Vysoké učení technické v Brně Ministerstvo vnitra ČR

Fakulta informačních technologií

***FACIS: Sada forenzních analytických nástrojů ke zpracování obrazu a videa pro službu kriminální policie a vyšetřování***

**Identifikační kód *VJ020100041***

**Název předkládaného výsledku:**

***MAXTOR - Multimedia Artifact eXTractOR***

|  |  |  |
| --- | --- | --- |
| **Typ výsledku dle UV č. 837/2017** | **Evidenční číslo (příjemce)** | **Rok vzniku** |
| ***R***software | R1 | 2022 |
| **ISBN-ISSN** | **Webový odkaz na výsledek** | **Kde a kdy publikováno** |
|  | https://www.fit.vut.cz/research/product/748/ | web |

**Stručná anotace k výsledku:**

MAXTOR je nástroj pro extrakci multimediálních dat z obrazů disků různých počítačových, ale i mobilních systémů. Podporované formáty disků jsou RAW a E01. Disková tabulka particií může byt buďto Master Boot Record (GPT), nebo GUID Partition Table (GPT). Mezi podporované souborové systémy patří EXT2, EXT3, EXT4, NTFS, FAT12, FAT16, FAT32, UFS, dále pak APFS a HFS.

Úlohou nástroje je rekurzivních průchod vstupních obrazů systému a následná extrakce multimediálních data. Množina formátů multimediálních dat je konfigurovatelná a patří mezi ně např. jpeg, bmp, tiff, png, ale tak i mpeg, mkv, mp4. Nástroj může volitelně extrahovat i kontejnery, které mohou potenciálně nést multimediální obsah a to např. doc, docx, epub, pdf, ods, odp, ppt, pptx, rtf, xls, xlsx, nebo i archívy jako zip, rar, 7z, zlib, tar, gz, atd.

**Řešitelský tým:**

*Jan Pluskal, Dominika Regéciová, Dušan Kolář, Ondřej Ryšavý*

Obsah

[Úvod 1](#_Toc148460229)

[Technická dokumentace 1](#_Toc148460230)

[Instalace 1](#_Toc148460231)

[Spuštění aplikace 2](#_Toc148460232)

[Uživatelský manuál 3](#_Toc148460233)

[Runtime aplikace 3](#_Toc148460234)

[Programátorská dokumentace 5](#_Toc148460235)

## Úvod

Tento projekt si klade za cíl vytvořit nástroj schopný rychle a efektivně extrahovat multimediální obsah z obrazů pevných disků. Primárním cílem této implementace je umožnit automatizovaný způsob extrakce zájmových souborů.

## Technická dokumentace

Technická dokumentace je rozprostřena do sekcí Instalace, Uživatelský manuál a Programátorská dokumentace.

## Instalace

Nástroj je balíčkován v Nix [[1]](#footnote-1)správci balíčků. Tento přístup umožňuje mít plnou kontrolu nad závislostmi nástroje, který se tímto stane naprosto nezávislý na operačním systému. Nástroj je možné tedy provozovat na strojích s architekturou amd64 jako program, vytvořit LXC/Docker/Podman kontejner či specializovaný virtuální stroj.

Nativní instalace se provádí importem Maxtor flake modulu a přidáním balíčku maxtor.packages."x86\_64-linux".maxtor mezi systémové balíčky viz Obrázek 1 Ukázka základního nix flake pro nástroj Maxtor.

Alternativně, případně pro vývoj je možné naklonovat repozitář projektu a následně vstoupit do vývojářského shell.

nix develop

Ve vývojářském shellu jsou připraveny všechny závislosti projektu dle deklarace modulu Maxtor[[2]](#footnote-2).

Prostředí Nix je možné použít i k sestavení projektu pomocí:

čímž vznikne výsledek sestavení dostupný v ./result.

nix build .#maxtor

V případě potřeby je možné sestavit spustitelný nástroj i čistě pomocí standardních kontejnerizačních nástrojů jako je Docker, nebo Podman a provozovat nástroj Maxtor v kontejneru.

[podman|docker] build . -t facis/maxtor:latest

Obrázek 1 Ukázka základního nix flake pro nástroj Maxtor.

{

 description = "FACIS Nixos configuration";

 inputs = {

 nixpkgs.url = "github:nixos/nixpkgs/nixos-unstable";

 maxtor = {

 url = "git+https://gitlab.nesad.fit.vutbr.cz/facis/Maxtor.git";

 inputs.nixpkgs.follows = "nixpkgs";

 };

 };

 outputs = inputs @ {

 self,

 nixpkgs,

 maxtor

 }: let

 devShellOutputs = import ./devShell.nix {inherit nixpkgs;};

 in {

 inherit (devShellOutputs) devShell devShells;

 nixosConfigurations = {

 facis-pc = nixpkgs.lib.nixosSystem {

 system = "x86\_64-linux";

 modules = [

 {

 config,

 pkgs,

 inputs,

 ...

}: {

 environment.systemPackages = [

 inputs.maxtor.packages."x86\_64-linux".maxtor

 ];

 ];

 specialArgs = {inherit inputs;};

 };

 };

 };

}

### Spuštění aplikace

Pokud je nástroj Maxtor využíván v NixOS a je tedy importován pomocí flake viz Obrázek 1, stává se systémovým nástrojem a je možné ho spustit jako jiný libovolný systémový nástroj:

./result/bin/Maxtor.Cli

Pokud byl nástroj Maxtor sestaven ze zdrojových kódů získaných z repozitáře projektu a přeložen pomocí Nix balíčkovacího správce pak je výsledek spustitelný následovně:

V případě, že je nástroj Maxtor sestaven jako kontejner, spuštění je následovné:

[podman|docker] run --rm facis/maxtor:latest

./result/bin/Maxtor.Cli

Nástroj Maxtor očekává vstupní a výstupní argumenty, které udávají, jaké obrazy má zpracovat, parametr –inputDirectory a kam má uloží výsledky, parametr –outputDirectory.

## Uživatelský manuál

Tato aplikace je primárně určena pro systémové integrátory, kteří automatizují extrakci multimediálních dat z obrazů disků. Aktuální kapitola stručně vysvětlí proces běhu aplikace a načítání rozšíření.

### Runtime aplikace

Vstupem nástroje jsou obrazy pevných disků. Nástroj zpracovává jak celistvé obrazy jednotlivých disků, tak i obrazy, které jsou rozděleny do po sobě jdoucích celků, tzv. split. Ukázka z běhu nástroje zpracovávající obraz disku se systémem Windows 11 v formátu .dd je k dispozici na Obrázek 2.



Obrázek 2 Ukázka běhu nástroje MAXTOR extrahující multimediální data a kontejnery potenciálně obsahující zájmová data z obrazu systému Windows 11 ve formátu .dd.

Nástroj MAXTOR generuje podrobný výpis zpracovaných souborů ve formě strojově zpracovatelného vstupu v jazyce JSON. Ukázka tohoto výstupu je k dispozici na Obrázek 3.

Obrázek 3 Ukázka logu nástroje MAXTOR a jeho extrakčního procesu obrahující "chain of custody".

{

"OriginalPath": "Users/facis/Documents/extracto-master\_on\_facis\_documents/ extractor-master/facis\_test\_data/videos/mp4/video\_mp4.mp4",
"ArchivedPath": "Users/facis/Documents/extractor-master\_on\_facis\_documents/ extractor-master/facis\_test\_data/videos/mp4/video\_mp4.m4v",

"FileExt": ".m4v",
"FileTypeInfo": "MPEG-4 video|QuickTime file",

"FileSize": 38273258,

"Md5Hash": "D225BD12CDBDDEF0572D71481C6E9327",

"CreationTime": "0053-11-14T06:04:20",
"LastWriteTime": "0053-11-14T05:59:28",

"LastAccessTime": "0053-11-14T06:04:20"

},

Výsledek extrakce zájmových dat je pak k dispozici v adresářové struktuře

Obrázek 4 Ukázka adresářové struktury výstupu nástroje MAXTOR.

/data/facis\_w11.e01\_\_7/viditelna\_on\_C/ └── viditelna

├── 1255183-profimedia-0528660780-base\_16x9\_jpg.jpg ├── 2021 IPMS SHOW LAS VEGAS Part 1.m4v

├── 2021 IPMS SHOW- LAS VEGAS Part 2.m4v

├── 3c1f99cee72f32eeae9eaea2dcb5697c\_.jpg

├── 473109f9d0803364a5673d7ca27fa884.jpg

├── 7dda15764889397d880e446e9b3173db\_resize=1256,1005\_.jpg

├── 9d99750b392d3b01b0ef2b7cd9d17165\_resize=356,410\_.jpg

├── ab9a260383a936f482bf495316b294cb.jpg

├── Automated number plate recognitionfrom low quality video-sequences.pdf ├── Automatic Number Plate Recognition in LowQuality Videos.pdf

├── b90b509056ae34d4891f=bd70deae61d\_2.jpg

├── baby-groot-from-guardians-of-the-galaxy\_4216x2372\_xtrafondos.com.jpg ├── c66281ce8f083a2c9fa=b68fe70f5e1b\_.jpg

├── d35d0536704e383ab7e0e296298569f0\_resize=564,376\_.jpg

├── d8eab284a4fe3909aed1c55d0c4b4030\_.jpg

├── eba8f47e813c34dbb99418b3263819f6\_.jpg

├── f80210add79b31a09a5b3add26316273\_.jpg

├── info-eduard-2022-05cz-ma.pdf

├── INIOCHOS 2022

│ ├──F-16ISufa876IsraelAirForce

│ │ ├──51993952317\_e6bae656c7\_k.jpg

│ │ ├──51993952447\_1ab9020615\_k.jpg

│ │ ├──51993953107\_a1f0f1399b\_k.jpg

│ │ ├──51994958141\_b32c874b75\_k.jpg

## Programátorská dokumentace

Nástroj MAXTOR je implementován jako .NET aplikace s využitím SDK .NET 6. Veškeré závislosti, na nativních .NET knihovnách, jsou dobře zdokumentované v rámci deklarace jednotlivých .csproj nebo také v souboru deps.nix[[3]](#footnote-3). Nástroj vyžaduje závislost na externí knihovně TheSheuthKit[[4]](#footnote-4), která je popsána v NixOS modulu[[5]](#footnote-5). Knihovna TheSleuthKit je volně dostupná pod licencí viz repositář projektu[[6]](#footnote-6).

Pro zajištění konzistentního deklarativně konfigurovaného prostředí je k dipozici Nix modul definující veškeré závislosti. Zpřístupnění vývojového shell je pak možné následovně

nix develop

což předpřipraví SDK .NET 6 a další závislosti v rámci cesty v PATH.

Následný vyvoj je pak možné provádět s využitím libovolného IDE. Nástroj MAXTOR byl vyvíjen v prostředí Visual Studio Code.

Kód dodržuje zásady CleanCode[[7]](#footnote-7) a další běžně používané zásady programování v prostředí .NET, jako např. zásady S.O.L.I.D. Kód je členěn do minimalistických celků zachycující danou úroveň abstrakce ve formě projektů. Diagram architektury, viz Obrázek 5, popisuje toto členění. Kód je tedy dle zásad CleanCode samodokumentující.

Základem je projekt Maxtor.TskLib, který pomocí C/C++ Interop volání využívá knihovnu TheSleuthKit (TSK) a vytváří pro ní C# wrapper. Dalším projektem je 0xbrock.FiletypeChecker, který slouží pro identifikaci souboru dle signatury, nebo obsahu v závislosti na daném typu souboru.

Jádrem nástroje MAXTOR je Maxtor.Lib projekt, který obsahuje veškerou aplikační logiku. Uživatel je schopný využívat rozhraní příkazové řádky implementované v projektu  Maxtor.Cli, nebo importovat PowerShell moduly implementované v projektu Maxtor.Ps.

Aby bylo zajištěno, že při vývoji se neobjeví regresní chyby tak řešení obsahuje projekt Maxtor.TskLib.Tests, který pokrývá funkcionalitu exportovanou pomocí rozhraní Maxtor.TskLib. Tento typ jednotkových (unit) testů je zvlášť důležitý v případě aktualizace knihovny Tsk, aby bylo zajištěno, že funkcionalita této knihovny spolu s jednotlivými typy i rozhraními se nezměnila a je stále kompatibilní s nástrojem MAXTOR.

Pro demonstraci a zajištění systémových testů je k dipozici projekt Maxtor.Ps.Commands.Tests, který pokrývá testy rozhraní exportované pro PowerShell.



Obrázek 5 Diagram architektury nástroje MAXTOR.

1. https://nixos.org/learn [↑](#footnote-ref-1)
2. https://gitlab.nesad.fit.vutbr.cz/facis/Maxtor/-/blob/master/nix/default.nix [↑](#footnote-ref-2)
3. https://gitlab.nesad.fit.vutbr.cz/facis/Maxtor/-/blob/master/nix/deps.nix [↑](#footnote-ref-3)
4. https://github.com/sleuthkit/sleuthkit [↑](#footnote-ref-4)
5. https://gitlab.nesad.fit.vutbr.cz/facis/Maxtor/-/blob/master/nix/default.nix [↑](#footnote-ref-5)
6. https://github.com/sleuthkit/sleuthkit/tree/develop/licenses [↑](#footnote-ref-6)
7. https://www.amazon.com/Clean-Code-Handbook-Software-Craftsmanship/dp/0132350882 [↑](#footnote-ref-7)