VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

FORENZNÍ ANALÝZA DISKŮ A METADAT POD OS LINUX

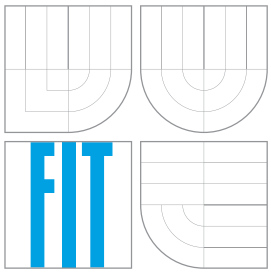
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

JITKA KOCNOVÁ

BRNO 2014

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

FORENZNÍ ANALÝZA DISKŮ A METADAT POD OS LINUX

FORENSIC ANALYSIS OF DISCS AND METADATA UNDER OS LINUX

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — JITKA KOCNOVÁ
AUTHOR

VEDOUCÍ PRÁCE — Ing. PAVEL OČENÁŠEK, Ph.D.
SUPERVISOR

BRNO 2014

Abstrakt

Tato práce se věnuje forenzní analýze paměťových prvků a metadat. Její součástí je tvorba linuxové aplikace zaměřené na obnovení dat z paměťových médií se souborovým systémem EXT, UDF a ISO 9660; a také několik výukových příkladů. Aplikace je vytvořená v jazyce C++ a obnovuje soubory pomocí vyhledávání jejich začátků a konců. Testování aplikace proběhlo na reálných datech a také bylo provedeno srovnání aplikace s již existujícími nástroji soustředěnými na tuto problematiku. Na základě testů bylo zjištěno, že aplikace dokáže uspět i v případech, kdy se nedají k obnově souborů využít informace ze žurnálu (tak pracují některé jiné aplikace).

Abstract

This bachelor's thesis is about forensic analysis of memory devices and their metadata. It's part is also an application for Linux system which focuses on restoring data from memory devices using EXT, UDF and ISO 9660 file systems; and also some examples for educational use. The application was written in the C++ language and restores files by searching for their start and end tags. The application was tested on real data and it was also compared with already existing similar applications. As a result, it was found out that the application is successful even if there is no chance to work with informations stored in journal file of EXT file system, that is used by some of the other programs the application was compared with.

Klíčová slova

EXT2, EXT3, EXT4, FAT, forenzní analýza, ISO 9660, i-uzel, Linux, metadata, NTFS, obraz disku, obnova dat, paměťová zařízení, superblok, UDF.

Keywords

EXT2, EXT3, EXT4, FAT, forensic analysis, ISO 9660, i-node, Linux, metadata, NTFS, disk image, data restoring, memory devices, superblock, UDF.

Citace

Jitka Kocnová: Forenzní analýza disků a metadat pod OS Linux, bakalářská práce, Brno, FIT VUT v Brně, 2014

Forenzní analýza disků a metadat pod OS Linux

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracovala samostatně pod vedením pana Ing. Pavla Očenáška, Ph.D. Uvedla jsem všechny literární prameny a publikace, ze kterých jsem čerpala.

.....
Jitka Kocnová
19. května 2014

Obsah

Seznam obrázků

Seznam tabulek

Kapitola 1

Úvod

Výpočetní technika urazila od svého zrodu velký kus cesty a stala se přirozenou součástí našeho života. Slouží nám při práci, zábavě, vzdělávání; neobešly by se bez ní zdravotnictví, armáda, mediální prostředky a mnohá další odvětví. Uplatnění našla bohužel i v kriminalitě — ať už jako cíl trestné činnosti nebo prostředek k jejímu páchání.

Podobně, jako se vyšetřují například vraždy nebo loupeže, probíhá vyšetřování zločinů spojených s výpočetní technikou — této oblasti se věnuje forenzní analýza.

1.1 Kybernetické zločiny z právního hlediska

V České republice je počítačová zločinnost definována jako majetkový trestný čin, který může souviset i s porušováním autorských práv, neoprávněným nakládáním s osobními údaji, nebo s trestným činem proti hospodářské kázni. Je postihována podle následujících paragrafů § 230, § 231 a § 232 trestního zákoníku, zákona č. 40/2009 Sb. V nich je možné nalézt například následující informace:
Za překonání bezpečnostního opatření, neoprávněný přístup k počítačovému systému nebo k nosiči informací je v České republice trest odnětí svobody na jeden rok, zákaz činnosti nebo propadnutí věci či jiné majetkové hodnoty.

Odnětí svobody na dva roky, zákaz činnosti nebo propadnutí věci či jiné majetkové hodnoty je trestem za neoprávněné použití dat, jejich smazání, změnu, s

Pokud je trestný čin uvedný v předchozích dvou odstavcích spáchán v úmyslu způsobení škody, újmy, získání neoprávněného prospěchu nebo omezení funkčnosti počítače, bude potrestán odnětím svobody na šest měsíců až tři roky.

Na šest měsíců smí být odňata svoboda tomu, kdo poruší povinnost vyplývající ze zaměstnání, postavení nebo funkce, a tím poškodí, zničí nebo pozmění data uložená na datovém nosiči, nebo provede zásah do programového nebo technického vybavení počítače. [5]

1.2 Forenzní analýza v informatice

Forenzní analýza je postup sloužící k vyšetřování různých subjektů souvisejících s řešením trestného činu a k získávání důkazů pro soudní řízení. Její počátky je možné nalézt v 70. letech s narůstající zločinností ve finančním sektoru. V 80. letech začaly vznikat první forenzní nástroje. V informatice má forenzní analýza mnoho odvětví, jako například

- forenzní analýza sítí,
- forenzní analýza diskových pamětí,
- forenzní analýza audia,
- forenzní analýza videa,
- forenzní analýza databází,
- forenzní analýza mobilních zařízení.[1]

Při vyšetřování je potřeba zodpovědět si několik důležitých otázek:

- Jak byl zločin spáchán?
- Jde o krádež dat nebo peněz, nebo pouze o vniknutí do systému?
- Byla poškozena něčí práva?
- Byl někdo ohrožen, obtěžován, vydírán?

Podstatné je, že úkolem forenzní analýzy není pouze například najít a obnovit skrytá nebo smazaná data, ale také zajistit, že tato data budou validní a tím pádem i použitelná pro další vyšetřování.

1.3 Linux a forenzní analýza

Linux je založen na unixovém operačním systému, jehož počátky sahají do 60. let 20. století, kdy se započalo s vývojem jádra unixového systému. Kromě Linuxu jsou na stejném principu postaveny i další operační systémy, jako je FreeBSD či MacOS od firmy Macintosh. Linux prošel dlouhým vývojem a v dnešní době je k dispozici množství jeho distribucí, lišící se grafickým prostředím, programovým vybavením a možností přizpůsobení pro různé potřeby (osobní počítače, servery, mobilní zařízení atd.). [2]

Linuxové distribuce využitelné pro forenzní analýzu mají tu výhodu, že mohou být spuštěny například z CD, DVD, nebo flash paměti bez toho, aby měnily obsah datových médií zkoumaného počítače. Navíc se dají zdarma stáhnout z internetu a mají v sobě předinstalované některé speciální nástroje sloužící pro forenzní analýzu; další programy je možné doinstalovat.

Většina forenzních nástrojů je snadno použitelná i pro vyšetřovatele, kteří nejsou do detailů seznámeni s vnitřní strukturou a činností hardwaru a softwaru počítače, a tím snižují časové nároky na průběh vyšetřování.

Nástroje forenzní analýzy jsou ovšem dvousečnou zbraní. Mohou posloužit k vyřešení kriminálního případu, ale stejně tak je možné, že se stanou pomůckami při páchání trestného činu.

1.4 Data a metadata

V této části první kapitoly budou představeny prvky, které jsou hlavními předměty zkoumání celé bakalářské práce.

Data jsou veškeré informace, které jsou uložené na nějakém paměťovém médiu v podobě souborů. Soubory se od sebe liší nejen například svou velikostí, ale především typem (textový soubor, obrázek, video, hudba, systémový soubor,...). Každý soubor má svůj název a ukazatel na metadata.

Metadata slouží k bližšímu popisu souborů, se kterými souvisí. Jinými slovy jsou to data, která popisují data. Obsahují informace o velikosti souboru, data a časy změn, a ukazatel do paměti na začátek obsahu souboru.

Metadata jsou důležitým prvkem při obnovování smazaných souborů — existuje-li možnost, jak se dostat k informacím v metadatech, je i šance, že ukazatel do paměti na začátek obsahu souboru nebyl zničen. Ovšem může nastat situace, kdy sice existuje záznam v metadatech, ale data na adrese, na kterou odkazují, byla přepsána jiným souborem nebo byla nahrazena nulami, případně náhodným shlukem dat (a tudíž nelze obsah původního smazaného souboru obnovit, případně lze získat jen jeho fragment).

Při pokusu obnovení dat pomocí jejich metadat nastává další problém, pokud při smazání souboru dojde k vynulování ukazatele do paměti nebo jeho přepsání náhodnými daty — bez prozkoumání každého sektoru disku je pak téměř nemožné najít začátek požadovaného souboru.

Kapitola 2

Paměťové prvky a souborové systémy

Tato bakalářská práce se zaměřuje na forenzní analýzu paměťových prvků pomocí linuxových nástrojů. V této kapitole proto bude čtenář seznámen se základními typy paměťových zařízení a využívaných souborových systémů.

2.1 Paměťová zařízení

V současné době můžeme rozlišit několik druhů paměťových zařízení, které se od sebe liší nejen svou kapacitou, ale hlavně vnitřní strukturou, zpracováním, způsobem ukládání informací a využívanými souborovými systémy.

2.1.1 Pevné disky

Asi nejznámnějším a v počítačích se nejčastěji vyskytujícím typem nevolatilního paměťového média[1] je magnetický pevný disk. Skládá se z jednotlivých záznamových ploten a čtecích/záznamových hlaviček pohybujících se těsně nad povrchem plotny. Na každou plotnu připadají dvě hlavičky, protože plotna smí obsahovat datový záznam z obou svých stran. Každá plotna obsahuje stopy tvořící soustředné kružnice. Cylindrem je označována množina stop nacházejících se na jednotlivých plotnách nad sebou. Stopy obsahují sektory, které jsou nejmenšími zapisovatelnými jednotkami pevného disku. Typická velikost sektoru je 512 B, u moderních disků 4 kB (s emulací 512B). Disk je možné k počítači připojit například přes rozhraní ATA, SATA, SCSI, USB, FireWire.

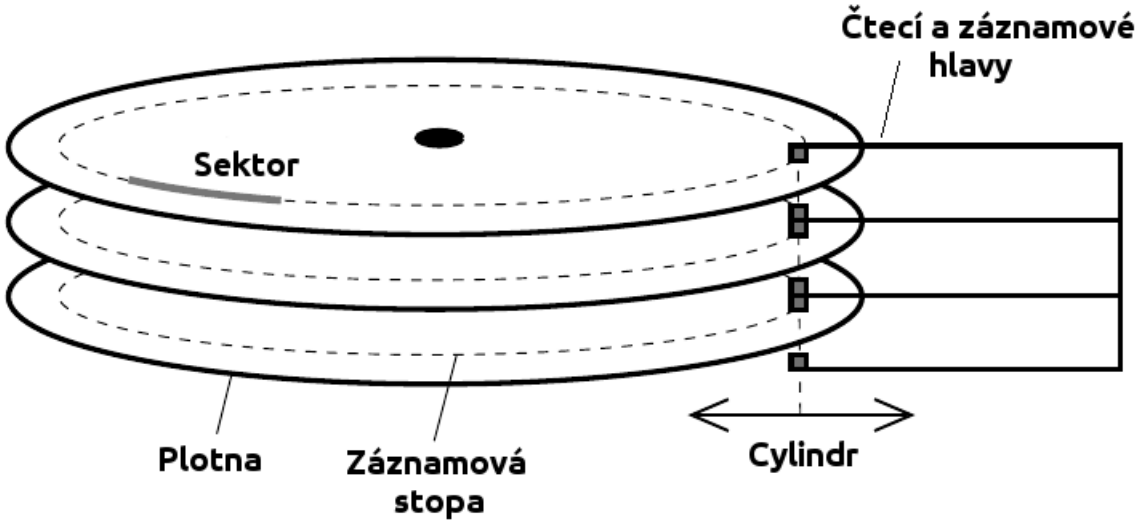
Kapacita pevného disku se dnes pohybuje ve stovkách gigabytů až jednotek terabytů.

Zápis probíhá pomocí magnetorezistivní čtecí/zápisové hlavy, která díky elektromagnetické indukci zmagnetuje místo na disku. Čtení probíhá obdobně: různě zmagnetovaná místa disku, nad kterými se hlava pohybuje, ovlivňují elektrický odpor hlavy — a tím pádem i elektrický proud, který hlavou protéká. Tento proud se dále měří a převádí do podoby požadované informace.

Magnetický způsob záznamu má ovšem i své nevýhody — čím větší hustoty záznamu chceme dosáhnout, tím pravděpodobnější je kolize mezi jednotlivými bity: záznamová místa budou tak blízko u sebe, že se navzájem mohou magneticky ovlivňovat a může tak dojít

[1] Nepotřebuje napájecí napětí pro udržení informace

k poškození dat. Řešením by v blízké budoucnosti mohla být momentálně testovaná technologie nazvaná HAMR[2], která využívá zahřátí zapisovaného místa laserem a následné ochlazení po zápisu, čímž se informace stabilizuje



Obrázek 2.1: Pevný disk[3]

Důležitými pojmy spojenými s pevnými disky v počítačích jsou tabulka rozdělení disku[4], hlavní spouštěcí záznam[5] a chráněná oblast disku[6].

Tabulka rozdělení disku obsahuje informace o rozčlenění disku na jednotlivé logické nebo fyzické oddíly a je uložena v hlavním spouštěcím záznamu v prvním sektoru pevného disku spolu se zavaděčem, který rozhoduje o tom, ze kterého diskového oddílu bude při spuštění počítače zaveden operační systém (tento se na daném oddílu samozřejmě musí vyskytovat).

Výše zmíněné diskové oddíly poskytují možnost mít na jednom disku více souborových či operačních systémů (každý v samostatném oddílu), dále pak získat prostor pro databázi nebo oddělit operační systém od dat (tím je uživatelským programům a datům poskytnuta větší ochrana například při poškození a následné nutnosti přeinstalování systému — změny se provedou pouze v oddílu s operačním systémem a data zůstanou netknutá). Oddíly je možné i skrýt pomocí modifikace tabulky rozdělení disku (daný oddíl se do ní nezahrne), a ukrýt tak citlivá data, což je jedna z technik používaných ve zločinech, které mají určitou spojitost s počítači.

Chráněnou oblastí disku rozumíme část disku, která není přístupná běžnému operačnímu systému a jeho uživateli. Původním záměrem bylo poskytovat na disku místo, které by dokázalo uchovat informace i po naformátování disku (ať už vědomém nebo následkem chyby uživatele). Je obvykle využívána výrobci počítačů pro uchovávání informací pro instalaci nebo zotavení systému; dá se ale zneužít i rootkitem[7] nebo pro skrytí citlivých

[2]Heat Assisted Magnetic Recording
[3]Převzato z: [4] (15. 11. 2013)
[4]Partition Table
[5]Master Boot Record
[6]Host Protected Area
[7]Technika maskování přítomnosti viru skrýváním jeho adresářů a instalačních souborů

či inkriminujících dat — proto je vhodné věnovat se jí při vyšetřování v rámci forenzní analýzy.[3]

2.1.2 Flash disky

Tento typ nevolatilního paměťového média, který nahradil diskety a je odvozen od EEPROM[8], není diskem v pravém slova smyslu. Pro uložení dat je využito tranzistorové pole — každá paměťová buňka je reprezentována jedním tranzistorem s plovoucím hradlem. Připojuje se přes USB konektor.
Existují dva typy flash pamětí:

- NOR FLASH — paměťové buňky lze adresovat (číst, zapisovat) samostatně, mazání se provádí po blocích (několik buněk najednou),
- NAND FLASH — buňky nejsou samostatně adresovatelné, musí se adresovat po stránkách (skupinách), mazání se provádí po blocích buněk (několik stránek najednou).

Hlavní výhodou flash pamětí jsou jejich malé rozměry, cenová dostupnost a kapacita, která se v současnosti pohybuje v řádu desítek GB.

2.1.3 CD a DVD disky

CD a DVD disky jsou datovými nosiči využívajícími optickou záznamovou techniku. Obě média ukládají data pouze z jedné strany do spirály s konstatní hustotou zápisu. Kapacita CD se pohybuje okolo 700 MB, u DVD pak kolem 4.7 GB (může mít jednu nebo dvě zápisové vrstvy, lze zapisovat i na obě strany do kapacity 17,1 GB). Pro čtení i zápis se využívá mechanika s laserovým paprskem. DVD jsou zpětně kompatibilní s CD.

CD i DVD disky používají pro správu dat obvykle následující souborové systémy:

- ISO 9660 — starší systém, nepodporuje přidávání dat na disk, při každém zápisu na CD-RW se celý disk vytváří znovu (přemaže se),
- UDF — náhrada za ISO 9660, standardizováno pro disky pouze pro čtení i přepisovatelné disky, umožňuje přidávání dalších dat u CD/DVD

Oba typy disků lze využít pro uložení audia, videa, textů, programů, jako instalační média, nebo jako tzv. live CD/DVD pro zavádění operačního systému bez nutnosti jeho instalace.

2.1.4 RAM pamět a rychlá vyrovnávací paměť

Jedná se o volatilní typ paměti, která k udržení dat potřebuje být připojena ke zdroji elektrického napětí.
RAM paměť využívá přímý přístup ke svým paměťovým buňkám — narozdíl třeba od magnetických pásek nebo CD/DVD disků, u kterých se uplatňuje sekvenční přístup. Je to rychlá paměť s kapacitou v řádu jednotek gigabytů. V praxi se používá jako dočasné úložiště dat z pevného disku po dobu, kdy je počítač zapnutý. Z pevného disku se do RAM nahrají data, se kterými se momentálně pracuje, případně i data, která leží blízko těchto dat (a je tedy velká pravděpodobnost, že budou za nějakou dobu vyžadována také). K datům se pak

[8] Elektricky mazatelná programovatelná ROM, dva tranzistory tvoří jednu paměťovou buňku

přistupuje ne přímo na disku, ale na rychlejší RAM. Po ukončení práce se data z RAM přesunou zpět na disk (provede se skutečný zápis dat na disk). Taková paměť je dnes konstuována jako DRAM[9] — jednu paměťovou buňku zastupuje jeden tranzistor. Protože je vhodné, aby data v takto sestrojené RAM vydržela určitou dobu, je nutné obnovovat informaci v ní uloženou pomocí podpůrného elektrického obvodu.

Rychlé vyrovnávací paměti se používají pro vyrovnání rozdílných rychlostí čtení z disku a RAM. Jejich kosntrukce je podobná jako u RAM, jen s tím rozdílem, že pro jednu paměťovou buňku používají dva tranzistory — jedná se o tzv. SRAM: statickou RAM, která nepotřebuje podpůrný elektrický obvod pro zachování obsažené informace. Od toho se odvíjí i jejich vyšší cena.

2.2 Souborové systémy

Souborový systém popisuje způsob uložení a organizace dat (obvykle ve formě hierarchické struktury souborů a adresářů) na disku.

2.2.1 FAT

Jeden z nejjednodušších souborových systémů vytvořila firma Microsoft původně pro použití na disketách, ale našel své uplatnění i na discích, případně flash pamětech. Dnes existují 4 typy FAT:

- FAT12 — verze využívaná pro diskety,
- FAT16 — podporuje pevné disky do velikosti 2 GB,
- FAT32 — podoporuje pevné disky do velikosti až 2 TB, využíván v operačních systémech Windows 95, 98, Me, 2000, XP, Vista,
- FATX — souborový systém pro herní platformu Xbox.

Struktura FAT systému je následující:

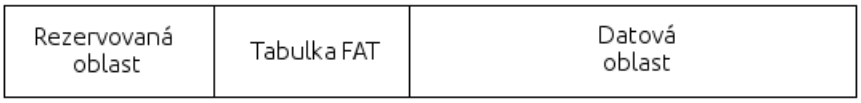
- rezervovaná oblast,
- souborová alokační tabulka,
- datová oblast.

Rezervovaná oblast začíná na prvním sektoru disku a přímo v tomto sektoru obsahuje tzv. zaváděcí sektor (boot sector). První tři byty prvního sektoru jsou vyhrazeny pro informaci o umístění kódu potřebného ke startu operačního systému.
Za rezervovanou oblastí následuje souborová alokační tabulka[10], která obsahuje záznamy o každém clusteru (viz dále) datové oblasti souborového systému.

Datová oblast tohoto souborového systému začíná na prvním sektoru za souborovou alokační tabulkou. Je tvořena clustery — skupinami za sebou následujících sektorů disku. Počet sektorů v jednom clusteru určuje záznam v zaváděcím sektoru v rezervované oblasti; vždy je to však 2^n sektorů. S clustery se pracuje pouze v datové oblasti souborového systému FAT, v ostatních dvou oblastech se využívají sektory. Ve FAT12 a FAT16 je počáteční

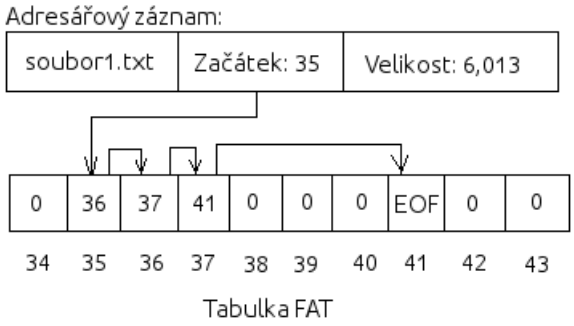
[9] Dynamická RAM
[10] FAT table

cluster datové oblasti vyhrazen pro kořenový adresář[11] ; ve FAT32 se kořenový adresář smí nacházet v libovolném clusteru (jeho umístění je dáno v zaváděcím sektrou).



Obrázek 2.2: Struktura souborového systému FAT.[12]

Informace o každém clusteru jsou dostupné v souborové alokační tabulce (viz výše). Je-li cluster volný (a je-li možné ho použít pro uložení dat), má ve svém záznamu v tabulce číslo 0. Je-li cluster poškozen a nelze ho využít, v tabulce je označen číslem `0xFF7` (FAT12), `0xFFF7` (FAT16), nebo `0x0FFF FFF7` (FAT32). Všechny ostatní hodnoty značí, že je cluster alokován pro potřeby souboru nebo složky. Těmito hodnotami jsou adresy následujících clusterů, které souvisí s jedním souborem či složkou, případně mají hodnotu konce souboru[13].[3]



Obrázek 2.3: Adresování clusterů v souborovém systému FAT.[14]

Clustery, které jsou alokované pro potřeby složky, obsahují tzv. adresářový záznam. První dva záznamy v této struktuře jsou "."(složka samotná) a ".."(rodičovský adresář složky). Dojde-li k vytvoření nové podsložky nebo souboru, pak se v adresářovém záznamu vytvoří nová položka právě pro tuto novou entitu. Každá položka obsahuje jméno souboru/podsložky, se kterou souvisí, její velikost (u složky vždy nulová), atributy (jen pro čtení, skrytý, systémový soubor, příznak, zda se jedná o složku), data i časy vytvoření a posledního přístupu a zápisu, a počáteční cluster souboru/podsložky.

2.2.2 NTFS

Tento souborový systém byl představen firmou Microsoft společně s příchodem operačního systému Windows NT jako reakce na zvyšující se kapacity pevních disků; stal se taktéž primárním souborovým systémem pro Windows Vista a je podporován většinou linuxových

[11] root directory
[12] Převzato z: [3] (15. 11. 2013)
[13] angl. EOF - End Of File
[14] Převzato z: [3] (15. 11. 2013)

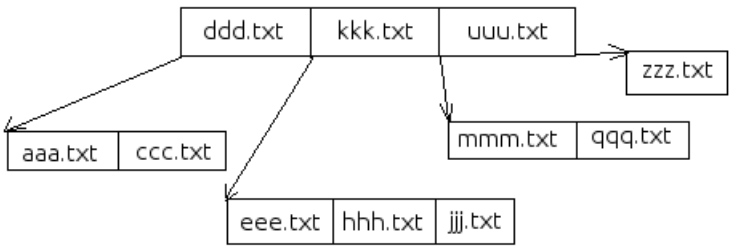
distribucí. NTFS považuje veškerá uložená data na disku za soubory.

Po naformátování disku souborovým systémem NTFS se vytváří následující části[3]:

- zaváděcí oddíl[15] - nachází se v prvním sektoru souborového systému,
- tabulka MFT[16] se záznamy o všech souborech a adresářích,
- prostor pro samotná data.

Nejdůležitější částí NTFS je tabulka MFT, která plní podobnou funkci jako tabulka FAT v souborovém systému FAT. Prvních 16 záznamů je určeno pro uložení informací o samotném souborovém systému. Následující záznamy jsou určené pro atributy jednotlivých souborů a adresářů. Každý atribut má hlavičku a obsah. Hlavička určuje typ, jméno a velikost atributu. Obsah atributu může být rezistentní nebo nerezistentní. Obsah rezistentního atributu se zaznamenává do MFT tabulky spolu s hlavičkou atributu; pro nerezistentní obsah se alokuje cluster v datové části souborového systému. Atributy taktéž uchovávají metadata (data a časy vytvoření, modifikace a posledního přístupu, modifikace záznamu v MFT tabulce; soubor je pouze pro čtení; systémový soubor; komprimovaný soubor; informace o vlastníkovi souboru).

Spolu s atributy se v adresářích využívá indexování. Indexem se rozumí skupina seřazených atributů. V praxi slouží pro snadnější a rychlejší vyhledávání souborů a využívá strukturu zvanou B-strom.



Obrázek 2.4: Struktura B-stromu v NTFS.[17]

Systém si taktéž uchovává přehled o operacích, které nad souborem proběhly (ukládání, mazání). Toto je základní vlastností tzv. žurnálování — informace o probíhající operaci se zaznamená žurnálu dříve, než se samotná operace provede a změní se data i informace v metadatech, což pomáhá při obnovování a opravách například po pádu operačního systému. V takových případech se procházejí záznamy v žurnálu a zjišťuje se (podle nastaveného příznaku), zda před poruchou transakce proběhly, nebo ne, a provádí se náprava. Žurnálovací záznamy jsou umístěny ve druhé položce MFT tabulky.

NTFS zavádí pevné odkazy. Jedná se o odkazy na jeden soubor, které smí být umístěné kdekoli na disku v rámci tohoto souborového systému. V tabulce MFT je uložen počet vytvořených pevných odkazů na jednotlivé soubory nebo složky. Při smazání pevného odkazu

[15] Partition boot sector
[16] MFT - Master File Table
[17] Převzato z: [3] (20. 11. 2013)

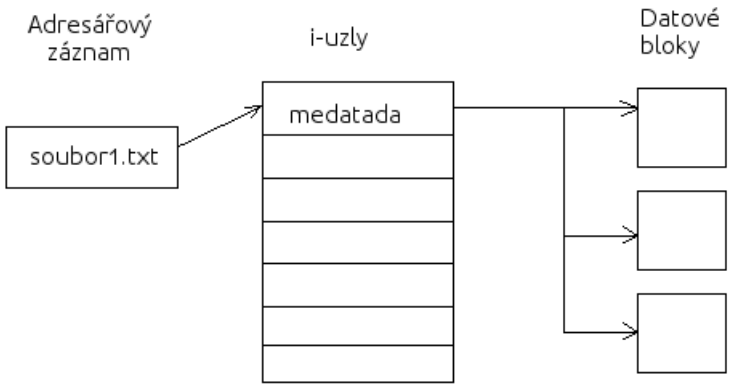
se jejich počet pro daný soubor zmenší o 1; pokud je počet pevných odkazů na soubor nulový, soubor se smaže. Pokud by došlo ke smazání originálního souboru, ale pořád by existoval alespoň jeden pevný odkaz na něj, pak na disku k uvolnění místa alokovaného pro obsah souboru nedojde a pouze se sníží počet pevných odkazů, které na něj ukazují.

2.2.3 EXT2, EXT3, EXT4

Standardním linuxovým souborovým systémem je tzv. Second Extended File System vycházející z prvního souborového systému vytvořeného pro unixové systémy UFS (Unix File System). Podoporuje práci s pevnými disky do velikosti až 1 exabyte a soubory o velikosti 16 terabytů (u EXT4).[3]

Tento souborový systém považuje vše na disku za soubor. Využívá taktéž i-uzly (informační uzly), uložené v tabulce i-uzlů, které obsahují informace o každém souboru nebo adresáři (metadata; čas vytvoření souboru, jeho změny, přístupu k jeho obsahu, identifikace vlastníka a skupiny). Číslo i-uzlu je svázáno s názvem souboru, o kterém i-uzel obsahuje informace. I-uzel taktéž obsahuje ukazatel na další i-uzly nebo přímo datové bloky (v i-uzlu může být až 12 přímých adres na datové bloky; pokud taková kapacita souboru nestačí, místo jednoho či více přímých adres na datové blok se vytvoří odkazy na datové bloky s adresami datových bloků — jedná se o tzv. nepřímé adresování). Využívá se také pro tvorbu symbolických odkazů, kdy není nutné soubor na různá místa v adresářové struktuře kopírovat, ale stačí zde umístit právě odkaz na jeden tento požadovaný soubor na disku (pevné odkazy) nebo se odkaz na datový blok souboru umístí přímo do i-uzlu (symbolické odkazy).[4]

Každá složka má — tak jako v NTFS — svůj adresářový záznam, ve kterém se nacházejí záznamy o jejím obsahu (názvy všech jejich souborů a podsložek).



Obrázek 2.5: Přímé adresování datových bloků z i-uzlu.[18]

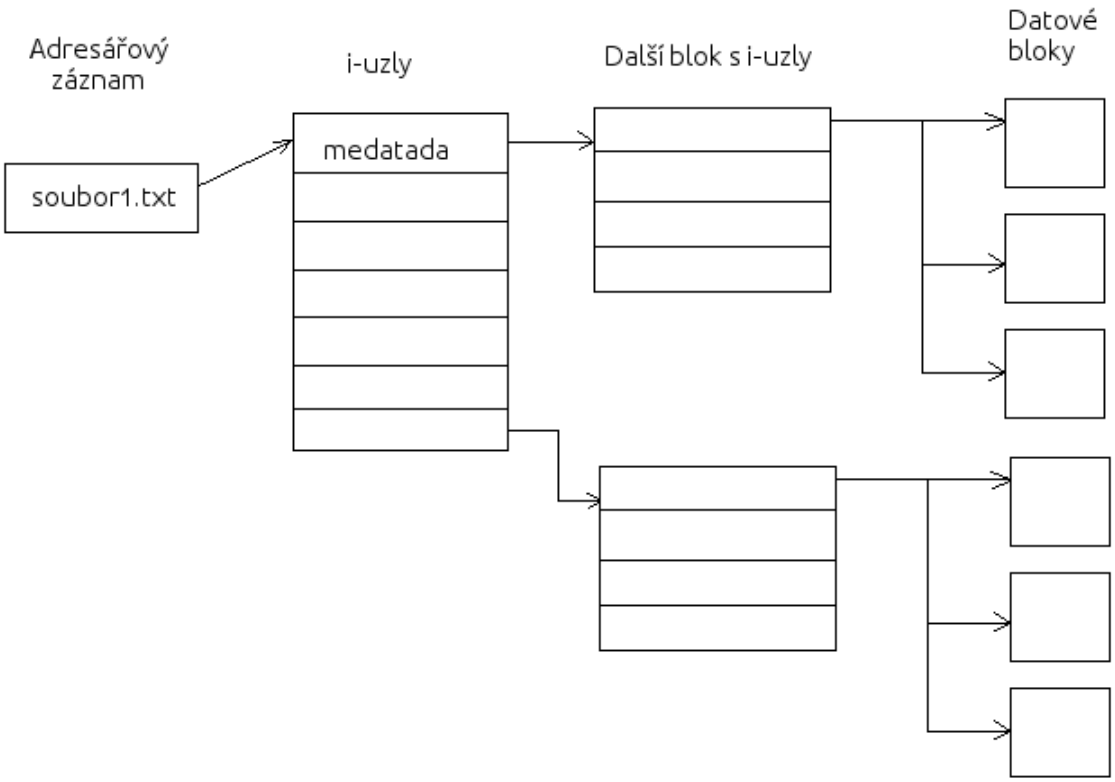
Prvních 1024 bytů souborového systému je využito pro zaváděcí kód operačního systému. Data jsou uložena v blocích o velikostech 2 nebo 4 kilobyty (připomínají clustery v NTFS). Informace o disku jsou umístěny v superbloku, který se nachází za oblastí pro kód pro zavedení operačního systému (1 kilobyte od začátku souborového systému). Zde jsou informace o velikosti bloků, jejich počtu, počet i-uzlů, název oddílu, časové údaje (poslední zápis, čas připojení svazku) a místo, kam byl datový svazek připojen. Za superblokem následuje

[18] Převzato z: [3] (28. 11. 2013)

blok tabulek popisovačů skupin bloků[19]. Kromě obsahu souboru zahrhnuje i tabulku i-uzlů a další informační záznamy.

U EXT3 se objevuje bezpečnější způsob mazání souborů, kdy se vymaže ukazatel v i-uzlu na daný datový blok obsahující soubor. Z toho plyne, že pokud uživatel neví, na kterém bloku soubor začíná, nemůže ho obnovit nebo se k datům dostat.

EXT4 zavádí nový způsob alokace místa na disku po tzv. extentech, což jsou navazující fyzické datové bloky. Tento způsob usnadňuje práci s velkými soubory a zmenšuje fragmentaci disku.



Obrázek 2.6: Nepřímé adresování pomocí alokování nového bloku i-uzlů.[20]

2.2.4 HFS+

Tento souborový systém je používán zařízeními firmy Macintosh. Jeho struktura obsahuje zaváděcí oddíl[21], MDB[22] (zahrhnuje datum a čas vytvoření HFS+ oddílu a umístění dalších systémových souborů), mapu oddílů[23] (záznamy o použitých a volných blocích disku) a katalog (seznam všech souborů a adresářů v oddílu). [1]

[19] Group Descriptor Tables
[20] Převzato z: [4] (28. 11. 2013)
[21] boot block
[22] Master Directory Block
[23] Volume Bitmap

Kapitola 3

Forenzní analýza paměťových prvků a metadat

Jak už bylo zmíněno výše, forenzní analýza je postupem ke zkoumání a vyšetřování počítačové kriminality. Její hlavní body budou shrnuty v následujících podkapitolách.

3.1 Zajištění paměťových prvků

V první fázi vyšetřování zločinu spjatého nějakým způsobem s informačními technologiemi je třeba zajistit místo činu a zabezpečit ho před vlivem okolí a samotných pachatelů — to by mohlo mít nepříznivý vliv na pravost a pravdivostní hodnotu hledaných dat. Podstatné pro další práci forenzního analytika jsou např. počítače, notebooky, PDA, tablety, mobilní telefony, CD, DVD, flash disky, externí disky. Všechna tato zařízení se musí zaevidovat, popsat jejich stav v okamžiku nalezení (zda bylo zapnuté nebo vypnuté, jaké programy na něm běžely, co bylo vidět na obrazovce atp.) a je-li to možné, bezpečně dopravit do výzkumné laboratoře. Taktéž je vhodné uložit před vypnutím počítače neuložené změny v souborech do jejich nových verzí (vytvořit nový soubor s jiným jménem), aby při následném uložení nedošlo ke ztrátě potenciálně důležitých informací, které se mohly nacházet v předchozí verzi souboru.

Někdy ovšem není možné odnést paměťové zařízení z místa činu. V takovém případě se musí data zajistit na okamžitě[1]. S tím ovšem souvisí i zvýšené riziko neplatnosti získaných dat. Na analyzovaném počítači mohou běžet nejrůznější skripty, účelně měnící data ukládaná na disk (což ztěžuje či přímo znemožňuje jejich čtení bez znalosti činnosti takového skriptu), nebo se během zajišťování důkazů může podezřelý subjekt připojit na svůj počítač vzdáleně a podsouvat tak vyšetřovatelům znehodnocená nebo odlišná data, než jaká by se jinak dala z počítače získat. Proto je při tomto způsobu získávání důkazů důležité ukončit všechny procesy a programy, které by mohly ohrozit výpovědní hodnotu dat jak na pevných discích a flash discích, tak v paměti RAM. Vhodné je také odpojit počítač od sítě internet, aby bylo zamezeno ohrožení důkazů vzdáleným přístupem útočníka. Další možností, jak zvýšit ochranu dat, je využít pro jejich zajištění svůj vlastní důvěryhodný linuxový systém, který lze spustit například z CD nebo flash disku[2]. Výhodou je, že takový operační systém nebude automaticky připojovat disky a další paměťová média počítače, a tím pádem ani ohrožovat data před jejich modifikací.

[1] angl. live acquisition
[2] tzv. live CD/flash disk

Získaná data je taktéž možné chránit pomocí hardwarového či softwarového blokování zápisu do paměťového zařízení. V případě hardwaru se jedná o prvek umístěný mezi pevným diskem a jeho řadičem. Programově řešené blokování zápisu pracuje na základě sledování žádostí o vykonání obslužné rutiny (tzv. programové přerušení). Pokud se objeví žádost o přerušení související s diskem, blokovací program ho vyhodnotí a pokud se jedná o požadavek na zápis na disk, provede program vlastní akci (nenechá transakci proběhnout), v opačném případě (nehrozí-li porušení dat — např. čtení z disku) nechá blokovací program obslužnou rutinu dokončit svou činnost.

3.2 Získávání důležitých dat

Když chceme podrobit obsah zajištěného paměťového zařízení, například pevného disku, bližšímu zkoumání pomocí forenzních nástrojů, je důležité, aby nedošlo ke ztrátě či porušení originálních dat, která by pak nemohla sloužit jako případný důkazný materiál. Proto je potřeba před samotným výzkumem prosvést kopii těchto dat z původního nosiče na nosič jiný. Existují čtyři následující metody kopírování: [3]

- vytvoření obrazu disku do souboru (nejčastější metoda),
- kopie z disku na disk (využití při kopiích starších disků s jinou geometrií),
- logická kopie disku na nový disk (kopírování pouze těch souborů, které by se mohly při vyšetřování hodit, a ne celého disku; nelze ovšem dále využít pro obnovování souborů),
- řídká kopie (jako logická kopie, ale kopírují se i fragmenty smazaných dat).

Pokud máme více nástrojů, kterými můžeme provést kopii dat, je vhodné to udělat a vytvořit více, než jen jednu kopii. Lze tak zmenšit riziko nepoužitelnosti získaných dat v případě, že například kopírování pomocí jednoho nástroje proběhne špatně.

Někdy je velikost dat, která chceme zkoumat, neúnosně velká a kopírování by trvalo dlouho. Proto je možné využít bezztrátvou kompresi, a následně vzniklá komprimovaná data ověřit přes jejich kontrolní součet.

3.3 Validace získaných dat

Velice podstatné je zachování integrity mezi originálními daty a jejich forenzními kopiemi. K jejímu ověření se dají využít digitální podpisy získaných souborů (třeba pomocí algoritmů MD5, SHA-1 či SHA-256).

3.4 Získávání informací ze získaných dat

Paměťová zařízení ovšem nemusí obsahovat pouze ta data, která jsou podstatná pro vyšetřovaný případ. V praxi je tomu většinou naopak — pevný disk je plný nejrůznějších souborů (textových dokumentů, fotek, videí, programů, systémouvých souborů) a mezi nimi se ukrývají hledaná data. Prohledávání celého obsahu paměťového prvku může být zdlouhavé,

a proto je třeba zvážit i povahu případu, na kterém se pracuje. Může jít o internetovou zločinnost (pak se zaměřujeme na historii, záložky a záznamy webových prohlížečů a poštovních klientů), útok z vnějšku (pokusíme se najít známky přítomnosti rootkitů a virů) či podezření na úmyslné poškození nebo krádež dat (sledujeme data a časy změn a přístupů k souborům).

Neméně důležité je také vědět, jaký operační systém a programy (a jejich verze) jsou na počítači nebo jiném zkoumaném zařízení k dispozici. Je pak jasnější, čeho byl majitel zařízení schopen. Tím se může usnadnit práce vyšetřovatele i následná rekonstrukce případu.

Výstupem forenzní analýzy paměťového média jsou jak soubory, tak jejich fragmenty a metadata.

3.5 Rekonstrukce případu

V poslední části práce forenzního analytika se ze získaných důkazů rekonstruuje řešený případ nebo jeho část. Zejména je zaměřena na způsob práce podezřelého subjektu, informace, které se snažil získávat, a škody, které svou činností napáchal.

3.6 Závěrečná zpráva

Konečným výstupem celé forenzní analýzy je tzv. závěrečná zpráva, která shrnuje případ, dokumentuje zkoumaná zařízení, použité postupy a dosažené výsledky.

Kapitola 4

Nástroje forenzní analýzy disků a metadat

V této kapitole bude představeno několik Linuxových nástrojů, které je možné využít pro práci na forenzní analýze paměťových zařízení. Jedná se pouze o základní informace o nástrojích; jejich praktické srovnání bude provedeno v kapitole 5.

4.1 dd a dcfldd

`dd`[1] je unixovým nástrojem spouštěným z terminálu či příkazové řádky, který se dá využít pro různé potřeby:

- tvorba bitové kopie paměťových zařízení,
- oprava hlavního zaváděcího záznamu [2],
- modifikace dat - například přepsání prvních bytů souboru,
- zabezpečení disku vynulováním nebo přehráním náhodnými daty,
- obnova dat,
- testování disku,
- generování náhodných dat do souboru,
- vytváření prázdných souborů nebo zvětšení jejich velikosti.

`dcfldd`[3] je rozšířenou verzí dd. Poskytuje navíc výpočet kontrolních součtů, ověření totožnosti souborů a procentuální vyjádření probíhající operace.

4.2 guymaker

Pro získávání dat z paměťových zařízení je možné použít také linuxový nástroj `guymaker`. Podporuje klonování disků a kopírování obrazů disků do AFF formátu.

[1] http://ss64.com/bash/dd.html
[2] MBR — Master Boot Record
[3] http://dcfldd.sourceforge.net

4.3 aimage

`aimage` je dalším nástrojem, který je možné využít pro tvorbu kopií paměťových zařízení do souborů typu raw, AFD, AFF, AFM. Během kopírování si lze také nechat vygenerovat MD5 nebo SHA-1 kontrolní součty.

4.4 S.M.A.R.T. a smartmontools

`S.M.A.R.T.`[4] je technologie vyvinutá za účelem sledování, testování a vyhodnocování stavu pevných disků. Během provozu disku se sledují různé parametry a na základě nich se pak vyhodnocuje, jaká je pravděpodobnost selhání disku. Tuto technologii využívají dva nástroje ze skupiny smartmontools: smartctl a smartd.

4.5 Second Look

Tento linuxový program[5] byl vytvořen pro zachytávání tzv. volatilních informací (např. data v paměti cache, RAM), které nemusí existovat na disku, a po jisté době (nebo po vypnutí počítače) mohou být ztracena. Analýza těchto dat je podstatná z hlediska vyhledávání rootkitů či útočníkových „zadních vrátek“ v systému, kudy si může posílat citlivá data na své zařízení.

4.6 SleuthKit a Autopsy

`SleuthKit`[6] je kolekcí nástrojů použitelných na unixových a linuxových systémech a na Windows. Používá se prostřednictvím příkazů v terminálu, ale lze ovšem využít i grafickou nástavbu `Autopsy`.

Pomocí jednotlivých nástrojů, které jsou členěny do kategorií (diskové nástroje, nástroje pro diskové oddíly, pro souborové systémy, vyhledávací nástroje) lze například zobrazovat informace o paměťovém médiu nebo jeho obrazu, procházet záznamy v metadatech, zjišťovat stav datových bloků souborového systému, nebo odhalovat chráněné oblasti disku a jejich obsah.

4.7 LiME

`LiME`[7] je modulem pro jádro linuxového systému (byl přizpůsoben i pro Android), který umožňuje přístup k volatilní paměti a zachycení celého jejího obsahu. To může pomoci např. při odhalování přítomnosti malwaru v počítači nebo při analýze činnosti různých programů.

[4] http://sourceforge.net/apps/trac/smartmontools/wiki
[5] http://secondlookforensics.com
[6] http://www.sleuthkit.org/
[7] http://code.google.com/p/lime-forensics

4.8 Volatility

`Volatility`[8] je kolekcí souborů pod licencí GPU pro získávání dat z volatilní paměti (RAM paměti), jejichž činnost je nezávislá na systému, na kterém jsou používány.

4.9 Memfetch

Tento program je využitelný pro sledování potenciálně podezřelého chování programů. `Memfetch`[9] pracuje na způsobu získávání paměti programu přímo za běhu a nijak jeho činnost nenarušuje.

4.10 DFF - Digital Forensics Framework

`DFF`[10] je otevřený software[11] program určený pro forenzní analýzu. Poskytuje programovou ochranu proti přepsání, výpočet kontrolních součtů, bezpečný přístup k diskům a souborovým systémům, obnovování souborů, analýzu volatilních pamětí (RAM paměť, paměť cache), umožňuje práci s formáty raw a AFF, a je využitelný při forenzní analýze na Linuxu i Windows.

4.11 e2undel

Konzolový program `e2undel`[12] slouží pro získání smazaných dat v souborovém systém EXT2. Při obnovování dat nemanipuluje se strukturou souborového systému; vyžaduje poze práva pro čtení.

4.12 ntfsundelete

`ntfsundelete`[13] se využívá taktéž pro obnovu smazaných dat, ale na souborových systémech NTFS a FAT. Podporuje všechny typy souborů, dokáže pracovat s pevnými disky, paměťovými kartami, flash disky i disketami. Zaznamenává jména souborů, cestu k nim, jejich stav, velikost, datum vytvoření a změny, a druh souboru.

4.13 foremost a Scalpel

`foremost`[14] je linuxový program sloužící pro obnovu smazaných souborů. Dokáže prohledávat kopie disků v jakémkoli formátu a vyhledávat soubory podle jejich počátečních a koncových znaků a vnitřní struktury. Podpporuje mnoho typů souborů: `pdf, bmp, jpg, doc, docx, ppt, zip, wav, exe, htm, mov, mp4`, atd.
Na `foremost` je založen další nástroj pro obnovu dat: `Scalpel`[15]. Jde v podstatě o dokonalejší a rychlejší verzi `foremost`.

[8] https://code.google.com/p/volatility
[9] http://ostatic.com/memfetch/home/1
[10] http://www.digital-forensic.org
[11] angl. Open Source
[12] http://e2undel.sourceforge.net
[13] http://ntfsundelete.com
[14] http://foremost.sourceforge.net
[15] https://github.com/sleuthkit/scalpel

4.14 gpart

gpart[16] je nástrojem, který je využitelný při pokusu o obnovu tabulky rozdělení disku, pokud je tato porušená, smazaná nebo obsahuje nesprávné záznamy.

4.15 testdisk

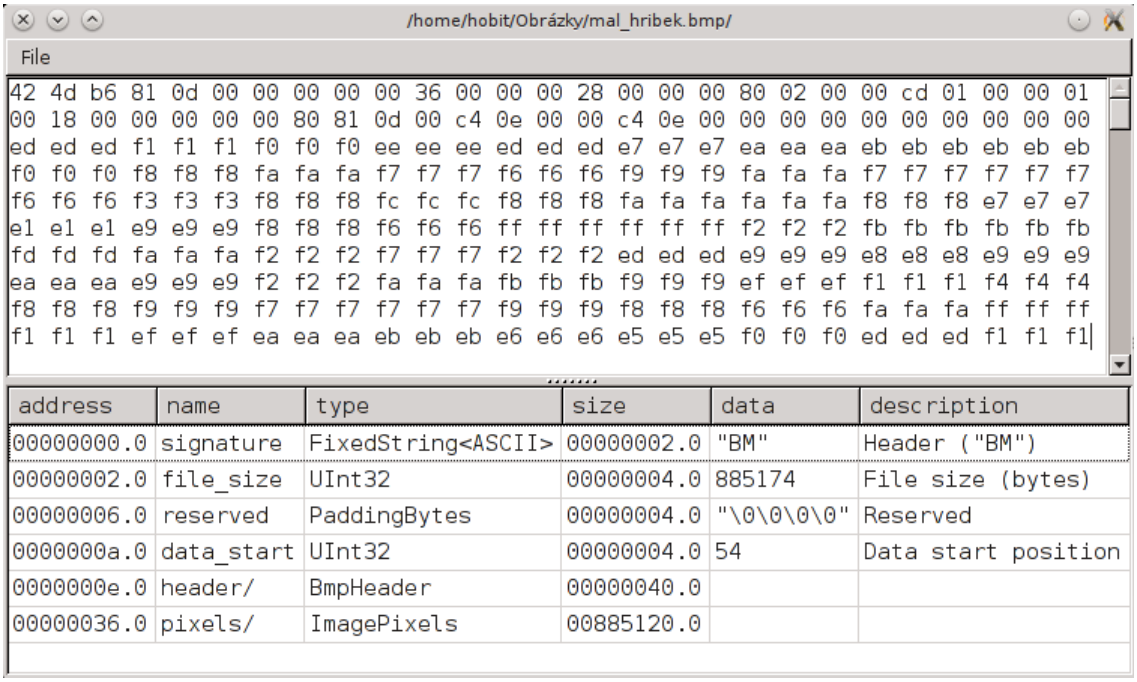
`testdisk`[17] se používá pro řešení problémů s diskovými oddíly a tabulkou rozdělení disku — tuto tabulku dokáže opravit nebo ji celou vytvořit znova, pokud byla zničena či smazána, a to na základě kontroly diskového prostoru počítače. Součástí tohoto programu je utilita photorec, která slouží pro obnovu smazaných souborů.

4.16 GNU Libextractor

`GNU Libextractor`[18] je knihovnou pro extrahování metadat ze souborů.

4.17 hachoir

`hachoir`[19] je knihovna napsaná v jazyce Python pro modifikaci a zkoumání souborů na binární úrovni a taktéž pro extrakci metadat. Podporuje přibližně šedesát souborových formátů.



Obrázek 4.1: Program `hachoir` má grafickou nadstavbu.

[16] http://www.freebsd.org/cgi/man.cgi?query=gpart&sektion=8
[17] http://www.cgsecurity.org/wiki/TestDisk
[18] http://www.gnu.org/software/libextractor
[19] https://bitbucket.org/haypo/hachoir/wiki/Home

4.18 vinetto

Operační systémy Windows ukládají metadata obrazových souborů do Thumbs.db souborů. Nachází se v nich jak popis, tak zmenšený obrázek. Pokud se soubor s obrázkem smaže, Thumbs.db zůstává zachován. Vinetto[20] je nástrojem, který se dá využít na prozkoumání Thumbs.db souborů a extrahování metadat o obrazových souborech, což může posloužit při jejich obnovování.

4.19 KaliLinux, BackTrack Linux, Knoppix

Jak už bylo řečeno v kapitole, která se věnovala postupu při provádění forenzní analýzy, jsou Linuxové distribuce využitelné jak přímo na místě činu pro získávání dat z počítače, tak i při dalším výzkumu získaných dat. Některé distribuce byly vytvořeny a přizpůsobeny právě pro takovou práci — mají malou velikost, jsou spustitelné přímo z CD, DVD nebo flash disku, a obsahují množství nástrojů pro forenzní analýzu. Nejpoužívanějšími distribucemi jsou například KaliLinux[21] (nástupce distribuce BackTrack Linux[22]) nebo Knoppix[23].

4.20 truecrypt a truecrack

`truecrypt`[24] je nástroj ovladatelný přes grafické rozhraní a využitelný pro šifrování oddílů disku nebo celého disku. Lze zvolit šifrování AES, Serpent, Twofish a jejich kombinace. Toto zabezpečení lze prolomit pomocí programu `truecrack`[25] (ovládaný přes příkazovou řádku), který funguje na principu útoku hrubou silou, kdy generuje hesla ze zadané vstupní abecedy do maximální zadané délky a aplikuje je na zašifrovaný oddíl. Nalezené heslo následně oznámí uživateli.

[20] http://vinetto.sourceforge.net
[21] http://www.kali.org
[22] http://www.backtrack-linux.org
[23] http://www.knoppix.org
[24] http://www.truecrypt.org/
[25] http://code.google.com/p/truecrack/

Kapitola 5

Praktické srovnání vybraných forenzních nástrojů

Součástí této práce je srovnání volně dostupných forenzních nástrojů, které je možné používat na operačním systému Linux a které souvisejí se získáváním dat z paměťových prvků. Vybrané programy byly testovány na operačním systému ArchLinux s verzí linuxového jádra 3.14.1-1 s využitím flash paměti o kapacitě 8 GB.

5.1 Nástroje pro tvorbu kopie paměťového média

Pro forenzního analytika je velmi důležité bezpečné uchování originálních dat a možnost práce s jejich kopií; tak se totiž snižuje riziko poškození důkazního materiálu. Za tímto účelem je možné použít například programy `dd, dcfldd` či `partimage`, které budou následně srovnány.

Všechny tři vybrané nástroje nabízí možnost kopie kopie média do souboru s uživatelsky zvoleným jménem a umístěním. Programy `dd` a `dcfldd` se ovládají pomocí příkazové řádky; nástroj `partimage` poskytuje jednoduché textové uživatelské rozhraní. Při použití `dd` či `dcfldd` není možné vytvořit koprimovanou kopii, narozdíl od `partimage`, který poskytuje tři možnosti komprese (žádnou, gzip, bzip).

Program	**Doba běhu**	**Velikost výstupního souboru**	**Objem získaných dat**
`dd`	7 min 25 s	7680,4 MB	7655,88 MB
`dcfldd`	7 min 15 s	7680,7 MB	7556 MB
`partimage` (bez komprese)	28 s	421,7 MB	420,58 MB
`partimage` (komprese gzip)	32 s	156,6 MB	420,58 MB
`partimage` (komprese bzip)	33 s	149,3 MB	420,58 MB

Tabulka 5.1: Srovnání nástrojů pro kopii dat.

Jak je patrné z tabulky, největší nároky na čas i na paměť klade tvorba kopie bez použití komprese, jako je tomu v případě použití nástrojů `dd, dcfldd` a prvního testu `partimage`. `Partimage` také – narozdíl od zbylých dvou nástrojů – kopíruje pouze nevynulované bloky (čili bloky, které obsahují nějaká data), a tím razantně snižuje velikost výsledné kopie; při využití koprese je pak výstupní soubor ještě menší.

5.2 Nástroje pro obnovu souborů

Mezi důležité nástroje forenzní analýzy patří také programy umožňující obnovit smazané soubory či jejich fragmenty. Pro srovnání byly vybrány nástroje `foremost`, `extundelete` a `photorec`.

Program	Doba běhu	Požadavek	Počet obnovených souborů
`foremost`	10 min 22 s	Obnova všech souborů	2171
`photorec`	13 min 36 s	Obnova všech souborů	2483
`extundelete`	7,6 s	Obnova všech souborů	36

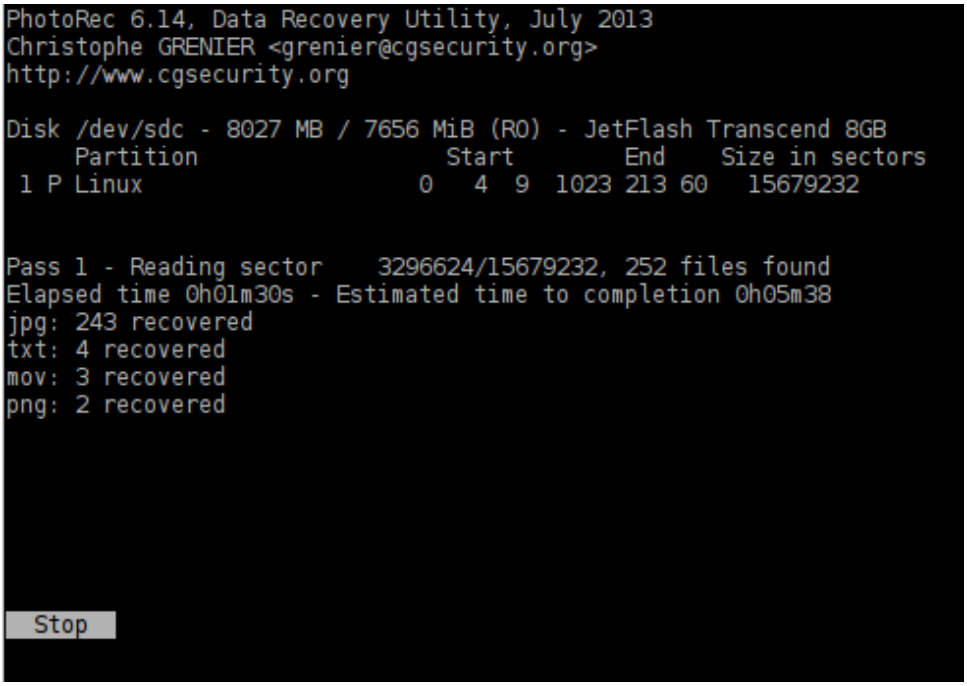
Tabulka 5.2: Srovnání nástrojů pro obnovu dat.

`foremost` dovoluje obnovit buď všechny soubory, nebo jen soubory určitého typu; poskytuje též možnost neobnovovat žádný soubor a podat pouze hlášení o zkoumaném médiu (případně jeho obrazu). Pracuje na základě vyhledávání začátků a konců souborů. Tento program se ovládá pomocí příkazové řádky.
`photorec` při své práci ignoruje souborový systém paměťového média. Nejprve najde prvních 10 souborů, z jejichž umístění a velikosti vypočítá velikost bloků souborového systému a dále prochází médium blok po bloku a vyhledává začátky a konce souborů. Pokud by byl obnovený soubor menší, než je uvedeno v jeho hlavičce, pak se takový soubor neobnoví (došlo pravděpodobně k jeho poškození nebo fragmentaci). Je ovládán přes jednoduché textové rozhraní. Narozdíl od `foremost` nedovoluje specifikovat typ hledaných souborů a proto se pokouší obnovit soubory všech typů, které se na médiu nacházejí.

`extundelete` je program, který se dá využít při práci se souborovými systémy EXT3 a EXT4. Pracuje na základě analýzy žurnálovacího souboru a i-uzlů. Umožňuje obnovovat buď všechny soubory z paměťového média, nebo vybrané soubory podle zadané cesty či jejich čísla i-uzlu.

5.3 Nástroje pro opravu paměťových médií

Mezi forenzní nástroje lze zahrnout i ty, které slouží k opravám či analýze paměťových prvků. Do této kategorie patří například program `testdisk`, se kterým je možné provádět analýzu geometrie paměťových médií, změnit geometrii hlav, cylindrů či sektorů disku, obnovit superblok ze záložní kopie (u systému EXT3 nebo EXT4), změnit souborový systém, vytvořit zálohu média pomocí utility photorec či zobrazit seznam souborů obsažených



Obrázek 5.1: Rozhraní programu `photorec`.

na médiu (včetně smazaných souborů). `testdisk` je ovládán pomocí textového rozhraní. Všechny provedené akce se zaznamenávají do výstupního logovacího souboru.

Kapitola 6

Praktická realizace forenzní aplikace

Součástí bakalářské práce je návrh a tvorba aplikace, která bude využitelná při forenzí analýze. Tato aplikace nazvaná `bp_undelete` pro operační systém Linux bude zaměřená na obnovování dat. Bude možné ji použít při zkoumání pevných disků, flash disků, a taktéž CD/DVD disků; bude podporovat souborové systémy EXT2, EXT3 a EXT4 (které jsou typické právě pro linuxové systémy), ISO 9660 a UDF. Výstupem aplikace bude obnovený soubor či množina souborů a soubor s informacemi o průběhu a výsledku práce aplikace (kolik souborů bylo obnoveno a dále ke každému souboru informace o velikosti a trvání jeho obnovy).

Druhou částí praktické stránky bakalářské práce bude několik demonstračních příkladů použití forenzních nástrojů, které by mohly být využity pro výuku na FIT VUT v Brně.

6.1 Implementace

Některé aplikace věnující se podobnému úkolu, které byly zmiňovány a popsány výše, a které dovolují práci se souborovými systémy EXT2, EXT3 či EXT4, pracují na základě analýzy žurnálovacího souboru. Z toho plyne, že mají k dispozici určitá metadata o hledaných souborech — například které bloky má soubor přidělen, jaké je jeho jméno nebo kdy byla provedena jeho poslední změna (obvykle jeho smazání). Žurnálovací soubor může ovšem být u některých médií poškozen a ke zmíněným informacím se poté není možné dostat. Proto byl při tvorbě aplikace zvolen odlišný přístup: podle nalezených informací o paměťovém médiu se zjistí velikost bloků, na které je médium rozděleno a tyto bloky se budou postupně procházet a budou v nich hedány počáteční a koncové značky souborů.

Pro tvorbu aplikace byl zvolen modulární přístup a jazyk C++, který umožňuje nízkoúrovňovou práci se zkoumanými objekty. V C++ je možné využít knihovny a funkce, které usnadňují práci s datovými médii. Bylo uvažováno nad využitím objektového přístupu, ovšem aplikace nemá zapotřebí mít za běhu uloženo velké množství informací a navíc není žádaná vyšší abstrakce nad daty, jakou objektový přístup poskytuje.

6.2 Popis částí

Výsledná aplikace se skládá z několika dílčích modulů, které budou popsány níže.

6.2.1 bp_main

Tento modul je vstupním bodem aplikace. Obsahuje funkci `main(int argc, char*argv[])`, ve které je nejdříve vyřešeno načtení a zpracování parametrů, se kterými byl program spuštěn. Parametry jsou trojího typu. První parametr je vždy název oddílu či disku, nad kterým se aplikace spouští. Druhá sada parametrů určuje typy souborů, jenž chce uživatel obnovit (tyto typy jsou uvedeny v podsekci 6.2.3). Je-li požadována obnova všech podporovaných formátů, je třeba zadat parametr -all. Tyto údaje jsou zaznamenány do položek struktury `paramFileTypes`, odvozené od typu `fileTypes` (tento je definován v hlavičkovém souboru *bp_main.h*):

```
typedef struct fileTypes
{
 int jpeg;
 int png;
 int gif;
 int odt;
 int bmp;
 int pdf;
 int rtf;
 int html;
 int mpg;
 int msoff;
}fileTypes;
```

Třetím typem parametrů jsou ty, které dovolují omezit velikost obnovovaných souborů. Uživatel může chtět získat soubory větší či menší, než je zadaná velikost souboru v bytech. Získané hodnoty se ukládají do struktury `valueFP`, která je typu `fileParams`:

```
typedef struct fileParams
{
 uint64_t max_size;
 uint64_t min_size;
}fileParams;
```

Navíc je možné zadat parametr –help, který vypíše nápovědu.

Současně se zpracováním parametrů se v místě, odkud je aplikace spuštěna, vytváří i adresářová struktura, která bude později sloužit jako úložiště obnovených souborů. Tato struktura obsahuje adresáře pojmenované podle jednotlivých podporovaných typů souborů, které jsou umístěny do aplikací taktéž vytvářeného adresáře OUTPUT.

Následně je zavolána funkce `diskInfo(argv[1], paramFileTypes,valueFP)`, patřící do následujícího modulu.

6.2.2 bp_diskInfo

Druhou částí programu je modul `bp_diskInfo`, obsahující funkci `int diskInfo(char* diskNameParam, fileTypes paramFileTypes, fileParams valueFP)`. V této funkci se jako první načte obsah systémového souboru */proc/mounts*, ve kterém se nachází informace

o připojených paměťových svazcích. Nejprve je pomocí C++ funkce getline vyhledán řádek s údaji patřícími k názvu svazku, se kterým pracujeme (ten byl předán funkci `diskInfo()` jako parametr `char* diskNameParam`). Z tohoto řádku je podstatný třetí sloupec — typ souborového systému přítomného na svazku. Není-li v */proc/mounts* nalezen odpovídající záznam, uživatel chce pracovat nad médiem, které není připojeno, a program končí svou činnost s chybovým hlášením.

Dále se zjistí, zda nalezený souborový systém je aplikací podporován. Jde-li o systém UDF či ISO 9660, aplikace se pokusí otevřít si disk pro čtení, a zavolá funkci `findFiles (cd_dvd, paramFileTypes, sectorSize, valueFP)`, sloužící pro vyhledávání souborů. Bude-li se pracovat nad souborovým systémem EXT2, EX3 či EXT4, provede se nejprve připojení disku pro čtení a poté se do pole `unsigned char superblock[1024]` načte prvních 1024 bytů od začátku disku, kde se nacházejí detailní informace o médiu (jedná se o oblast zvanou superblok, viz 2.2.3). Odtud se zjistí velikost bloků, do kterých je celý svazek organizován. Důležitost tohoto kroku bude vysvětlena v podsekci 6.2.3. Poté se již zavolá funkce `findFiles(disk, paramFileTypes, blockSize, valueFP)`.

6.2.3 bp_findFiles

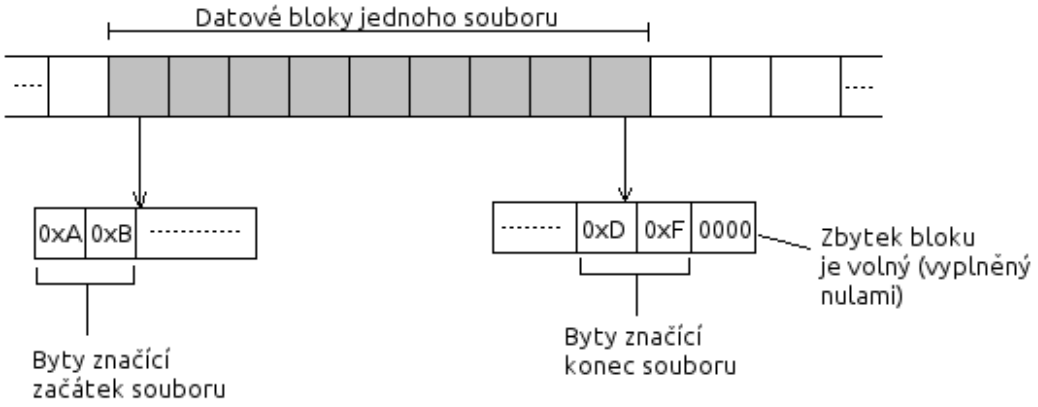
V tomto modulu dochází ve funkci `int findFiles(FILE *disk, fileTypes paramFileTypes, uint64_t blockSize, fileParams valueFP)` k jednomu průchodu obsahem analyzovaného média (parametr disk). V cyklu se postupně načítají za sebou jdoucí datové bloky do pole `unsigned char* data` o velikosti dané parametrem `blockSize`. Velikost načítaného bloku je důležitá zejména z toho důvodu, že soubory uložené na datovém médiu začínají vždy současně se začátkem bloku souborového systému — proto je pak snadné podle analýzy několika prvních bytů bloku určit, zda byl nalezen začátek některého z podporovaných typů souborů.

V takovém případě se do proměnné `uint32_t startPosition` zaznamená pořadové číslo bloku se začátkem souboru, a nastaví se příznaky, které určí, že se mají ostatní typy souborů ignorovat, a že se bude dále vyhledávat konec právě nalezeného souboru. Konec se ovšem může nacházet na libovolné pozici v bloku, a proto je nutné při jeho hledání projít postupně byte po bytu celý právě načtený blok a hledat sekvenci za sebou jdoucích znaků odpovídajících konci daného typu souboru.

V momentě, kdy je konec souboru nalezen, se zaznamená do proměnné `uint32_t endPosition` číslo bloku, ve kterém se vyskytuje. Poté je volána funkce `restoreFile (disk, startPosition, endPosition, name.c_str(), blockSize, valueFP, fileType)`, ve které dojde k obnovení daného souboru. Následně se nastaví příznaky tak, aby bylo možné pokračovat v hledání začátků dalších požadovaných souborů a taktéž se pomocí funkce `fseek()` posune ukazatel na další datový blok, který bude v dalším cyklu načten a analyzován.

Aplikace je schopná rozpoznat následující typy souborů: `jpeg`, `gif`, `png`, `bmp`, `rtf`, `odt`, `html`, `mpeg`, a dále pak soubory vytvořené pomocí kancelářského balíku MS-Office (přípony `doc`, `ppt` a `xls`). Tyto typy byly vybrány, protože je u nich snadné najít jak jejich počáteční, tak koncovou sekvenci bytů, a není proto nutné prohledávat například hlavičky souborů za účelem zjištění velikosti souboru.

Jediný takový problém nastává v případě typu `bmp`. Ten začíná dvěmi byty s hodnotou `0x42` a `0x4D`, ovšem konec nijak označen není. Proto je potřeba vyčíst velikost souboru z druhého až pátého bytu jeho prvního bloku. Tento údaj je ale uložen v pořadí tzv. Little-endian, kdy nejlevější bit je nejméně významný a nejpravější bit je nejvíce významný (má



Obrázek 6.1: Umístění souboru na disku.

nejvyšší mocninu). Pro účely aplikace je třeba převést tyto čtyři byty s informací o velikosti do pořadí tzv. Big-endian, kdy nejlevější bit je nejvíce významný a nejpravější bit nejméně výzamný.

Pro převod nebyla vybrána žádná funkce, která je k dispozici v jazyku C++, ale probíhá následujícím způsobem. Nejprve jsou všechny čtyři byty celočíselně vyděleny číslem 16 (pracuje se s šestnáctkovou číselnou soustavou), a také je nad nimi provedena operace modulo (taktéž číslem 16). Tak získáme čísla, která v dalším kroku budou násobena číslem 16 s patřičnou mocninou a sčítána v potřebném pořadí.

```
prvni = (uint32_t)data[0x2] % 16;
druhy = (uint32_t)data[0x2] / 16;
treti = (uint32_t)data[0x3] % 16;
ctvrty = (uint32_t)data[0x3] / 16;
paty = (uint32_t)data[0x4] % 16;
sesty = (uint32_t)data[0x4] / 16;
sedmy = (uint32_t)data[0x5] % 16;
osmy = (uint32_t)data[0x5] / 16;
```

```
bmpLen = osmy*pow(16.0,7.0) + sedmy*pow(16.0,6.0) + sesty*pow(16.0,5.0)
+ paty*pow(16.0,4.0) + ctvrty*pow(16.0,3.0) + treti*pow(16.0,2.0) + druhy*16
+ prvni;
```

Mějme například velikost souboru v jeho hlavičce zaznamenánu jako tato čtyři hexadecimální čísla: `0xB6 0x81 0x0D 0x00`. Pro naše účely potřebujeme jejich pořadí otočit na: `0x00 0x0D 0x81 0xB6`. Nejprve čísla vydělíme následujícím způsobem:

$$\begin{aligned} prvni &= 182 \,\%\, 16 = 6 \\ druhy &= 182 \,/\, 16 = 11 \\ treti &= 129 \,\%\, 16 = 1 \\ ctvrty &= 129 \,/\, 16 = 8 \end{aligned}$$

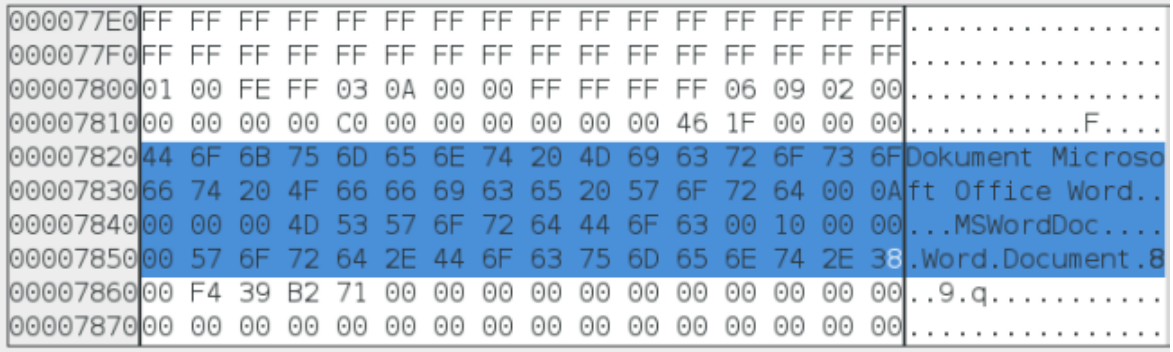
$$paty = 13 \,\%\, 16 = 13$$
$$sesty = 13 \,/\, 16 = 0$$
$$sedmy = 0 \,\%\, 16 = 0$$
$$osmy = 0 \,/\, 16 = 0$$

Poté už stačí jen tato čísla vynásobit číslem 16 s patřičnou mocninou a sečíst:

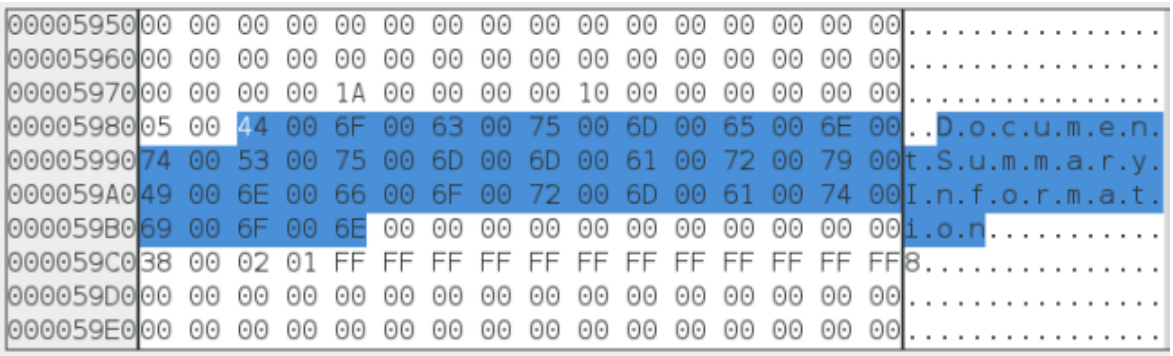
$osmy * 16^7 + sedmy * 16^6 + sesty * 16^5 + paty * 16^4 + ctvrty * 16^3 + treti * 16^2 + druhy * 16^1 + prvni * 16^0$

Výsledkem je číslo 852406, což je velikost analyzovaného souboru v bytech. Toto číslo je poté vyděleno velikostí (taktéž v bytech) jednoho datového bloku přítomného souborového systému a díky tomu je jasné, v kolika blocích se nacházejí data související se souborem a ve kterém bloku od bloku se začátkem souboru se nachází jeho konec.

Zajímavým případem je obnovování souborů vytvořených pomocí kancelářského balíku MS-Office. Všechny takové soubory jsou aplikací ukládány do jedné složky (`ms-office`) z následujícího důvodu. Textové (`doc`), tabulkové (`xls`) a prezentační (`ppt`) dokumenty mají totiž schodné počáteční sekvence bytů: 0xD0 0xCF 0x11 0xE0 0xA1 0xB1 0x1A 0xE1, a proto není zřejmé, o který typ souboru se jedná. Liší se pouze v koncových sekvencích:



Obrázek 6.2: Koncové byty dokumentu Word.



Obrázek 6.3: Koncové byty dokumentů Excel a PowerPoint.

6.2.4 bp_restore

V posledním modulu se nachází funkce `void restoreFile(FILE *disk, uint32_t startPosition, uint32_t endPosition, const char* name, uint64_t blockSize, fileParams valueFP, int fileType)`, která slouží k obnovení nalezeného souboru z datového nosiče.
V první části se otestuje, zda soubor, který chceme obnovit, splňuje velikostní limit zadaný uživatelem (pokud byl takový limit nastaven ve struktuře předané jako parametr `fileParams valueFP`). Poté se k názvu souboru připojí cesta do adresáře, kam bude po obnovení umístěn — a to na základě typu souboru identifikovaného podle parametru int fileType.

Samotné obnovení souboru probíhá následujícím způsobem. Nejprve se vytvoří výstupní datový tok `std::ofstream restoredFile(name, std::ios::binary)`. Funkcí `fseek()` se nastaví ukazatel na datový blok (s pořadovým číslem zadaným v parametru `uint32_t startPosition`) obsahující začátek souboru na datovém médiu. Pak je tolik bloků, kolik určuje číslo v proměnné `uint32_t range`, v cyklu načítáno do pole `char* buffer[blockSize]` a zapisováno do výstupního datového toku `restoredFile`:

```
uint32_t range = endPosition - startPosition +1; /*po odecteni bychom prisli
o naceni posledniho datoveho bloku - je potreba pricist 1*/
for(uint32_t i = 0; i< range; i++)
{
  //vypocet ukazatele na spravny blok na disku
  uint64_t off = (uint64_t)(i+startPosition)*(uint64_t)blockSize;
  int errSeek = fseek(disk,off,SEEK_SET);
  if(errSeek != 0)
  {
   std::cerr <<"RESTORE: Error reading data content!"<<std::endl;
   exit(1);
  }
  //nacteni datoveho bloku
  unsigned int errRead = fread(buffer, sizeof(unsigned char), blockSize, disk);
  if (errRead != blockSize)
  {
   std::cerr <<"RESTORE: Error reading datablock!"<<std::endl;
   exit(1);
  }
  /*** kopie datoveho bloku s casti dat obnovovaneho souboru***/
  restoredFile.write(buffer,len);

}
```

Po zapsání posledního bloku s daty souboru je vypsáno hlášení o úspěšném obnovení souboru a řízení je předáno zpět do funkce `findFiles`, která je popsána v podsekci 6.2.3.

6.3 Výstup aplikace

Jak už bylo naznačeno v předchozí sekci, jedním z výstupů aplikace je adresářová struktura obsahující všechny nalezené a obnovené soubory. Druhou částí je pak informační soubor *outLog.txt*. Sem aplikace zapisuje údaje o každém obnoveném souboru – jeho nové umístění, velikost, typ a dobu, za kterou byl obnoven:

Restored file: OUTPUT/bmp/bmp40649 type: bmp size: 664 kB time: 0.001055s

6.4 Testování aplikace a srovnání s jinými nástroji

Popisovaná aplikace byla vytvořena a taktéž testována na operačním systému Arch Linux s verzí linuxového jádra 3.14.1-1. Testování probíhalo nad flash diskem o velikosti 2 GB, na který bylo nahráno 478 souborů o celkové velikosti 834,6 MB.

Následující tabulka shrnuje několik testovacích běhů programu nad flash pamětí. Protože se tato práce věnuje též srovnání linuxových forenzních nástrojů, byly do tabulky zahrnuty i hodnoty získané při jejich použití na stejných či podobných testech.

Program	**Typy souborů**	**Doba běhu (reálný čas)**	**Doba běhu (systémový čas)**	**Obnoveno souborů**	**Neobnoveno souborů**
`bp_undelete`	všechny podporované	3 min 56,779 s	0 min 3,193 s	478	0
`bp_undelete`	jpeg, gif, png, bmp	2 min 31,575 s	0 min 2,558 s	442	0
`bp_undetele`	html, doc, ppt, xls, odt, rtf	3 min 1,929 s	0 min 1,787 s	34	0
`foremost`	všechny podporované	2 min 36,699 s	0 min 3,753 s	554	11
`foremost`	jpeg, bmp, png, gif	2 min 16,333 s	0 min 3,717 s	519	6
`foremost`	doc, ppt, xls, htm	2 min 11,429 s	0 min 3,143 s	34	0
`photorec`	všechny podporované	3 min 06,536 s	0 min 7,093 s	512	0

Tabulka 6.1: Porovnání aplikace s ostatními nástroji.

Testovaná aplikace pracuje podobným způsobem jako programy `foremost` a `photorec` — prochází médium blok po bloku a vyhledává začátky a konce souborů. Přesto jsou mezi těmito programy jisté rozdíly.

Jak je patrné z tabulky, `bp_undelete` vyžaduje pro svou činnost nejvíce času oproti ostatním programům. To může být zapříčiněno způsobem vyhledání a obnovy souboru. `bp_undelete` nejprve nalezne na médiu začátek a konec souboru a poté se vrátí do bloku se začátkem a kopíruje data z následujících bloků až do bloku s koncem souboru. Takto se vlastně každý blok, který nese data, načte dvakrát, než se soubor obnoví. Oproti tomu `foremost` a `photorec` načítají data ihned po nalezení začátku souboru a tak projdou každým blokem pouze jednou — proto byly také rychlejší, i když nalezly více souborů (viz dále).

Stejně tak se programy liší v počtu úspěšně obnovených souborů, na což má vliv jak způsob práce programů tak počet podporovaných typů souborů. `bp_undelete` dokáže nalézt všechny soubory, jejichž typy podporuje, pokud na datovém médiu nedošlo k fragmentaci či porušení začátku nebo konce souboru. Fragmentace souboru znemožňuje jeho obnovení, protože jednotlivé datové bloky v sobě nenesou žádnou informaci o tom, který blok s daty souboru je následující (o to se starají i-uzly společně s žurnálem). Protože `bp_undelete` nevyužívá žurnálovací záznam a i-uzly jsou po odstranění souboru vynulovány, není možné nijak zjistit, které bloky je nutné načíst.

`foremost` a `photorec` nejenže dokáží pracovat s více typy souborů, ale také umí nalézt a obnovit například obrázky, které jsou součástí prezentací či textových souborů. Proto z analyzovaného fash disku extrahovaly více souborů, než aplikace `bp_undelete`, která takto nepracuje.

Jak již bylo zmíněno v popisu aplikace, s programem `bp_undelete` je možné pracovat také nad systémy ISO 9660 a UDF, které jsou používány zejména na CD a DVD discích. Při pokusech s diskem CD-RW bylo však zjištěno, že dojde-li ke smazání obsahu nosiče (pomocí speciálního nástroje — například `brasero, k3b`), neodstraní se pouze medatada, ale i data samotná. Celý disk je totiž naplněn nulami a tím dojde ke ztrátě veškerých dat. Z toho důvodu není možné například z omylem vymazaného CD-RW disku obnovit jakákoli data. Pomocí aplikace se tedy dají extrahovat pouze ty soubory, které jsou v současné době na disku vypáleny.

6.5 Metriky kódu

- Počet souborů: 6.
- Počet řádků zdrojového textu: 1065.
- Velikost statických dat: 33,2 kB.
- Velikost spustitelného souboru: 10,9 kB(systém Linux).

6.6 Výstupy do výuky

Druhou praktickou částí této práce je — vedle tvroby aplikace — i několik demonstračních videí, na kterých je ukázáno využití některých forenzních nástrojů zmíněných v kapitole 4 v praxi. Tato videa jsou umístěna na přiloženém CD disku v adresáři `video`.

Kapitola 7

Závěr

Tato bakalářská práce měla za cíl provést čtenáře odvětvím forenzní analýzy, která se věnuje práci s datovými úložišti a získávání dat z nich.
Nejprve byl vysvětlen samotný pojem „forenzní analýza“ a uveden obvyklý postup práce, který s ním souvisí. Taktéž byl uveden do souvislostí s pojmem „počítačová kriminalita“ a s postihy definovanými v trestním zákoníku České republiky.

Dále pak byly popsány nejběžnější datové nosiče, jako jsou např. pevné disky, flash paměti, CD a DVD disky, a paměťi typu RAM. Práce obsahuje i stručný popis nejčastějších souborových systémů (FAT, NTFS, EXT).

Součástí textu je také výčet a srovnání volně dostupných forenzních nástrojů použitelných na operačním systému Linux. Kromě toho byla vytvořena aplikace věnující se obnově dat odstraněných z paměťových prvků se souborovým systémem EXT, UDF nebo ISO 9660. Tato aplikace pracuje na mírně odlišném způsobu a s menším počtem typů souborů, než jiné dostupné programy, a proto se při srovnání s nimi lišila jak v časovém hledisku, tak v počtu nalezených souborů. Aplikace funguje na principu hledání počátečních a koncových značek souborů. Zvolený způsob je výhodný, pokud je potřeba získat data z poškozených disků, kde není možné nalézt informace o souborech v i-uzlech nebo žurnálovacím souboru. Dalším možným rozšířením tohoto programu by mohlo být právě podpora dolování dat z žurnálu (i poškozeného) a tím i větší počet obnovitelných typů souborů. Aplikaci a její zdrojové kódy je možné dále volně šířit a upravovat. Poslední částí práce je několik videí, které je možné použít při výuce na FIT VUT v Brně, nacházejících se na přiloženém CD disku.

Literatura

[1] Bill Nelson, Amelia Phillips, Christopher Steuart: *Guide to computer forensics and investigation*. Course Technology, 2010, iSBN 978-1-435-49883-9.

[2] Brian Hatch, J. L.: *Hacking bez tajemství: Linux*. Computer Press, 2003, iSBN 80-7226-869-4.

[3] Carrier, B.: *File System Forensic Analysis*. Pearson Education, 2005, iSBN 0-32-126817-2.

[4] Vojnar, T.: IOS - Operační systémy [online]. `http://www.fit.vutbr.cz/study/courses/IOS/public/`, 2012-3-01 [cit. 2008-11-28].

[5] WWW stránky: Ley.cz - zákony online, právní poradna. `http://zakony-online.cz/?s10&q10=all`.

Příloha A

Obsah CD

- Technická zpráva ve formátu PDF v adresáři `/thesis/`
- Zdrojové kódy technické zprávy ve formátu LaTeXv adresáři `/thesis-latex/`
- Zdrojové kódy aplikace v adresáři `/src/`
- Soubor `readme.txt` s návodem na použití aplikace
- Videa s příklady v adresáři `/video/`