FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání diplomové práce

Název:	Bezpečnostní analýza Drive Snapshot
Student:	Bc. Michal Bambuch
Vedoucí:	Ing. Josef Kokeš
Studijní program:	Informatika
Obor / specializace:	Počítačová bezpečnost
Katedra:	Katedra informační bezpečnosti
Platnost zadání:	do konce letního semestru 2022/2023

Pokyny pro vypracování

1) Nastudujte problematiku zálohování diskových oddílů.
2) Seznamte se s nástrojem Drive Snapshot (http://www.drivesnapshot.de/).
3) Proveďte reverzní analýzu klíčových částí programu Drive Snapshot, např. formátu vytvářených souborů.
4) Analyzujte použití kryptografie při vytváření diskových obrazů. Popište použité algoritmy a jejich nastavení.
5) Vyhodnoťte bezpečnostní vlastnosti algoritmů i jejich implementace.
6) Naleznete-li nějaké zranitelnosti, demonstrujte je napsáním nástroje, který na ně zaútočí, nebo popište podmínky, za kterých by takový útok uspěl. Navrhněte opravná opatření.

Elektronicky schválil/a prof. Ing. Róbert Lórencz, CSc. dne 28. února 2021 v Praze.

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Diplomová práce

Bezpečnostní analýza Drive Snapshot

Bc. Michal Bambuch

Katedra informační bezpečnosti
Vedoucí práce: Ing. Josef Kokeš

6. května 2021

Poděkování

Rád bych poděkoval zejména mému vedoucímu Ing. Josefu Kokešovi za cenné rady při zpracování této diplomové práce. Dále bych rád poděkoval mé rodině za podporu při studiu.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principu při přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisu, zejména skutečnost, že České vysoké učení technické v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 citovaného zákona.

V Praze dne 6. května 2021 . . . . . .

Odkaz na tuto práci

Bambuch, Michal. *Bezpečnostní analýza Drive Snapshot*. Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2021.

Abstrakt

Tato diplomová práce se zabývá bezpečnostní analýzou programu Drive Snapshot. Práce prezentuje výsledky reverzní analýzy klíčových částí programu, popisuje použité kryptografické algoritmy a vyhodnocuje bezpečnost programu. Během bezpečnostní analýzy byla objevena řada bezpečnostních zranitelností, které mohou oslabit použitou kryptografii nebo ohrozit bezpečnost hesel anebo vytvořených záloh.

Klíčová slova Bezpečnostní analýza, Drive Snapshot, reverzní inženýrství, bezpečnostní zranitelnost.

Abstract

This thesis addresses the security analysis of Drive Snapshot. It presents the results of the reverse analysis of the key parts of the program, describes the used cryptographical algorithms, and evaluates the application security. During the security analysis, several security vulnerabilities were discovered that could weaken the used cryptography or compromise the security of passwords or created backups.

Keywords Security analysis, Drive Snapshot, reverse engineering, security vulnerability.

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Zálohování disků je naprosto nezbytné opatření k minimalizaci rizika ztráty potřebných dat. Počítačové disky mají omezenou životnost a k jejich selhání, a tedy i ke ztrátě dat, může kdykoliv dojít. Dle výzkumu společnosti Backblaze je po 4 letech nepřetržitého provozu 20 % disků nefunkčních [1]. Pro operační systém Microsoft Windows je nabízeno mnoho programů, které slouží k vytváření diskových záloh a jejich případnému obnovení. Jedním z nich je program Drive Snapshot, jehož bezpečnostní analýze se věnuji v této diplomové práci.

Drive Snapshot je program s více než 15letou historií, který nabízí jednoduché řešení pro vytváření záloh disků a jejich případnou obnovu. Zálohy disku navíc umí vytvářet i za běhu operačního systému, a to i v případě systémového disku. Program navíc nabízí možnost šifrování vytvořených záloh, které má zajistit důvěrnost zálohovaných informací.

Cílem této diplomové práce je provést bezpečnostní analýzu programu Drive Snapshot. V bezpečnostní analýze se zaměřuji zejména na vlastní souborový formát pro vytvářené zálohy a detekci použitých kryptografických algoritmů včetně jejich nastavení. Dále se také podrobně věnuji práci s uživatelskými hesly a popisuji technické provedení jejich možného ukládání.

V první kapitole stručně shrnuji problematiku zálohování disků. V druhé kapitole představuji program Drive Snapshot, popisuji jeho funkce a možnosti. Ve třetí kapitole prezentuji výsledky reverzní analýzy programu Drive Snapshot, ve které jsem se zaměřil zejména na formát souboru s vytvořenou zálohou, způsob šifrování zálohy včetně všech použitých kryptografických algoritmů a práci programu s hesly uživatele. V poslední čtvrté kapitole vyhodnocuji bezpečnost programu a prezentuji nalezené bezpečnostní zranitelnosti.

Kapitola 1

Problematika zálohování disků

Při zálohování disků existuje mnoho možností, jak zálohy vytvářet, různé programy nabízejí různé možnosti zálohy. Před zálohováním si je potřeba rozmyslet, co se od zálohy očekává a jaké řešení bude nejvhodnější. V této kapitole uvádím základní pojmy z oblasti zálohování disků a uvádím různé alternativy k analyzovanému programu Drive Snapshot.

1.1 Základní pojmy z oblasti disků

Ve spojitosti s disky se běžně používají následující pojmy – disk (*drive*), diskový oddíl (*partition*), svazek (*volume*) a soubor (*file*). Tyto pojmy jsou vůči sobě v určitém hierarchickém postavení. Na samotném vrcholu je disk – fyzické zařízení sloužící k uložení dat.

Operační systémy Microsoft Windows umožňují dva způsoby, jak s disky pracovat. Buď se s diskem pracuje jako s běžným diskem (*basic disk*), anebo jako s dynamickým diskem (*dynamic disk*). Běžné disky umožňují klasické rozdělení disku na jeden či více diskových oddílů, které jsou po naformátování použity pro uložení dat. Dynamické disky[1] oproti běžným diskům nabízejí pokročilé druhy svazků, které umožňují rozložení jednoho svazku přes více disků, zrcadlení svazků nebo vytvoření svazku typu RAID-5 [2].

Běžný disk lze tedy rozdělit na několik diskových oddílů, kdy každý z nich se chová jako samostatný disk [3]. Informace o diskových oddílech obsahuje MBR[2] nebo modernější GPT[3] tabulka. Na diskových oddílech lze následně vytvářet svazky.

Svazek (také disková jednotka) je nejvyšší jednotkou v hierarchii souborového systému [4]. Disková jednotka je naformátována souborovým systémem,

[1] V současné době jsou dynamické disky z části označené jako zastaralé. Jejich náhradou jsou ve Windows 10 Prostory úložiště (*Storage Spaces*) [2].

[2] Master boot record

[3] GUID Partition Table

který může dostat přidělené písmeno jednotky a uživatel s ní poté pracuje jako se samotným diskem ve správci souborů. Svazek je složen minimálně z jednoho diskového oddílu (typicky v případě běžného disku), ale může být složen i z více diskových oddílů nacházejících se i na rozdílných discích. Pod diskovou jednotkou si uživatel vytváří strukturu složek, do kterých ukládá své soubory.

Soubor je základní jednotkou dat, se kterou pracuje uživatel v rámci souborového systému [5]. V souborech rozmístěných ve složkách jsou uložena veškerá uživatelská data. Zálohování slouží právě jako prevence před ztrátou těchto dat.

1.2 Druhy záloh

Záloha je kopie dat pořízená v určitém čase, která je uložena nezávisle na původních datech. Zálohy slouží k obnově dat v případě jejich ztráty [6]. Existuje několik druhů záloh, základním druhem je však plná záloha, která obsahuje kompletní kopii všech dat. Plná záloha může být následně základem pro rozdílové (diferenciální) zálohy nebo přírůstkové (inkrementální) zálohy, které při opakovaném zálohování mohou uspořit místo i čas.

V případě rozdílových záloh se ukládají pouze provedené změny od poslední plné zálohy. Pro obnovení tedy stačí plná záloha a poslední rozdílová záloha. Při použití přírůstkových záloh se také zaznamenávají pouze změny od poslední zálohy, ale ta na rozdíl od rozdílové zálohy nemusí být vždy plnou zálohou. Pro obnovení je tedy nutné mít plnou zálohu a k ní i všechny další přírůstkové, které byly od doby vzniku plné zálohy vytvořeny.

V souvislosti se zálohováním se často uvádí pojem archivace. Nejedná se však o totéž. Smyslem zálohování je zabránění ztráty potřebných dat, z tohoto důvodu se u zálohování řeší i proces obnovy dat. Zálohy neslouží k dlouhodobému uchování dat, dle knihy *Pro Data Backup and Recovery* [6] by neměly být zálohy použity pro uložení dat na dobu delší než 3 roky – k tomuto účelu slouží právě archivace, která se zaměřuje na dlouhodobé uchování dat.

1.3 Plánování záloh

Z pohledu aktuálnosti vytvořených záloh je nutné plánovat, jak často bude zálohování probíhat, z pohledu bezpečnosti je také nutné plánovat, po jak dlouhou dobu a na jakém místě se budou zálohy uchovávat. Dle místa uložení se rozdělují zálohy na *onsite* zálohy nebo *offsite* zálohy. V případě *onsite* zálohy je záloha uložena ve stejné lokaci jako zálohovaná data, v případě *offsite* zálohy je záloha umístěna v jiné geografické lokaci. Z pohledu rizika ztráty dat je uložení *offsite* bezpečnější (při krizové události v jedné lokaci jsou data uložena i v jiné lokaci), avšak na provedení je náročnější.

Pro výběr vhodného zálohovacího řešení je nutné znát i typ zálohovaných dat. Zálohu lze provést buď ve formě binární kopie celého disku, nebo lze data zálohovat i po jednotlivých souborech. Určitým kompromisem může být i hybridní přístup, při kterém se zálohují například jen použité bloky disku, k tomu je však nutná znalost zálohovaného souborového systému.

Důležitý je i způsob provedení zálohy – zálohy lze provádět buď *online* (za běhu systému), nebo *offline* (když je systém vypnutý). Některé programy umožňují provedení zálohy za běhu systému, jiné provedení zálohy za běhu neumožňují. Problematické bývá zejména provedení *offline* zálohy systémových disků, z kterých je spuštěn operační systém a další programy. Existují však řešení, která umí zálohovat systémové disky i za běhu operačního systému.

Při návrhu zálohování je nezbytné zvolit vhodné médium, na které budou zálohy ukládány. V současnosti jsou k dispozici dvě běžné možnosti – disky a magnetické pásky [6]. Pro běžného uživatele jsou vhodnější disky, které jsou jednodušší na použití a pro připojení k počítači není vyžadována speciální mechanika. Pro použití magnetických pásek je nezbytná větší počáteční investice do speciální mechaniky, avšak samotné pásky jsou levnější než pevné disky. Tato dvě média se také zásadně liší možností čtení a zápisu dat – na rozdíl od pevných disků neumožňují pásky náhodný přístup, avšak vyžadují sekvenční přístup.

Další možnosti bývají nabízeny i samotným zálohovacím softwarem. Z pohledu bezpečnosti zálohovaných dat může být zajímavé šifrování záloh, které zajistí důvěrnost ukládaných informací. Z uživatelského hlediska může být zajímavé například plánování a automatické vytváření záloh.

1.4 Zálohovací software

Na zálohování disků existuje celá řada programů pro různé operační systémy. Kromě komerčních řešení je na trhu i celá řada bezplatných alternativ. Určité základní řešení pro zálohování bývá i součástí moderních operačních systémů. Velmi často se však programy zaměřují na konkrétní způsob zálohování a pro různé druhy záloh může být nezbytné použít jiný software.

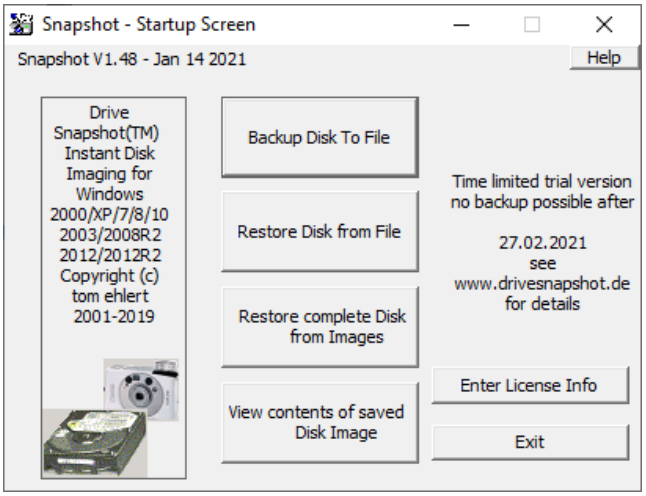
Z komerčních nástrojů je kromě programu Drive Snapshot známý například program Acronis True Image, který nabízí kompletní řešení problematiky zálohování disků. Dalším placeným nástrojem může být například EaseUS Todo Backup nebo Veeam Backup & Replication.

Alternativou z bezplatných nástrojů může být například nástroj FreeFileSync, který zajišťuje zálohování na úrovni jednotlivých souborů. Další možností může být například program Cobian Backup. Pro vytváření a obnovení *offline* záloh celých obrazů disků a diskových oddílů lze použít například bezplatný software Clonezilla.

Kapitola 2

Nástroj Drive Snapshot

Drive Snapshot[4], vyvíjený společností Tom Ehlert Software, je nástroj na zálohování disků pro operační systémy Microsoft Windows. Program umožňuje vytvořit obraz (zálohu) disku v konkrétním časovém okamžiku a uložit jej do jednoho či více souborů. Následně umí obnovit stav disku dle vytvořeného obrazu nebo obraz připojit jako virtuální disk. Program má dlouholetou historii, stále je však také vyvíjen. Poslední verze číslo 1.48 vyšla v prosinci 2019 [7].



Obrázek 2.1: Drive Snapshot – úvodní obrazovka

Program poskytuje nejen jednoduché grafické uživatelské rozhraní (GUI), ale nabízí i rozhraní pro ovládání z příkazové řádky (CLI) například pro použití ve skriptech. Pro použití není nutné program instalovat, celý je tvořen jedním spustitelným souborem `snapshot.exe` o velikosti pouhých 450 kB. Dle autora

[4] `http://www.drivesnapshot.de/en/index.htm`

program nijak nemodifikuje operační systém ani nijak nezasahuje do nastavení operačního systému [8]. Drive Snapshot má následující systémové požadavky:

- operační systém Microsoft Windows 2000, XP, Vista, 7, 8, 8.1, 10, Windows Server 2003, 2008, 2008 R2, 2012, 2016, 2019, Windows PE;
- 3 MB místa na disku (při instalaci);
- 32 MB RAM;
- uživatel musí mít administrátorská práva.

Mezi plně podporované souborové systémy se řadí FAT16, FAT32, NTFS a ReFS [9]. V omezené míře (např. bez podpory připojení) jsou podporovány souborové systémy Ext2, Ext3, Ext4, ReiserFS a XFS.

Drive Snapshot je komerční aplikace. K dispozici je funkčně neomezená zkušební verze, kterou je možné používat po dobu 30 dní. Po uplynutí této doby dojde k zablokování funkce na vytváření záloh a uživatel je pro další používání programu povinen zakoupit plnou verzi programu – licence stojí 39 eur za verzi pro počítač nebo 89 eur za verzi pro server (v případě operačního systému Windows Server) [10]. Uvedené ceny jsou platné pro elektronické licence (verze e-mail), licence na CD jsou o 10 eur dražší.

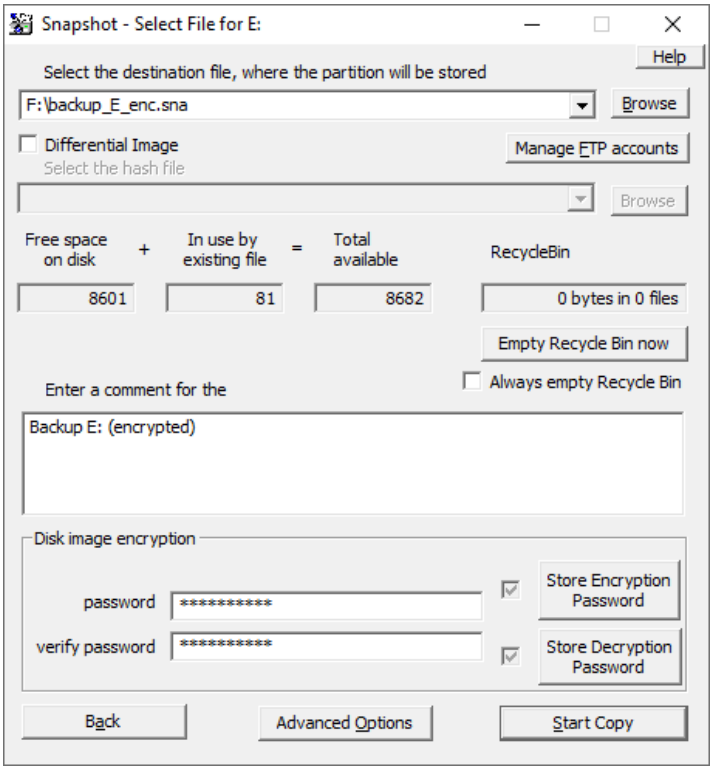
2.1 Zálohování disku

Jednou z hlavních výhod programu Drive Snapshot je schopnost vytváření záloh za běhu systému. Pro vytvoření obrazu disku není nutné provádět zálohu z jiného operačního systému, vše lze provést za běžného provozu, a to i v případě zálohy samotného systémového disku.

Vytvořený obraz disku bude vždy konzistentní a bude odpovídat stavu disku v čase spuštění zálohy. Při spuštění zálohy začíná Drive Snapshot odchytávat požadavky mezi souborovým systémem a ovladačem disku [9]. Pokud má dojít k zápisu na disk poté, co bylo spuštěno zálohování, provede se nejprve záloha daného místa a až poté bude zapsáno na disk. Během zálohování dochází také ke komprimaci dat, úspora paměti má dosahovat průměrně 50 % (závisí na typu dat na disku).

Při vytvoření zálohy z GUI obsahuje program jednoduchého průvodce, který provede uživatele celým procesem. Nejprve si uživatel vybere disk k zálohování a následně se mu zobrazí dialog (zobrazen na obrázku 2.2), ve kterém může upřesnit další nastavení zálohy.

Při spuštění z příkazové řádky lze možnosti zálohy upřesnit pomocí parametrů, které jsou popsány v podrobné dokumentaci [11]. CLI programu nabízí pokročilé funkce, které nelze z grafického rozhraní použít. Jednou z nich je například možnost vynechání konkrétních souborů či složek ze zálohy nebo použití asymetrického šifrovacího klíče.



Obrázek 2.2: Drive Snapshot – dialog nastavení zálohy

Vybrané možnosti nastavení zálohování, které Drive Snapshot umožňuje, popisuji v následujících podkapitolách.

2.1.1 Rozdělení zálohy do více souborů

Obraz disku může být uložen do jednoho souboru s příponou `.sna`, případně může být rozdělen do více souborů podle maximální velikosti jednoho souboru. To může být vhodné při uložení zálohy na více médií (například DVD). Výsledné soubory mohou být uloženy buď na disk v počítači (lokální či síťový), anebo mohou být nahrány na FTP server.

2.1.2 Vytváření rozdílových záloh

Kromě běžných záloh umí program vytvářet i rozdílové zálohy, kdy se při opakovaném zálohování neukládá vždy kompletní obraz disku, ale jen případné změny. Tyto rozdílové zálohy lze případně později opět sloučit do jednoho obrazu disku.

Při použití rozdílových záloh je nutné mít k danému `.sna` souboru se zálohou vygenerovaný také `.hsh` soubor s otisky sektorů. Ten se v základním nastavení generuje automaticky, případně jej lze vygenerovat dodatečně.

2.1.3 Šifrování záloh pomocí hesla

Drive Snapshot umožňuje kromě automatické komprese ukládaných dat tato data také šifrovat. Dle dokumentace [12] je pro šifrování dat použita symetrická šifra AES, která je v dnešní době standardem. Při obnovení nebo připojení šifrované zálohy je nezbytné zadat heslo, které bylo použito pro šifrování dané zálohy.

2.1.4 Šifrování záloh pomocí veřejného šifrovacího klíče

Kromě šifrování zálohy pomocí hesla je možné použít šifrování pomocí veřejného šifrovacího klíče [12]. Tato možnost, která byla do programu přidaná v minulé verzi 1.47, je dostupná pouze přes CLI.

Pro použití je nejprve nutné vygenerovat z hesla veřejný šifrovací klíč, jenž je uložen do textového souboru. Následně uživatel může vytvářet šifrované zálohy pomocí tohoto klíče. Takto vytvořená záloha odpovídá záloze, která by byla šifrována heslem, z nějž byl vygenerován klíč. Pro manipulaci se zálohou (čtení či obnovení) je nutné zadat heslo, které bylo použito pro vygenerování klíče. Šifrovací klíč nelze pro dešifrování použít a nelze jej převést zpátky na heslo.

2.1.5 Uložení výchozího hesla

Drive Snapshot umožňuje uložit heslo pro šifrování a dešifrování do registrů operačního systému. Šifrovací heslo slouží k vytváření záloh, dešifrovací heslo je použito k obnovení nebo čtení záloh. Šifrovací heslo je uloženo do registrů ve formě veřejného klíče a jeho uložení je dle autora bezpečné [12]. Dešifrovací heslo je v registrech uloženo v zašifrované formě. Autor v dokumentaci upozorňuje, že uložení hesla je sice pohodlné z uživatelského pohledu, avšak může to představovat určité bezpečnostní riziko.

2.1.6 Nastavení VSS

Volume Shadow Copy Service (VSS) je systémová služba, která umožňuje zajištění komunikace mezi zálohovací aplikací a běžícími aplikacemi. Pro vytvoření korektního obrazu disku za běhu systému může být tato komunikace nezbytná, příkladem může být potřeba databáze provést před zálohováním všechny transakce, aby byla data na disku v konzistentním stavu. VSS byla představena v operačním systému Windows Server 2003 a je součástí moderních verzí operačního systému Microsoft Windows. Tato služba se skládá z následujících součástí [13]:

VSS service je služba operačního systému, která má na starosti spolupráci a komunikaci ostatních částí VSS.

VSS requester je program, který si vyžádal vytvoření zálohy či jinou operaci se zálohami. Tím může být například systémová aplikace Windows Server Backup nebo zálohovací aplikace jako je Drive Snapshot.

VSS writer je komponenta, která je zodpovědná za to, že zálohovaná data budou v konzistentním stavu. Tyto komponenty jsou součástí operačního systému (například VSS writer pro registry), anebo bývají dodávané s aplikacemi, které je pro zálohování dat potřebují (Exchange Server).

VSS provider je komponenta zodpovědná za vytvoření obrazu disku. To může probíhat na úrovni software nebo hardware.

Vytvoření zálohy pomocí VSS probíhá následovně. VSS requester požádá VSS service o vytvoření zálohy. VSS writers zajistí konzistenci dat a připraví je k záloze. VSS provider provede vytvoření obrazu disku.

Pro zálohování disků používá Drive Snapshot buď vlastní systémový ovladač, který zajišťuje konzistenci dat a jejich kopírování, anebo umí využít právě službu VSS. Ve výchozím nastavení používá Drive Snapshot vlastní ovladač, který funguje i na starších verzích Windows a dle autora může být rychlejší než použití VSS [14]. Služba VSS je použita pouze v situaci, kdy uživatel zvolí zálohu více disků najednou nebo Drive Snapshot detekuje Microsoft Exchange 2003 či službu Active Directory, které je potřeba o záloze informovat a nechat je uvést do konzistentního stavu. Více informací o možnostech použití VSS lze nalézt v dokumentaci programu [14].

2.2 Obnovení disku ze zálohy

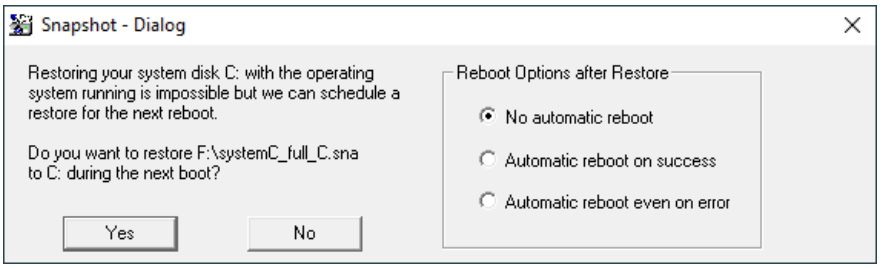
Obnovení disku ze zálohy lze (stejně jako ostatní úkony) provést buď z grafického rozhraní aplikace s pomocí jednoduchého průvodce, anebo z příkazové řádky. Obnovení disků lze provést i za provozu systému, pokud je splněna podmínka, že na cílovém disku není otevřený žádný soubor. Při spuštění obnovy Drive Snapshot tuto podmínku ověřuje a případně upozorní na procesy s otevřenými soubory na cílovém disku a vyzve k jejich ukončení.

Obnovení systémového disku za běhu operačního systému tedy není možné. Drive Snapshot pro tyto situace nabízí několik řešení. Systémový disk lze obnovit během restartu operačního systému, ze systému Windows PE (Preinstallation Environment) či Windows RE (Recovery Environment) nebo z operačního systému DOS.

2.2.1 Obnovení disku během restartu systému

Drive Snapshot od verze 1.40 umožňuje obnovení systémového disku během restartu operačního systému [15]. Pokud při obnově disku Drive Snapshot detekuje, že má být obnoven systémový disk, zahlásí chybu a nabídne obnovení

během restartu počítače. Následně, pokud uživatel souhlasí, dojde k restartu počítače a během startu operačního systému dojde k obnovení dat na systémovém disku. Toto řešení je plně automatizované a uživatel do něj nemusí nijak zasahovat.



Obrázek 2.3: Drive Snapshot – obnovení systémového disku během restartu

2.2.2 Obnovení disku z Windows PE nebo Windows RE

Windows PE (Preinstallation Environment) je speciální verze operačního systému Windows, která slouží pro instalaci či opravu systémů Windows nebo Windows Server. Pro svůj běh nepotřebuje pevný disk, lze jej spustit z USB disku, CD nebo DVD [16].

Windows RE (Recovery Environment) je nouzové prostředí pro opravy operačního systému Windows, které je založené na Windows PE. Ve výchozím nastavení je součástí standardní instalace Windows 10 a Windows Server 2016. Pokud není součástí systému, je možné jej spustit z instalačního média operačního systému [17].

V obou těchto prostředích lze spustit Drive Snapshot a obnovit (nejen) systémový disk ze zálohy. Na rozdíl od běžné verze Windows nemusí být možné použít 32bitovou verzi `snapshot.exe` na 64bitovém systému. V případě potíží je nutné použít 64bitovou verzi `snapshot64.exe`.

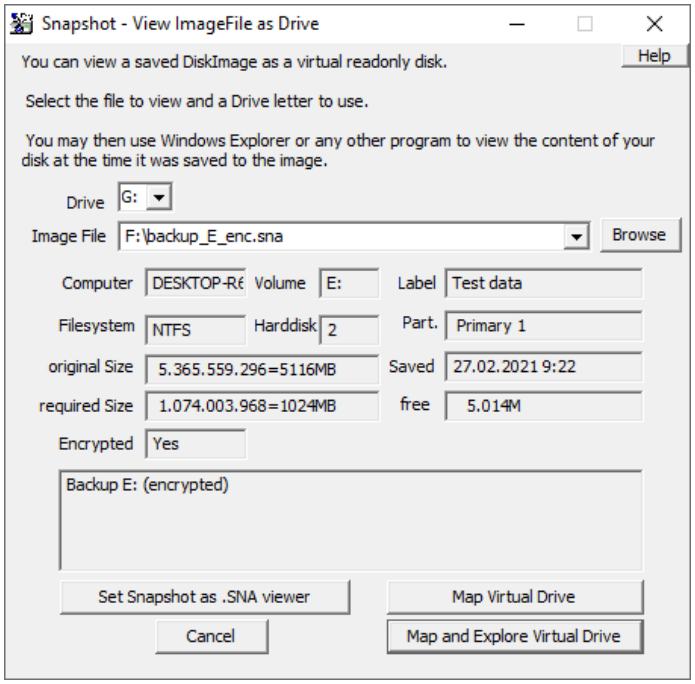
2.2.3 Obnovení disku ze systému DOS

Kromě předchozích možností je možné zálohu obnovit i ze systému DOS. Spustitelný soubor `snapshot.exe` obsahuje omezenou verzi programu Drive Snapshot pro DOS, která umožňuje zobrazovat informace o discích a souborech se zálohami, testovat, zda zálohy nejsou poškozené, a obnovovat je na disky. Podrobné informace o této možnosti jsou k nalezení na webu programu [18].

2.3 Prohlížení zálohy

Pro prohlížení diskových obrazů má Drive Snapshot zabudovaný nástroj, který umí připojit soubor se zálohou jako virtuální disk. Tento disk je připojen pouze

s právy ke čtení, nelze na něj zapisovat. Dostane však přidělené písmeno jednotky a lze s ním pracovat v dalších programech jako s běžnými disky. Tímto způsobem lze pohodlně prohlížet a případně (částečně) obnovit zálohovaná data.



Obrázek 2.4: Drive Snapshot – dialog otevření zálohy

Před otevřením souboru se zálohou zobrazuje Drive Snapshot základní informace o záloze – název počítače a disku, souborový systém zálohy, datum uložení, velikost zálohovaných dat, zda se jedná o šifrovanou zálohu a případnou poznámku připojenou k záloze. Kromě připojení disku přímo z programu Drive Snapshot lze nastavit připojení disku jako výchozí akci po otevření `.sna` souboru se zálohou ze správce souborů.

Kapitola 3

Reverzní analýza

Reverzní inženýrství (též reverzní analýza) je proces analýzy určitého předmětu, v této práci počítačového programu, bez znalosti jeho návrhu. Cílem reverzního inženýrství je vytvořit reprezentaci zkoumaného předmětu na vyšší úrovni abstrakce [19]. Během vývoje aplikací dochází k přeložení zdrojového kódu programu do strojového kódu – při tom se ztrácí velké množství informací, které činí zdrojový kód programu pro programátora čitelným. Pomocí technik reverzní analýzy lze zkoumat počítačový program i bez znalosti zdrojového kódu programu, tento postup je však časově náročný a vyžaduje často znalost instrukční sady procesoru.

Cílem této diplomové práce je provést bezpečnostní analýzu programu Drive Snapshot. Jelikož k programu nejsou dodávány zdrojové kódy, pro podrobnou analýzu funkčnosti programu je nutné použít techniky reverzního inženýrství. V následujících podkapitolách nejprve obecně zanalyzuji spustitelný soubor programu `snapshot.exe` a následně provedu reverzní analýzu těch částí programu, které jsem vyhodnotil z hlediska bezpečnosti programu jako kritické. Během reverzní analýzy jsem se zaměřil zejména na:

- analýzu souboru s vytvořenou zálohou;
- způsob generování náhodných čísel;
- odvození šifrovacího klíče ze zadaného hesla;
- zjištění použitých kryptografických algoritmů a jejich nastavení;
- práci s heslem a šifrovacím klíčem v paměti programu;
- možnosti ukládání hesel pro opětovné použití.

Reverzní analýzu programu Drive Snapshot jsem prováděl na operačním systému Windows 10 a využíval jsem množství nástrojů, zejména pak freeware

verzi IDA v7.0[5], x32dbg[6], Resource Hacker[7], CFF Explorer[8], Exeinfo PE[9], HxD[10] a další.

3.1 Analýza exe souboru

Drive Snapshot je tvořen jedním spustitelným souborem `snapshot.exe`. Program není nutné instalovat, stačí si stáhnout jeden soubor a ten spustit. Pro 64bitové prostředí (například Windows PE) nabízí autor 64bitovou verzi programu `snapshot64.exe`. Kromě těchto dvou variant je možné si stáhnout instalační soubor `setup.exe`, který provede klasickou instalaci. Součástí instalace je zkopírování 32bitové i 64bitové verze programu, dokumentace (kopie webu programu) a obrazu zaváděcí diskety operačního systému DOS do zvolené složky a nastavení asociace pro otevírání `.sna` souborů se zálohami.

V této diplomové práci jsem prováděl reverzní analýzu zkušební verze[11] programu Drive Snapshot ve verzi 1.48, staženou z webu[12] programu jako spustitelný soubor `snapshot.exe` o velikosti 445 kB. Tento soubor byl digitálně podepsán dne 14. ledna 2021 v 8:49:46 společností Tom Ehlert Software e.K. Detailní informace o verzi programu získané ze struktury `VERSIONINFO` obsahuje tabulka 3.1.

Tabulka 3.1: Detaily o analyzovaném programu ze struktury `VERSIONINFO`

CompanyName	Tom Ehlert Software
FileDescription	Drive Snapshot - Diskimaging for WindowsNT
FileVersion	1.48.18861
InternalName	Snapshot
LegalCopyright	Copyright © 2001-2019 by tom ehlert
LegalTrademarks	Drive Snapshot is a trademark of Tom Ehlert
OriginalFilename	snapshot.exe
ProductName	Drive Snapshot for WindowsNT
ProductVersion	1.48.18861

Soubor `snapshot.exe` obsahuje nejen 32bitový program pro operační systém Microsoft Windows, ale i omezenou verzi programu pro DOS (pro obnovu disku ze souboru zálohy), které se nebudu dále věnovat. Dále je součástí souboru `snapshot.exe` i vlastní systémový ovladač (ve 32bitové a 64bitové verzi), který Drive Snapshot využívá.

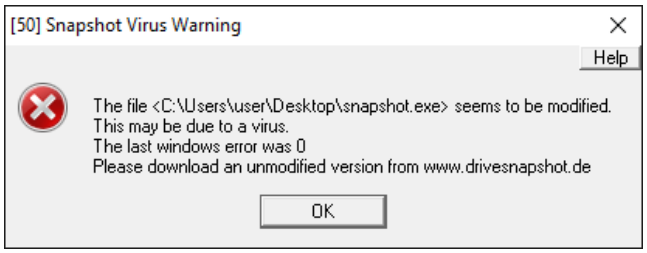
[5] `https://www.hex-rays.com/`
[6] `https://x64dbg.com/`
[7] `http://www.angusj.com/resourcehacker/`
[8] `https://ntcore.com/?page_id=388`
[9] `http://www.exeinfo.byethost18.com/`
[10] `https://mh-nexus.de/en/hxd/`
[11] Tato verze poskytuje plnou funkcionalitu programu po dobu 30 dní.
[12] `http://www.drivesnapshot.de/en/down.htm`

Program má v PE hlavičce v sekci DLL Characteristics označenou kompatibilitu s ASLR[13] a NX-bitem[14]. Tyto technologie, dle podpory a nastavení operačního systému a počítače, umožňují zvýšení bezpečnosti běžící aplikace.

Dle Exeinfo PE je nástroj Drive Snapshot napsaný v Microsoft Visual C++ verze 8, nástroj nepoužívá žádné externí knihovny (ani pro kryptografii či tvorbu GUI), využívá pouze Windows API. Spustitelný soubor programu je zkomprimovaný pomocí UPX[15]. Drive Snapshot je zkompilovaný s povolením funkce ochrany zásobníku (Stack Guard), většina funkcí programu obsahuje ve svém prologu vytvoření kanárka a v epilogu jeho kontrolu.

3.2 Ověření velikosti programu

Kromě výše zmíněných ochran má Drive Snapshot implementovanou jednoduchou bezpečnostní funkci, která během spuštění programu ověřuje velikost souboru s programem. Ve zkoumané verzi programu se testuje, zda je velikost menší nebo rovna 470 013 bytů. Jelikož soubor se zkoumanou verzí programu má velikost 455 960 bytů, tímto testem projde. V případě, že by byla velikost větší, tedy program by byl výrazně modifikován, spuštění programu selže, zobrazí se varování (zobrazené na obrázku 3.1) a Drive Snapshot se ukončí.



Obrázek 3.1: Drive Snapshot – upozornění na modifikaci programu

3.3 Formát zálohy

Nástroj Drive Snapshot používá pro uložení obrazu disku vlastní souborový formát. Vytvořený binární soubor se zálohou má příponu `.sna`, případně, pokud je záloha rozdělena do více souborů, mají tyto soubory stejný název, ale liší se příponou. První soubor zálohy má příponu `.sna`, následují přípony `.sn1` až `.sn9`, dále v případě potřeby `.s10` až `.s99` a v případě vyššího počtu souborů je přípona tvořena pouze číslicemi.

[13] ASLR (Address space layout randomization) je metoda, která umožňuje umístění programu na náhodné místo v paměti.

[14] NX-bit (Non eXecute bit) je technika pro rozlišení sekcí dat a kódu v operační paměti, resp. pro zakázání spuštění kódu ze sekce pro data.

[15] the Ultimate Packer for eXecutables (`https://upx.github.io/`)

Při použití rozdílových záloh je potřeba mít kromě souboru se zálohou vytvořený i hashovací soubor s příponou `.hsh`. Jelikož jsem se v této práci nezabýval analýzou fungování rozdílových záloh, budu se dále věnovat pouze popisu formátu `.sna` souboru s obrazem disku.

Soubor `.sna` se vnitřně dělí do různých sekcí, kdy každá sekce má svůj účel a její struktura se liší od ostatních sekcí. Každá sekce se však dá rozdělit na dvě části – hlavičku sekce a tělo sekce.

Hlavičky sekcí jsou pro všechny sekce stejné. Velikost hlavičky je vždy 12 bytů a obsahuje 3 hodnoty o velikosti 4 byty. První 4 byty jsou identifikátorem celé sekce, tyto 4 byty se dají interpretovat jako 4 ASCII znaky, dle kterých jsem převzal názvy sekcí. Za identifikátorem následuje 4bytové číslo bez znaménka, které představuje velikost těla sekce – pro velikost kompletní sekce (hlavička plus tělo sekce) je nutné k tomuto číslu přičíst velikost hlavičky, tj. 12 bytů. Poslední 4 byty hlavičky obsahují kontrolní součet celé sekce.

Tabulka 3.2: Struktura sekce v souboru se zálohou

adresa	obsah
sekce+0h	hlavička – identifikátor sekce (4 B)
sekce+4h	hlavička – velikost těla sekce (4 B)
sekce+8h	hlavička – kontrolní součet (4 B)
sekce+Ch	tělo sekce

Součástí programu je zabudovaná detekce poškozených souborů, ta umí detekovat například neplatné identifikátory či neplatné sekce. Přestože každá sekce obsahuje v hlavičce kontrolní součet CRC[16], není tento kontrolní součet kontrolován vždy. U běžné zálohy se kontroluje pouze u sekce SND0.

Za hlavičkou sekce následuje tělo sekce. V souboru se zálohou lze nalézt sekce SNTE, SDRI, SNV0, SND0, SNO0 a END_ nebo NEXT. V případě, že je soubor se zálohou šifrován, obsahuje dále sekci SNC2 nebo, pokud byla použita stará verze programu Drive Snapshot, sekci SNCR. Pokud se jedná o soubor s rozdílovou zálohou, může obsahovat ještě další sekce (například SNN0, SNI0, SNH0), analýze těchto sekcí jsem se však nevěnoval.

V následujících podkapitolách popisuji jednotlivé sekce, které obsahuje soubor s běžnou šifrovanou či nešifrovanou zálohou. Důraz je kladen zejména na sekci SNC2, která obsahuje data pro kryptografické algoritmy.

3.3.1 Sekce SNTE

Sekcí SNTE začíná každý `.sna` soubor s vytvořenou zálohou. Její identifikátor je číslo 0x45544E53 (pořadí bytů little endian). Při čtení po bytech interpretovaných jako znaky se získá právě řetězec SNTE.

[16] Cyclic redundancy check

Na začátku těla sekce se vyskytují informace o použité verzi programu Drive Snapshot – číslo verze, datum a čas kompilace a číslo sestavení. Po informacích o verzi programu následují textové řetězce obsahující informace o jméně souboru, verzi operačního systému, na kterém byla vytvořena záloha, informace o použité paměti a obsah příkazové řádky při spuštění programu.

3.3.2 Sekce SDRI

Po sekci SNTE navazuje sekce SDRI (identifikátor 0x49524453 při pořadí bytů little endian). Název sekce by mohl znamenat Snapshot Drive Information – začátek sekce obsahuje informace o disku, jako je číslo disku, počet cylindrů, počet stop na jeden cylindr, počet sektorů na jednu stopu a velikost sektorů a jejich množství. Za těmito informacemi následuje kopie MBR nebo GPT tabulky zálohovaného disku.

3.3.3 Sekce SNV0

Třetí sekce v souboru se zálohou bývá SNV0 (identifikátor 0x30564E53 při pořadí bytů little endian). Název sekce by mohl představovat Snapshot Volume Information, jelikož obsahuje informace o zálohované diskové jednotce.

Informace o disku z této sekce se zobrazují v dialogu při výběru souboru k obnovení disku či připojení zálohy jako virtuálního disku (obrázek 2.4). Sekce obsahuje název počítače na kterém byla záloha vytvořena, název a písmeno zálohované jednotky, název souborového systému, číslo disku a diskového oddílu, typ diskového oddílu, informace o využití disku (volné místo, obsazené místo), čas uložení zálohy a případnou uživatelskou poznámku k záloze.

3.3.4 Sekce SNC2

Sekce SNC2 (identifikátor 0x32434E53 při pořadí bytů little endian) je přítomná v souboru se zálohou pouze tehdy, je-li záloha šifrovaná. Název by mohl znamenat Snapshot Cryptography Information, sekce obsahuje všechny potřebné informace pro šifrování a dešifrování zálohy. Sekce se skládá z hlavičky (identifikátor sekce, velikost, kontrolní součet) a těla sekce. Detailní strukturu těla sekce zobrazuje tabulka 3.3.

Na začátku těla SNC2 sekce se nachází sůl, která byla použita pro výpočet otisku (hash) hesla. Při výpočtu otisku hesla se hashovací funkce volá opakovaně, počet opakování je uložen ve dvou 32bitových číslech obsahujících nižší a vyšší polovinu 64bitového čísla.

Proměnné *sncData1* a *sncData2* představují 2 čísla – každé o velikosti 19 937 bitů. Tato čísla jsou určitým mezivýsledkem při odvození klíče pro zašifrování či dešifrování hlavního šifrovacího klíče *AESkey*. Jejich podrobný popis je v podkapitole 3.5.

Součástí SNC2 sekce je i prvních 20 bitů šifrovacího klíče *AESkey*, který byl použit k zašifrování zálohy. V dokumentaci programu se uvádí, že to je z toho

Tabulka 3.3: Obsah těla SNC2 sekce

adresa	obsah	velikost
SNC2_body+0h	konstanta 0x11420	4 B
SNC2_body+4h	konstanta 0x00	4 B
SNC2_body+8h	sůl pro hashovací funkci	16 B
SNC2_body+18h	počet iterací hashovací funkce (nižší 4 B)	4 B
SNC2_body+1Ch	počet iterací hashovací funkce (vyšší 4 B)	4 B
SNC2_body+20h	*sncData1*	2500 B
SNC2_body+9E4h	*sncData2*	2500 B
SNC2_body+13A8h	prvních 20 bitů *AESkey* (zbytek nuly)	32 B
SNC2_body+13C8h	konstanta 0x14	4 B
SNC2_body+13CCh	konstanta 0x00	4 B
SNC2_body+13D0h	počet iterací šifrovací funkce (nižší 4 B)	4 B
SNC2_body+13D4h	počet iterací šifrovací funkce (vyšší 4 B)	4 B
SNC2_body+13D8h	zašifrovaný *AESkey*	32 B

důvodu, aby mohl program detekovat zadání špatného hesla [12]. Konkrétně se tam zmiňuje, že se jedná o otisk hesla, avšak tato informace není pravdivá.

Jako poslední je do sekce uložen zašifrovaný *AESkey* použitý k šifrování zálohy. Jelikož je toto šifrování *AESkey* prováděno opakovaně, je nutné kvůli zpětnému dešifrování uložit do SNC2 sekce počet těchto opakování. Počet opakování je uložen před samotným zašifrovaným *AESkey*, opět ve formě dvou 32bitových čísel představující jedno 64bitové číslo (obdobně jako u počtu iterací hashování).

Přesné vysvětlení principu odvození šifrovacího klíče z hesla, a tedy i vysvětlení proměnných z této kapitoly (*sncData1*, *sncData2*, počty iterací, klíč *AESkey* a další), je v podkapitole 3.5 Odvození šifrovacího klíče.

3.3.5 Sekce SND0

Soubor se zálohou je z většiny tvořen sekcemi SND0 (identifikátor 0x30444E53 při pořadí bytů little endian). Jako jediný druh se tyto sekce vyskytují v souboru opakovaně. Plný název sekce by mohl být Snapshot Data, jelikož každá z těchto sekcí obsahuje část dat ze zálohovaného disku. Struktura sekce je standardní, skládá se z hlavičky, za kterou následuje tělo sekce. Strukturu těla sekce zobrazuje tabulka 3.4.

Každá SND0 sekce obsahuje blok dat z disku. Tyto bloky bývají veliké 64 kB (odpovídá konstantě 0x10000 v tabulce 3.4). Adresa zálohovaného bloku představuje umístění dat na disku. Toto 64bitové číslo je rozdělené do dvou 32bitových čísel. *Velikost dat* představuje velikost uložených dat v sekci SND0. Jelikož jsou data z disku před zálohováním komprimována, je velikost dat v SND0 sekci běžně menší než 64 kB. Jako poslední jsou před uloženými daty

Tabulka 3.4: Obsah těla SND0 sekce

adresa	obsah	velikost
SND0_body+0h	adresa zálohovaného bloku (nižší 4 B)	4 B
SND0_body+4h	adresa zálohovaného bloku (vyšší 4 B)	4 B
SND0_body+8h	konstanta 0x10000 (velikost bloku)	4 B
SND0_body+Ch	*velikost dat*	4 B
SND0_body+10h	typ uložených dat	1 B
SND0_body+11h	nevyužito	3 B
SND0_body+14h	uložená data	*velikost dat*

uloženy informace o těchto datech, a to ve formátu bitového pole. Pokud je například nastavený 5 bit zprava (maska 0001000b), jsou uložená data šifrována.

3.3.6 Sekce SNO0

Sekce SNO0 (identifikátor 0x304F4E53 při pořadí bytů little endian) bývá předposlední sekcí v souboru se zálohou. Její název by mohl být Snapshot Offset. Sekce obsahuje informace o rozmístění jednotlivých SND0 sekcí v souboru se zálohou.

Tato sekce je nezbytná pro funkci obnovení celého disku z jednotlivých záloh a pro připojení souboru se zálohou jako virtuálního disku. Pro obnovení disku (obnovení jedné zálohy na jednu jednotku) tato sekce není v souboru potřeba, soubor ji nemusí ani obsahovat.

Sekce má (na rozdíl od předchozích sekcí) na svém konci speciální ukončení. Na pozici 8 bytů od konce se nachází 32bitové číslo vyjadřující velikost těla SNO0 sekce minus 8 bytů. Na posledních 4 bytech se nachází tag SRNE (0x45524E53 při pořadí bytů little endian).

3.3.7 Sekce END_ a NEXT

Sekcí END_ (identifikátor 0x5F444E45 při pořadí bytů little endian) nebo NEXT (identifikátor 0x5458454E při pořadí bytů little endian) bývá ukončen soubor se zálohou. V případě, kdy je záloha rozdělena do více souborů a pokračuje v dalším souboru, je použita sekce NEXT. Pokud současným souborem záloha končí, je použita sekce END_.

Obě sekce mají stejné rozložení, za hlavičkou sekce následuje tělo sekce, které slouží pouze jako výplň pro zaokrouhlení velikosti souboru na násobek 4 096 bytů. Na konci této sekce se na pozici posledních 8 bytů (obdobně jako u sekce SNO0) nachází velikost těla sekce minus 8 bytů a tag SRNE.

Při rozdělení zálohy do více souborů začíná navazující soubor sekcí SNCO (identifikátor 0x4F434E53 při pořadí bytů little endian). Obsah těla této sekce

je totožný se sekcí SNV0 – obsahuje informace o zálohovaném disku. Za touto sekcí poté následují datové SND0 bloky.

3.4 Generování náhodných čísel

Použití kvalitního (kryptograficky bezpečného) generátoru náhodných čísel je nezbytné pro zajištění bezpečnosti použitého kryptografického systému. Program Drive Snapshot používá generátor pseudonáhodných čísel pro generování šifrovacího klíče *AESkey* a odvození šifrovacího klíče z hesla.

Generování pseudonáhodných čísel zajišťuje vlastní funkce programu, která využívá pro generování buď systémovou funkci, anebo, pokud ta není k dispozici, implementuje vlastní generátor.

Ve výchozím stavu se Drive Snapshot snaží volat systémovou funkci *SystemFunction036* z knihovny advapi32.dll. Tato funkce představuje funkci *RtlGenRandom* z hlavičkového souboru ntsecapi.h [20].

Pokud není funkce *SystemFunction036* k dispozici (tato situace může nastat ve starších verzích operačního systému Windows), používá Drive Snapshot vlastní řešení pro generování pseudonáhodných čísel. Nejprve je volána funkce *srand*, kde je jako seed použit aktuální systémový čas (funkce *_time64*). Následně je pro každý generovaný náhodný byte proveden následující postup:

1. je zavolána funkce *rand*;
2. výstup z funkce *rand* je vložen do FPU[17] jednotky;
3. obsah FPU jednotky je vynásobený hodnotou $\frac{1}{32768}$;
4. obsah FPU jednotky je vynásobený hodnotou 255,0;
5. výsledek předchozích operací je převeden na celé číslo a nejnižší byte je použit jako výsledek.

3.5 Odvození šifrovacího klíče

Program Drive Snapshot umožňuje šifrovat zálohy disku pomocí zadaného hesla. V případě obnovení či připojení šifrované zálohy pak uživatel musí znát toto tajné heslo. V této podkapitole se zaměřuji na vztah mezi heslem a šifrovacím klíčem, kterým je záloha šifrována.

Pro šifrování a dešifrování zálohovaných dat používá Drive Snapshot symetrickou blokovou šifru AES-256, která pro šifrování i dešifrování používá stejný klíč o velikosti 256 bitů. Tento klíč, použitý pro šifrování či dešifrování zálohovaných dat, nazývám v celé práci *AESkey*.

[17] floating-point unit

Šifrovací klíč *AESkey* je při vytváření zálohy náhodně vygenerovaný funkcí pro generování náhodných čísel popsanou v předchozí podkapitole. Tento klíč tedy nijak nesouvisí s heslem, které zadal uživatel – nelze jej z hesla nijak odvodit. Z tohoto důvodu je nezbytné jej uložit v `.sna` souboru se zálohou.

Na konci SNC2 sekce se nachází zašifrovaná verze *AESkey*, která je zašifrována klíčem odvozeným z uživatelského hesla. Tento klíč budu nazývat *passwdKey*. V případě otevření či obnovení zálohy se z hesla uživatele odvodí právě *passwdKey* a pomocí něj se dešifruje zašifrovaná verze *AESkey* ze SNC2 sekce.

V následujících podkapitolách popisuji proces získání *AESkey* z hesla jak při vytvoření zálohy, tak při čtení (obnovení nebo připojení) zálohy.

3.5.1 Odvození šifrovacího klíče při vytvoření šifrované zálohy

Šifrovací klíč *AESkey* o velikosti 256 bitů je vygenerován náhodně. Při vytváření zálohy je nutné odvodit z hesla klíč *passwdKey*, pomocí kterého bude náhodně generovaný *AESkey* zašifrován.

Nejprve je nutné vytvořit otisk hesla. Pro to se používá vlastní implementace hashovací funkce SHA-256, která ze zadaného hesla vytvoří řetězec o délce 256 bitů. Hashovací funkce je implementována ve dvou funkcích – *SHA256processChunks* a *SHA256final*. První z nich, *SHA256processChunks*, postupně kopíruje vstup do hashovací funkce a vždy, když je naplněný celý chunk (64 bytů), provede hashování. Tato funkce pracuje se strukturou obsahující současný stav hashování předanou v argumentu funkce, v případě opakovaného volání se stejnou strukturou navazuje na předchozí stav. *SHA256final* provede poslední hashování, které zpracuje (dle definice SHA-256) poslední nezpracovaný chunk. Postup vytvoření otisku z hesla je následující:

1. vygeneruje se 16 bytů náhodných dat, které se použijí jako sůl při vytváření otisku hesla;
2. vytvoří se řetězec sůl+heslo;
3. opakovaně se volá funkce *SHA256processChunks* tolikrát, aby výpočet trval alespoň 500 ms, jako vstup se do funkce předává řetězec sůl+heslo;
4. zavolá se funce *SHA256final*, která dokončí hashování.

Výsledný otisk hesla vypočítaný funkcí SHA-256 se použije k dalšímu zpracování, tento otisk označuji jako *hashPasswd*. Dále se bude provádět modulární umocňování modulo prvočíslo $m = 2^{19937} - 1$. Pro toto umocňování se používá známý algoritmus Square&Multiply. Postup je následující:

1. $sncData1 = (0x1267)^{hashPasswd} \bmod m$;
2. vygeneruje se náhodné číslo *nonce* o velikosti 32 bytů;

3. $sncData2 = (0x1267)^{nonce} \bmod m$;

4. $passwdKey = (sncData1)^{nonce} \bmod m$, kde se jako výsledek použije pouze 32 nejnižších bytů.

Tím je vygenerován klíč *passwdKey*, který se použije pro zašifrování šifrovacího klíče *AESkey*. Pro šifrování je opakovaně volána funkce na zašifrování jednoho bloku šifry AES-256 tolikrát, aby toto opakované šifrování probíhalo alespoň 100 ms.

Tímto postupem byly vygenerovány všechny potřebné hodnoty pro šifrování a v budoucnu dešifrování souboru se zálohou. Do tohoto souboru se v sekci SNC2 uloží sůl pro hashovací funkci SHA-256, počet opakování hashování, *sncData1*, *sncData2*, 20 prvních bitů *AESkey* (pro ověření hesla), počet opakování šifrování *AESkey* a zašifrovaný *AESkey*. Náhodné číslo *nonce* se do souboru se zálohou neukládá.

3.5.2 Odvození šifrovacího klíče při čtení šifrované zálohy

Při čtení (obnovení či připojení) šifrované zálohy je nutné načíst ze SNC2 sekce zašifrovaný *AESkey* a ten dešifrovat. Postup je následující:

1. ze zadaného hesla se pomocí funkce SHA-256 vypočítá otisk hesla *hashPasswd* (pro výpočet se použije sůl a počet opakování uložené v sekci SNC2);

2. vypočte se $sncData1 = (0x1267)^{hashPasswd} \bmod m$ a ověří se, zda se rovná *sncData1* ze souboru se zálohou;

3. ze SNC2 sekce se načte *sncData2* a vypočte se klíč pro dešifrování *AESkey*, tedy $passwdKey = (sncData2)^{hashPasswd} \bmod m$;

4. ze souboru se zálohou se načte zašifrovaný *AESkey* a počet opakování šifrování, pomocí těchto údajů a šifrovacího klíče *passwdKey* se dešifruje;

5. pokud se shoduje prvních 20 bitů výsledku s hodnotou uloženou v SNC2 sekci, je získán (pravděpodobně) správný *AESkey*, tedy uživatel zadal správné heslo a může dojít k dešifrování zálohy.

Výpočet šifrovacího klíče *passwdKey* je založený na protokolu Diffie-Hellmanovy výměny klíčů, který využívá následující rovnosti:

$$\begin{aligned} passwdKey = (sncData1)^{nonce} &= ((0x1267)^{hashPasswd})^{nonce} = \\ &= ((0x1267)^{nonce})^{hashPasswd} = (sncData2)^{hashPasswd} \end{aligned}$$

Protokol Diffie-Hellman je do algoritmu odvození šifrovacího klíče zabudován kvůli možnosti použití asymetrické kryptografie při vygenerování šifrovacího klíče. Tato možnost je popsána v následující podkapitole.

3.6 Způsob asymetrického šifrování

Drive Snapshot obsahuje od předchozí verze 1.47 možnost používat k šifrování záloh veřejný šifrovací klíč [7]. Pomocí tohoto klíče lze vytvářet šifrované zálohy bez znalosti hesla, které je nutné použít k dešifrování. Tato funkcionalita je dostupná pouze přes CLI.

Veřejný šifrovací klíč lze vygenerovat pomocí CLI příkazem `snapshot.exe --pwgen=key.txt -pw=password`, kde *key.txt* je jméno souboru, do kterého bude klíč uložen, a *password* je heslo, které bude následně sloužit k dešifrování zálohy vytvořené tímto klíčem.

Soubor s klíčem je uložen v textové podobě, obsahuje 5 112 znaků, které představují hexadecimální číslice. 2 hexadecimální číslice tvoří jeden byte, celý soubor tedy obsahuje 2 556 bytů dat. Soubor s klíčem obsahuje tato data:

- sůl pro hashovací funkci (16 bytů);
- počet opakování hashování (8 bytů);
- *sncData1* (2 500 bytů);
- SHA-256 otisk předchozích dat (32 bytů).

Generování souboru s klíčem probíhá stejným způsobem jako při zálohování s heslem. Z hesla se výše popsaným způsobem vypočítá otisk hesla (volání hashovací funkce probíhá tolikrát, aby to trvalo alespoň 500 ms) a následně je proveden výpočet $sncData1 = (0x1267)^{hashPasswd} \bmod m$. Jako veřejný šifrovací klíč je uložena sůl, počet iterací pro hashovací funkci SHA-256 a číslo *sncData1*. Součástí veřejného šifrovacího klíče není heslo.

Pro vytvoření šifrované zálohy lze následně použít takto vytvořený veřejný šifrovací klíč. Zálohování je nutné spustit přes CLI a přidat parametr `-pwuse=key.txt`, kde *key.txt* je soubor s veřejným šifrovacím klíčem. Drive Snapshot ověří, zda je klíč validní (dle otisku SHA-256), uloží si proměnné z klíče a pokračuje v procesu zálohy. Tedy vygeneruje *nonce*, *AESkey*, vypočítá *sncData2*, *passwdKey*, zašifruje *AESkey* a zálohu a uloží potřebné údaje do SNC2 sekce v souboru se zálohou.

Záloha vytvořená pomocí veřejného šifrovacího klíče je ve stejném formátu jako záloha vytvořená pomocí hesla. Práce s takto vytvořenou zálohou tedy probíhá stejně – uživatel musí zadat heslo, které bylo použito k vygenerování použitého veřejného šifrovacího klíče.

3.7 Šifrování a dešifrování zálohy

Šifrování záloh probíhá pomocí symetrické blokové šifry AES (Advanced Encryption Standard), která byla v USA standardizována Národním institutem standardů a technologie (NIST) publikací FIPS PUB 197 [21]. Šifra AES je

standardizovaná ve třech variantách, velikost bloku je vždy 128 bitů, verze se liší délkou klíče a počtem rund. Přehled variant zobrazuje tabulka 3.5.

Tabulka 3.5: Varianty šifry AES [21]

verze AES	velikost bloku	velikost klíče	počet rund
AES-128	128 bitů	128 bitů	10
AES-192	128 bitů	192 bitů	12
AES-256	128 bitů	256 bitů	14

3.7.1 Implementace šifry AES

Šifra AES se skládá z několika operací. Blok šifry (16 bytů) je vložen do matice 4x4 označované jako stav, se kterou se během několika rund pracuje. Před první rundou se provedou operace *Key Expansion* a *AddRoundKey*. Poté se během každé rundy s výjimkou poslední provádí za sebou operace *SubBytes*, *ShiftRows*, *MixColumns*, *AddRoundKey*. Během poslední rundy se neprovádí operace *MixColumns*, provede se tedy pouze *SubBytes*, *ShiftRows* a *AddRoundKey*. Operace sčítání a násobení jsou definované na konečném tělese GF(2^8), operace sčítání je definována jako bitový xor, operace násobení je definována jako násobení polynomů modulo ireducibilní polynom $x^8 + x^4 + x^3 + x + 1$.

Key Expansion Před začátkem šifrování je nutné z šifrovacího klíče vygenerovat sadu rundovních klíčů – pro každou rundu a první volání *AddRoundKey* 16 bytů rundovního klíče. To má na starosti funkce *Key Expansion*.

SubBytes Tato operace je nelineární transformací, která se aplikuje na každý byte stavu. Transformace je prováděna pomocí substituční tabulky nazývané S-box.

ShiftRows Během operace *ShiftRows* dochází ke transformaci matice stavu na úrovni řádků matice. Řádky stavu, indexované od 0 do 3, jsou cyklicky posunuty o tolik pozic, jaký je jejich index.

MixColumns Operace *MixColumns* provádí transformaci matice stavu na úrovni sloupců. Každý sloupec stavové matice je vynásobený maticí 4x4 definované standardem.

AddRoundKey Při této operaci je kombinováno stavové slovo s odpovídajícím rundovním klíčem. Ke každému bytu stavového slova se přičte (operace xor) odpovídající rundovní klíč.

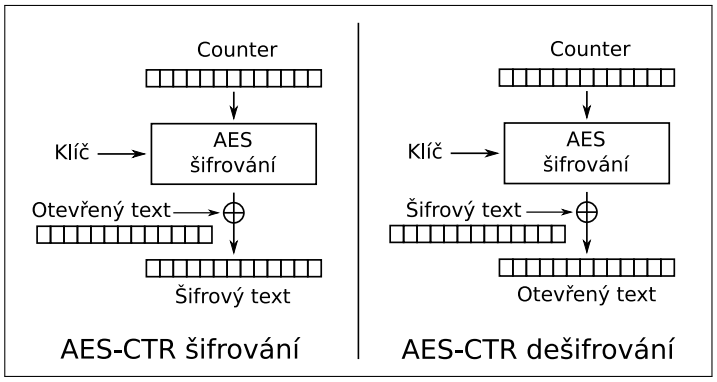
Drive Snapshot používá vlastní implementaci šifry AES. Ta je rozdělena do několika funkcí – zvlášť jsou implementovány funkce *Key Expansion* a její inverzní varianta pro expanzi klíče, zvlášť jsou pak implementovány funkce, které

zajišťují šifrování a dešifrování jednoho bloku (16 bytů) šifry AES. Součástí programu je podpora pro všechny varianty šifry dle délky klíče – AES-128, AES-192 i AES-256. Implementace šifry AES nevyužívá moderních instrukcí AES-NI[18] pro urychlení výpočtu, využívá však optimalizovanou implementaci pomocí T-boxů, které obsahují předpočítané výsledky po aplikaci operací *SubBytes*, *ShiftRows* a *MixColumns*.

3.7.2 Provozní režim šifry AES

Pro šifrování záloh se používá v současné verzi varianta AES-256 v režimu CTR[19]. V minulých verzích se dle dokumentace programu [12] používala varianta AES-128 CBC[20], kvůli zachování zpětné kompatibility umí pracovat Drive Snapshot i s touto konfigurací, avšak pro nové zálohy se již nepoužívá.

Při provozním režimu CTR je čítač použit jako vstup do šifrovací funkce AES. Takto zašifrovaný čítač je následně přičten (operace xor) na blok otevřeného textu a výsledek této operace je uložen jako šifrový text. V případě dešifrování je opět zašifrována hodnota čítače a přičtením (operace xor) výsledku k šifrovanému textu vznikne otevřený text [22]. Tento mód v podstatě převádí blokovou šifru na proudovou, kdy zašifrováním hodnoty čítače vzniká keystream použitý pro šifrování dat.



Obrázek 3.2: Šifrování a dešifrování v režimu CTR, odvozeno [23]

3.7.3 Průběh šifrování zálohy

Pokud uživatel vytváří šifrovanou zálohu, dojde k zašifrování uložených dat v SND0 sekcích pomocí šifry AES-256 v režimu CTR. K šifrování je jako klíč použit *AESkey*. Jiné sekce nejsou v souboru se zálohou šifrovány.

[18] Intel AES New Instructions
[19] Counter mode
[20] Cipher block chaining

Při vytváření SND0 sekce je volána funkce, která má na starosti zašifrování uložených dat (jedná se o diskové bloky po kompresi). V argumentech je šifrovací funkcí předán *AESkey* a jeho délka, buffer s daty k zašifrování, délka tohoto bufferu a výchozí hodnota čítače.

Jako výchozí hodnota čítače je použito nižších 32 bitů z adresy zálohovaného bloku (první 4 byty na začátku těla SND0 sekce). Těchto 32 bitů je následně rozšířeno na 128 bitů čítače tak, že je tato hodnota 4x zřetězena sama za sebe. Samotné šifrování jedné SND0 sekce pokračuje následovně:

1. jsou vypočítány rundovní klíče pro *AESkey* (funkce *Key Expansion*);
2. k hodnotě čítače (128bitové číslo) je připočteno číslo 1;
3. předpočítají se hodnoty čítače pro 8 bloků šifry AES (čítač se navýší vždy o hodnotu 1);
4. každý z 8 čítačů je (nezávisle, velikost čítače odpovídá velikosti bloku šifry AES) zašifrován šifrou AES-256 s klíčem *AESkey*;
5. je postupně zašifrováno 8 bloků ($8 * 16$ bytů) dat z bufferu tím způsobem, že je ke každému bytu přičten (operace) xor odpovídající byte zašifrovaného čítače;
6. pokud zbývá k zašifrování 8 nebo více bloků textu, pokračuje se od bodu číslo 3, pokud zbývá méně než 8 bloků, zašifrují se zbylá data.

3.7.4 Průběh dešifrování zálohy

Dešifrování dat ze sekce SND0 má díky vlastnostem režimu CTR na starosti stejná funkce jako v případě šifrování. Jako výchozí hodnota čítače se použije nižších 32 bitů z adresy zálohovaného bloku dat, tato 32bitová hodnota je následně rozšířena na 128 bitů opakováním své hodnoty. Zašifrováním postupně inkrementovaných hodnot čítače pomocí klíče *AESkey* vzniká keystream, jehož přičtením (operace xor) k šifrovaným datům dojde k dešifrování.

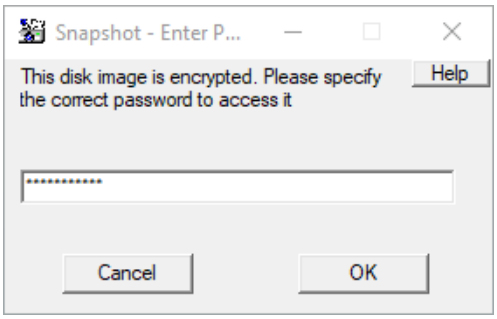
3.8 Načítání hesel k šifrování a dešifrování záloh

Existuje několik možností, jak lze programu předat heslo pro šifrování nebo dešifrování zálohy – zadáním do dialogu přes GUI, zadáním přes CLI nebo může být heslo uloženo do registrů (popsáno v kapitole 3.9).

Jednou z možností, kde uživatel může v GUI zadat heslo, je dialog pro vytvoření zálohy (obrázek 2.2). Zde jsou 2 textová pole pro zadání hesla, do kterých uživatel musí napsat stejné heslo – místo zadaných znaků jsou v polích hvězdičky, které nelze přepnout na znaky. Přestože tato textová pole mají omezenou velikost vstupu na 250 znaků, jsou hesla z polí čtena funkcí *GetDlgItemTextA*, a to do maximální délky 999 znaků.

Při práci se zašifrovanou zálohou může zadat uživatel heslo přes speciální dialogové okno (obrázek 3.3). Okno obsahuje jedno textové pole, které opět místo zadaných znaků zobrazuje hvězdičky. Maximální velikost vstupu do tohoto pole není omezena, jeho obsah je následně načtený funkcí *GetDlgItemTextA*, a to opět do maximální délky 999 znaků.

Další z možností je zadat heslo přes CLI parametrem `-pw=` při spuštění programu. V případě, že je potřeba použít v hesle mezeru, je nutné uzavřít heslo do uvozovek. Takto zadané heslo je (spolu s celým obsahem příkazové řádky) načteno pomocí funkce *GetCommandLineA*. Programem Drive Snapshot není aplikován žádný limit na velikost hesla, délka hesla je omezena pouze maximální délkou řetězce, kterou lze z příkazové řádky načíst (v případě cmd.exe to je 8191 znaků [24]). Tímto způsobem lze šifrovat a dešifrovat zálohy s delším heslem než je 999 znaků.



Obrázek 3.3: Drive Snapshot – dialog pro zadání hesla k dešifrování zálohy

3.9 Uložení hesla k zálohám do registrů

Aby uživatel, který chce pracovat s šifrovanými zálohami, nemusel opakovaně zadávat heslo, umožňuje Drive Snapshot uložit heslo do registrů operačního systému. V kontextu uložení hesla používá současná verze programu 2 termíny – šifrovací heslo a dešifrovací heslo.

Před verzí 1.47 měl program funkci pouze pro uložení výchozího hesla (nerozlišovalo se šifrovací a dešifrovací heslo). Toto heslo je použito jako výchozí během šifrovaní i dešifrování záloh. Tato možnost není v současné verzi přes GUI přístupná, tímto způsobem lze heslo uložit jen přes CLI parametrem `--setdefaultpwd=`. Při uložení hesla tímto způsobem je zadané heslo v současné verzi uložené zároveň jako šifrovací i dešifrovací heslo.

3.9.1 Šifrovací heslo

Šifrovací heslo lze uložit z GUI v dialogu pro vytvoření zálohy při zadání hesla nebo jej lze uložit přes CLI parametrem `--setdefaultencpwd=`. Délka ulože-

ného šifrovacího hesla není (kromě omezení daných způsobem načítání hesla) omezena. Pokud je uloženo šifrovací heslo, mělo by být automaticky použito při šifrování záloh [12]. To však platí pouze při zálohování přes CLI a pouze v případě, že není uloženo dešifrovací heslo. Pokud je uloženo v registrech dešifrovací heslo, je pro šifrování zálohy použito právě to (uložené šifrovací a dešifrovací heslo nemusí být stejné).

Šifrovací heslo není uložené ve formě textu, ale je uložené ve formě veřejného šifrovacího klíče, který byl odvozený z ukládaného hesla. Uložení klíče do registrů je bezpečné, veřejný šifrovací klíč nelze zpátky převést na heslo. Z tohoto důvodu lze uložené šifrovací heslo použít pouze k šifrování.

Veřejný šifrovací klíč (resp. šifrovací heslo) je uloženo ve formě binárních dat v registru HKCU\Software\SnapShot\DefaultEncryptionPassword. Odvození šifrovacího klíče probíhá obdobným způsobem jako při generování veřejného šifrovacího klíče do textového souboru. Struktura šifrovacího hesla uloženého v registrech je lehce odlišná, zobrazuje ji tabulka 3.6. Velikost šifrovacího hesla je vždy 5 024 bytů.

Tabulka 3.6: Struktura uloženého šifrovacího klíče v registrech

adresa	obsah	velikost
EncryptionPassword+0h	sůl pro hashovací funkci	16 B
EncryptionPassword+10h	počet iterací hashovací funkce	8 B
EncryptionPassword+18h	*sncData1*	2 500 B
EncryptionPassword+9DCh	*sncData2* (vždy samé nuly)	2 500 B

3.9.2 Dešifrovací heslo

Dešifrovací heslo lze uložit z GUI obdobně jako šifrovací heslo nebo jej lze uložit přes CLI parametrem `--setdefaultdecpwd=`. Dešifrovací heslo představuje textové heslo, které je v registrech uloženo v zašifrované podobě. Délka uloženého dešifrovacího hesla je omezena na 250 znaků, delší heslo nelze uložit.

Pokud je v registrech uloženo dešifrovací heslo, je automaticky použito při dešifrování záloh. V případě použití GUI je toto heslo automaticky vyplněno i v polích pro zadání hesla v dialogu pro vytvoření zálohy, bez zásahu uživatele tedy dojde k zašifrování zálohy s uloženým heslem. Při spuštění zálohy přes CLI je automaticky použito dešifrovací heslo k zašifrování zálohy také.

Dešifrovací heslo je uloženo ve formátu textového řetězce o 512 znacích v registru HKCU\Software\SnapShot\DefaultDecryptionPassword. Tento řetězec složený ze znaků hexadecimálních číslic představuje binární data o velikosti 256 bytů obsahující zašifrované heslo. Dešifrovací heslo je zašifrováno šifrou AES-128 v režimu CBC pomocí klíče, který je součástí programu. Na potenciální možnost dešifrování uloženého hesla pomocí klíče získaného ze

samotného programu je upozorněno v dokumentaci programu [12]. Samotný proces uložení hesla do registrů vypadá následovně:

1. Nejprve je nutné připravit data k šifrování:
   a) připraví se pole o velikosti 256 bytů inicializované na nuly;
   b) na pozici prvních 4 bytů se do pole uloží výsledek volání funkce *GetTickCount* (tato hodnota se dále k ničemu nepoužívá);
   c) na pozici 5. bytu se uloží velikost ukládaného hesla;
   d) od 6. bytu dále je vloženo ukládané heslo.
2. Následně jsou připravená data zašifrována tímto způsobem:
   a) vygeneruje se inicializační vektor (IV) tak, že se provede dešifrování jednoho bloku šifry AES-128 (16 bytů) plného nul klíčem[21], který je součástí programu Drive Snapshot;
   b) data z bodu 1 se zašifrují šifrou AES-128 v režimu CBC, pro šifrování se použije klíč a inicializační vektor z předchozího bodu.
3. Zašifrovaná data se překonvertují z binární podoby do hexadecimálního textového řetězce, který se uloží do registru.

3.10 Uložení hesla při obnově disku během restartu počítače

Obnovení systémového disku není možné provést za běhu operačního systému. Pokud je obnovení vyvoláno z GUI programu, nabídne Drive Snapshot obnovení systémového disku během restartu počítače, respektive během následujícího startu operačního systému Windows. V případě, že je záloha disku zašifrována, musí uživatel zadat během plánování obnovení heslo k rozšifrování zálohy. Toto heslo se již po restartu počítače při samotném obnovení nemusí znovu zadávat – musí tedy být někde uloženo.

Při obnovení systémového disku z GUI se zobrazí uživateli dialog informující o možnosti obnovy během restartu počítače (obrázek 2.3). V tomto dialogu může uživatel také upřesnit, zda se má po obnově disku automaticky spustit obnovený systém, anebo má Drive Snapshot počkat na reakci uživatele. Při souhlasu uživatele dojde k naplánování obnovy disku během příštího startu operačního systému. Následně je uživatel tázán, zda si přeje provést restart systému ihned, anebo až později.

Naplánování obnovy systémového disku probíhá následovně. Do klíče registru HKLM\System\CurrentControlSet\Control\Snapshot\ se zapíše několik

[21] Klíč použitý k zašifrování hesla jsem z programu Drive Snapshot úspěšně extrahoval a pomocí něj nezávisle otestoval tento postup. Z bezpečnostních důvodů jej však neuvádím.

hodnot obsahujících informace potřebné k obnovení zálohy. Patří mezi ně například hodnoty *Destination0* (cíl pro obnovení), *FullImage0* (cesta k souboru se zálohou), *Timeout* (doba po kterou lze přerušit proces obnovení) a *DefaultPassword* (heslo k záloze).

Heslo k záloze je v registru uloženo v zašifrované podobě – ve stejném formátu, v jakém je případně ukládáno dešifrovací heslo. Textový řetězec s heslem je zašifrovaný pomocí šifry AES-128 v režimu CBC pomocí tajného klíče, který je součástí programu Drive Snapshot. Binární data jsou v registru uložena jako hexadecimální textový řetězec. Velikost uloženého hesla je stejně jako u dešifrovacího hesla omezena na maximálně 250 znaků. V případě použití delšího hesla pro zašifrování zálohy nelze naplánovat obnovení této zálohy.

Během plánování zálohy je dále ze spustitelného souboru programu Drive Snapshot načten pomocí funkce *FindResourceA* binární zdroj s číslem 189 nebo 190 (dle detekované architektury procesoru) a tento zdroj je uložen jako soubor s názvem `snapnative.exe` do systémového adresáře (získaného pomocí funkce *GetSystemDirectoryA*). Soubor `snapnative.exe` je nativní aplikací[22], která bude spuštěna během následujícího startu systému a která provede obnovení systémového disku ze souboru se zálohou. Spuštění aplikace `snapnative.exe` během startu operačního systému je naplánováno pomocí přidání textového řetězce *snapnative.exe* do hodnoty *Boot execute* v klíči registru HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\.

Během následujícího startu systému má uživatel 10 vteřin na to, aby případně přerušil obnovení disku. Pokud obnovení přeruší, dojde k odstranění celého klíče HKLM\System\CurrentControlSet\Control\Snapshot\ a k odstranění naplánování spuštění obnovy z hodnoty *Boot execute*. K samotnému odstranění spustitelného souboru `snapnative.exe` ze systémového adresáře nedojde. Pokud uživatel obnovu nepřeruší, dojde k obnovení systémového disku ze souboru se zálohou a po dokončení celého procesu dojde ke spuštění obnoveného systému.

3.11 Načítání a uložení hesla k FTP serveru

Kromě možnosti uložení šifrovacího a dešifrovacího hesla do registru operačního systému nabízí Drive Snapshot možnost uložení přihlašovacích údajů k FTP serverům, na které umí program ukládat nebo z nich stahovat vytvořené zálohy. Údaje lze uložit buď přes CLI parametrem `--AddFTPAccount:`, nebo přes GUI skrze správu FTP účtu dostupnou z dialogového okna pro vytvoření zálohy.

[22]Nativní aplikace (native application) využívají přímo *native NT API* operačního systému, na rozdíl od běžných aplikací využívající Win32 API. Díky tomu mohou běžet před samotným zavedením Win32 subsystému. Jednou z nativních aplikací je například program Autochk (nástroj pro kontrolu disku) [25].

Pokud uživatel přidává FTP účet přes GUI, vyplní příslušné údaje do jednoduchého formuláře. Pole pro heslo zobrazuje místo zapsaných znaků hvězdičky, jeho velikost není omezena, ale při získání znaků z tohoto pole pomocí funkce *GetDlgItemTextA* je načteno maximálně 127 znaků. Při ukládání nejsou přihlašovací údaje vůči serveru otestovány.

Údaje se ukládají do registru KHCU\Software\SnapShot\FtpAccounts*id*, kde *id* představuje řetězec *username*+„.at.“+*server*+„.“+*port* (znak plus představuje operaci zřetězení). V tomto registru je uložena adresa FTP serveru (*server*), port serveru (*port*), uživatelské jméno (*username*) a heslo v zašifrované podobě. Šifrování hesla k FTP serveru probíhá způsobem lehce odlišným od šifrování dešifrovacího hesla:

1. vytvoří se řetězec *server*+*username*+*id*+*FTPkey*, kde *FTPkey* je řetězec, který je součástí programu[23];

2. řetězec z bodu 1 se zpracuje hashovací funkcí MD5 a uloží se jeho otisk o velikosti 16 bytů;

3. vypočítá se délka ukládaného hesla a vytvoří se řetězec, kde první byte představuje délku hesla a od 2. bytu je ukládané heslo;

4. řetězec s heslem vytvořený v bodu 3 se zašifruje pomocí šifrovací funkce AES-128 v režimu CBC, kde se jako šifrovací klíč použije otisk z hashovací funkce MD5 z bodu 2 a inicializační vektor je vytvořen dešifrováním jednoho bloku šifry AES-128 plného nul stejným šifrovacím klíčem;

5. takto zašifrované heslo je uloženo ve formě binárních dat o velikosti 256 bytů do registru.

3.12 Práce s citlivými údaji v paměti programu

Drive Snapshot může při svém běhu pracovat s velkým množstvím citlivých informací. Těmi může být například šifrovací klíč, vypočítané rundovní klíče, uživatelská hesla a další. Z pohledu bezpečnosti by se citlivé údaje měly v paměti vyskytovat pouze po nezbytně nutnou dobu a poté by měly být bezpečně smazány.

Pokud je v registrech uložené dešifrovací heslo, je do paměti načteno hned při samotném startu programu. V opačném případě je do paměti načteno až v moment zadání uživatelem. Toto heslo je následně přítomno v paměti po celou dobu programu, dle dokumentace například kvůli vytváření záloh více diskových oddílů [12].

[23] Řetězec *FTPkey* jsem také z programu Drive Snapshot extrahoval, avšak z bezpečnostních důvodů jej neuvádím.

Stejná situace nastává i s šifrovacím klíčem *AESkey* nebo s klíčem *passwdKey*, kterým je zašifrován *AESkey* v souboru se zálohou. V paměti jsou přítomny od vygenerování po celou dobu běhu programu, případně do doby, než jsou nahrazeny novými klíči (při šifrování či dešifrování další zálohy).

K bezpečnému mazání paměti až na výjimečné případy (kopie hesla uživatele, která se posílá do hashovací funkce) nedochází. Například vypočítané rundovní klíče pro AES zůstávají vždy na zásobníku, obdobné platí pro dešifrované dešifrovací heslo načtené z registru nebo číslo *nonce* použité při výpočtu *passwdKey*. Nejedná se jen o paměť na zásobníku, při načítání dešifrovacího hesla z registru je udělána kopie tohoto hesla na haldě. V průběhu procesu je následně tato kopie později dealokována pomocí funkce *free* bez bezpečného smazání.

Kapitola 4

Vyhodnocení bezpečnosti

V předchozí kapitole jsem popsal výsledky reverzní analýzy programu Drive Snapshot. Identifikoval jsem použité kryptografické algoritmy a jejich nastavení a zanalyzoval jsem z mého pohledu kritické části programu, které by mohly ohrozit bezpečnost. Vyhodnocení použitých algoritmů a objevené bezpečnostní chyby popisuji v následujících podkapitolách.

4.1 Použité kryptografické algoritmy

Pro šifrování uložených dat v souboru se zálohou je použita šifra AES-256 v provozním režimu CTR. Tato symetrická šifra je od roku 2002 přijata jako americký standard [21], ve verzi AES-256 je možné ji v USA použít pro zabezpečení vládních dokumentů až do úrovně přísně tajné [26].

Drive Snapshot používá vlastní implementaci šifry AES-256. Během analýzy jsem objevil závažné implementační chyby, které způsobují oslabení této šifry – jedná se například o vyzrazení nadpoloviční části šifrovacího klíče v souboru s uloženou zálohou (podkapitola 4.2.1) a nesprávné použití provozního režimu CTR (podkapitola 4.2.2).

Jako šifrovací klíč *AESkey* je použito 256 bitů náhodně vygenerovaných dat. Generování probíhá pomocí funkce *RtlGenRandom* nebo, pokud není tato funkce k dispozici, pomocí funkce *rand*. Bohužel dokumentace [20] k funkci *RtlGenRandom* nezmiňuje způsob generování náhodných čísel, pouze upozorňuje, že se místo této funkce má používat funkce *CryptGenRandom* (u které je ovšem v dokumentaci zmíněno, že je zastaralá [27]). Dle *Microsoft SDL Cryptographic Recommendations* [28] je však funkce *RtlGenRandom* kryptograficky bezpečná. Funkce *rand* není kryptograficky bezpečná [29].

Náhodně vygenerovaný šifrovací klíč *AESkey* je v souboru se zálohou uložen v zašifrované podobě. Šifrování probíhá (opakovaně) pomocí šifry AES-256. V implementaci šifrování klíče je bezpečnostní chyba popsaná v podkapitole 4.2.1.

Šifrovací klíč *AESkey* je zašifrován pomocí klíče *passwdKey*, který je odvozen z uživatelem zadaného hesla. K heslu je připojena náhodně vygenerovaná sůl (generována stejným algoritmem jako šifrovací klíč *AESkey*) a dochází k opakovanému hashování pomocí vlastní implementace funkce SHA-256. Následně je použit protokol Diffie-Hellmanovy výměny klíčů, pomocí kterého je ustanoven šifrovací klíč *passwdKey*.

Hashovací funkce SHA-256 není vhodná pro vytváření otisků hesel kvůli rychlosti jejího výpočtu. Pro tyto účely je doporučeno využívat nějakou speciální *key derivation function* (například PBKDF2), která je náročná na výpočetní výkon počítače [30]. Drive Snapshot kvůli zpomalení výpočtu provádí opakované volání hashovací funkce SHA-256, avšak počet iterací je navázán na výpočetní výkon počítače. Tento problém popisuji v podkapitole 4.2.3.

Bezpečnost Diffie-Hellmanova algoritmu pro výměnu šifrovacích klíčů je založena na problému řešení diskrétního logaritmu. Drive Snapshot ve výpočtu používá generátor $g = 0\text{x}1267$ a prvočíslo $p = 2^{19937} - 1$. Použitá kombinace g a p je nestandardní, nelze tedy jednoduše prohlásit tuto kombinaci i přes velikost čísla p za bezpečnou. Z hlediska bezpečnosti by bylo vhodné použít nějakou doporučenou ověřenou grupu a číslo g pro algoritmus Diffie-Hellman uvedenou například v dokumentech RFC 3526 nebo RFC 5114 [31, 32].

Drive Snapshot využívá vlastní implementaci hashovací funkce SHA-256 a symetrické šifry AES. Správnost implementace SHA-256 jsem ověřil pomocí knihovny OpenSSL, která pro několik náhodně vygenerovaných řetězců o délce 1 000 znaků vrátila stejný výsledek jako implementace hashovací funkce v programu Drive Snapshot. Správnost implementace šifry AES jsem ověřil pomocí úspěšného dešifrováním dat zašifrovaných programem Drive Snapshot pomocí knihovny OpenSSL.

4.2 Nalezené bezpečnostní chyby

Během reverzní analýzy programu Drive Snapshot jsem objevil celou řadu více či méně závažných bezpečnostních chyb. Jejich podrobný popis a návrhy na opravu zmiňuji v následujících podkapitolách.

4.2.1 Velká část šifrovacího klíče je zveřejněna v souboru se zálohou

Pro šifrování dat uložených v souboru se zálohou se používá šifra AES-256 v provozním režimu CTR, použitý šifrovací klíč *AESkey* má délku 256 bitů. Jelikož je šifrovací klíč vygenerován náhodně, je kvůli následnému dešifrování uložen v zašifrované formě na pozici 0x13E4 od hlavičky SNC2 sekce v souboru se zálohou. Kromě těchto 256 bitů je dále na pozici 0x13B4 od hlavičky SNC2 sekce uloženo v nezašifrované podobě prvních 20 bitů šifrovacího klíče kvůli možnosti ověření zadání správného hesla. Z důvodu špatné implementace

šifrování klíče *AESkey* a zveřejněním jeho prvních 20 bitů je v `.sna` souboru se zálohou zveřejněno celkem 148 z 256 bitů šifrovacího klíče.

Pro zašifrování klíče *AESkey* je také použita šifra AES-256. Při šifrování je volána funkce na zašifrování jednoho bloku šifry AES, které je na vstupu předaný ukazatel do paměti na *AESkey*. Protože velikost jednoho bloku šifry AES je 16 bytů, dochází k (opakovanému) šifrování pouze prvních 16 bytů šifrovacího klíče *AESkey*. Druhá půlka klíče zůstává nezašifrována.

Tato chyba vznikla pravděpodobně v předchozí verzi programu Drive Snapshot, kdy se pro šifrování začala používat šifra AES-256 namísto AES-128. U verze se 128bitovým klíčem se velikost klíče rovnala velikosti bloku – proto stačilo provést šifrování pouze jednoho bloku. To už nyní neplatí – AES-256 má 256bitový klíč, avšak velikost bloku je stále 16 bytů.

Přestože oslabení šifry z úrovně bezpečnosti 256 bitů na 108 bitů je razantní, stále by měla být ochrana před prolomením šifry dostatečná (dle doporučení NIST musí být od roku 2015 bezpečnost alespoň na úrovni 112 bitů, dříve platilo 80 bitů [33]). Prolomení šifry hrubou silou by zatím na současných počítačích nemělo být možné.

Oprava této chyby se skládá ze dvou částí. Jako první je nutné správně šifrovat oba dva bloky (256 bitů) šifrovacího klíče. To může být provedeno použitím šifrovacího režimu ECB[24] – tedy nezávislým šifrováním obou dvou částí šifrovacího klíče. Dále není nutné zveřejnit prvních 20 bitů šifrovacího klíče kvůli kontrole zadaného hesla – pro ověření správnosti stačí vypočítat proměnnou $sncData1 = (0x1267)^{hashPasswd} \bmod m$) a tu porovnat oproti uložené hodnotě v souboru se zálohou.

4.2.2 Chybné použití šifrovacího režimu CTR

Drive Snapshot používá pro šifrování zálohovaných dat operační režim CTR. Pro zajištění bezpečnosti tohoto operačního režimu je nezbytné, aby se použitý čítač (counter) během šifrování se stejným šifrovacím klíčem neopakoval [22].

Každá sekce SND0 má na začátku těla sekce uloženou adresu zálohovaného bloku. Tato hodnota je uložena jako dvě 32bitová čísla – nižší a vyšší část adresy. Do šifrovací funkce je jako výchozí hodnota čítače předána nižší část adresy, ta je následně rozšířena na 128 bitů tím způsobem, že je tato hodnota 4x sama za sebe zřetězena. Během šifrování je následně toto 128bitové číslo při šifrování každého bloku šifry AES inkrementováno o hodnotu 1.

Obraz disku je vytvářen po blocích o velikosti 64 kB, každý takový zálohovaný blok tvoří v souboru se zálohou jednu SND0 sekci. Jelikož zálohované bloky dat vždy začínají na adrese, která je násobkem 2^{16}, obsahuje vždy nižší půlka výchozí hodnoty čítače předávané do šifrovací funkce samé nuly. Z 32bitové výchozí hodnoty čítače se tak stává efektivně pouze 16bitová hodnota. Po

[24] Electronic code book

zálohování 2^{16} SND0 sekcí (resp. 4 GB dat z disku) dojde k opakování výchozí hodnoty čítače.

Pokud dojde k opakování čítače, tedy 2 či více SND0 sekcí je šifrováno pomocí stejného keystreamu, lze na tuto chybu zaútočit pomocí *known-plaintext attack* [34]. Z principu fungování operačního režimu CTR platí, že pokud se provede operace xor na dvou šifrových textech, které vznikly použitím stejného keystreamu, výsledkem této operace bude odpovídat operaci xor dvou otevřených textů – dojde k odstranění informace o keystreamu. Toho lze využít k různým variantám útoků.

Pokud zná útočník celý otevřený text, je odhalení použitého keystreamu, a tedy i dešifrování dalších bloků se stejným keystreamem, triviální. K této hypotetické variantě útoku je vhodné doplnit, že na začátku zálohy se bude nacházet MBR nebo GPT tabulka, která je zveřejněna (v původní nekomprimované podobě) v sekci SDRI. Pro úspěšný útok ale nemusí být nezbytně nutné znát celý otevřený text. Pomocí techniky zvané *crib-dragging* může být možné provést dešifrování pomocí odhadování možného obsahu šifrového textu.

Výsledkem opravy této chyby musí být zajištění jedinečnosti použitých čítačů v celé záloze. Používání 32bitové hodnoty (která efektivně obsahuje jen 16 bitů informace) pro inicializaci výchozí hodnoty čítače je nedostatečné. Řešením může být použití celé 64bitové adresy zálohovaného bloku jako inicializační hodnoty, kdy například vyšší polovina 128bitového čítače bude inicializovaná touto hodnotou a nižších 64 bitů bude od nulové hodnoty postupně inkrementováno.

4.2.3 Síla šifrování je závislá na výkonu počítače

Při vypočítání otisku ze zadaného hesla je hashovací funkce volána tolikrát, aby výpočet trval alespoň 500 ms. Obdobně při šifrování klíče *AESkey* je šifrování prováděno tolikrát, aby výpočet trval alespoň 100 ms. V obou případech je počet iterací určen při vytváření zálohy. Před zahájením výpočtu otisku hesla či před šifrováním klíče je uložen výsledek funkce *GetTickCount* a následně je po každém hashování či šifrování dalším voláním funkce *GetTickCount* srovnáno, jestli už nedošlo k překročení dříve zmíněného limitu. Chybou je, že není nastavena žádná minimální hodnota – tedy v nejhorším případě může být počet opakování roven pouze jedné.

Toto opatření má snížit počet hesel, které útočník zvládne při útoku v určitém čase vyzkoušet. Bohužel uživatel může být vystaven v omyl ohledně úrovně zabezpečení, protože například má slabý výkon počítače, což povede k oslabení zabezpečení.

Nejjednodušším řešením je přidat do algoritmu minimální počet iterací, který se musí (nezávisle na výkonu počítače) provést. Na implementaci složitějším, ale bezpečnějším řešením je výměna hashovací funkce SHA-256 za nějakou *key derivation function* (například PBKDF2 nebo Argon2), která je navržena tím způsobem, aby její výpočet byl pomalý a paměťově náročný.

4.2.4 Chybná práce s citlivými údaji v paměti programu

Citlivé údaje by se měly z bezpečnostních důvodů nacházet v paměti pouze po nezbytně nutnou dobu a poté, co tyto údaje nejsou potřeba, by mělo dojít k jejich bezpečnému smazání. Při dealokaci paměti pomocí funkce *Free* nebo při zrušení lokálních proměnných po návratu z funkce nedochází k bezpečnému smazání uložených dat – obsah paměti může zůstat nezměněný. Tato paměť, ve které mohly zůstat citlivé údaje, může být později programu opět přidělena. Z tohoto důvodu je nutné citlivé údaje před uvolněním paměti přepsat.

Drive Snapshot to však až na výjimečné situace nedělá. Po návratu z funkce zůstávají na zásobníku například vypočítané rundovní klíče pro šifru AES, načtené heslo z registru v otevřeném textu nebo vygenerované číslo *nonce*. Při uvolňování alokované paměti zůstává na haldě například kopie dešifrovacího hesla načteného z registru nebo šifrovací a dešifrovací heslo.

Kromě výše zmíněného je po celou dobu běhu programu (respektive od chvíle, kdy jsou citlivé informace načteny) uložen například hlavní šifrovací klíč *AESkey*, šifrovací a dešifrovací heslo a šifrovací klíč, kterým je v souboru se zálohou zašifrovaný klíč *AESkey*. Ani tato data nejsou při ukončení programu bezpečně smazána, navíc lze diskutovat o tom, zda je potřeba, aby byly v paměti uloženy po dokončení šifrování či dešifrování.

Přítomnost citlivých dat v paměti je možné jednoduše ukázat při použití programu Drive Snapshot ve virtuálním prostředí. Pomocí programu provedu zálohu disku, kterou budu šifrovat zvoleným heslem. Pokud po provedení šifrování uložím obraz paměti virtuálního prostředí, naleznu v něm použité heslo v čitelné podobě.

Uložení hesla a dalších citlivých informací v paměti po celou dobu běhu programu lze chápat kvůli pohodlnějšímu použití programu – při vytváření více šifrovaných záloh nemusí uživatel vícekrát zadávat stejné heslo, případně po provedení šifrované zálohy si ji může bez zadání hesla připojit jako virtuální disk a zkontrolovat ji. Program by však měl mít tuto možnost jako volitelnou a nabízet alternativu, kdy kvůli vyšší bezpečnosti budou citlivá data uložena pouze po nezbytně nutnou dobu.

Kromě omezení doby výskytu citlivých dat v paměti by měl program bezpečně zacházet s těmito daty poté, co nejsou dále potřeba. Citlivé proměnné by měly být vždy bezpečně přepsány. Heslo by po vygenerování otisku mělo být bezpečně smazáno, šifrovací klíč *AESkey* by měl být po dokončení šifrování bezpečně smazán. Obdobné platí pro veškeré citlivé údaje.

4.2.5 Délka hesla zadaného přes CLI je zveřejněna v souboru se zálohou

Pokud uživatel vytváří šifrovanou zálohu přes CLI a nemá uložené šifrovací či dešifrovací heslo a ani nepoužívá šifrovací klíč, musí zadat heslo k záloze přes parametr `-pw=`. Celý obsah příkazové řádky se následně uloží v čitelné

podobě do SNTE sekce, která se nachází na začátku souboru se zálohou. Heslo (respektive řetězec za `-pw=`) je zaměněno za hvězdičky, avšak počet hvězdiček odpovídá počtu znaků v hesle. Útočník tedy přímo ze souboru se zálohou získává informaci o hesle. Pokud by se útočník snažil zjistit použité heslo pomocí útoku hrubou silou nebo slovníkovým útokem, informace o počtu znaků použitých v hesle snižuje počet možných hesel a tím zvyšuje pravděpodobnost úspěšného útoku.

Pro opravení této bezpečnostní chyby je nutné upravit funkci, která má na starosti skrytí hesla z uložené příkazové řádky. Řešením může být nahrazení hesla jedinou hvězdičkou, čímž se odstraní informace o délce hesla.

4.2.6 Část hesla zadaného přes CLI může být zveřejněna v souboru se zálohou

Tato chyba navazuje na předchozí chybu popsanou v podkapitole 4.2.2. Každý `.sna` soubor s vytvořenou zálohou obsahuje v SNTE sekci obsah příkazové řádky. Pokud uživatel spustí vytvoření šifrované zálohy přes CLI a zadá heslo přes parametr `-pw=`, je zadané heslo nahrazeno hvězdičkami (počet hvězdiček odpovídá počtu použitých znaků). Funkce pro skrytí hesla však nefunguje vždy správně.

V případě, že chce uživatel pro šifrování použít heslo obsahující mezeru, musí při použití CLI uzavřít heslo do uvozovek. Pokud uživatel například zadá na příkazový řádek příkaz `C:\>snapshot -pw="A very strong password" E: F:\backup.sna`, vytvoří zálohu disku E: zašifrovanou pomocí hesla `A very strong password`. Do vytvořeného souboru se zálohou je poté v SNTE sekci zapsán textový řetězec `*CmdLine="C:\snapshot.exe" -pw=** very strong password" E: F:\backup.sna*`.

Na výše zmíněném příkladu lze pozorovat, že funkce pro skrytí hesla nepočítá s možností, že může být heslo z důvodu použití mezery v hesle uzavřeno do uvozovek. Funkce pouze nalezne řetězec `-pw=` a znaky za ním mění na hvězdičky, dokud nenačte znak mezery nebo konec řetězce. Za první použitou mezerou tak již není heslo nahrazeno hvězdičkami.

Špatná práce s řetězcem hesla platí jenom pro funkci na zakrytí hesla hvězdičkami. Vzniklá záloha je zašifrovaná celým heslem (bez uzavíracích uvozovek). V kombinaci s předchozí chybou však může být prolomení i velmi dlouhého hesla s mezerami triviální, protože útočník bude muset prolomit pouze část hesla před první mezerou. Oprava tohoto chování by měla být proto provedena společně s opravou chyby z předchozí kapitoly, a to tak, aby v souboru se zálohou nebyla žádná informace o hesle.

4.2.7 Konverze kódování hesla může snížit bezpečnost hesla

Drive Snapshot pracuje interně se všemi řetězci v kódování ANSI, které je závislé na použité kódové stránce. Pokud uživatel použije v hesle znaky, jež

neobsahuje jeho kódová stránka, dojde ke konverzi, která může vést k oslabení hesla. Dešifrování takto vytvořené zálohy pak může být velmi jednoduché.

Pro načítání řetězců (včetně zadávaného hesla) z dialogového okna GUI používá Drive Snapshot funkci *GetDlgItemTextA*. Při zadání znaků, které nejsou součástí kódové stránky uživatele, dochází ke konverzi. Pokud použitý znak nemá v dané kódové stránce podobnou náhradu, dojde k jeho náhradě za znak otazníku (kód 0x3F). V případě zadávání hesla o tom ani uživatel není informován, protože obsah pole s heslem je skrytý (nahrazený hvězdičkami). Ke stejné chybě dojde, pokud je Drive Snapshot spuštěn z příkazové řádky, na které bylo zadáno heslo přes parametr `-pw=`. Obsah příkazové řádky je načten pomocí funkce *GetCommandLineA*, která provede konverzi znaků nenacházejících se v použité kódové stránce.

Pokud například uživatel používající českou lokalizaci Windows (resp. kódovou stránku Windows-1250) použije v hesle znaky azbuky, dojde k nahrazení všech znaků hesla za otazníky. Tedy pokud uživatel zadal jako heslo k šifrování řetězec `Приветмир`, dojde skrytě ke konverzi na řetězec `??????????`. Pro dešifrování této zálohy stačí zadat heslo skládající se z 9 otazníků.

V kombinaci s chybou popsanou v kapitole 4.2.5, která dává informaci o počtu znaků v hesle, může být prolomení hesla značně zjednodušeno. Pro opravu chyby by bylo nejlepší nahradit všechny ANSI verze knihovních funkcí pracujících s textovými řetězci za Unicode verze. Tedy například nahradit funkci *GetDlgItemTextA* za funkci *GetDlgItemTextW*. Rychlým řešením může být použití kontroly, zdali nedochází při zadání hesla ke konverzi pomocí načtení hesla v Unicode a srovnání s výsledkem konverze z Unicode do ANSI a zpátky do Unicode.

4.2.8 Nekonzistentní načítání hesla

Při šifrování či dešifrování zálohy může uživatel zadat heslo několika různými způsoby – při použití CLI přes parametr `-pw=`, v GUI přes dialog pro vytvoření zálohy nebo přes dialog pro zadání hesla při otevření zálohy. Další způsob může být práce s heslem přes uložené dešifrovací heslo. Kromě hesla k šifrování může uživatel zadávat také heslo k FTP účtu v dialogu pro správu FTP účtů. Tyto různé způsoby mají různé podmínky kladené na heslo a v určitých případech může nastat situace, že je pro šifrování použito heslo kratší než uživatel očekával.

Při zadání hesla přes dialog pro vytvoření hesla je vstupní pole omezeno na 250 znaků, funkce, která z pole heslo načítá, je omezena na 999 znaků. Dialog pro zadání hesla při otevření zálohy nemá omezenou vstupní velikost, ale je z něj načteno maximálně 999 znaků. Při načtení hesla z příkazové řádky je délka načteného hesla omezena systémovým nastavením (pro cmd.exe přes 8 000 znaků [24]). Uložené dešifrovací heslo může mít maximálně 250 znaků. Pole pro zadání FTP hesla nemají délku vstupu omezenou, ale je z nich načítáno maximálně 127 znaků.

Pokud chce uživatel například použít své dlouhé heslo (delší než 250 znaků) ze správce hesel pro zašifrování zálohy a toto heslo vloží ze schránky do dialogu pro vytvoření zálohy, dojde ke zkrácení hesla na 250 znaků a uživatel o tom není nijak informován. Vzhledem ke zkrácení hesla nebude uživatel moci s daným heslem otevřít vytvořenou zálohu, protože v dialogu na zadání hesla při dešifrování není omezena délka na 250 znaků, takže nedojde ke zkrácení hesla na stejný řetězec jako při vytvoření zálohy. Nejen že dojde k nepochopitelné chybě pro uživatele, ale také je bez jeho vědomí použito kratší heslo a tím je oslabena bezpečnost šifrování.

Program by měl mít sjednocené požadavky na délku hesla a v případě překročení maximální délky hesla by měl být uživatel na tento fakt upozorněn. Informace o omezeních kladených na heslo by měly být zmíněny v dokumentaci programu.

4.2.9 Špatné zpracování argumentů na příkazovém řádku

Zpracování argumentů předaných přes CLI nefunguje vždy tak, jak by bylo očekávané. Různé druhy argumentů reagují rozdílně na nevalidní vstupy, čímž mohou způsobit použití mnohem slabšího hesla, než uživatel očekával.

Problém tvoří zejména mezery. Pokud chce uživatel použít v hesle mezeru, musí řetězec s heslem uzavřít do uvozovek. Toto chování není v dokumentaci popsané. Následující tabulka 4.1 zobrazuje různé druhy příkazů zadané přes CLI a jejich vyhodnocení programem Drive Snapshot.

Tabulka 4.1: Vyhodnocení různých příkazů zadaných přes CLI

#	příkazový řádek	výsledek operace
1	`snapshot.exe C: D:\image.sna -pw=two words`	Chyba *invalid argument*. Očekávané chování.
2	`snapshot.exe C: D:\image.sna -pw="two words"`	Vytvoří se záloha šifrovaná heslem `two words`. Očekávané chování.
3	`snapshot.exe --setdefaultpwd=two words`	Uloží se `two` jako výchozí heslo. Neočekávané chování, očekávána chyba *invalid argument*.
4	`snapshot.exe --setdefaultpwd='two words'`	Uloží se `'two` jako výchozí heslo. Neočekávané chování, očekávána chyba *invalid argument*.
5	`snapshot.exe --setdefaultpwd="two words"`	Uloží se `two words` jako výchozí heslo. Očekávané chování.
6	`snapshot.exe -pwgen=key.txt -pw=two words`	Vytvoří se veřejný šifrovací klíč pro heslo `two`. Neočekávané chování, očekávána chyba *invalid argument*.

První a druhý příkaz vytvoří šifrovanou zálohu disku C:, kdy soubor se zálohou je uložen na disk D: do souboru `image.sna`. Příkaz číslo 1 hned po spuštění zobrazí chybu a ukončí program. To je očekávané chování, protože je heslo zadané ve špatném formátu, a tedy na příkazové řádce je slovo navíc. Druhý příkaz vytvoří zálohu šifrovanou celým řetězcem `two words`, tedy použije správně celé heslo s mezerou.

Příkazy 3, 4 a 5 ukazují rozdílné chování oproti předchozímu příkazu. Ve 3. případě se nastaví jako výchozí heslo řetězec `two`. Heslo je bez varování nečekaně oslabeno, očekávané chování by bylo zobrazení chyby *invalid argument* z důvodu nadbytečného slova `words`. Pro vstup číslo 4 platí totéž, co pro vstup číslo 3. Jednoduché uvozovky nelze použít pro ohraničení řetězce s heslem, znak jednoduché uvozovky se použije jako znak hesla. Příkaz číslo 5 zobrazuje správné chování, kdy je uloženo celé heslo.

Jako poslední je v tabulce uveden příkaz pro vytvoření veřejného šifrovacího klíče a jeho uložení do souboru. Tento příkaz se chová obdobně jako příkaz pro uložení výchozího hesla, v tabulce zmiňuji pouze verzi, která způsobí to, že použité heslo bude bez upozornění znatelně oslabeno. Očekávané chování by bylo vypsání chyby *invalid argument* kvůli nadbytečnému slovu na příkazové řádce. Pokud by bylo prohozeno pořadí `-pwgen=` a `-pw=`, program se zachová stejně – vygeneruje klíč pro heslo skládající se pouze z prvního slova bez jakéhokoliv upozornění.

Stejným způsobem, jakým se chová argument `--setdefaultpwd=`, se chovají i argumenty `--setdefaultencpwd=`, `--setdefaultdecpwd=` a `-pwgen=` ve spojení s `--pw=`.

Kromě výše zmíněného problému má Drive Snapshot ještě problém se zpracováním samotných uvozovek, které mohou uzavírat řetězec s heslem. Všechny následující řetězce se při předání přes argument `-pw=` interpretují jako `heslo` – `"heslo"`, `"he"slo`, `"h"e"s"l"o"`, `""heslo"` i `""hes"lo"`. V jiných případech však dojde na jinou interpretaci (takto vytvořené zálohy nelze dešifrovat s heslem `heslo` – `"he""slo`, `""hes"lo""` nebo například `h"""eslo`).

Program by měl sjednotit kód a jeho logiku na zpracování vstupu z příkazové řádky. Ve všech případech by se měl program chovat konzistentně a ideálně upozorňovat na špatně zadané heslo (nadbytečná slova na příkazové řádce). Kromě toho je potřeba opravit kód na zpracování uvozovek.

4.2.10 Heslo k FTP účtu může být zveřejněno v souboru se zálohou

Při vytváření zálohy disku umí Drive Snapshot uložit vytvořený soubor se zálohou na FTP server. Pokud uživatel spouštěl zálohu pomocí CLI programu, může být heslo k FTP serveru, na který je záloha nahrávána, uloženo v čitelné podobě v souboru se zálohou.

Pro uložení souboru se zálohou na FTP server je nutné zadat název souboru ve tvaru `ftp://username:password@server:port/path/name.sna` [35].

Při vytváření zálohy pomocí CLI je sice v konzoli při informačních výpisech nahrazeno heslo v názvu souboru hvězdičkami (ovšem jejich počet odpovídá počtu znaků v hesle – obdoba bezpečnostní chyby 4.2.5), avšak plný obsah příkazové řádky je uložen v textové podobě do SNTE sekce. Pokud se útočník dostane k souboru se zálohou, získá z něj jednoduše heslo k FTP serveru. To platí i pro šifrované zálohy, jelikož SNTE sekce není nikdy zašifrována.

Této chybě se lze vyhnout uložením přihlašovacích údajů k FTP účtu. Poté již není nutné v názvu souboru udávat heslo k FTP účtu. Uložení účtu lze provést přes GUI i CLI programu. Je ovšem nutné brát v potaz, že údaje k FTP účtům jsou v registrech uloženy sice v zašifrované podobě, avšak není problém je dešifrovat (podrobněji bezpečnostní chyba 4.2.2). Nejjednodušší opravou této chyby je neukládat obsah příkazové řádky do SNTE sekce v souboru se zálohou. Kromě hesla k FTP účtu může odhalovat další citlivé údaje, jako je například adresářová struktura.

Kromě zveřejnění hesla v SNTE sekci je toto heslo dále zobrazováno v čitelné podobě v informačních výpisech při obnovení zálohy či pokusu o její připojení jako virtuálního disku (které vždy skončí s chybou). Heslo by po zadání nemělo být zobrazováno, případně může být nahrazeno například jednou hvězdičkou tak, aby nedošlo k úniku informace o délce hesla.

4.2.11 Heslo k FTP účtu může být v čitelném formátu uloženo v registrech

Funkce pro nahrávání a ukládání záloh na FTP server je dostupná i z GUI programu Drive Snapshot. Pokud uživatel nepoužije uložený FTP účet, ale zadá název souboru v požadovaném formátu[25] pro použití FTP serveru, je heslo, respektive celý název souboru, uloženo v čitelné podobě v registrech.

Textová pole na zadávání jména souboru se zálohou v GUI programu si pamatují historii posledních 10 použitých souborů. V případě, že je použit soubor na FTP serveru a je v názvu specifikováno heslo (tedy účet není uložen), je v historii posledních použitých souborů uložený celý název souboru včetně hesla k FTP serveru. Tyto záznamy o posledních souborech navíc nelze z programu smazat (jsou přítomny, dokud se historie nepřepíše). Informace o posledních použitých souborech je uložena v čitelné podobě v registrech v klíči HKCU\Software\SnapShot\.

Obdobně jako v předchozí chybě je v případě použití hesla v názvu souboru toto heslo někdy v oknech GUI informujících o průběhu zálohy nahrazeno hvězdičkami a někdy zůstává v čitelné podobě. V případě vytvoření zálohy je heslo nahrazeno hvězdičkami (počet hvězdiček udává délku hesla), v případě ověření zálohy je v poli název souboru heslo v čitelné podobě, v informační hlášce o začátku ověřování je však nahrazeno hvězdičkami. Při obnovení disku

[25] ftp://username:password@server:port/path/name.sna

je heslo k FTP účtu vždy zobrazeno čitelně, při pokusu o připojení souboru jako virtuálního disku, které vždy skončí s chybou, je zobrazeno také.

Pro opravení chyby navrhuji odstranit heslo z řetězce ukládaného do historie posledních souborů. Nemá-li program Drive Snapshot k danému FTP účtu uložené heslo, měl by otevřít dialogové okno pro zadání hesla (v současné verzi skončí s chybou). Toto dialogové okno by mohlo obsahovat i možnost uložení FTP účtu, avšak v případě uložení hesla k účtu by mělo obsahovat varování, že heslo lze z registrů získat (popsáno v následující podkapitole 4.2.12).

4.2.12 Nedostatečné varování uživatele při ukládání hesel

Drive Snapshot nabízí možnost uložit dešifrovací heslo a hesla k FTP účtům do registru operačního systému. Uživatel však není dostatečně informován o bezpečnosti tohoto řešení – hesla jsou sice v registru zašifrována, ale šifrovací klíče jsou součástí spustitelného souboru programu. Pro potencionálního útočníka není problém hesla dešifrovat.

V případě dešifrovacího hesla je jako šifrovací klíč použit textový řetězec, který je součástí programu. Dešifrovací heslo je zašifrováno vždy stejným šifrovacím klíčem. Pro šifrování hesel k FTP účtům je jako šifrovací klíč použita kombinace informací o serveru, přihlašovacího jména a šifrovacího klíče, který je také součástí programu. Hesla k různým FTP účtům jsou tedy sice zašifrovány jiným šifrovacím klíčem, ale opět není problém hesla dešifrovat – přihlašovací jméno a údaje o účtu jsou volně uloženy v registrech a klíč lze extrahovat ze spustitelného souboru programu.

V současné verzi programu Drive Snapshot se nachází pouze jediné varování ohledně bezpečnosti uložených hesel, a to pouze při uložení dešifrovacího hesla. Varování se zobrazí ve formě *ballon tooltip* tehdy, podrží-li uživatel kurzor nad tlačítkem pro uložení dešifrovacího hesla. Varování se mu tedy nemusí vůbec zobrazit. V případě uložení FTP hesla program žádné upozornění nezobrazuje. Vzhledem k tomu, že pro útočníka není problém hesla z registru dešifrovat, měl by Drive Snapshot o tomto riziku informovat důrazněji, a to i v případě hesel k FTP účtům. Řešením může být vyskakovací okno, ve kterém by uživatel musel potvrdit, že si je vědom rizika spojeného s uložením hesla.

Kromě lepší informovanosti uživatele může být dobré použít na zašifrování uložených hesel například systémovou funkci *CryptProtectData* z hlavičkového souboru *dpapi.h* (Data Protection API) [36]. Tato funkce zašifruje data takovým způsobem, že pouze stejný uživatel (a většinou na stejném počítači) může data dešifrovat. Na různých počítačích tedy bude pro šifrovaní použit jiný šifrovací klíč.

4.2.13 Žádné varování při použití slabého hesla

Současná bezpečnostní doporučení zmiňují, že pro bezpečnost hesla je zásadní zejména jeho délka [30]. Program Drive Snapshot nezobrazuje ani v případě použití GUI žádné informace o bezpečnosti zadaného hesla. Použití slabých hesel o nedostatečné délce oslabuje bezpečnost šifrování vytvořené zálohy.

Program by měl (minimálně při použití GUI) uživatele varovat v případě použití krátkého hesla (dle doporučení NIST [30] musí mít hesla alespoň 8 znaků). V případě použití méně znaků by měl uživatel potvrdit, že si je vědom použití slabého hesla a tedy možných rizik.

4.2.14 Odvození klíče z hesla je zranitelné útokem postranním kanálem

Při odvození klíče *passwdKey* sloužícího k zašifrování hlavního šifrovacího klíče *AESkey* se používá Diffie-Hellmanův algoritmus pro ustanovení společného klíče. Tato část je do algoritmu zakomponována z důvodu možnosti používání veřejného šifrovacího klíče. Během výpočtu je použita vlastní implementace modulárního umocňování, která je zranitelná vůči různým druhům útoků postranním kanálem.

Při odvození šifrovacího klíče se provádí následující postup. Z hesla se vypočítá jeho otisk pomocí opakovaného volání hashovací funkce SHA-256. Výsledkem této funkce je otisk o velikosti 32 bytů. Poté se vygeneruje *nonce* – náhodné číslo o velikosti 32 bytů. Tato dvě čísla se poté používají v algoritmu Diffie-Hellman jako tajné exponenty, číslo 0x1267 je použito jako základ pro umocnění a celý výpočet probíhá modulo prvočíslo $m = 2^{19937} - 1$. Výsledkem umocňování jsou poté čísla *sncData1*, *sncData2* a následně samotný šifrovací klíč *passwdKey*.

Pro modulární umocňování se používá známý algoritmus *Square&Multiply*. Tento algoritmus prochází exponent po jednotlivých bitech, pro každý bit provede operaci *square* (umocnění na druhou) a navíc, pokud je bit exponentu roven 1, provede operaci *multiply* (násobení). Jelikož postup výpočtu je závislý na vstupních datech, existuje potenciální možnost útoku například postranním časovým kanálem.

Přestože je zneužití této chyby málo pravděpodobné, navrhuji následující úpravy. Jedním z řešení může být taková úprava algoritmu *Square&Multiply*, že k provedení operace *multiply* dojde vždy nezávisle na hodnotě bitu exponentu – v případě, že je bit roven nule, se vypočítaná hodnota nepoužije. Při tomto řešení je potřeba si zkontrolovat, aby při kompilaci programu nedošlo k odstranění nepotřebného kódu při optimalizacích. Dalším řešením může být nahrazení algoritmu *Square&Multiply* za algoritmus *Montgomery ladder* (Montgomeryho žebřík).

4.2.15 Dokumentace obsahuje zavádějící informace o šifrování

Dokumentace, respektive web programu Drive Snapshot, obsahuje špatné informace o použité variantě šifry AES a použitém operačním režimu, navíc anglická a německá jazyková mutace dokumentace uvádí odlišné údaje. Anglická verze dokumentace uvádí použití šifry AES-128 v operačním režimu CBC [12], německá verze dokumentace zmiňuje použití šifry AES-256 v operačním režimu CBC [37]. Ani jedna z těchto informací není zcela správná, Drive Snapshot v současné verzi 1.48 používá pro šifrování zálohovaných dat šifru AES-256 v operačním režimu CTR.

Dále například německá verze stránky o šifrování v odstavci o slovníkových útocích zmiňuje použití 128 bitového klíče [37], avšak anglická verze zmiňuje použití klíče o velikosti 256 bitů [12]. Na stejné stránce anglická verze zmiňuje uložení 20 bitů z otisku hesla v souboru se zálohou kvůli potřebě ověření hesla, avšak v souboru se zálohou je pro tento účel uložených prvních 20 bitů šifrovacího klíče. Německá verze tuto informaci vůbec nezmiňuje.

Obě jazykové mutace dokumentace by měly obsahovat shodné údaje – v současné verzi jsou některé informace dostupné pouze v jednom jazyce. Detaily o šifrování by měly být aktualizovány tak, aby byly platné pro současnou verzi programu – současné informace pravděpodobně platily pro předchozí verze programu Drive Snapshot.

Dále dokumentace neobsahuje informaci o tom, jaké informace jsou v souboru se zálohou zašifrovány. V případě šifrované zálohy dojde pouze k zašifrování SND0 sekcí obsahující samotná uživatelská data. Sekce obsahující technické informace, jako je například velikost disku, čas vytvoření zálohy, použitý souborový systém, nejsou zašifrovány ani v případě šifrované zálohy, obdobné platí i pro uloženou MBR nebo GPT tabulku disku. Kromě těchto informací je například v šifrované záloze v nešifrované podobě uložena uživatelská poznámka k záloze či obsah příkazové řádky při spuštění programu.

4.2.16 Web programu není dostupný přes zabezpečený protokol HTTPS

Web[26] programu Drive Snapshot není dostupný přes zabezpečený protokol HTTPS. Kvůli použití klasického nezabezpečného protokolu HTTP může útočník provést útok *man-in-the-middle*, během kterého může dovést uživatele ke stažení upravené verze programu. Spuštění upravené aplikace s administrátorskými právy může vést k plnému ovládnutí uživatelova počítače útočníkem.

Spustitelný soubor programu Drive Snapshot je sice digitálně podepsán autorem programu, avšak výstraha operačního systému při spuštění souboru s neplatným podpisem není nijak výrazná (odlišná grafika okna UAC[27] při žádosti o eskalaci práv). Běžný uživatel nemusí postřehnout, že se děje něco

[26] `http://www.drivesnapshot.de/`
[27] User Account Control

nepatřičného. Pokud by byl web dostupný přes zabezpečený protokol HTTPS s certifikátem vydaným uznávanou certifikační autoritou, tak v případě podvržení certifikátu nezobrazí internetový prohlížeč uživateli danou stránku bez komplikovaného přidání výjimky a realizace potencionálního útoku by byla mnohem náročnější.

Řešením tohoto problému je změna konfigurace webového serveru, která povolí používání protokolu HTTPS. Pro zabezpečení je nezbytné získat certifikát od uznávané certifikační autority. Ten lze získat i automatizovaně zadarmo například od autority Let's Encrypt[28].

4.3 Ostatní nalezené chyby

V této kapitole popisuji další nalezené chyby, které jsem nevyhodnotil jako nutně bezpečnostní hrozbu – prvotně způsobují pouze nefunkčnost programu Drive Snapshot. To však nemusí znamenat, že tyto chyby neskrývají bezpečnostní zranitelnost nebo nemůžou uživatele uvést v omyl a způsobit například ztrátu dat. Zejména pád programu může naznačovat zranitelnost typu *buffer overflow*, která může mít pro bezpečnost fatální důsledky.

4.3.1 Nevalidní chování při zavření okna pro zadání hesla

V případě, že chce uživatel připojit přes GUI šifrovanou zálohu jako virtuální disk, zobrazí se dialogové okno (obrázek 3.3) pro zadání hesla. Pokud toto okno uživatel zavře kliknutím na tlačítko *cancel* nebo kliknutím na křížek pro zavření okna, dialog se správně ukončí a k pokusu o připojení nedojde. Pokud se však uživatel pokusí stejné dialogové okno zavřít při ověření nebo obnovení šifrované zálohy, dochází k různým chybám.

Po zavření okna s výzvou pro zadání hesla totiž dojde většinou k pokusu o zpracování souboru s nerozšifrovanou zálohou. Zobrazí se okno, které informuje o průběhu ověření nebo obnovení zálohy a pak dojde k zobrazení chybové hlášky, jako je například *„defekt header data, to wr 197 != expand 10000“*, za kterou následuje dialog pro otevření souboru s další částí zálohy. V některých případech se zobrazí jiná chybová hláška, v jiných dokonce dojde k pádu celého programu.

Při připojení šifrované zálohy jako virtuálního disku pomocí CLI bez parametru `-pw=` je zobrazeno dialogové okno na zadání hesla 2x po sobě. Pokud uživatel obě okna zavře, končí program s chybovou hláškou *„No valid volume file“*. Při spuštění obnovení disku z šifrované zálohy přes CLI se vypíše na konzoli, že je nutné zadat správné heslo přes parametr `-pw=`.

Program by měl správně pracovat v případech, kdy se uživatel rozhodne nezadat heslo k šifrované záloze. Správné chování vykazuje Drive Snapshot u připojení zálohy jako virtuálního disku přes GUI a u obnovení zálohy přes

[28] `https://letsencrypt.org/`

CLI. V jiných případech dochází k zobrazení pro uživatele nepochopitelných chybových hlášek a zvláštnímu chování, v určitých případech dokonce k pádu programu.

4.3.2 Vytvoření neplatné zálohy

Tato chyba se vyskytuje v případě, kdy uživatel vytváří pomocí GUI programu více záloh disku. Pokud vytvoří nejprve šifrovanou zálohu a poté chce vytvořit nešifrovanou zálohu, je nešifrovaná záloha vytvořena chybně a je nepoužitelná, přestože projde úspěšně ověřením. Přesný postup pro reprodukci chyby je následovný:

1. uživatel spustí GUI programu Drive Snapshot;
2. uživatel zvolí možnost vytvoření zálohy a vytvoří šifrovanou zálohu;
3. po dokončení zálohy se uživatel vrátí do hlavního menu a zvolí opět vytvoření zálohy;
4. v dialogovém okně je předvyplněné heslo použité pro šifrování předchozí zálohy, uživatel toto heslo smaže, protože nechce vytvořit šifrovanou zálohu, a spustí zálohování;
5. dojde k vytvoření nešifrované zálohy, která sice úspěšně projde ověřením, avšak je poškozená a tedy nepoužitelná.

Chyba, která poškodila nešifrovanou zálohu vytvořenou v bodě 4 a 5, vznikla pravděpodobně tím, že Drive Snapshot uložil data v sekcích SND0 zašifrovaně kvůli vygenerovanému klíči *AESkey* uloženém v paměti z předchozího šifrování, avšak nevytvořil v souboru se zálohou SNC2 sekci (údaje o kryptografii), protože v GUI neměl zadané heslo. Teorii o zašifrování dat v SND0 sekcích potvrzuje nastavení bitu označujícího šifrování dat v proměnné obsahující informace o typu uložených dat, která je umístěna na začátku SND0 sekce.

Při pokusu o připojení takto poškozené zálohy jako virtuálního disku sice dojde k připojení disku, avšak při pokusu o jeho čtení se zobrazí chybová hláška operačního systému ohledně poškozené struktury disku. Při pokusu o obnovu dat z této zálohy dojde k pádu programu Drive Snapshot.

4.4 Shrnutí nalezených zranitelností

Kromě výše uvedených zranitelností jsem nenašel v programu Drive Snapshot žádné další bezpečnostní chyby. Během mojí analýzy jsem například nenalezl žádné případy přetečení bufferu na haldě či zásobníku, které bývají častou bezpečnostní chybou.

Nalezené bezpečnostní chyby detailně popsané v předchozích podkapitolách jsem ohodnotil pomocí metodiky CVSS (Common Vulnerability Scoring System) v aktuální verzi 3.1. CVSS poskytuje metodu pro vyhodnocení nalezených zranitelností a umožňuje jejich ohodnocení číselnou hodnotou vyjadřující závažnost dané bezpečnostní zranitelnosti [38]. Tuto číselnou hodnotu, nabývající hodnot od 0 (nejnižší závažnost) do 10 (nejvyšší závažnost), lze následně převést pomocí převodní tabulky (tabulka 4.2) na slovní hodnocení (none, low, medium, high, critical).

Tabulka 4.2: Převedení CVSS skóre na slovní hodnocení [39]

CVSS skóre	závažnost
0.0	none
0.1–3.9	low
4.0–6.9	medium
7.0–8.9	high
9.0–10.0	critical

Kromě číselného skóre (respektive jednoduchého slovního ohodnocení) lze zranitelnost popsat pomocí takzvaného *vector string*, který obsahuje přesnou klasifikaci hodnocené bezpečnostní zranitelnosti v krátkém textovém řetězci. Klasifikace zranitelností je založena na několika parametrech, povinně se hodnotí [39]:

- Attack vector (vektor útoku);
- Attack Complexity (složitost útoku);
- Privileges Required (potřeba oprávnění);
- User Interaction (nutnost interakce uživatele);
- Scope (rozsah útoku);
- Confidentiality, Integrity, Availability (dopad na důvěryhodnost, integritu a dostupnost dat).

Kromě výše zmíněných základních parametrů je ještě možné zpřesnit výpočet pomocí dalších volitelných parametrů. Vyhodnocení bezpečnostní zranitelnosti, včetně výpočtu skóre a vygenerování *vector string*, lze provést v interaktivní kalkulačce umístěné na webu[29] Forum of Incident Response and Security Teams (organizace spravující CVSS).

V následující tabulce 4.3 uvádím vypočítané hodnocení pro nalezené zranitelnosti dle metodiky CVSS ve verzi 3.1. Pro výpočet jsem použil výše zmíněnou kalkulačku. Přesné zdůvodnění všech zvolených parametrů dle použité

[29] https://www.first.org/cvss/calculator/3.1

metodiky uvádím v dodatku A. Bezpečnostní chyby popsané v kapitole 4.2.13 (žádné varování při použití slabého hesla) a 4.2.15 (dokumentace obsahuje zavádějící informace o šifrování) nebyly dle metodiky hodnoceny, protože se nejedná přímo o zranitelnosti, které by mohl potenciální útočník zneužít.

Tabulka 4.3: Vyhodnocení nalezených bezpečnostních zranitelností pomocí metodiky CVSS ve verzi 1.3

kapitola	název bezpečnostní zranitelnosti	CVSS skóre
4.2.1	Velká část šifrovacího klíče je zveřejněna v souboru se zálohou	7.0 (high)
4.2.2	Chybné použití šifrovacího režimu CTR	9.1 (critical)
4.2.3	Síla šifrování je závislá na výkonu počítače	7.4 (high)
4.2.4	Chybná práce s citlivými údaji v paměti programu	6.0 (medium)
4.2.5	Délka hesla zadaného přes CLI je zveřejněna v souboru se zálohou	6.8 (medium)
4.2.6	Část hesla zadaného přes CLI může být zveřejněna v souboru se zálohou	8.1 (high)
4.2.7	Konverze kódování hesla může snížit bezpečnost hesla	8.1 (high)
4.2.8	Nekonzistentní načítání hesla	6.5 (medium)
4.2.9	Špatné zpracování argumentů na příkazovém řádku	8.8 (high)
4.2.10	Heslo k FTP účtu může být zveřejněno v souboru se zálohou	9.6 (critical)
4.2.11	Heslo k FTP účtu může být v čitelném formátu uloženo v registrech	7.7 (high)
4.2.12	Nedostatečné varování uživatele při ukládání hesel	7.7 (high)
4.2.13	Žádné varování při použití slabého hesla	nehodnoceno
4.2.14	Odvození klíče z hesla je zranitelné útokem postranním kanálem	4.9 (medium)
4.2.15	Dokumentace obsahuje zavádějící informace o šifrování	nehodnoceno
4.2.16	Web programu není dostupný přes zabezpečený protokol HTTP	8.3 (high)

Závěr

Cílem této diplomové práce bylo provedení bezpečnostní analýzy programu Drive Snapshot – nástroje pro zálohování disků. Hlavní částí práce je reverzní analýza klíčových částí zkoumaného programu. V třetí kapitole jsem mimo jiné podrobně popsal vlastní formát souboru s vytvořenou zálohou, způsob generování náhodných čísel, které jsou použity v kryptografických algoritmech, postup odvození šifrovacího klíče z hesla zadaného uživatelem a celý postup šifrování a dešifrování vytvářených záloh včetně identifikace použitých kryptografických algoritmů. Dále jsem popsal způsoby práce programu s hesly včetně způsobů načítání hesel a jejich případného ukládání. Kromě metod nakládání programu s hesly jsem se zaměřil i na práci programu s dalšími citlivými údaji.

Během reverzní analýzy jsem nalezl celou řadu bezpečnostních zranitelností a chyb. Objevil jsem například chyby v implementaci a použití kryptografických algoritmů, chybnou práci s citlivými údaji nebo problémy spojené s kódováním znaků hesla. Drive Snapshot obsahuje chyby, které způsobují například uložení více než poloviny šifrovacího klíče použitého pro zašifrování zálohy do souboru se zálohou nebo chyby, které při specifických situacích mohou způsobit prozrazení části informací o hesle, jako je jeho délka nebo část obsahu. Kromě chyb v samotném programu jsem nalezl neaktuální informace ohledně použitých kryptografických algoritmech v dokumentaci programu a použití nezabezpečeného protokolu HTTP na webu programu.

Všechny nalezené bezpečnostní zranitelnosti a chyby popsané ve čtvrté kapitole byly nahlášeny autorům programu Drive Snapshot. Autoři ocenili nahlášení nalezených zranitelností a slíbili jejich opravení. V době odevzdání textu této diplomové práce byly dokončovány opravy nahlášených chyb a mělo by brzy dojít k vydaní opravené verze programu Drive Snapshot.

Bibliografie

1. BEACH, Brian. *How Long Do Disk Drives Last?* [Online]. Backblaze, 2013-11 [cit. 2021-05-01]. Dostupné z: https://www.backblaze.com/blog/how-long-do-disk-drives-last/.
2. MICROSOFT. *Windows app developer documentation: Basic and Dynamic Disks* [online]. 2018-05 [cit. 2021-05-02]. Dostupné z: https://docs.microsoft.com/en-us/windows/win32/fileio/basic-and-dynamic-disks.
3. MICROSOFT. *Windows app developer documentation: Disk Devices and Partitions* [online]. 2018-05 [cit. 2021-05-02]. Dostupné z: https://docs.microsoft.com/en-us/windows/win32/fileio/disk-devices-and-partitions.
4. MICROSOFT. *Windows app developer documentation: Volume Management* [online]. 2018-05 [cit. 2021-05-02]. Dostupné z: https://docs.microsoft.com/en-us/windows/win32/fileio/volume-management.
5. MICROSOFT. *Windows app developer documentation: Files and Clusters* [online]. 2018-05 [cit. 2021-05-02]. Dostupné z: https://docs.microsoft.com/en-us/windows/win32/fileio/files-and-clusters.
6. NELSON, Steven. Introduction to Backup and Recovery. In: *Pro Data Backup and Recovery*. Berkeley, CA: Apress, 2011, s. 1–16. ISBN 978-1--4302-2663-5. Dostupné z DOI: 10.1007/978-1-4302-2663-5_1.
7. TOM EHLERT SOFTWARE. *Drive Snapshot – What's new* [online] [cit. 2021-03-01]. Dostupné z: http://www.drivesnapshot.de/en/news.htm.
8. TOM EHLERT SOFTWARE. *Snapshot – Trialware Download* [online] [cit. 2021-03-01]. Dostupné z: http://www.drivesnapshot.de/en/down.htm.

9. TOM EHLERT SOFTWARE. *Snapshot – DiskImage Backup for Windows NT* [online] [cit. 2021-03-02]. Dostupné z: http://www.drivesnapshot.de/en/backup.htm.

10. TOM EHLERT SOFTWARE. *Drive Snapshot – Order* [online] [cit. 2021-03-01]. Dostupné z: http://www.drivesnapshot.de/en/order.htm.

11. TOM EHLERT SOFTWARE. *Snapshot – command line options* [online] [cit. 2021-03-03]. Dostupné z: http://www.drivesnapshot.de/en/commandline.htm.

12. TOM EHLERT SOFTWARE. *Snapshot – Encryption Details* [online] [cit. 2021-03-03]. Dostupné z: http://www.drivesnapshot.de/en/icrypt.htm.

13. MICROSOFT. *Windows Server Storage documentation: Volume Shadow Copy Service* [online]. 2019-01 [cit. 2021-03-04]. Dostupné z: https://docs.microsoft.com/en-us/windows-server/storage/file-server/volume-shadow-copy-service.

14. TOM EHLERT SOFTWARE. *Drive Snapshot – VSS* [online] [cit. 2021-03-05]. Dostupné z: http://www.drivesnapshot.de/en/snapshot_vss.htm.

15. TOM EHLERT SOFTWARE. *Restoring the system drive during restart* [online] [cit. 2021-03-04]. Dostupné z: http://www.drivesnapshot.de/en/restboot.htm.

16. MICROSOFT. *Windows hardware developer documentation: Windows PE (WinPE)* [online]. 2018-10 [cit. 2021-03-04]. Dostupné z: https://docs.microsoft.com/en-us/windows-hardware/manufacture/desktop/winpe-intro.

17. MICROSOFT. *Windows hardware developer documentation: Windows Recovery Environment (Windows RE)* [online]. 2017-05 [cit. 2021-03-04]. Dostupné z: https://docs.microsoft.com/en-us/windows-hardware/manufacture/desktop/windows-recovery-environment--windows-re--technical-reference.

18. TOM EHLERT SOFTWARE. *Snapshot – Restoring a volume from Dos* [online] [cit. 2021-03-03]. Dostupné z: http://www.drivesnapshot.de/en/restdos.htm.

19. CHIKOFSKY, E.J.; CROSS, J.H. Reverse engineering and design recovery: a taxonomy. *IEEE Software*. 1990, roč. 7, č. 1, s. 13–17. Dostupné z DOI: 10.1109/52.43044.

20. MICROSOFT. *Programming reference for the Win32 API: RtlGenRandom function (ntsecapi.h)* [online]. 2018-12 [cit. 2021-03-14]. Dostupné z: https://docs.microsoft.com/en-us/windows/win32/api/ntsecapi/nf-ntsecapi-rtlgenrandom.

21. NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY. *Advanced Encryption Standard (AES)*. Washington, D.C., 2001. Tech. zpr., Federal Information Processing Standards Publications (FIPS PUBS) 197. Dostupné z DOI: 10.6028/NIST.FIPS.197.

22. DWORKIN, Morris. Recommendation for Block Cipher Modes of Operation: Methods and Techniques. *NIST Special Publication*. 2001, roč. 800, 38A. Dostupné z DOI: 10.6028/NIST.SP.800-38A.

23. WHITETIMBERWOLF. *CTR encryption 2* [online]. Wikimedia Commons, 2013 [cit. 2021-03-21]. Dostupné z: https://commons.wikimedia.org/wiki/File:CTR_encryption_2.svg.

24. MICROSOFT. *Shell Experience troubleshooting documentation for Windows clients: Command prompt (Cmd.exe) command-line string limitation* [online]. 2021-02 [cit. 2021-03-19]. Dostupné z: https://docs.microsoft.com/en-US/troubleshoot/windows-client/shell-experience/command-line-string-limitation.

25. RUSSINOVICH, Mark. *Inside Native Applications* [online]. 2006-11 [cit. 2021-04-07]. Dostupné z: https://docs.microsoft.com/en-us/sysinternals/resources/inside-native-applications.

26. NATIONAL SECURITY AGENCY. *Commercial National Security Algorithm Suite* [online]. 2015-08 [cit. 2021-03-25]. Dostupné z: https://apps.nsa.gov/iad/programs/iad-initiatives/cnsa-suite.cfm.

27. MICROSOFT. *Programming reference for the Win32 API: CryptGenRandom function (wincrypt.h)* [online]. 2018-12 [cit. 2021-03-25]. Dostupné z: https://docs.microsoft.com/en-us/windows/win32/api/wincrypt/nf-wincrypt-cryptgenrandom.

28. MICROSOFT. *Microsoft security engineering documentation: Microsoft SDL Cryptographic Recommendations* [online]. 2018-12 [cit. 2021-03-25]. Dostupné z: https://docs.microsoft.com/en-us/security/sdl/cryptographic-recommendations.

29. MICROSOFT. *Microsoft C runtime library (CRT) reference: rand* [online]. 2020-04 [cit. 2021-03-25]. Dostupné z: https://docs.microsoft.com/en-us/cpp/c-runtime-library/reference/rand.

30. GRASSI, P.; FENTON, J.; NEWTON, E.; PERLNER, R.; REGENSCHEID, A.; BURR, W.; RICHER, J.; LEFKOVITZ, N.; DANKER, J.; CHOONG, Y.; GREENE, K.; THEOFANOS, M. Digital Identity Guidelines: Authentication and Lifecycle Management. *NIST Special Publication*. 2017, roč. 800, 63B. Dostupné z DOI: 10.6028/NIST.SP.800-63b.

31. KIVINEN, T.; KOJO, T. *More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE)* [Internet Requests for Comments]. RFC Editor, 2003-05. RFC, 3526. Dostupné z DOI: 10.17487/RFC3526.

32. LEPINSKI, M.; KENT, S. *Additional Diffie-Hellman Groups for Use with IETF Standards* [Internet Requests for Comments]. RFC Editor, 2008-01. RFC, 5114. Dostupné z DOI: 10.17487/RFC5114.

33. BARKER, E; ROGINSKY, A. Transitioning the Use of Cryptographic Algorithms and Key Lengths. *NIST Special Publication*. 2019, roč. 800, 131A Rev.2. Dostupné z DOI: 10.6028/NIST.SP.800-131Ar2.

34. BIRYUKOV, Alex. Known Plaintext Attack. In: *Encyclopedia of Cryptography and Security*. Ed. TILBORG, Henk C. A. van; JAJODIA, Sushil. Boston, MA: Springer US, 2011, s. 704–705. ISBN 978-1-4419-5906-5. Dostupné z DOI: 10.1007/978-1-4419-5906-5_588.

35. TOM EHLERT SOFTWARE. *Using an FTP Server* [online] [cit. 2021-04-08]. Dostupné z: http://www.drivesnapshot.de/en/ftp.htm.

36. MICROSOFT. *Programming reference for the Win32 API: CryptProtectData function (dpapi.h)* [online]. 2018-12 [cit. 2021-03-31]. Dostupné z: https://docs.microsoft.com/en-us/windows/win32/api/dpapi/nf-dpapi-cryptprotectdata.

37. TOM EHLERT SOFTWARE. *Snapshot – Verschlüsselung Details* [online] [cit. 2021-04-06]. Dostupné z: http://www.drivesnapshot.de/de/icrypt.htm.

38. FORUM OF INCIDENT RESPONSE AND SECURITY TEAMS. *Common Vulnerability Scoring System SIG* [online] [cit. 2021-04-23]. Dostupné z: https://www.first.org/cvss/.

39. FORUM OF INCIDENT RESPONSE AND SECURITY TEAMS. *Common Vulnerability Scoring System version 3.1: Specification Document (Revision 1)* [online] [cit. 2021-04-23]. Dostupné z: https://www.first.org/cvss/v3-1/cvss-v31-specification_r1.pdf.

Dodatek A

Vyhodnocení zranitelností dle metodiky CVSS

Tento dodatek obsahuje detaily k ohodnocení parametrů nalezených bezpečnostních zranitelností dle metodiky CVSS. Komentáře k ohodnocení jsou zveřejněny v tabulkách na následujících stranách.

Tabulka A.1: Velká část šifrovacího klíče je zveřejněna v souboru se zálohou (4.2.1) – ohodnocení parametrů dle metodiky CVSS. Vector string: CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:N/E:P

parametr	hodnota	komentář
Attack Vector	Network	Pro provedení útoku musí útočník získat soubor se zálohou. Ten může získat různými způsoby – z lokálního média nebo i přes síť.
Attack Complexity	High	Více než polovinu šifrovacího klíče lze sice jednoduše vyčíst ze souboru se zálohou, prolomení šifrování hrubou silou je však zatím nemožné.
Privileges Required	None	Pro otevření souboru nejsou nutná žádná speciální oprávnění, jedinou podmínkou je získání souboru se zálohou.
User Interaction	None	K útoku není nutná spolupráce uživatele.
Scope	Unchanged	Zranitelnou i ovlivněnou komponentou je soubor se zálohou.
Confidentality	High	V případě prolomení šifrování souboru se zálohou získá útočník plný přístup k zálohovaným datům.
Integrity	Hight	V případě prolomení šifrování souboru se zálohou získá útočník plný přístup k zálohovaným datům, útočník tedy může zálohu i upravit.
Availability	None	Prolomení šifrování neohrozí dostupnost dat.
Exploit Code Maturity	Proof-of-Concept	K úspěšnému prolomení šifrování je nutné provést útok na šifru AES. Kvůli zranitelnosti je délka šifrovacího klíče oslabena z 256 bitů na 108 bitů. Takto krátký klíč již sice nesplňuje současná doporučení na délku šifrovacího klíče, přesto však bude pravděpodobně neprolomitelný.

Tabulka A.2: Chybné použití šifrovacího režimu CTR (4.2.2) – ohodnocení parametrů dle metodiky CVSS. Vector string: CVSS:3.1/AV:N/-AC:L/PR:N/UI:N/S:U/C:H/I:H/A:N

parametr	hodnota	komentář
Attack Vector	Network	Pro provedení útoku musí útočník získat soubor se zálohou. Ten může získat různými způsoby – z lokálního média nebo i přes síť.
Attack Complexity	Low	Pro provedení útoku je nezbytné znát (anebo odhadovat), jaká data byla zašifrována. Kvůli uložení MBR nebo GPT tabulky v SDRI sekci však bude dešifrování části zálohy jednoduché.
Privileges Required	None	Pro otevření souboru se zálohou nejsou nutná žádná speciální oprávnění, jedinou podmínkou je získání souboru se zálohou.
User Interaction	None	K útoku není nutná spolupráce uživatele.
Scope	Unchanged	Zranitelnou i ovlivněnou komponentou je soubor se zálohou.
Confidentality	High	Prolomením šifrování souboru se zálohou získá útočník plný přístup k zálohovaným datům.
Integrity	High	Prolomením šifrování souboru se zálohou získá útočník plný přístup k zálohovaným datům.
Availability	None	Prolomení šifrování neohrozí dostupnost dat.

Tabulka A.3: Síla šifrování je závislá na výkonu počítače (4.2.3) – ohodnocení parametrů dle metodiky CVSS. Vector string: CVSS:3.1/AV:N/-AC:H/PR:N/UI:N/S:U/C:H/I:H/A:N

parametr	hodnota	komentář
Attack Vector	Network	Pro provedení útoku musí útočník získat soubor se zálohou. Ten může získat různými způsoby – z lokálního média nebo i přes síť.
Attack Complexity	High	Oslabení hashovací funkce sice zjednodušuje útok hrubou silou na slabá hesla, avšak výpočetní náročnost je stále značná. Z tohoto důvodu hodnotím složitost útoku jako vysokou.
Privileges Required	None	Pro otevření souboru nejsou nutná žádná speciální oprávnění, jedinou podmínkou je získání souboru se zálohou.
User Interaction	None	K útoku není nutná spolupráce uživatele.
Scope	Unchanged	Zranitelnou i ovlivněnou komponentou je soubor se zálohou
Confidentality	High	Prolomením šifrování souboru se zálohou by došlo k plnému přístupu k zálohovaným datům.
Integrity	High	Prolomením šifrování souboru se zálohou získá útočník plný přístup k zálohovaným datům.
Availability	None	Prolomení šifrování neohrozí dostupnost dat.

Tabulka A.4: Chybná práce s citlivými údaji v paměti programu (4.2.4) – ohodnocení parametrů dle metodiky CVSS. Vector string: CVSS:3.1/AV:L/AC:L/PR:H/UI:N/S:U/C:H/I:H/A:N

parametr	hodnota	komentář
Attack Vector	Local	Pro provedení útoku musí mít útočník lokální přístup k počítači.
Attack Complexity	Low	Získání obsahu operační paměti procesu není náročné, pouze jsou pro to nutná vysoká oprávnění.
Privileges Required	High	Z důvodu běhu Drive Snapshot s administrátorskými právy je nutné mít také administrátorská práva pro čtení paměti tohoto procesu.
User Interaction	None	Pro provedení útoku není nutná spolupráce uživatele.
Scope	Unchanged	Zranitelnou i ovlivněnou komponentou je program Drive Snapshot a vytvořené zálohy.
Confidentality	High	Získáním šifrovacích klíčů a hesel získá útočník plný přístup k vytvořeným zálohám.
Integrity	High	Získáním šifrovacích klíčů a hesel získá útočník plný přístup k vytvořeným zálohám.
Availability	None	Dostupnost není ovlivněna.

Tabulka A.5: Délka hesla zadaného přes CLI je zveřejněna v souboru se zálohou (4.2.5) – ohodnocení parametrů dle metodiky CVSS. Vector string: CVSS:3.1/AV:N/AC:H/PR:N/UI:R/S:U/C:H/I:H/A:N

parametr	hodnota	komentář
Attack Vector	Network	Pro provedení útoku musí útočník získat soubor se zálohou. Ten může získat různými způsoby – z lokálního média nebo i přes síť.
Attack Complexity	High	I přes znalost délky hesla je složitost provedení útoku hrubou silou na heslo uživatele stále velmi vysoká.
Privileges Required	None	Pro otevření souboru nejsou nutná žádná speciální oprávnění, jedinou podmínkou je získání souboru se zálohou.
User Interaction	Required	Při vytváření zálohy musí uživatel zadat heslo pomocí parametru `-pw=`. Toto chování je však očekávatelné.
Scope	Unchanged	Zranitelnou i ovlivněnou komponentou je program Drive Snapshot a vytvořené zálohy.
Confidentality	High	Prolomením šifrování souboru se zálohou dojde k plnému přístupu k zálohovaným datům.
Integrity	High	Prolomením šifrování souboru se zálohou dojde k plnému přístupu k zálohovaným datům.
Availability	None	Prolomení šifrování neohrozí dostupnost dat.

Tabulka A.6: Část hesla zadaného přes CLI může být zveřejněna v souboru se zálohou (4.2.6) – ohodnocení parametrů dle metodiky CVSS. Vector string: CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:N

parametr	hodnota	komentář
Attack Vector	Network	Pro provedení útoku musí útočník získat soubor se zálohou. Ten může získat různými způsoby – z lokálního média nebo i přes síť.
Attack Complexity	Low	Znalost části použitého hesla značně usnadní útok hrubou silou nebo slovníkový útok na heslo.
Privileges Required	None	Pro otevření souboru nejsou nutná žádná speciální oprávnění, jedinou podmínkou je získání souboru se zálohou.
User Interaction	Required	Při vytváření zálohy musí uživatel zadat heslo pomocí parametru `-pw=` a použít v hesle mezeru.
Scope	Unchanged	Zranitelnou i ovlivněnou komponentou je program Drive Snapshot a vytvořené zálohy.
Confidentality	High	Prolomením šifrování souboru se zálohou dojde k plnému přístupu k zálohovaným datům.
Integrity	High	Prolomením šifrování souboru se zálohou dojde k plnému přístupu k zálohovaným datům.
Availability	None	Prolomení šifrování neohrozí dostupnost dat.

Tabulka A.7: Konverze kódování hesla může snížit bezpečnost hesla (4.2.7) – ohodnocení parametrů dle metodiky CVSS. Vector string: CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:N

parametr	hodnota	komentář
Attack Vector	Network	Pro provedení útoku musí útočník získat soubor se zálohou. Ten může získat různými způsoby – z lokálního média nebo i přes síť.
Attack Complexity	Low	Při splnění předpokladu na použití znaků v hesle nenacházejících se v kódové stránce je značně usnadněn útok hrubou silou.
Privileges Required	None	Pro otevření souboru nejsou nutná žádná speciální oprávnění, jedinou podmínkou je získání souboru se zálohou.
User Interaction	Required	Při vytváření zálohy musí uživatel použít v hesle znaky, které se nenachází v použité kódové stránce.
Scope	Unchanged	Zranitelnou i ovlivněnou komponentou je program Drive Snapshot a vytvořené zálohy.
Confidentality	High	Prolomením šifrování souboru se zálohou dojde k plnému přístupu k zálohovaným datům.
Integrity	High	Prolomením šifrování souboru se zálohou dojde k plnému přístupu k zálohovaným datům.
Availability	None	Prolomení šifrování neohrozí dostupnost dat.

Tabulka A.8: Nekonzistentní načítání hesla (4.2.8) – ohodnocení parametrů dle metodiky CVSS. Vector string: CVSS:3.1/AV:N/AC:L/PR:N/-UI:R/S:U/C:N/I:N/A:H

parametr	hodnota	komentář
Attack Vector	Network	Pokud by chtěl útočník zneužít zkrácení hesla na 250 znaků, musel by mít k dispozici soubor se zálohou. Ten může získat různými způsoby – z lokálního média nebo i přes síť. Dopad na dostupnost však tato zranitelnost bude mít vždy i bez zásahu útočníka.
Attack Complexity	Low	Při splnění předpokladu o způsobu vložení hesla dojde ke zkrácení délky hesla na maximálně 250 znaků. Prolomení takto dlouhého hesla je extrémně náročné a v dopadech na důvěrnost a integritu hodnotím tento útok jako neproveditelný. Tato zranitelnost však bude mít vliv i na dostupnost zálohy, a to bez jakéhokoliv zásahu útočníka. Proto hodnotím složitost jako nízkou.
Privileges Required	None	Pro provedení útoku nejsou nutná žádná speciální oprávnění.
User Interaction	Required	Při vytváření zálohy musí uživatel použít v hesle zadaném v GUI programu více než 250 znaků.
Scope	Unchanged	Zranitelnou i ovlivněnou komponentou je program Drive Snapshot a vytvořené zálohy.
Confidentality	None	Přestože je heslo zkráceno, prolomení hesla o délce 250 znaků je nejspíše neproveditelné.
Integrity	None	Přestože je heslo zkráceno, prolomení hesla o délce 250 znaků je nejspíše neproveditelné.
Availability	High	Pokud uživatel použije heslo delší než 250 znaků, bude mít tato zranitelnost vliv na dostupnost vytvořené zálohy. Kvůli zkrácení hesla nebude uživatel moci svoji zálohu rozšifrovat. Z tohoto důvodu hodnotím dopad na dostupnost jako vysoký.

Tabulka A.9: Špatné zpracování argumentů na příkazovém řádku (4.2.12) – ohodnocení parametrů dle metodiky CVSS. Vector string: CVSS:3.1/AV:N/-AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H

parametr	hodnota	komentář
Attack Vector	Network	Pro provedení útoku musí útočník získat soubor se zálohou. Ten může získat různými způsoby – z lokálního média nebo i přes síť.
Attack Complexity	Low	Vlivem špatného zpracování argumentů může být pro šifrování použito mnohem kratší heslo, než uživatel předpokládal. Kvůli tomu je útok hrubou silou či slovníkový útok na heslo zjednodušen. Z tohoto důvodu hodnotím složitost útoku jako nízkou.
Privileges Required	None	Pro otevření souboru nejsou nutná žádná speciální oprávnění, jedinou podmínkou je získání souboru se zálohou.
User Interaction	Required	Uživatel si musí například uložit heslo do registrů, a to takovým způsobem, kdy kvůli zranitelnosti dojde k jeho zkrácení (v místě první mezery).
Scope	Unchanged	Zranitelnou i ovlivněnou komponentou je program Drive Snapshot a vytvořené zálohy.
Confidentality	High	Prolomením šifrování souboru se zálohou dojde k plnému přístupu k zálohovaným datům.
Integrity	High	Prolomením šifrování souboru se zálohou dojde k plnému přístupu k zálohovaným datům.
Availability	High	Vedlejším efektem této zranitelnosti je, že pokud jsou naplněny předpoklady, může dojít i bez zásahu útočníka k omezení dostupnosti. Kvůli zkrácení hesla totiž uživatel nebude moci rozšifrovat svoji zálohu.

Tabulka A.10: Heslo k FTP účtu může být zveřejněno v souboru se zálohou (4.2.10) – ohodnocení parametrů dle metodiky CVSS. Vector string: CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:H/I:H/A:H

parametr	hodnota	komentář
Attack Vector	Network	Pro provedení útoku musí útočník získat soubor se zálohou. Ten může získat různými způsoby – z lokálního média nebo i přes síť.
Attack Complexity	Low	Heslo k FTP serveru je uloženo v čitelné podobě v souboru se zálohou.
Privileges Required	None	Pro otevření souboru nejsou nutná žádná speciální oprávnění, jedinou podmínkou je získání souboru se zálohou.
User Interaction	Required	Uživatel musí vytvořit pomocí CLI zálohu nahranou na FTP server, přihlašovací údaje musí být zadány pomocí speciálního formátu názvu souboru.
Scope	Changed	Zranitelnou komponentou je soubor se zálohou, ovlivněnou komponentou je FTP server.
Confidentality	High	Získáním uživatelského jména a hesla k FTP serveru získá útočník plný přístup.
Integrity	High	Získáním uživatelského jména a hesla k FTP serveru získá útočník plný přístup.
Availability	High	Získáním uživatelského jména a hesla k FTP serveru získá útočník plný přístup.

Tabulka A.11: Heslo k FTP účtu může být v čitelném formátu uloženo v registrech (4.2.11) – ohodnocení parametrů dle metodiky CVSS. Vector string: CVSS:3.1/AV:L/AC:L/PR:H/UI:R/S:C/C:H/I:H/A:H

parametr	hodnota	komentář
Attack Vector	Local	Útočník musí mít přístup k registrům napadeného uživatele.
Attack Complexity	Low	Heslo k FTP serveru je uloženo v čitelné podobě v registrech.
Privileges Required	High	Útočník musí mít přístup k účtu uživatele nebo účet s právy administrátora.
User Interaction	Required	Uživatel musí vytvořit pomocí GUI zálohu nahranou na FTP server, přihlašovací údaje musí být zadány pomocí speciálního formátu názvu souboru.
Scope	Changed	Zranitelnou komponentou je program Drive Snapshot, ovlivněnou komponentou je FTP server.
Confidentality	High	Získáním uživatelského jména a hesla k FTP serveru získá útočník plný přístup.
Integrity	High	Získáním uživatelského jména a hesla k FTP serveru získá útočník plný přístup.
Availability	High	Získáním uživatelského jména a hesla k FTP serveru získá útočník plný přístup.

Tabulka A.12: Nedostatečné varování uživatele při ukládání hesel (4.2.12) – ohodnocení parametrů dle metodiky CVSS. Vector string: CVSS:3.1/AV:L/AC:L/PR:H/UI:R/S:C/C:H/I:H/A:H

parametr	hodnota	komentář
Attack Vector	Local	Útočník musí mít přístup k počítači uživatele.
Attack Complexity	Low	Pro dešifrování hesel stačí extrahovat klíč ze spustitelného souboru programu.
Privileges Required	High	Útočník musí mít přístup do registrů daného uživatele.
User Interaction	Required	Uživatel si musí uložit hesla do registrů.
Scope	Changed	Zranitelnou komponentou je program Drive Snapshot, ovlivněnou komponentou můžou být vytvořené šifrované zálohy nebo FTP server.
Confidentality	High	Získáním hesel dojde k možnosti přístupu k chráněným datům.
Integrity	High	Získáním hesel dojde k možnosti úpravy chráněných dat.
Availability	High	Získáním hesel k FTP účtu může útočník uložené zálohy smazat a tím omezit dostupnost.

Tabulka A.13: Odvození klíče z hesla je zranitelné útokem postranním kanálem (4.2.14) – ohodnocení parametrů dle metodiky CVSS. Vector string: CVSS:3.1/AV:P/AC:H/PR:H/UI:R/S:U/C:H/I:H/A:N/E:U

parametr	hodnota	komentář
Attack Vector	Physical	Pro provedení útoku musí mít útočník fyzický přístup k danému počítači.
Attack Complexity	High	Provedení útoku je velmi náročné, pravděpodobně bude potřeba provést více měření.
Privileges Required	High	Útočník musí mít zařízení pod plnou kontrolou.
User Interaction	Required	Uživatel musí vytvořit šifrovanou zálohu v době, kdy se bude snažit útočník zaútočit.
Scope	Unchanged	Zranitelnou i ovlivněnou komponentou je program Drive Snapshot a vytvořené zálohy.
Confidentality	High	V případě úspěšného útoku může dojít k odhalení šifrovacího klíče, a tedy dešifrování dat.
Integrity	High	V případě úspěšného útoku může dojít k odhalení šifrovacího klíče, a tedy modifikaci zálohy.
Availability	None	Prolomení šifrování neohrozí dostupnost dat.
Exploit Code Maturity	Unproven	Tento útok je pouze teoretický, jeho provedení by bylo velmi náročné.

Tabulka A.14: Web programu není dostupný přes zabezpečený protokol HTTPS (4.2.16) – ohodnocení parametrů dle metodiky CVSS. Vector string: CVSS:3.1/AV:N/AC:H/PR:N/UI:R/S:C/C:H/I:H/A:H

parametr	hodnota	komentář
Attack Vector	Network	Útočník potřebuje provést *man in the middle* útok mezi webem a uživatelem.
Attack Complexity	High	Útočník musí úspěšně provést *man in the middle* útok a nahradit stahovaný program Drive Snapshot upravenou verzí.
Privileges Required	None	Útočník nepotřebuje žádná oprávnění ani na serveru, ani na klientovi.
User Interaction	Required	Útočník musí stáhnout podvržený soubor s programem Drive Snapshot a spustit jej.
Scope	Changed	Zranitelnou komponentou je server programu. Postiženou komponentou je operační systém uživatele.
Confidentality	High	Vzhledem k tomu, že Drive Snapshot běží vždy s administrátorskými právy, může s těmito právy spustit útočník vlastní kód a kompromitovat celý systém uživatele.
Integrity	High	Útočník může kompromitovat celý systém uživatele.
Availability	High	Útočník může kompromitovat celý systém uživatele.

Dodatek B

Seznam použitých zkratek

AES Advanced Encryption Standard

AES-NI Intel AES New Instructions

ASLR Address space layout randomization

API Application Programming Interface

CBC Cipher block chaining

CLI Command line interface

CRC Cyclic redundancy check

CTR Counter mode

CVSS Common Vulnerability Scoring System

ECB Electronic Codebook

FPU Floating-point unit

GPT GUID Partition Table

GUI Graphical user interface

HTTP Hypertext Transfer Protocol

HTTPS Hypertext Transfer Protocol Secure

MBR Master boot record

NIST National Institute of Standards and Technology

NX-bit Non eXecute bit

RAID Redundant Array of Independent Disks

UAC User Account Control

UPX The Ultimate Packer for eXecutables

VSS Volume Shadow Copy Service

Windows PE Windows Preinstallation Environment

Windows RE Windows Recovery Environment

Dodatek C

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `examples` ..... ukázky nalezených zranitelností
- `text` ..... text práce
  - `src` ..... zdrojový kód práce ve formátu XeLaTeX
  - `DP_Bambuch_Michal_2021.pdf` ..... text práce ve formátu PDF