové stránky `clipboard` API poskytují je možné jej využít uvnitř content scriptu, který je vložen do dané stránky. Pokud ale přesuneme funkcionalitu spojenou s prací se schránkou do content scriptu, narazíme na problém, kdy je uživateli zobrazena žádost o udělení přístupu ke schránce při každém vložení content scriptu. Uživatel v tento moment má možnost přístup neudělit, v takovém případě content script se schránkou nemůže pracovat a tím veškerá funkcionalita spojená se schránkou nebude fungovat. V druhém případě, kdy uživatel přístup povolí, získá přístup ke schránce celá stránka a ne pouze konkrétní content script. Dalším důsledkem tohoto je, že zmíněná žádost bude zobrazena při každém

[4] https://bugs.chromium.org/p/chromium/issues/detail?id=813285

[5] https://developer.mozilla.org/en-US/docs/Mozilla/Add-ons/WebExtensions/Interact_with_the_clipboard

[6] https://bugs.chromium.org/p/chromium/issues/detail?id=1160302

obnovení stránky či otevření nové stránky, což značně ovlivňuje uživatelskou přívětivost při používání prohlížeče. Nejzásadnějším faktorem je, že z podstaty řešení ochrany proti ztrátě dat, nechceme uživateli udělit možnost tento systém obejít a možnost neudělení oprávnění tuto možnost poskytuje.

Další možnost jak pracovat se schránkou je návrat k zastaralému `document.execCommand` API, nicméně princip práce s tímto API je založen na přítomnosti DOM a vytvoření elementu, do kterého lze vložit text (např. `textarea`) a následně pomocí něho číst nebo zapisovat obsah schránky. Detailní popis této funkcionality je uveden v podsekci 5.3.1. Nicméně jak již bylo zmíněno, tento přístup vyžaduje ke správnému fungování DOM a service worker žádným nedisponuje. Pro využití tohoto přístupu by tedy bylo nezbytné otevírat záložku obsahující prázdnou stránku, se kterou by rozšíření pracovalo nebo opět využít content script, který by vytvořil element přímo ve stránce, která tento script obsahuje. Ani jeden z těchto přístupů neevokuje dobrý přístup k uživatelské přívětivosti. V prvním případě by musel mít uživatel neustále, jemu z neznámého důvodu, otevřenou záložku. V tom druhém by mohlo např. docházet ke konfliktům identifikátorů elementů, či narušit vzhled stránky. Proto se nabízí využití manifestu verze 2 a background page, která má svůj DOM a tedy obsahuje jak `clipboard` API tak možnost vytváření elementů přímo v rámci rozšíření bez nutnosti otevírání zvláštní záložky.

Problém se schránkou, ale přetrvává i v případě manifestu verze 2 a sice v případě již několikrát zmiňovaného `clipboard` API. Pro přístup ke schránce pomocí metod `clipboard` API je nutné, aby stránka, které tyto metody volá, byla v popředí, což background page již z principu být nemůže, jelikož běží v jiném procesu než samotný prohlížeč. Bylo tedy využito přístupu s využitím `document.execCommand` API, jehož implementace je popsána v podsekci 5.3.1.

5.3 Popis implementace komponent

Tato sekce je věnována technickému popisu implementace jednotlivých komponent celého řešení. V podsekci 5.3.1 je popsána implementace samotného rozšíření včetně souboru `manifest.json`, implementace ochrany vůči uploadu souborů, kopírování do schránky a pořizování snímku okna Google Chrome. V podsekci 5.3.2 je popsán protokol, kterým rozšíření komunikuje s nativním procesem a implementace nativního procesu, který ukládá informace o bezpečnostních incidentech do databáze. Nakonec v podsekci 5.3.3 je popsána struktura a implementace SQLite databáze.

5.3.1 Rozšíření

Manifest

Jak je uvedeno v podsekci 3.1.1 manifest rozšíření musí být sepsán v souboru `manifest.json`, který se nachází v kořenovém adresáři balíčku s rozšířením. Následuje výčet vybraných položek použitých v manifestu vytvořeného rozšíření s konkrétními hodnotami a odůvodněním výběru těchto položek a hodnot.

- `manifest_version` určuje verzi manifestu, v implementovaném řešení je použit manifest verze 2 a to z důvodů uvedených v sekci 5.2.
- `default_locale` určuje výchozí lokalizaci jejíž hodnota je textový řetězec `"en"` určující angličtinu jako výchozí lokalizaci. Rozšíření lze také lokalizovat do češtiny.

- `background` položka je JSON objekt, který obsahuje položky `page` a `persistent`.
  - `page` určuje HTML soubor s background page. Background page ve vytvořeném řešení je pojmenována `background.html` a nachází se v kořenovém adresáři balíčku. Soubor `background.html` obsahuje hlavičku, ve které je přidán element `script`, který importuje JavaScript kód ze souboru `background.js`, který je popsán níže v rámci této podsekce. Tělo tohoto HTML souboru obsahuje `textarea` element, který je využíván pro práci se schránkou, viz níže nadpis „Kopírování do schránky“.
  - `persistent` obsahuje boolean hodnotu určující, zda má background page běžet po celou dobu běhu prohlížeče bez toho, aniž by byla „unloaded“.
- `permissions` je pole řetězců, které specifikují oprávnění potřebné pro správné fungování rozšíření.
  - `*://*/*` řetězec představuje masku URL, ke kterým může rozšíření přistupovat. Tato maska zachytí URL se všemi podporovanými schématy, všechna doménová jména a všechny cesty v rámci dané domény. Toto oprávnění je potřeba pro vložení content scriptů do libovolných stránek.
  - `storage` povoluje využití `chrome.storage` API, které je využívané ke čtení konfigurace rozšíření.
  - `nativeMessaging` umožňuje komunikaci s nativním procesem.
  - `identity` a `identity.email` poskytuje přístup `chrome.identity` API, díky kterému lze získat údaje o uživateli přihlášeném do Google Chrome. Pro získání údajů o uživateli, je nutné uvést obě tyto oprávnění .
  - `webRequest` a `webRequestBlocking` poskytuje přístup k `chrome.webRequest` API, aby bylo možné blokovat HTTP dotazy je nutné uvést i `webRequestBlocking`.
  - `notifications` poskytuje přístup k `notifications` API, které umožňuje vytvářet notifikace a zobrazovat je v oznamovací oblasti operačního systému Windows.
- `storage` položka je JSON objekt obsahující položku `managed_schema`, která uvádí název souboru se schématem úložiště. Schéma úložiště musí být specifikováno, aby rozšíření mohlo využívat `managed` úložiště ze `storage` API. Způsob využití úložiště je popsán níže v této podsekci.

Background script

Background script je vložen do background page pomocí HTML elementu `script` uvedeného v hlavičce. Background script je implementován v souboru `background.ts` a obsahuje kód, který inicializuje třídu `PolicyHelper` a další na ní závislé. Třída `PolicyHelper` zprostředkovává aktuálně nastavené politiky pro jednotlivé kanály či nastavená bezpečná úložiště. Inicializace této třídy spočívá v načtení politik ze `storage.managed` úložiště a následné inicializace komponent na nich závislých pomocí callback funkcí, které jsou volány po dokončení čtení nastavení z úložiště.

Kromě inicializace některých tříd background script obsahuje registrace obslužných rutin k událostem významným pro navrženou funkcionalitu. Konkrétní zpracovávané události jsou uvedeny v příslušných podsekcích, které následují dále.

Konfigurace rozšíření

Rozšíření je konfigurováno pomocí lokálního úložiště, jak je popsáno v sekci 4.3. V sekci 4.3 je také zmíněno, že konkrétní nastavení je vytvořeno pomocí enterprise policy. Enterprise policy vytvoří nastavení zapsáním příslušných registrů na koncové stanici. Toto chování lze emulovat zápisem odpovídajících registrů, stejně jako by to udělala enterprise policy.

Pro tento účel byl vytvořen script `setSettings.ps1`, který vezme JSON soubor s názvem `settings.json` umístěný ve složce s nainstalovaným řešením a vytvoří odpovídající registry. Registry, ke kterým má rozšíření následně přístup, jsou uloženy pod registrovým klíčem `HKLM\SOFTWARE\Policies\Google\Chrome\3rdparty\extensions\<EXTENSION_ID>`. Struktura tohoto klíče pak odpovídá schématu, které je uvedené v manifestu v položce `storage.managed_schema`.

Výpis 5.1 obsahuje definované schéma pro implementované rozšíření. Formát, který by měl JSON soubor definující schéma úložiště dodržovat, je popsán v dokumentaci Google[7]. Ze zde uvedeného souboru vyplývá, že výše zmíněný registrový klíč by měl obsahovat další dva klíče s názvy `SafeStorage` a `StoragePolicySettings`. Klíč `SafeStorage` je typu pole, jehož položky jsou typu string. Tento klíč tedy může obsahovat registry typu `REG_SZ` pojmenované číslem odpovídajícím indexu v poli začínající od hodnoty 1, data v nich uložená jsou řetězce znaků. Klíč `StoragePolicySettings` je typu objekt, který má položky `clipboard`, `screenCapture` a `upload`. které mohou nabývat číselných hodnot. Rozšíření rozpozná hodnoty 0, 1 a 2, kde 0 znamená, že má být daná operace pouze zaznamenána, 1 znamená, že při incidentu bude vytvořena notifikace a 2 určuje, že má být daná operace zablokována.

```
{
    "type": "object",
    "properties": {
        "SafeStorage": {
            "type": "array",
            "items": { "type": "string" }
        },
        "StoragePolicySettings": {
            "type": "object",
            "properties": {
                "clipboard": { "type": "number" },
                "screenCapture": { "type": "number" },
                "upload": { "type": "number" }
            }
        }
    }
}
```

Výpis 5.1: Schéma `managed` úložiště

Zasílání zpráv nativnímu procesu

Zasílání nativních zpráv je implementováno statickou třídou `Messenger` v souboru `Native-Messaging.ts`. Zprávu nativnímu procesu lze odeslat pomocí metody `sendNativeMessage`.

[7] https://developer.chrome.com/docs/extensions/mv2/manifest/storage/

Tato metoda nejprve zkontroluje, zda již existuje spojení mezi rozšířením a nativním procesem a následně zprávu odešle. Pokud spojení neexistuje, je voláním metody `chrome.runtime.connectNative` vytvořen port typu `chrome.runtime.Port`. Vytvořenému portu je následně přiděleno jméno a obslužné rutiny událostí informujících o přijetí zprávy či odpojení portu. Port slouží k udržování spojení mezi nativním procesem a rozšířením. Stačí ho vytvořit pouze při první zprávě a následující zprávy jsou odesílány přes tento port bez nutnosti navazování nového spojení.

Metoda `sendNativeMessage` po vytvoření portu připraví zprávu pro odeslání. Odchozí zpráva je typu `NativeMessage`, který je také definovaný v souboru `NativeMessaging.ts`. Příprava spočívá v získání údajů o přihlášeném uživateli pomocí metody `chrome.identity.getProfileUserInfo`. Po získání těchto informací je zpráva odeslána. Zpráva obsahuje informace o typu zprávy, přihlášeném uživateli, incidentu a akci, která byla aplikována při detekci incidentu. Nativní proces odpovídá zprávou obsahující informaci, zda byl záznam úspěšně uložen do databáze a chybovou hláškou v případě neúspěchu.

Vkládání content scriptů

Vkládání content scriptů je prováděno dynamicky pomocí `chrome.tabs` API. Vkládání je prováděno v rámci obslužné rutiny události `chrome.tabs.onUpdated`. Při každé aktualizaci záložek prohlížeče je provedena kontrola, zda je otevřená stránka využívána jako bezpečné úložiště. Pokud je načtená stránka jedním z bezpečných úložišť, jsou content scripty vloženy pomocí metody `chrome.tabs.executeScript`. Tato funkce přijímá dva parametry, prvním z nich je identifikátor záložky, do které mají být scripty vloženy a druhým je objekt, který obsahuje informace o vkládaném content scriptu. Hlavní položkou tohoto objektu je položka `file`, která určuje cestu ke danému content scriptu. Další položkou je volitelná položka `runAt`, která specifikuje, kdy má být daný content script načten. Tato položka je nastavena na hodnotu `document_start` udávající, že by měl být daný content script vložen po zahájení načítání stránky.

Rozšíření si v mapě uchovává informace o tom, zda již byl content script do dané záložky vložen. V případě, že již vložen byl, není vkládání opakováno, aby nedocházelo k opakovanému hlášení stejných incidentů a tím vytváření redundantních notifikací a hlavně duplikátních záznamů o incidentech. V případě, že by rozšíření vytvářelo duplikátní záznamy, databáze by byla zbytečně zaplňována a mohlo by docházet k nepřesnému vyhodnocování bezpečnostních politik.

Upload

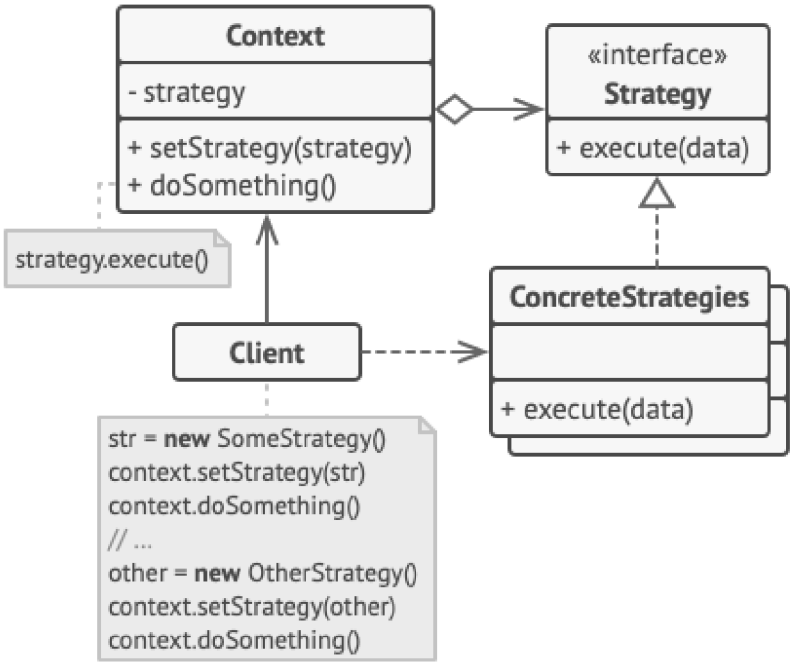
Ochrana vůči nahrávání souborů mimo bezpečná úložiště je postavena na analýze HTTP dotazů. Analýzu dotazů lze provádět pomocí obslužné rutiny události `chrome.webRequest.onBeforeRequest`. Tato událost je vytvořena před odesláním dotazu, ještě než je navázáno TCP spojení a umožňuje tedy dotaz blokovat nebo přesměrovat. Registrace obslužné rutiny události `onBeforeRequest` vyžaduje dva povinné argumenty `callback` a `filter`, při registraci lze také specifikovat volitelný argument `extraInfoSpec`.

- `callback` - funkce se vstupním argumentem typu `chrome.webRequest.WebRequestBodyDetails`, která v případě, že má být dotaz zablokován či přesměrován vrací objekt typu `BlockingResponse` nebo `undefined` v opačném případě.
- `filter` - objekt typu `RequestFilter`. Slouží k filtrování dotazů, pro které má být provedena `callback` funkce.

- `extraInfoSpec` - pole řetězců, které může obsahovat řetězce `"blocking"`, `"requestBody"` a `"extraHeaders"`
  - `"blocking"` - udává, že `callback` může vrátit `BlockingResponse`
  - `"requestBody"` - udává, že vstupní argument `callback` funkce bude obsahovat i tělo dotazu
  - `"extraHeaders"` - udává, že `callback` může upravovat hlavičky dotazu způsobem, který porušuje CORS protokol[8]

Obslužná rutina je zaregistrována s funkcí `processWebRequest` jako `callback` argument. Jako argument `filter` je předán objekt `{ urls: ['<all_urls>'] }`, který indikuje, že `callback` má být aplikován na všechny dotazy. Nakonec jako volitelný argument `extraInfoSpec` je předáno pole `["blocking", "requestBody"]`, což umožňuje blokovat dotazy a číst tělo dotazu.

Zpracování dotazů je postaveno na návrhovém vzoru strategie, který je využíván k poskytnutí možnosti volby různé implementace stejného algoritmu. Návrhový vzor strategie je tvořen kontextem zpracování a množinou konkrétních strategií, které implementují společné rozhraní. Uživatel mění aktuální strategii podle potřeby. Na obrázku 5.1 je uvedena struktura návrhového vzoru strategie. Třída `Context` na obrázku 5.1 je implementována



Obrázek 5.1: Struktura návrhového vzoru strategie, převzato[9]

třídou `UploadPageContext`. Tato třída si uchovává aktuálně nastavenou strategii v proměnné `page` typu `UploadPage`. Nová strategie je zvolena pokaždé, když je v prohlížeči otevřena nová záložka a navštívená stránka je jednou z podporovaných stránek. Jednotlivé strategie jsou implementovány třídami, které dědí od abstraktní třídy `UploadPage`. Konkrétní strategie jsou popsány níže v této podsekci. Třída `UploadPageContext` si v proměnné

[8] https://developer.chrome.com/docs/extensions/reference/webRequest/#life-cycle-of-requests
[9] https://refactoring.guru/design-patterns/strategy

`pageContexts` uchovává reference na instance strategií pro jednotlivé záložky prohlížeče. V případě, že je aktivní záložka přepnuta na jinou existující záložku a strategie pro ni již byla vytvořena, není vytvářená nová, ale na základě identifikátoru záložky je aktuální strategie přenastavena. V případě, že je záložka zavřena a existovala pro ni instance strategie, tato instance je uvolněna. Přepínání a mazání strategii je realizováno pomocí obslužných rutin událostí `chrome.tabs.onActivated` a `chrome.tabs.onRemoved`. Obslužné rutiny těchto událostí jsou implementovány pomocí metod `switchUploadpageContext` a `removeUploadPageContext` třídy `UploadPageContext`.

Jako rozhraní pro strategii slouží abstraktní třída `UploadPage`. Třída implementující `UploadPage` musí doplnit hodnoty pro vlastnosti `UploadUrls`, `UploadMethod` a `Name` a implementaci metody `getUploadData`.

- `UploadUrls` - pole hodnot typu `string | RegExp`, udává URL, na které daná stránka odesílá dotazy s nahrávanými soubory,
- `UploadMethod` - řetězec specifikující metodu použitou v dotazu provádějícím upload souboru, obvykle POST či PUT,
- `Name` - název dané strategie, slouží pro porovnávání typu strategie a pro identifikaci ve výpisech,
- `getUploadData` - metoda, která získává z dotazu data o nahrávaném souboru, např. název souboru, umístění na disku či obsah souboru.

Třída `UploadPage` poskytuje třídám, kter od ní dědí, implementaci metod

- `processRequest` - zpracuje dotaz, který obsahuje nahrávaný soubor na základě nastavené politiky,
- `blockOperation` - zablokuje dotaz, vytvoří notifikaci o zablokování a vytvoří záznam do databáze,
- `notify` - vytvoří notifikaci a záznam do databáze, dotaz je povolen a nahrávání proběhne bez problému,
- `logOperation` - pouze vytvoří záznam do databáze a
- `containsFileUpload` - rozhodne, zda dotaz obsahuje nahrávání souboru, toto je prováděno na základě `UploadUrls` a `UploadMethod`, kvůli problému ve `webRequest` API popsaném v podsekci 5.2.2

Pro přidání nové strategie je potřeba vytvořit třídu, která bude dědit od `UploadPage` a implementovat potřebné metody. Po dokončení implementace, je potřeba přidat novou strategii mezi podporované stránky. Podporované stránky si uchovává `UploadPageContext` v mapě `supportedUploadPages`, nově vytvořenou strategii do ní lze přidat v metodě `initSupportedUploadPages`.

Vytváření strategie pro každý cloud či službu pro sdílení souboru a dalšího obsahu zvlášť je nutné z toho důvodu, že kvůli neúplné podpoře `requestBody` ve `webRequest` API (viz podsekce 5.2.2) nelze spolehlivě získat obsah těla dotazu a určit, zda dotaz obsahuje soubor s citlivými daty. Implementované strategie se vztahují na cloudové úložiště Google Drive a na služby pro sdílení obsahu *uloz.to* a *uschovna.cz*. Tyto tři služby byly vybrány z toho důvodu, že demonstrují různý obsah položky `requestBody` v detailu dotazu. V případě

uloz.to tato položka obsahuje umístění souboru na disku. V případě *uschovna.cz* tělo dotazu obsahuje pouze položku `error`, která neobsahuje žádná data. V případě Google Drive se podařilo získat samotný obsah nahrávaného souboru, ale to až po zpracování binárních dat z těla zachyceného dotazu. Z pozorování nahrávání souborů na Google Drive se zdá, že je prováděno přes proxy. Dotaz zachycený v prohlížeči totiž využívá metodu PUT a jeho tělo obsahuje další dotaz s metodou POST a tělem obsahujícím samotný soubor. Obsah souboru ve vnořeném dotazu je zakódován pomocí `base64` kódování. Po dekódování je možné číst samotný obsah nahrávaného souboru.

Kopírování do schránky

K poskytnutí ochrany vůči kopírování do schránky je zapotřebí content script `clipboard.js`, který informuje background script o tom, že došlo ke kopírování obsahu ze stránky a že by na to měl background script reagovat. Content script zasílá zprávy background scriptu pomocí `chrome.runtime.sendMessage` metody, v momentě zachycení události `copy`, která je vygenerována právě tehdy, když uživatel iniciuje kopírování přes uživatelské rozhraní prohlížeče. Odeslaná zpráva obsahuje jméno scriptu, který tuto zprávu odeslal. Background script na přijaté zprávy reaguje obslužnou rutinou události `chrome.runtime.onMessage`. Obslužná rutina této události je implementována funkcí `onMessageHandler` v souboru `Messaging.ts`. Funkce `onMessageHandler` zpracovává všechny přijaté zprávy nehledě na to, ze kterého content scriptu přijdou. Tato funkce vybere příslušnou obslužnou rutinu podle toho, který script zprávu odeslal, což zjistí z přijaté zprávy. Následně provede zvolenou obslužnou rutinu.

Obslužná rutina pro script `clipboard.js` je implementována funkcí `onCopyHandler` v souboru `Clipboard.ts`. Tato funkce si nejprve přečte jaký režim politiky je nastaven pro kopírování do schránky a podle toho buď pouze vytvoří záznam o neoprávněném kopírování nebo také zobrazí notifikaci uživateli a nebo společně s vytvořením záznamu a zobrazením notifikace kopírování zablokuje.

Vytváření záznamu o incidentu je prováděno stejným způsobem jako v případě nahrávání souborů pomocí nativního procesu a zasíláním zpráv mezi rozšířením a nativním procesem. Vytváření notifikací je implementováno statickou třídou `Notifications`, která poskytuje metodu `showNotifications`. Tato metoda přijímá vstupní argumenty `mode` a `operation`, jejichž hodnota určuje obsah zobrazené notifikace. Blokování kopírování do schránky je prováděno přepisováním obsahu schránky.

Manipulace a čtení obsahu schránky je implementováno pomocí zastaralého `document.execCommand` API, viz podsekce 5.2.3. Tento přístup je závislý na přítomnosti DOM, který poskytuje background page `background.html`. Do těla background page je vložen element `textarea` s atributem `id`, který má hodnotu `"sandbox"`. Čtení schránky tedy spočívá v získání elementu uvnitř JavaScript kódu pomocí funkce `document.getElementById`. Hodnota tohoto elementu je nastavena na prázdný řetězec a pomocí metody `select` je označen obsah elementu a následně pomocí `document.execCommand("paste")` je obsah schránky vložen do textového pole a jeho hodnota načtena do proměnné. Toto je implementováno funkcí `getClipboardContent`. Zápis do schránky je implementován funkcí `setClipboardContent` a spočívá ve stejném přístupu jako v případě čtení, pouze se změnou při volání funkce `execCommand`, kde je v tomto případě jako první argument příkaz použit `"copy"`, který zapíše data do schránky. Tento princip byl převzat[10] a adaptován pro využití za účelem blokování kopírování do schránky.

[10] https://gist.github.com/srsudar/e9a41228f06f32f272a2

Pořizování snímků okna Google Chrome

Ochrana vůči pořizování snímků okna Google Chrome je založena na stejném principu jako ochrana vůči kopírování do schránky. Stejně jako v případě kopírování do schránky, je snímek okna Google Chrome pořízený pomocí klávesy `PrintScreen` uložen do schránky. Při načtení bezpečného úložiště je do stránky vložen content script monitorující událost `keyup`, která je vygenerována při uvolnění klávesy klávesnice. Pokud tuto událost vyvolala klávesa `PrintScreen`, content script odešle zprávu background scriptu. Background script zpracuje přijetí této zprávy na základě nastavené politiky. V případě, že je nastavená politika v režimu `Block`, je schránka přepsána pomocí funkce `setClipboardContent`. Notifikace a log o incidentu je vytvořen stejným způsobem jako při kopírování do schránky, který je uveden výše v podsekci „Kopírování do schránky“.

Tento způsob měl sloužit jako záložní mechanismus. Primární mechanismus pro blokování snímků okna Google Chrome měl fungovat pomocí Microsoft API `node-uwp`, které umožňuje pro dané okno nastavit proměnnou `IsScreenCaptureEnabled`. Okno s touto proměnnou nastavenou na hodnotu `false`, bude na snímku obrazovky zobrazené jako černý obdélník. Nicméně se mi nepodařilo toto API zprovoznit a proto byl využit pouze záložní mechanismus pro blokování snímků okna Google Chrome.

5.3.2 Nativní proces

Nativní proces je nastartován rozšířením při otevření portu a ukončen v momentě, kdy je port uzavřen. Rozšíření zasílá zprávy nativnímu procesu přes standardní vstup a naopak nativní proces zasílá zprávy rozšíření přes standardní výstup procesu. Nativní proces ve smyčce čeká na zprávy od rozšíření a po přijetí zprávy vytvoří dotaz na vložení záznamu do databáze.

Pro komunikaci s SQLite databází je využíván NuGet balíček `Microsoft.Data.Sqlite`. Komunikaci s databází zaštiťuje statická třída `Logger`, která implementuje návrhový vzor singleton. Vzor singleton je využit, aby nedocházelo k vytváření více spojení jedním procesem. `Logger` si uchovává instanci třídy `LoggerBase`, která provádí samotnou komunikaci s databází. `LoggerBase` při vytváření logu metodou `CreateLog` vytvoří spojení s databází, jejíž umístění získá ze souboru s nastavením `settings.json` a následně vloží záznam do databáze.

Po otevření spojení s databází je vytvořen log na základě typu přijaté zprávy. Jednotlivé typy logů jsou reprezentovány příslušnou třídou, která dědí od bázové třídy `LogBase`. Všechny třídy dědící od `LogBase` musí implementovat abstraktní metodu `GetInsertCommand`, která vrací dotaz pro vložení záznamu do příslušné tabulky. Třída `LogBase` poskytuje implementaci metody `Insert`. V rámci metody `Insert` je vytvořen záznam v tabulce `user_activity`, který je společný pro všechny typy logů a následně záznam pro konkrétní incident. Společné informace jsou ID uživatele, email uživatele, čas výskytu incidentu a vykonaná akce.

Protokol komunikace

Jak již bylo zmíněno výše, rozšíření předává zprávy na standardní vstup procesu. Data na vstupu mají předepsaný formát. Každá zpráva začíná čtyřmi bajty, které udávají délku přijímané zprávy. Samotná zpráva je řetězec kódovaný pomocí UTF8 obsahující serializovaný JSON objekt. Čtení, odesílání a zpracovávání zpráv je prováděno třídou `MessageHelper`, která využívá knihovnu `System.Text.Json` pro práci se soubory typu JSON.

Čtení zpráv je implementováno metodou `ReadMessage` třídy `MessageHelper`, která v případě úspěšné deserializace vrací instanci třídy `Message`, která reprezentuje přijatou zprávu.

Odesílání zpráv je realizováno metodou `WriteMessage`, která přijímá instanci třídy `ResponseMessage` jako vstupní argument. Vstupní zpráva je serializována a následně jsou na standardní výstup zapsány čtyři bajty udávající délku odesílané zprávu a serializovaný řetězec se zprávou.

Zpracování přijatých zpráv spočívá ve vytvoření logu metodou `CreateLog`, která přijímá instanci třídy `Message` jako vstupní argument a také dva výstupní argumenty `result` a `message`, které obsahují výsledek vložení záznamu do databáze a případně chybovou hlášku. V případě, že metoda `CreateLog` odchytí výjimku, která nesouvisí s SQLite je `result` nastaven na `false` a `message` na chybovou hlášku ze zachycené výjimky. V každém případě je rozšíření odeslána zpráva informující o výsledku operace.

5.3.3 Databáze

Databáze byla implementována podle schématu uvedeného v sekci 4.2.4. Script `db_init.sql` obsahuje kód v jazyce SQLite, který nejprve vytvoří tabulky databáze podle zmíněného schématu. Všechny tabulky mají sloupec pojmenovaný `Id`, jehož hodnoty slouží jako primární klíč. Tento klíč je automaticky inkrementován při vložení nového záznamu. Tabulky `operation_upload`, `operation_clipboard_copy` a `operation_screen_capture` obsahují sloupec `Activity`, který slouží jako cizí klíč. Tento klíč odkazuje na řádek v tabulce `user_activity`. V případě smazání záznamu v tabulce `user_activity` je díky `ON DELETE CASCADE` akci smazán také záznam, jehož cizí klíč na tento záznam odkazuje.

5.4 Instalace

Pro instalaci řešení je nezbytné nejdříve vytvořit soubor s databází pomocí batch scriptu `init_db.bat`. Tento script očekává vstupní parametr, který určuje cestu k souboru s databází. Pokud poskytnutý adresář neexistuje, script se zeptá, zda ho má vytvořit. Pokud je zvolena možnost, aby script adresář nevytvářel, nebude vytvořen ani soubor s databází.

Následně je nutné spustit další batch script s názvem `install.bat`. Tento script nejdříve vytvoří registrový klíč `HKEY_LOCAL_MACHINE\SOFTWARE\Google\Chrome\NativeMessagingHosts\com.dlp_for_saas.native_host` a nastaví jeho výchozí hodnotu na cestu k adresáři, ve kterém se tento script nachází. Předpokládá se tedy, že tento adresář obsahuje také manifest nativního procesu.

Instalační script dále vytvoří, pokud již neexistuje, registrový klíč `HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Google\Chrome\ExtensionInstallForcelist`, který slouží pro definování rozšíření, které jsou automaticky nainstalovány do Google Chrome při dalším spuštění prohlížeče. Rozšíření nainstalovaná tímto způsobem nemohou být uživatelem prohlížeče odinstalována. Odinstalovat je lze pouze smazáním příslušných registrů. Registrový klíč `ExtensionInstalForcelist` obsahuje hodnoty pojmenované celými čísly začínajícími od hodnoty 1. Data těchto hodnot jsou ve tvaru `<extension-id;update-url`. Část `update-url` obsahuje URL, kde je dostupný balíček s rozšířením. Pokud je rozšíření hostováno v Chrome Web Store hodnota této položky bude vždy nastavena na `https://clients2.google.com/service/update2/crx`. Rozšíření bude nainstalováno při dalším spuštení Google Chrome.

Kapitola 6

Experimenty

Tato kapitola je věnována experimentům zaměřeným na ověření funkčnosti vytvořeného řešení na praktických scénářích simulujících úniky dat z organizace. Následující sekce obsahují popis možného scénáře, jehož následkem může být únik firemních informací a způsob, jakým na ně bude rozšíření reagovat.

Testování všech scénářů proběhlo s nastavením politik tak, jak je uvedeno ve výpisu 6.1.

```
{
    "DatabaseLocation": "C:\\DP\\Tests",
    "Policy":{
        "SafeStorage": [ "drive.google.com" ],
        "StoragePolicySettings": {
            "clipboard": 2,
            "screenCapture": 2,
            "upload": 2
        }
    }
}
```

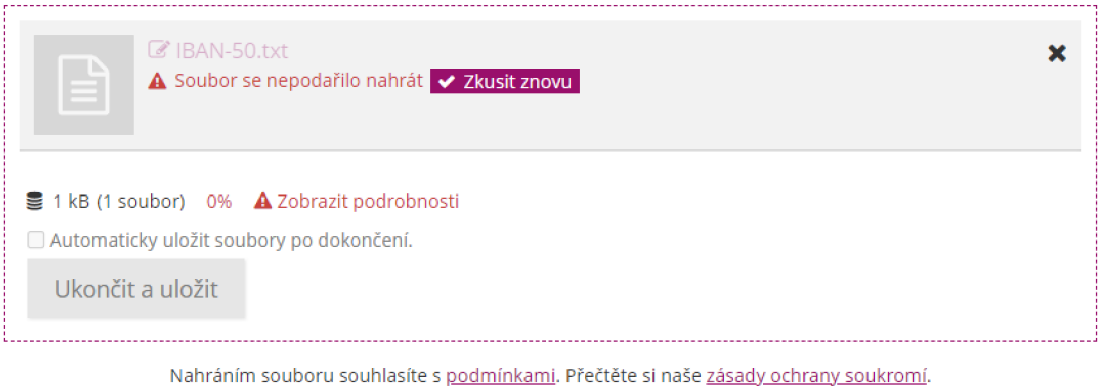
Výpis 6.1: Nastavení politik pro provádění experimentů

Toto nastavení uvádí, že soubor s SQLite databází je uložen v adresáři `C:\DP\Tests`. Jako bezpečné úložiště je používán Google Drive, jehož URL je uvedeno v položce `SafeStorage`. Bezpečnostní politiky jsou nastaveny tak, že při všech podporovaných operacích bude nahrávání, kopírování či pořízení snímku okna Google Chrome zablokováno, uživateli bude zobrazena notifikace informující o důvodu zablokování a záznam o incidentu bude uložen do databáze.

6.1 Nahrávání souborů mimo bezpečné úložiště

Tento scénář simuluje únik dat nahráním souboru s citlivými informacemi mimo bezpečné úložiště a tím úniku dat mimo bezpečnostní perimetr společnosti. V momentě, kdy zaměstnanec nahraje soubor mimo bezpečné úložiště, např. do služby pro sdílení obsahu uloz.to, dochází k úniku. Vytvořené rozšíření v případě načtení stránky uloz.to vytvoří monitorovací kontext a monitoruje dotazy, které jsou při práci s touto stránkou zasílány. V momentě, kdy je detekováno nahrávání souboru, je provedena korekční akce podle nastavené politiky.

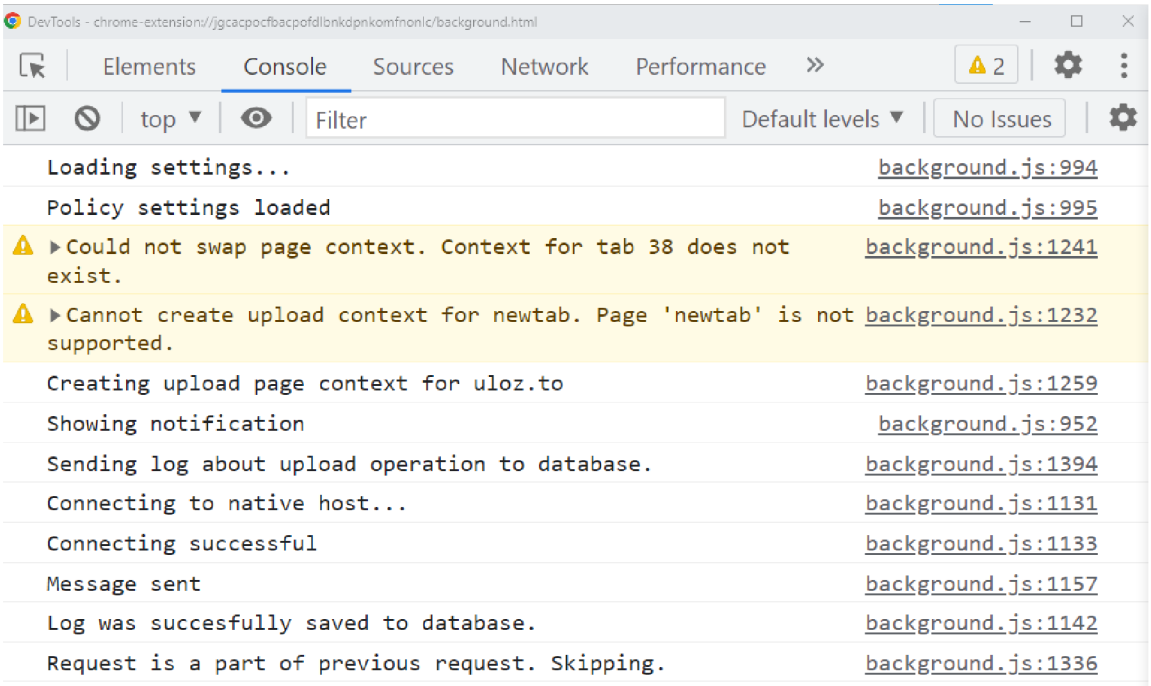
Provedení testu spočívá v načtení stránky uloz.to v prohlížeči, který má nainstalované vytvořené rozšíření a následném pokusu o nahrání souboru. Soubor lze nahrát kliknutím na tlačítko „Nahrát soubor“, následně vybrat soubor z disku či ho přetáhnout do vyznačeného pole. Rozšíření toto nahrávání, dle nastavených politik (viz výpis 6.1), zablokuje. Obrázek 6.1 obsahuje výstřižek ze stránky, kde byl upload zablokován. Uživatelské rozhraní stránky informuje o tom, že se nahrávání nezdařilo a v pravém dolním rohu je zobrazena notifikace informující uživatele o tom, že nahrávání bylo zablokováno podle nastavené bezpečnostní politiky. Na obrázku 6.3 je výpis z konzole rozšíření. První dva řádky jsou vypsány okamžitě



Obrázek 6.1: Snímek obrazovky ze stránky uloz.to se zablokovaným nahráváním

po startu rozšíření a informují o tom, že si rozšíření načetlo nastavení. Další dva řádky jsou vypsány v momentě, kdy je otevřena nová záložka. Třetí řádek informuje o tom, že pro záložku s identifikátorem 38 nebyl vytvořen kontext pro monitorování nahrávání a čtvrtý řádek informuje o tom, že pro stránku `newpage` nelze kontext vytvořit. Následně byla otevřena stránka uloz.to a pro ní byl vytvořen monitorovací kontext, viz pátý řádek. Následně byl nahrán soubor, ve výpisu je vidět, že byla vytvořena notifikace a log byl uložen do databáze. Pro uložení logu do databáze bylo vytvořeno spojení s nativním procesem a log byl zapsán do databáze. Záznam logu v databázi je uveden na obrázku 6.2. Poslední řádek výpisu konzole říká, že byl dotaz přeskočen. Tento výpis je zobrazen, protože po zablokování dotazu je dotaz opakován a pro zablokování nahrávání musejí být zablokovány i opakované dotazy. Přeskočení dotazu, znamená že je zablokován, ale již není vytvořena další notifikaci ani záznam v databázi.

Rozšíření zablokovalo nahrávání souboru, čímž bylo zabráněno úniku informací, uživateli byla zobrazena notifikace a informace o výskytu incidentu byly uloženy do databáze. Rozšíření se tedy zachovalo tak, jak bylo navrženo.



Obrázek 6.2: Snímek konzole rozšíření během nahrávání souboru

	UserId	UserEmail	TimeStamp	Operation	ActionTaken	lenarr	Url
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	113897...	dvorak.jan.dev@gmail.com	2022-05-15T16:03:03....	Upload	Blocked	C:...	https://upload.uloz.to/v1/file

Obrázek 6.3: Záznam v databázi o zablokovaném nahrávání na stránku uloz.to

6.2 Kopírování dat do schránky

Tento experiment ověřuje správnou funkcionalitu ochrany vůči kopírování dat do schránky. Test spočívá v otevření stránky s bezpečným úložištěm, po dokončení načítání stránky by měly být inkjektovány content scripty. V Google Drive je připraven testovací soubor, po otevření tohoto souboru a pokusu zkopírovat data by mělo být kopírování zablokováno a schránka by měla obsahovat řetězec `Copy blocked`. Dále by měla být vytvořena notifikace informující uživatele o tom, že bylo kopírování zablokováno a měl by být vytvořen záznam o incidentu.

Na obrázku 6.4 je výpis z konzole rozšíření zachycený při načítání bezpečného úložiště. První čtyři řádky jsou stejné jako na obrázku 6.2 a jsou popsány v sekci 6.1. Pátý až osmý řádek informuje o tom, že content scripty `clipboard.js` a `screenCapture.js` byly úspěšně vloženy. Poslední řádek informuje o tom, že bylo rozpoznáno bezpečné úložiště a nahrávání souborů nebude blokováno.

Na obrázku A.1 je snímek obrazovky s otevřeným PDF souborem, který obsahuje vygenerovaná čísla bankovních účtů. Po označení dat v souboru a vyvolání kopírování do schránky klávesovou zkratkou `Ctrl + c` nebo přes kontextové menu probíhá podobná procedura jako v případě nahrávání souboru. Operace je zablokována přepsáním obsahu schránky, byla vytvořena notifikace a záznam v databázi. Na obrázku A.2 je vidět záznam uložený do databáze po provedení této operace.

Obrázek 6.4: Výpis konzole rozšíření při načtení bezpečného úložiště

6.3 Pořizování snímků obrazovky

Test pořizování snímku okna prohlížeče probíhá stejným způsobem jako v případě testu kopírování do schránky, proto zde není popsán dopodrobna, ale jsou zde vloženy pouze obrázky A.3 a A.4 zachycující činnost rozšíření po vyvolání operace snímku obrazovky pomocí klávesy `PrintScreen`.

Kapitola 7

Závěr

Cílem této diplomové práce bylo nastudovat témata související s ochranou firemních dat a vývojem rozšíření pro webové prohlížeče, vytvoření návrhu rozšíření, které poskytuje ochranu firemních dat uložených v úložištích typu SaaS, implementace navrženého řešení a provedení experimentů, které ověřují vliv vytvořeného řešení na bezpečnost informací na praktických scénářích. Výsledkem diplomové práce je technická zpráva složená z kapitol obsahujících teoretické zpracování výše zmíněných témat, vypracovaný návrh konfigurovatelného rozšíření, dokumentace implementace řešení a popis provedených experimentů s jejich výsledky.

V práci se mi podařilo navrhnout konfigurovatelné řešení poskytující ochranu proti ztrátě dat, jehož součástí je ochrana proti nahrávání souborů mimo bezpečná úložiště, ochrana vůči kopírování dat ze souborů uložených v bezpečném úložišti pomocí systémové schránky, ochrana proti pořizování snímků okna prohlížeče Google Chrome a systém pro zaznamenávání a uchovávání informací o případných porušeních nastavených v rámci politik bezpečnosti informací.

Ochrana proti nahrávání souborů je k dispozici pouze na stránkách, pro které je implementována. Z důvodu komplikací zmíněných v sekci 5.2 nebylo možné vytvořit generické řešení nahrávání souborů na libovolné stránky. Přidání implementace pro další stránky či služby je díky návrhu snadné a rozšíření je snadno rozšiřitelné. Nicméně s množstvím existujících cloudových úložišť a služeb pro sdílení souborů a dalšího obsahu toto řešení neškáluje velmi dobře. Pro dosažení kompletní ochrany by tedy bylo nutné vytvořit vlastní implementaci pro všechny stránky, kam lze nahrávat soubory.

Ochrana vůči snímkům obrazovky funguje s určitými omezení a sice, aby bylo pořízení snímku zablokováno musí být okno prohlížeče aktivní, neposkytuje tedy ochranu při použití nástrojů jako jsou Windows Snipping Tool či pořizování snímku obrazovky pomocí telefonu či jiných zařízení.

Výstupem implementační části práce je rozšíření schopné blokování kopírování obsahu zobrazeného v okně prohlížeče Google Chrome, součástí rozšíření je také prototyp, který pro vybrané stránky poskytuje ochranu vůči nahrávání souborů mimo bezpečné úložiště. Rozšíření dále obsahuje modul pro ochranu vůči pořizování snímků obrazovky a modul pro zasílání zpráv nativnímu procesu, který zařizuje ukládání informací o incidentech do databáze. Ochrana proti nahrávání souborů a pořizování snímků obrazovky obsahuje výše zmíněná omezení. Součástí řešení je dále také nativní aplikace, která slouží pro zpracování zpráv od rozšíření a ukládání záznamů do databáze. V neposlední řadě byl vytvořen script pro inicializaci SQLite databáze a scripty pro instalaci a odinstalaci celého řešení. Rozšíření bylo nahráno do Chrome Web Store, kde bude podrobeno kontrole ze strany Google a následně bude

dostupné na adrese Chrome Web Store `https://chrome.google.com/webstore/detail/dlp-for-saas/jhmbhhblimfofhkblileomhbafhlldpk`.

Provedené experimenty ověřily, že navržená funkcionalita byla správně implementována a rozšíření se chová podle očekávání se známými omezeními uvedenými v sekcích 5.2 a 5.3.1.

Vývoj nástroje poskytujícího ochranu proti ztrátě firemních dat formou rozšíření webového prohlížeče se ukázal býti nepraktický z důvodu omezených možností poskytovaných vývojářům webových rozšíření. Tento problém je o to závažnější s příchodem manifestu verze 3, který ještě více omezuje možnosti vývojářů, např. nahrazením `webRequest` API deklarativním API, které neumožňuje inspekci HTTP dotazů, úplným zrušením podpory blokujícího `webRequest` API, nepřístupným `clipboard` API, atd., uvedené jsou pouze příklady relevantní pro vytvořené řešení a daný případ použití. To je pravděpodobně jedním z hlavních důvodů proč i existující řešení firem jako např. Microsoft, Symantec či McAfee využívají rozšíření pouze jako monitorovací komponentu robustnějšího řešení a vynucování politik obstarává vlastní infrastruktura.

Největší nedostatky vytvořeného řešení spočívají v nespolehlivém přístupu k tělu jednotlivých dotazů a tím nemožnosti čtení obsahu nahrávaných souborů pomocí `webRequest` API, toto by muselo být opraveno či změněno na straně Google, aby bylo možné vytvořit generické řešení, které by analyzovalo obsah nahrávaných souborů a na jeho základě rozhodovalo, zda soubor obsahuje důvěrná firemní data a zda je potřeba zablokovat jeho nahrávání či nikoli. Dalšími možnými rozšířeními stávajícího řešení je možnost přidat dodatečné kanály, kterými mohou data uniknout ven z organizace, např. pomocí webových emailových klientů nebo „instant messaging“ aplikací přístupných přes webový prohlížeč.

Literatura

[1] *2020 Year End Report Data Breach QuickView Report*. Risk Based Security, Inc, 2021.

[2] Apostu, A., Puican, F., Ularu, G., Suciu, G., Todoran, G. et al. Study on advantages and disadvantages of Cloud Computing–the advantages of Telemetry Applications in the Cloud. *Recent advances in applied computer science and digital services*. 2013, sv. 2103.

[3] Birthare, S. K. a Sharma, R. Study on migration of on-premise ERP to SaaS product. *International Research Journal of Modernization in Engineering Technology and Science*. 2020, sv. 2.

[4] *C-311/18 - Facebook Ireland and Schrems*. The Court of Justice of the European Union, červenec 2020. Navštíveno 13.4. 2022. Dostupné z: https://curia.europa.eu/juris/liste.jsf?num=C-311/18.

[5] Cheng, L., Liu, F. a Yao, D. D. Enterprise data breach: causes, challenges, prevention, and future directions. *WIREs Data Mining and Knowledge Discovery*. 2017, sv. 7, č. 5, s. e1211. DOI: https://doi.org/10.1002/widm.1211. Dostupné z: https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/widm.1211.

[6] *ČSN ISO/IEC 27000 (36 9790) Informační technologie - Bezpečnostní techniky - Systémy řízení bezpečnosti informací - Přehled a slovník*. Praha: Úřad pro technickou normalizaci, metrologii a státní zkušebnictví, 2010.

[7] *ČSN ISO/IEC 27001 (36 9797) Informační technologie - Bezpečnostní techniky - Systémy řízení bezpečnosti informací - Požadavky*. Praha: Úřad pro technickou normalizaci, metrologii a státní zkušebnictví, 2014.

[8] Developers, C. *Architecture overview*. Květen 2021. Navštíveno 16.5. 2022. Dostupné z: https://developer.chrome.com/docs/extensions/mv3/architecture-overview/.

[9] Dillon, T., Wu, C. a Chang, E. Cloud Computing: Issues and Challenges. In: *2010 24th IEEE International Conference on Advanced Information Networking and Applications*. 2010, s. 27–33. DOI: 10.1109/AINA.2010.187.

[10] Garuba, M., Liu, C. a Washington, N. A Comparative Analysis of Anti-Malware Software, Patch Management, and Host-Based Firewalls in Preventing Malware Infections on Client Computers. In: *Fifth International Conference on Information Technology: New Generations (itng 2008)*. 2008, s. 628–632. DOI: 10.1109/ITNG.2008.233. ISBN 978-0-7695-3099-4.

[11] Gul, I., Rehman, A. ur a Islam, M. H. Cloud computing security auditing. In: *The 2nd International Conference on Next Generation Information Technology*. 2011, s. 143–148.

[12] Han, P., Liu, C., Cao, J., Duan, S., Pan, H. et al. CloudDLP: Transparent and Scalable Data Sanitization for Browser-Based Cloud Storage. *IEEE Access*. 2020, sv. 8, s. 68449–68459. DOI: 10.1109/ACCESS.2020.2985870.

[13] Hill, D. G. *Data protection : governance, risk management, and compliance*. 1. vyd. Boca Raton: CRC Press, 2019. ISBN 978-0-367-38533-0.

[14] *How to Secure Your Network Using IDS/IPS Tools*. May 2020. Dostupné z: https://www.tek-tools.com/security/best-ids-and-ips-tools.

[15] Kandias, M., Virvilis, N. a Gritzalis, D. The Insider Threat in Cloud Computing. In: *Critical Information Infrastructure Security*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, s. 93–103. Lecture Notes in Computer Science. ISBN 9783642414756.

[16] Liu, S. a Kuhn, R. Data loss prevention. *IT professional*. IEEE. 2010, sv. 12, č. 2, s. 10–13.

[17] Marešová, S. *Hodnocení výkonnosti a kvality společnosti dle normy ISO*. Vysoké učení technické v Brně. Fakulta podnikatelská, 2017.

[18] Metalidou, E., Marinagi, C., Trivellas, P., Eberhagen, N., Skourlas, C. et al. The Human Factor of Information Security: Unintentional Damage Perspective. *Procedia, social and behavioral sciences*. Elsevier Ltd. 2014, sv. 147, s. 424–428. Procedia - Social and Behavioral Sciences. ISSN 1877-0428.

[19] Oppliger, R. Internet Security: Firewalls and Beyond. *Commun. ACM*. New York, NY, USA: Association for Computing Machinery. may 1997, sv. 40, č. 5, s. 92–102. DOI: 10.1145/253769.253802. ISSN 0001-0782. Dostupné z: https://doi.org/10.1145/253769.253802.

[20] Sandhu, R. a Samarati, P. Authentication, access control, and audit. *ACM Computing Surveys (CSUR)*. ACM New York, NY, USA. 1996, sv. 28, č. 1, s. 241–243.

[21] Sandhu, R. S. On Five Definitions of Data Integrity. In: Citeseer. *DBSec*. 1993, s. 257–267.

[22] Stallman, R. *Who does that server really serve?* [online]. Březen 2010. Navštíveno 13.2.2022. Dostupné z: https://bostonreview.net/articles/richard-stallman-free-software-drm/.

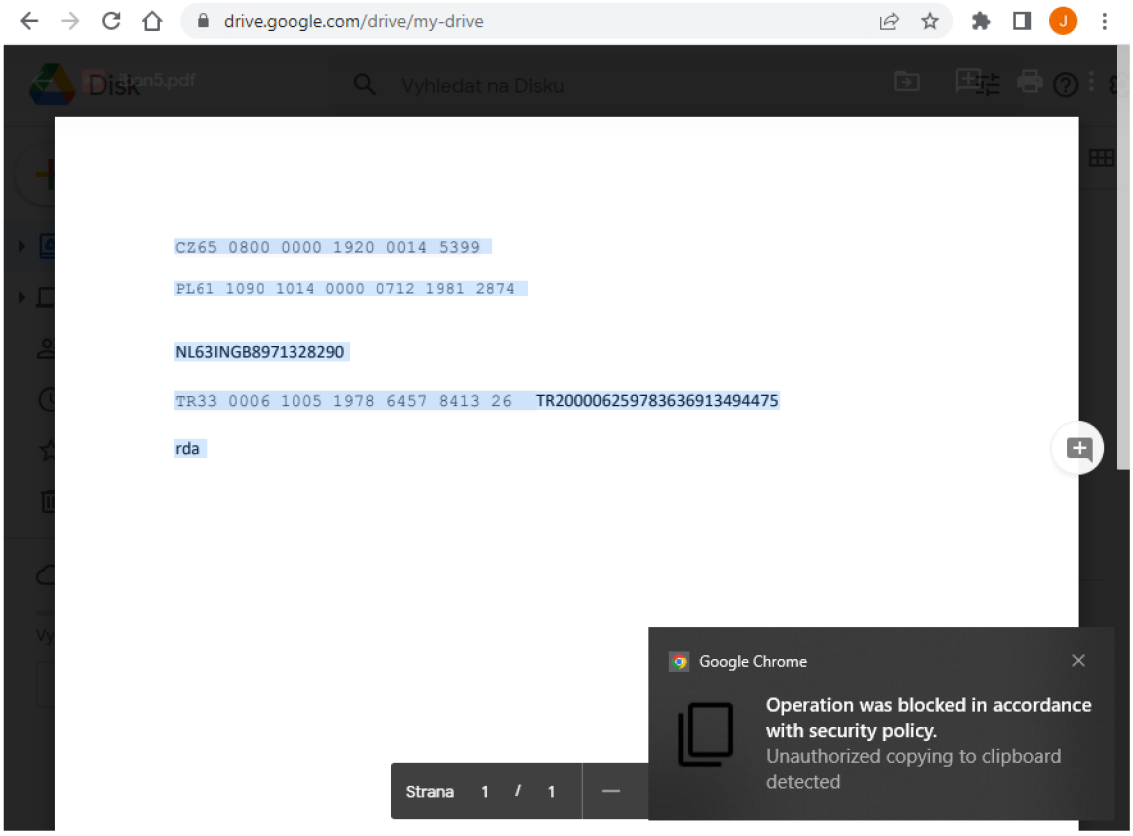
[23] Stamp, M. *Information security: principles and practice*. 2. vyd. John Wiley & Sons, 2011. ISBN 978-0-470-62639-9.

[24] Tahboub, R. a Saleh, Y. Data Leakage/Loss Prevention Systems (DLP). In: *2014 World Congress on Computer Applications and Information Systems (WCCAIS)*. 2014, s. 1–6. DOI: 10.1109/WCCAIS.2014.6916624.

[25] Zins, C. Conceptual approaches for defining data, information, and knowledge. *Journal of the American Society for Information Science and Technology.* Hoboken: Wiley Subscription Services, Inc., A Wiley Company. 2007, sv. 51, č. 4, s. 479–493. ISSN 0002-8231.

Příloha A

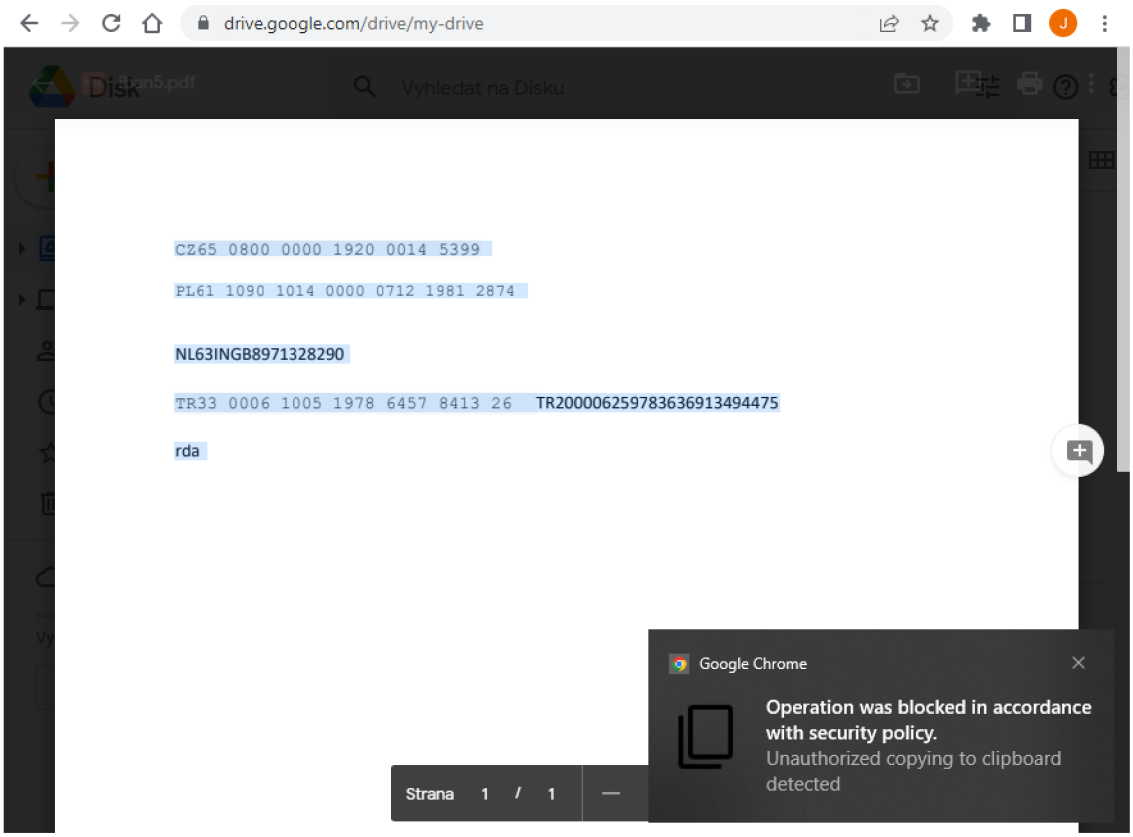
Experimenty



Obrázek A.1: Snímek obrazovky ze stránky Google Drive se zablokovaným kopírováním do schránky

UserId	UserEmail	TimeStamp	Operation	ActionTaken	Url
Filter	Filter	Filter	Filter	Filter	Filter
113897...	dvorak.jan.dev@gmail.com	2022-05-15T21:04:37...	Clipboard...	Blocked	https://drive.google.com/drive/my-drive

Obrázek A.2: Záznam v databázi o zablokovaném kopírování do schránky



Obrázek A.3: Snímek obrazovky ze stránky Google Drive se zablokovaným pořízením snímku obrazovky

UserId	UserEmail	TimeStamp	Operation	ActionTaken	Url
Filter	Filter	Filter	Filter	Filter	Filter
113897...	dvorak.jan.dev@gmail.com	2022-05-15T22:18:26...	ScreenCapture	Blocked	https://drive.google.com/drive/my-drive

Obrázek A.4: Záznam v databázi o zablokovaném pořizování snímku obrazovky

Příloha B

Struktura přiloženého nosiče

```
README.md
DlpForSaasSrc
    extension - zdrojové soubory rozšíření
        ...
        public - zdrojové soubory nepotřebné k překladu TS na JS
        src - zdrojové soubory v jazyce TS
    NativeHost - zdrojové soubory nativního procesu
DlpForSaasPackage.zip - instalační scripty a zkompilovaný nativní proces
Technická zpráva
    DP-xdvora2m.pdf
    DP-xdvora2m.zip - Latex soubory pro vysázení technické zprávy
```