UNIVERZITET U NIŠU
ELEKTRONSKI FAKULTET

ANALIZA FAJL SISTEMA

Digitalna Forenzika

Mentor:

Prof. dr Bratislav Predić

Student:

Stefan Aćimović, br. ind. 15495

Niš, Februar 2020

Sadržaj

1. Uvod

Analiza fajl sistema predstavlja ispitivanje podataka na particiji ili na disku. Postoji veliki broj rezultata ove analize od kojih su na primer, izlistavanje fajlova u folderima i pregled njihovih metapodataka i povraćaj obrisanih podataka.

U prvom delu ovoh rada biće objašnjeno šta ja i zašto se koristi fajl sistem, koje sve vrste fajl sistema postoje. Biće predstavljeni NTFS i Ext fajl sistemi koji se koriste kod Windows odnosno Linux operativnog sistema.

Drugi deo rada namenjen je prezentaciji već postojećih rešenja za analizu fajl sistema na Windos operativnom sistemu.

I treći deo rada predstavlja prezentaciju implementacije jednog vida analize fajl sistema uz kompletnu strukturu aplikacije.

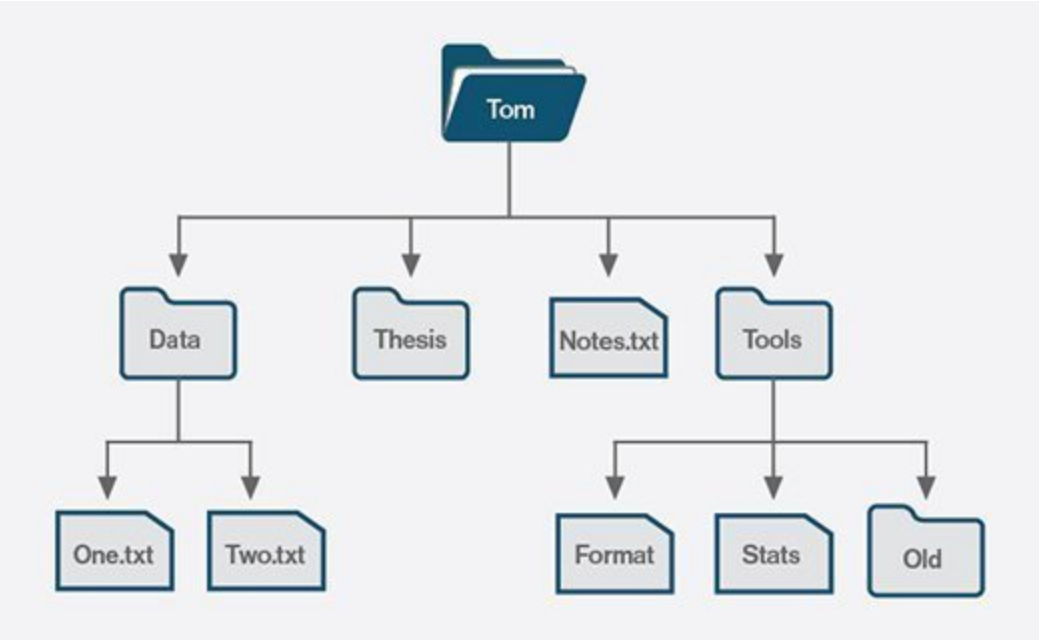
2. Analiza Fajl Sistema

2.1. Fajl Sistem

Fajl sistem predstavlja način na koji se podaci smeštaju (organizuju) i čitaju sa nekog medijuma za skladištenje podataka (npr. Hard Disk). Bez fajl sistema podacima bi se veoma teško pristupalo jer bi bilo potrebno ustanoviti gde je početak a gde kraj podatka. Zato deljenjem podataka u odvojene celine i dodeljivanjem imena tim celinama, lako možemo da razdvojimo i identifikujemo podatke koji su nam potrebni. Zbog toga se uvodeni pojmovi fajl i folder odnosno datoteka i direktorijum.

Način rada fajl sistema baziran je na primeru rada jedne poslovne kancelarije, gde papiri sa nekim podacima na njima predstavljaju fajlove, a fascikle i fioke gde se ti papiri čuvaju predstavljaju foldere. Kada nam je potreban neki papir mi tačno znamo u kojoj fascikli ili fioci se on nalazi. Isto tako i fajl sistem zna koji fajl i u kom folderu se nalazi i kako su oni smešteni na hard disku. Fajl sistem je hijararhijski organizovan, tako da se u jednom folderu mogu nalaziti fajlovi ali i drugi folderi u kojima se dalje mogu nalaziti isto fajlovi i folderi.

Svaki fajl i folder ima tri važne kategorije: ime, sadržaj i metadata. Ime fajla ili foldera se koristi za njihovo identifikovanje u fajl sistemu. Zato na primer nije moguće imati dva fajla ili foldera sa istim imenom u okviru nekog foldera. Sadržaj se više odnosi na fajlove jer su to ustvari podaci koji se čuvaju u ovkiru tog fajla. A metadata podaci su podaci koji sadrže dodatne informacije o fajlovima i folderima tj. to su podaci koji opisuju fajlove i foldere. Neki od metapodataka su na primer putanja do fajla ili foldera, njihova veličina, vreme kada su kreirani, vreme kada im je poslednji put pristupano za čitanje ili upis, itd…



Slika 1. Primer hijararhije fajl sistema

Operativni sistem mora da ima način na koji razume fajl sistem kako bi mogao da prikazuje, otvara i čuva podatke na njemu. Različiti operativni sistemi imaju različite načina za organizovanje i čuvanje podataka na hard diskovima ili nekim drugim skladištima podataka, zato se i njihovi fajl sistemi razlikuju. Windows, Mac i Linux svi koriste različite fajl sisteme, jer svako je razvijao fajl sistem onako kako je njima bilo potrebno. Tako da su neki fajl sistemi brži, neki stabilniji ili više skalabini.

Windows operativni sistem koristi fajl sisteme FAT32 i NTFS. FAT32 fajl sistem se više koristi za starije verzije Windowsa operativnog sistema (Windows XP), dok se NTFS fajl sistem koristi za novije verzije Windowsa. Mac operativni sistem koristi HFS+ fajl sistem, dok Linux koristi Ext2, Ext3 i Ext4 fajl sistem. Trenutno je u razvoju novi Btrfs fajl sistem za Linux koji bi trebao da zameni Ext4 jednog dana. U nastavku ovog rada biće opisani NTFS i Ext fajl sistemi.

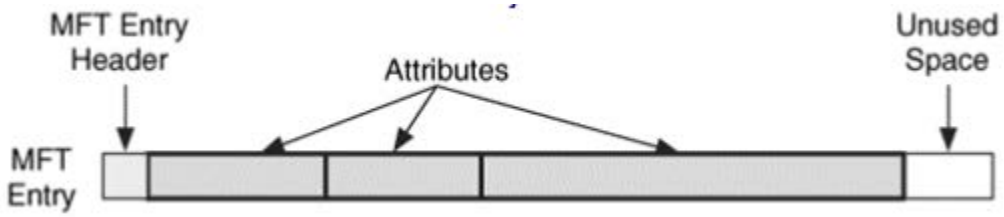
2.2. NTFS Fajl Sistem

NTFS (New Techonologies File System) je fajl sistem razvijen od strane Microsoft-a koji se koristi u svim Windows operativnim sistemima od Windows XP verzije. Pre NTFS-a korišćen je FAT32 fajl sistem koji se danas koristi kod eksternih skladišta za čuvanje podataka kao što je na primer USB fleš.

NTFS fajl sistem je dizajniran da bude pouzdan, siguran i da ima podršku za uređaje za skladištenje velike količine podataka.

Najvažniji koncept koji se mora razumeti u dizajnu NTFS fajl sistema jeste to da se svi podaci smeštaju i čuvaju kao fajlovi. Svi podaci koji su bitni za operativni sistem se takođe smeštaju u fajlove koji su najčešće sakriveni od korisnika, ali je i te fajlove moguće pronaći u ovkiru particije kao i svaki drugi običan fajl. Za razliku od drugih fajl sistema NTFS fajl sistem nema specifičan layout, već se ceo fajl sistem smatra kao oblast podataka i da bilo koji sektor može se dodeliti fajlu.

Glavni deo NTFS-a je MFT (Master File Table) tabela koja sadrži informacije o svim fajlovima i folderima. Na početku NTFS-a se prvo nalazi Boot Sector a zatim MFT tabela. Svaki fajl i folder su bar jednom uneti u ovu tabelu. Svaki vrsta u ovoj tabeli odnosi se na jedan fajl ili folder. Informacije koje se čuvaju u vrsti zauzimaju 1 KB prostora, a samo prvih 42B se imaju definisanu svrhu, dok ostali bajtovi sadrže attribute koji su ustvari manje structure podataka (npr. jedan atribut se koristi za čuvanje imena fajla). Kao i sve drugo u NTFS-u i MFT tabela je fajl. Čak i prva vrsta u MFT tabeli se odnosi na nju i ima ime *$MFT* i u ovoj vrsti se nalazi lokacija MFT tabele na disku i to je jedino mesto gde može da se pronađe lokacija MFT tabele. Veličina MFT tabele nije fiksa već se vremenom dinamički menja. Jednom dodata vrsta u MFT tabeli se više nikad ne briše, iako folder ili fajl je obrisan.



Slika 2. Izgled jedne vrste u MFT tabeli

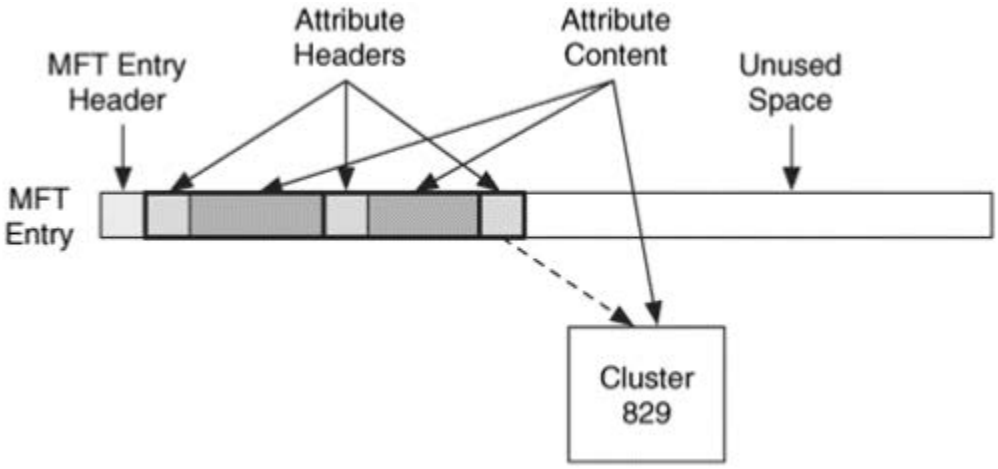
Ukoliko se desi da atributi jednog fajla ne mogu da stanu u jednu vrstu MFT tabele onda se kreiraju nove vrste sve dok se atributi ne sačuvaju, a tada prva vrsta koja se odnosi na taj fajl se naziva *base file record*, i svaka sledeća vrsta sadrži jedno polje u kome se nalazi adresa ove prve vrste.

Prvih 16 vrsta u MFT tabeli su rezervisana za sistemske fajlove. Na slici 3 su prikazani ti sistemski fajlovi.

Entry	File Name	Description
0	$MFT	The entry for the MFT itself.
1	$MFTMirr	Contains a backup of the first entries in the MFT. See the "File System Category" section in Chapter 12.
2	$LogFile	Contains the journal that records the metadata transactions. See the "Application Category" section in Chapter 12.
3	$Volume	Contains the volume information such as the label, identifier, and version. See the "File System Category" section in Chapter 12.
4	$AttrDef	Contains the attribute information, such as the identifier values, name, and sizes. See the "File System Category" section in Chapter 12.
5	.	Contains the root directory of the file system. See the "File Name Category" section in Chapter 12.
6	$Bitmap	Contains the allocation status of each cluster in the file system. See the "Content Category" section in Chapter 12.
7	$Boot	Contains the boot sector and boot code for the file system. See the "File System Category" section in Chapter 12.
8	$BadClus	Contains the clusters that have bad sectors. See the "Content Category" section in Chapter 12.
9	$Secure	Contains information about the security and access control for the files (Windows 2000 and XP version only). See the "Metadata Category" section in Chapter 12.
10	$Upcase	Contains the uppercase version of every Unicode character.
11	$Extend	A directory that contains files for optional extensions. Microsoft does not typically place the files in this directory into the reserved MFT entries.

Slika 3. Sistemski fajlovi koji se prvi nalaze u MFT tabeli

Svaki atribut u jednoj vrsti u MFT tabeli se sastoji iz zaglavlja i dela za sadržaj. U zaglavlju atributa se nalaze informacije o tipu atributa, njegovoj veličini i imenu. Takođe zaglavlje sadrži i flegove koji pokazuju da li je podatak kompresovan ili je enkriptovan. Deo koji se odnosi na sadržaj može imati bilo koji format i bilo koju veličinu. Tako da na primer za atribut koji se odnosi na sadržaj fajla koji može da bude nekoliko MB ili GB veličine, nije praktično da se to čuva kao vrste u tabeli koje su samo 1 KB veličine. Kako bi rešio ovaj problem NTFS ima dve lokacije gde može sadržaj atributa da skladišti. Ako je upitanju rezidentni atribut njegov sadržaj se čuva u MFT tabeli zajedno sa njegovim zaglavljem. Naravno ovo je moguće samo za male attribute. Za ne rezidentne atribute njihov sadržaj se čuva u eksternom klasteru na fajl sistemu. Zaglavlje atributa ukazuje na to da li je atribut rezidentan ili ne rezidentan. Ako je atribut rezidentan odmah posle zaglavlja sledi njegov sadržaj, a ako je ne rezidentan onda se u zaglavlju nalazi adresa klastera u kome se nalazi njegov sadržaj. Na slici 4 prikazan je primer ne rezidentnog atributa.



Slika 4. Izgled vrste ne rezidentnog atributa u MFT tabeli

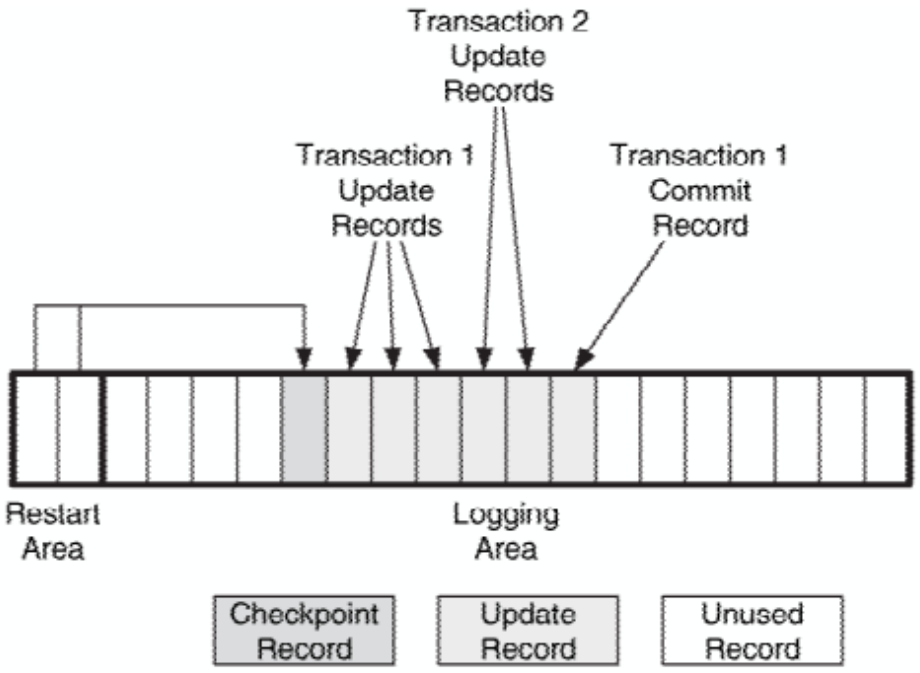
Kao što je prethodno rečeno svaki atribut ima svoj tip. Tip atributa se definiše preko njegovog numeričkog broja. Na slici 5 su prikazani svi standardni tipovi atributa u MFT tabeli. Pored numeričkog broja tip atributa ima ime koje je počinje sa znakom $ i sva su slova velika i opis koji tip šta predstavlja. Ovi atributi nisu dostupni za sve fajlove. Na primer folderi nemaju atribut tipa $DATA, a fajlovi nemaju atribut tipa $INDEX_ROOT.

Type Identifier	Name	Description
16	$STANDARD_INFORMATION	General information, such as flags; the last accessed, written, and created times; and the owner and security ID.
32	$ATTRIBUTE_LIST	List where other attributes for file can be found.
48	$FILE_NAME	File name, in Unicode, and the last accessed, written, and created times.
64	$VOLUME_VERSION	Volume information. Exists only in version 1.2 (Windows NT).
64	$OBJECT_ID	A 16-byte unique identifier for the file or directory. Exists only in versions 3.0+ and after (Windows 2000+).
80	$SECURITY_ DESCRIPTOR	The access control and security properties of the file.
96	$VOLUME_NAME	Volume name.
112	$VOLUME_ INFORMATION	File system version and other flags.
128	$DATA	File contents.
144	$INDEX_ROOT	Root node of an index tree.
160	$INDEX_ALLOCATION	Nodes of an index tree rooted in $INDEX_ROOT attribute.
176	$BITMAP	A bitmap for the $MFT file and for indexes.
192	$SYMBOLIC_LINK	Soft link information. Exists only in version 1.2 (Windows NT).
192	$REPARSE_POINT	Contains data about a reparse point, which is used as a soft link in version 3.0+ (Windows 2000+).
208	$EA_INFORMATION	Used for backward compatibility with OS/2 applications (HPFS).
224	$EA	Used for backward compatibility with OS/2 applications (HPFS).
256	$LOGGED_UTILITY_STREAM	Contains keys and information about encrypted attributes in version 3.0+ (Windows 2000+).

Slika 5. Standardni tipovi atributa u MFT tabeli

Kako bi povećali pouzdanost svog fajl sistema, Microsoft je dodao *journaling* u NTFS. Ovo je mehanizam u kome fajl sistem sve promene čuva u okviru jednog log fajla pre nego što izvrši te promene. Ovo je napravljeno sa idejom da ukoliko dođe do pada OS-a u toku ažuriranja nekog fajla taj fajl može da postane korumpiran. Ali ako pre toga upiše promene u log fajl i ako padne sistem, fajl sistem opet može kada se sistem podigne da izvrši te promene. Taj log fajl se nalazi u MFT tabeli na 2 mestu sa imenom $LogFile. Sve promene se čuvaju u okviru $DATA atributa. Po nekim istraživanjima otkriveno je da ovaj log fajl nosi dva posto veličine fajl sistema.

Log fajl ima dva glavna dela, jedan je restart deo a drugi je log deo. Kao što je prikazano na slici 6, restart deo sadrži informacije koje pomažu sistemu da se oporavi ako dođe do neke greške. Takođe sadrži pointer koji pokazuje na poslednju uspešno izvršenu transakciju koja se nalazi u log delu. Log deo sadrži različite tipove zapisa. Svaki zapis ima svoj LSN (Logical Sequence Number) koji je jedinstvena 64 bitna vrednost. LSN se inkrementira za svaki novi fajl. Pošto je log deo fiksne dužine i kada više nema mesta za nove zapise, novi zapisis se stavljaju na pocetak. Tako da zapisi na pocetku fajla mogu imati LSN veći nego oni na kraju. Postoji više tipova zapisa. Jedan od njih je update zapis koji se najviše koristi i koji se koristi za opis transakcije pre nego što se izvrši. Veliki broj transakcija zahteva više od jednog zapisa pošto su podeljene na više manjih operacija i svaka od tih operacija ima svoj zapis. Svaki update zapis pored LSN-a sadrži i informaciju o tome šta će operacija da uradi i drugi deo koji se odosi na to šta sistem da uradi kako bi se poništila ova operacija. Ovi zapisi se kreiraju pre nego što se transakcija izvrši i ažuriraju se kada se transakcija izvrši. Ovo se drugačije naziva commit zapisa. Drugi tip zapisa jeste checkpoint zapis koji se koristi da sistemu kaže odakle treba da krene kako bi verifikovao fajl sistem. Ovaj zapis se kreira na svakih 5 sekundi i on se čuva u restart delu log fajla. Da bi verifikovao fajl sistem, operativni sistem pronalazi poslednji checkpoint zapis i identifikuje poslednju transakciju koja je započela. Ako je transakcija završena i commit zapis postoji, operativni sistem proverava da li je sadržaj stvarno ažuriran na disku ili je izgubljen u kešu. Ako transakcija nije izvršena i ne postoji commit zapis, operativni sistem vraća sve u stanje pre nego što je transakcija započela.



Slika 6. Izgled $DATA atributa u okviru log fajla u MTF tabeli

2.3. Analiza NTFS Fajl Sistema

Analiza svakog fajl sistema može se svrstati u četri kategorije: analiza strukture fajl sistema, analiza sadržaja, analiza meta podataka i analiza imena fajla.

Analizom strukture fajl sistema otkriva se kako je taj fajl sistem konfigurisan kao i njegov layout. Kod NTFS fajl sistema prvi korak je analiza boot sektora koji je prvi sektor fajl sistema i koji je deo $Boot fajla. Analizom ovok sektora pronalazimo početnu lokaciju MFT tabele ka oi veličinu svake njene vrste. Sa tim informacijama možemo da procesiramo prvu vrstu MFT tabele koja se odnosi na $MFT fajl i koja će nam dati lokaciju ostatka MFT tabele. Ako je bilo koji podataka korumpiran, a znamo veličinu particije, možemo proračunati lokaciju središnjeg sektora fajl sistema gde se nalaze backup kopije prvih vrsta MFT tabele koji se nalaze u $MFTMirr fajlu. Kada smo to pronašli onda možemo da procesiramo $Volume and $AttrDef sistemske fajlove iz kojih dobijamo informaciju o verziji fajl sistema, particiji i specijalnim atributima.

Pod analizom sadržaja podrazumevamo lociranje određenog klastera, određivanje njegovog statusa i procesiranje njegovog sadržaja. Pronalaženje nekog klastera je lako zato što je prvi klister na početku fajl sistema i u njemu se nalazi definisana veličina svakog klastera. Status klastera se određuje lociranje $Bitmap fajla i procesiranjem njegovog $DATA atributa. Uobičajena praksa je da se izbaci nealociran proctor iz fajl sistema. To se radi tako što se ispituje $Bitmap fajl i sadržaj svakog klastera koji je 0 se izbacuje.

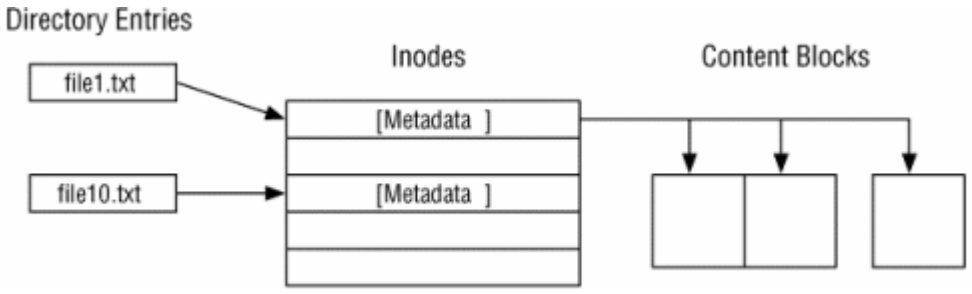
Analiza meta podataka se koristi kako bi se dobile više informacija o nekom fajlu ili folderu. Ona obuhvata process lociranja MFT vrste za taj određeni fajl ili folder i njeno procesiranje. Ali da bi pronašli tu vrstu potrebno je da prvo lociramo MFT koristeći početnu adresu u boot sektoru. Kada smo locirali tabelu onda je na redu da se analiziraju vrste. Neki fajl ili folder može da ima više vrsta zbog svoje veličine tako da je neophodno procesirati listu atributa $ATTRIBUTE_LIST za dodatne adrese koje trebaju da se procesiraju. Prilikom analize vrste prvo se locira prvi atribut. Analizira se njegovo zaglavlje, određuje se njegov tip i procesira se njegov sadržaj. Sledeci atribut se određuje dužinom prvog atributa i tako se redom dok se ne isprocesiraju svi atributi. Ako želimo neke osnovne informacije možemo odmah da tražimo $STANDARD_INFORMATION atribut i $DATA atribut koji sadrži sadržaj fajla.

Analiza imena fajla se koristi kako bi se fajlovi i folderi pronašli koristeći njihova imena. Ona obuhvata process lociranja foldera, procesiranja njegovog sadržaja i lokacija fajla. Analiza imena fajlova i indeksa kod NTFS-a je komplikovana stvar. Prvo je neophodno da se locira root folder koji se nalazi na 5-om mestu u MFT tabeli. Da bi procesirali folder prvo se analizira sadržaj $INDEX_ROOT i $INDEX_ALLOCATION atributa. Ovi atributi sadrže listu indeksa koji se odnose na čvorove u stablu. Svaki indeks predstavlja jedan fajl u okviru tog foldera. Takođe svaki indeks može imati jedan ili više indeksa. Svi ovi indeksi se pamte kao B stablo i moraju biti ponovo sortirani svaki put kada se dodaju novi fajlovi ili obrišu postojeći.

2.4. Ext Fajl Sistem

Ext je fajl sistem koji se koristi kod Linux operativnog sistema. Postoje tri verzije ovog fajl sistema: Ext2, Ext3 i Ext4. Ext2 je najstarija verzija ovog fajl sistema. Razlika između Ext2 i Ext3 je u tome što je kod Ext3 implentiran journaling ali po cenu performansi. Ext4 je noviji moderan fajl sistem koji je dosta brži od svojih prethodnika. Trenutno Ext4 je podrazumevan fajl sistem kod svih Linux operativnih sistema. Pošto je razlika između ovih verzija samo u nekim funkcionalnostima, a struktura je ista u nastavku ovog rada ovaj fajl sistem imaće ime Ext.

Ext fajl sistem je baziran na UFS (UNIX File System) fajl sistemu i dizajniran je da bude brz i pouzdan. Kod Ext fajl sistema svi podaci koji se odnose na neki fajl su lokalizovani kako glava hard diska ne bi morala da se pomera mnogo da bi ih pročitala. Layout fajl sistema počinje sa opcionim rezervisanim delom, a ostatak fajl sistema je podeljen na grupe blokova odnosno sekcije. Svaka sekcija osim poslednje ima isti broj blokova koji se koriste za smeštanje imena, meta podataka i sadržaja fajla. Meta podaci fajla ili foldera se čuvaju u strukturu podataka koja se naziva *inode*, koja je fiksne dužine i koja se nalazi u inode tabeli. Svaka grupa blokova sadrži po jednu inode tabelu. Imena fajlova se čuvaju u okviru strukture *directory entries* i one sadrže pored imena fajla i pointer na fajl inode vrstu u inode tabeli. Ova struktura se nalazi u blokovima koji su deo foldera kome pripada ovaj fajl. A sadržaj fajla se čuva u posebnim blokovima koji su ustvari grupa uzastopnih sektora.



Slika 7. Veza između osnovnih struktura kod Ext fajl sistema

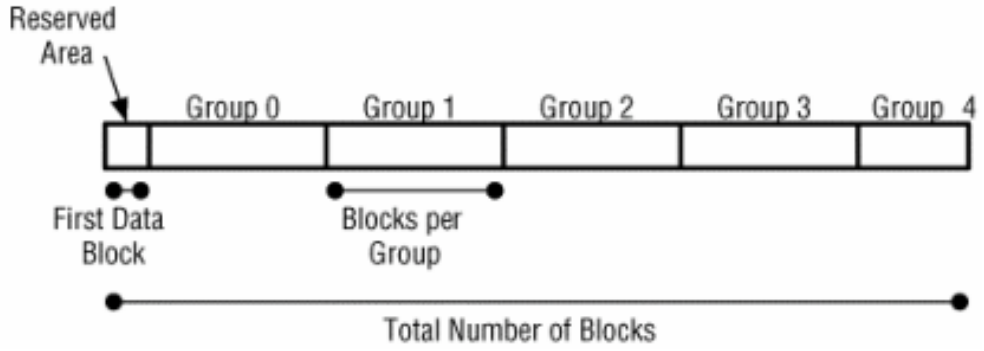
Informacije o layout strukturi Ext fajl sistema se čuva u superblok strukturi podataka, koja se nalazi na početku fajl sistema. Superblok sadrži sve osnovne informacije o veličini i konfiguraciji fajl sistema. Slično je boot sektoru kod NTFS fajl sistema. Takođe postoji i *group deskriptor* struktura podataka koja opisuje kako su organizovani blokovi u okviru grupe. Ovi deskriptori se nalaze u group deskriptor tabeli, koja se nalazi u bloku posle superbloka. Pošto su superblock i group deskritpor veoma bitni za fajl sistem više njihovih kopija se čuvaju u okviru fajl sistema ukoliko se nešto desi sa originalima.

Superblok ima veličinu od 1 KB, ali veliki broj bajtova je neiskorišćen, i nalazi se na 1 KB od starta fajl sistema. Kopije superbloka se najčešće nalaze na početku prvog bloka u svakoj grupi blokova. Superblok sadrži osnovne informacije kao što su veličina bloka, ukupan broj blokova, broj blokova po grupi i broj rezervisanih blokova. Takođe sadrži informacije o ukupnom broju inode-ova, kao i broj inode-ova po grupi blokova. Superblok sadrži i informacije o ukupnom broju slobodnih inode-ova i blokova koje se kasnije koriste prilikom alokacije novih inode-ova i blokova.

U Ext fajl sistemu blok može biti veličine 1, 2 i 4 KB i informacija o njihovoj veličini se čuva u superbloku. Adrese blokova kreću od 0, tako da blok 0 se nalazi u prvom sektoru fajl sistema. Svaki blok je deo neke grupe blokova. Grupe blokova počinju odmah nakon rezervisanih blokova kao što je prikazano na slici 8. Kojoj grupi blok pripada računa se po formuli:

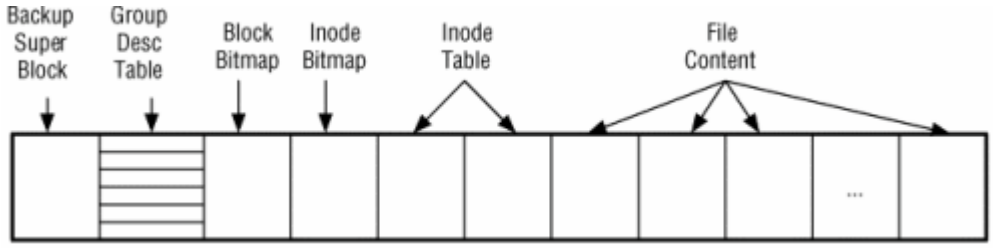
$$Grupa = \frac{Blok - Rezervisani\ Blokovi}{Broj\ Blokova\ Po\ Grupi}$$

Tako da na primer ako nema rezervisanih blokova i nalazi se 32768 blokova po grupi, blok 60000 nalaziće se u grupi 1.



Slika 8. Izgled blokova Ext fajl sistema na primeru 5 grupe blokova

Svaka grupa blokova sadrži kopiju superbloka, group deskriptor tabele, inode tabele, inode bitmape i blok bitmape.



Slika 9. Izgled blok grupe u okviru Ext fajl sistema

Group deskriptor tabela koja u fajl sistemu sledi odmah posle superbloka, koja sadrži group deskriptore za svaku grupu blokova u fajl sistemu. Kopija ove tabele se nalazi u svakoj grupi blokova kao i superblok. Group deskriptor sadrži informaciju gde se nalaze ovi blokovi, ka oi koji su free blokovi i indoe-ovi u okviru neke grupe blokova.

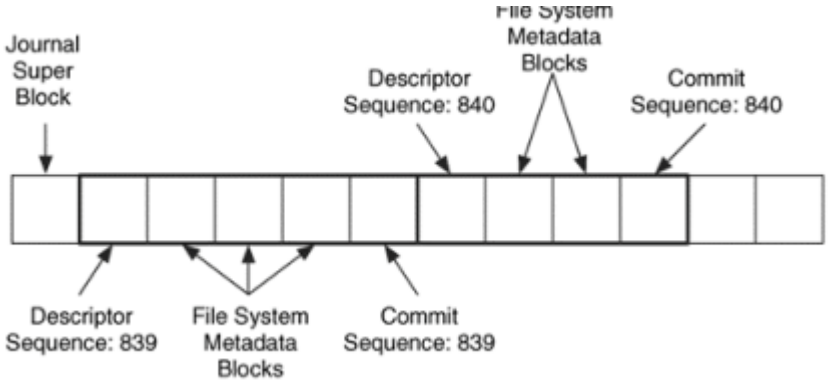
Blok bitmapa se koristi za status alokacije blokova u okviru grupe i njena lokacija se može naći u group deskriptoru. Veličina ovog bloka može da se izračuna deljenjem broja blokova u grupi sa osam. U današnjim Linux sistemima ovaj blok je tačno veličine jednog bloka, a veličina bloka se definiše prilikom kreiranja fajl sistema.

Inode bitmapa se koristi kao i blok bitmapa samo što se odnosi na inode-ive u grupi. I njena lokacija kao i lokacija blok bitmape se nalazi u group deskriptoru.

Ukoliko Ext fajl sistem sadrži kernel operativnog sistema, on onda sadrži boot kod, a svi ostali fajl sistemi ne bi trebali da imaju boot kod. Ukoliko boot kod postoji on se nalazi na 1 KB pre superbloka. Boot kod će se izvršiti tek kada dobije kontrolu od strane MBR (Master Boot Record) u nultom sektoru na disku. U Ext boot kod obično zna koji blokovi sadrže kernel operativnog sistema i njih onda učitava u memoriju. Mnogi današnji Linux sistemi nemaju boot kod u fajl sistemu gde se nalazi kernel, već postoji boot loader u MBR-u koji sadrži informaciju u kojim blokovima se kernel nalazi i u tom slučaju MBR će učitati kernel.

Kao i NTFS i Ext3 i Ext4 fajl sistemi koriste journaling tehniku. Journal se obično nalazi u inode-u 8 čija lokacija se nalazi u okviru superbloka. Journal sadrži zapise o operacijama koje će se izvršiti za ažuriranje blokova. Postoje dva moda u kojima journal radi. U jednom modu se samo

zapisuju ažuriranja metapodataka, a u drugom modu sva ažuriranja, uključujući i ažuriranja sadržaja blokova. Trenutne verzije Ext fajl sistema koriste prvi mod. Journaling u Ext fajl sistemu radi na nivou blokova, što znači da ukoliko se promeni jedan bit u okviru nekog inode-a, ceo taj blok u kome se nalazi inode biće sačuvan u journal-u. Prvi blok u journal-u je uvek superblok, a svi ostali blokovi journal-a se koriste za ostale blokove. Ažuriranja se kao i kod NTFS fajl sistema vrši u transakcijama i svaka transakcija ima svoj broj. Blokovi u journal-u sadrže ili informacije o transakciji ili ažurirani sadržaj. Svaka transakcija počinje sa blok deskriptorom koji sadrži informaciju o broju transakcije i listom blokova koji se ažuriraju. Ovog bloka idu ažurirani blokovi. Nakon što je transakcija izvršena i sve sačuvano na disk onda se dodaje i commit blok sa istim brojem kao i broj transakcije. Moguća je situacija da se pre commit bloka nađe novi blok deskriptor za sledeću transakciju.



Slika 10. Primer Ext journal-a sa dve transakcije

2.5. Analiza Ext Fajl Sistema

Isto kao i kod NTFS fajl sistema i u ovom delu biće opisana analiza Ext fajl sistema preko četri kategorije analize: analiza strukture fajl sistema, analiza sadržaja, analiza meta podataka, analiza imena fajla.

Analiza strukture fajl sistema se koristi kako bi se izanalizirala sve bitne strukture podataka Ext fajl sistema iz koji dobijamo informaciju o layout-u fajl sistema sa kojim kasnije možemo da primenimo ostale tehnike analize. Kod Ext fajl sistema prvi korak je da se locira superblok. To je lak posao pošto se on uvek nalazi na 1 KB od početka fajl sistema. Kada se on pronađe vrši se njegovo procesiranje čime dobijamo ključne informacije kao što su veličina i lokacija najvažnijih struktura podataka. Ako se desi da je originalni superblok korumpiran to nije problem pošto se njegove kopije nalaze na početku svake grupe blokova. Takođe u superbloku se nalaze i flegovi koji nam pokazuju da li fajl sistem ima neke jedinstvene funkcionalnosti koje nisu podržane od stane alata za analizu i operativnog sistema. Veličinu fajl sistema možemo da izračunamo množenjem veličine bloka i broja blokova u fajl sistemu. Ako je ova vrednost manja od veličine particije to znači da je moguće da postoje neki sakriveni podaci u okviru fajl sistema. Nakon superbloka nalazi se group deskriptor tabela i uz superblok sada možemo da lociramo sve strukture podataka koje se koriste za fajlove i foldere.

Kod analize sadržaja lociraju se blokovi, čitaju njihovi sadržaji i određuje njihov status. Pronalaženje određenog bloka je lako pošto prvi blok počinje u prvom sektoru fajl sistema, a informaciju o veličini blokova nalazimo u superbloku. Ukoliko želimo da izbacimo sve nealocirane blokove iz fajl sistema taj posao se radi obradom svake vrste u okviru group deskriptor tabele. Svaka vrsta učitava bitmapu bloka i pretražuje 0. Svi blokovi čiji su bitovi 0 se izbacuju.

Analiza meta podataka obuhvata lociranje i procesiranje strukture podataka koje sadrže meta podatke o fajlovima i folderima. Kod Ext fajl sistema meta podaci se čuvaju u inode strukturama podataka. Da bi locirali određeni inode prvo moramo da pronađemo kojoj grupi blokova pripada. Nakon toga kada se grupa pronađe neophodno je procesirati group deskriptor kako bi se pronašla inode tabela grupe. I na kraju iz inode tabele pronađemo vrstu koja nam je potrebna i nju procesiramo. Nekada je dobro procesirati i ne alocirane inode-ove pošto se u njima nalaze informacije o fajlu koji je obrisan.

Analiza imena fajlova podrazumeva da izlistamo imena svih fajlova u folderu kako bi mogli da pronađemo traženi fajl ili fajlove koji odgovaraju određenom šablonu. Prvi korak je da se locira root folder. Ovo je lako pošto se on u Ext fajl sistemu uvek nalazi u inode-u 2. Folderi u Ext fajl sistemu se tretiraju slično kao i fajlovi samo što imaju poseban tip u okviru njihovog inode-a. Kada pronađemo root folder onda se vrši sekvencijalno procesiranje svih njegovih fajlova i foldera tj. njihovih struktura podataka. Ako želimo da procesiramo samo alocirane fajlove i foldere onda nakon procesiranog jednog fajla skačemo dalje za veličinu tog procesiranog fajla i tako sve do kraja bloka. Ako želimo da vidimo nealocirane fajlove onda se ignoriše veličina procesiranog fajla već se uzme više bajtova.

3. Postojeća Rešenja

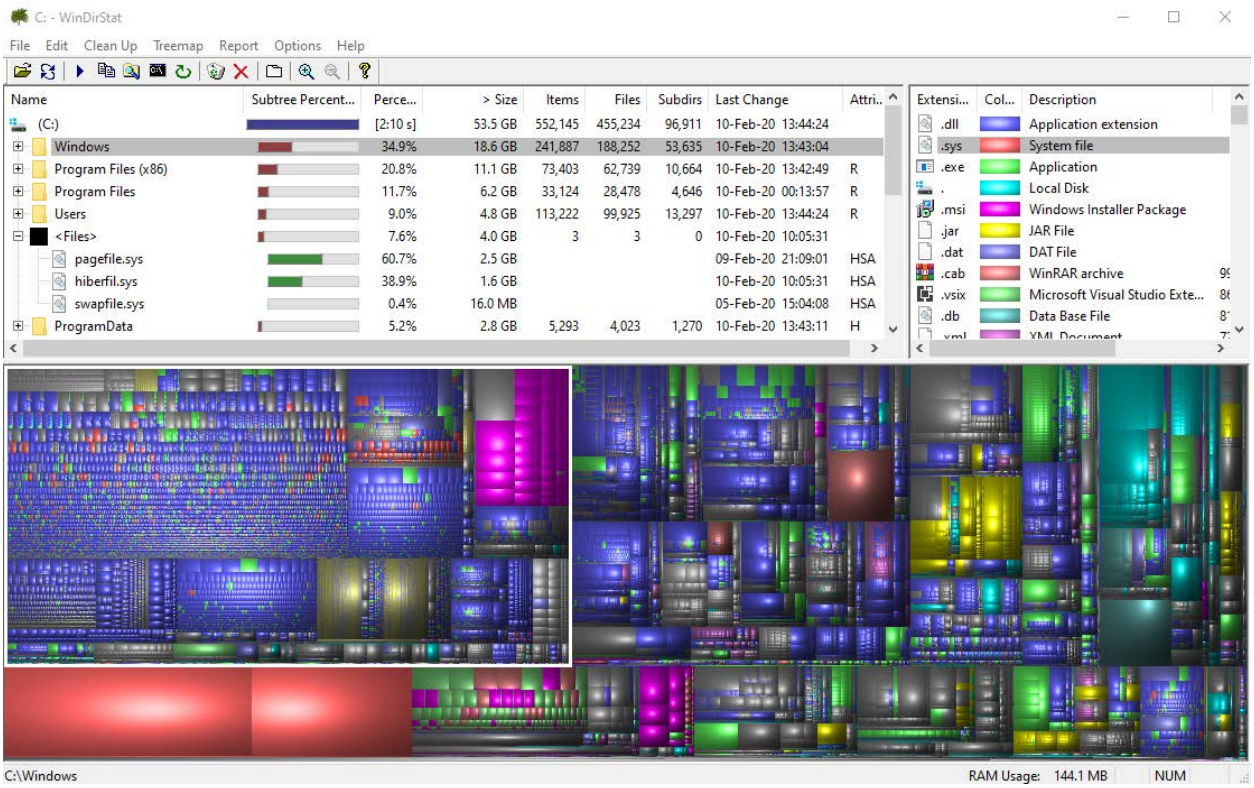
U današnje vreme postoji veliki broj alata koji se koriste za analizu fajl sistema. Neki od alata su besplatni a neki se pak plaćaju i međusobno imaju isti cilj da prikažu sve što se nalazi na fajl sistemu, ali se ipak razlikuju po dodatnim funkcionalnostima koje nude. U ovom delu rada biće opisana četri najpoznatija alata za analizu fajl sistema kod Windows operativnog sistema: WinDirStat, SpaceSniffer, TreeSize i Everything.

3.1. WinDirStat

Najstariji i najrasprostranjeniji alat za analizu fajl sistema jeste sigurno WinDirStat. Interfejs WinDirStat-a je veoma jednostavan, ali nam opet daje sve informacije koje su nam potrebne.

Prilikom pokretanja WinDirStat nudi opciju da se izabere da li da analizira samo jednu particiju ili sve particije. Ukoliko izaberemo sve particije onda će potrajati malo više vremena dok aplikacija ne obradi sve fajlove. Kada se završi skeniranje na raspolaganju su nam tri dela odnosno panela kao što je prikazano na slici 11. Prvi panel odnosi se na hijerarhijski prikaz foldera i fajlova kao i informacije o njihovoj veličini, ukupnom procentu koji zauzimaju, broju podfoldera i fajlova, kada su zadnji put ažurirani. Drugi panel predstavlja listu svih ekstenzija koje su pronađene u okviru particije. U prvoj koloni je naziv ekstenzije, u drugoj boja kojom su obojeni svi fajlovi koji imaju ovu eksteziju i koja se koristi u trećem panelu i u trećoj koloni je opis ekstenzije odnosno ceo naziv ekstenzije. Treći panel predstavlja grafički prikaz strukture fajl sistema. Svaki fajl i folder su predstavljeni kao pravougaonik i u zavisnosti od ekstenzije taj pravougaonik ima neku boju. Što je procenat koji folder odnosno fajl zauzima to je i pravougaonik veći i biće više vidljiviji u grafičkom prikazu. Klikom na bilo koji folder ili fajl iz prvog panela u drugom i trećem panelu se obeleži odgovarajuća ekstenzija odnosno pravougaonik, a takođe moguće je i obrnuto. Pored ovih informacija WinDirStat takođe omogućava i da se folderi i fajlovi otvore odnosno pokrenu direktno iz aplikacije. Takođe omogućava i otvaranje terminala za konkretan folder, kao i brisanje foldera i fajlova.

Možda jedina zamerka kod WinDirStat alata jeste to što mu je potrebno malo više vremena dok obradi ceo fajl sistem i dok izgeneriše grafički prikaz, ali ako posmatramo koliko informacija nam pruža u odnosu na druge brže alate verujem da ova mana može i da se zanemari.



Slika 11. Izgled WinDirStat aplikacije

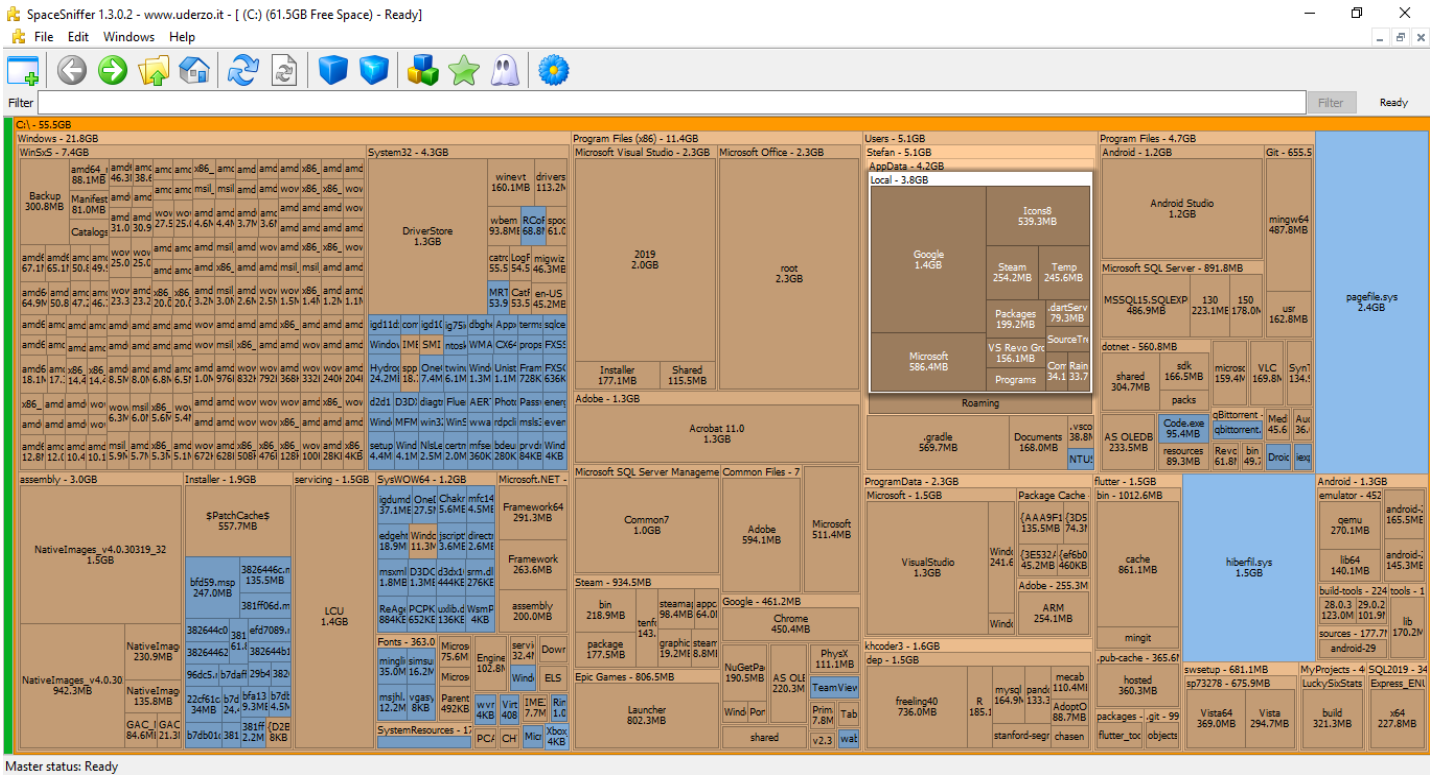
3.2. SpaceSniffer

SpaceSniffer za razliku od ostalih alata nema listu foldera i fajlova, ali je zato najbolji alat za grafički prikaz stanja fajl sistema.

Kao i kod WinDirStat alata i kod ovog alata prilikom pokretanja moguće je odabrati da se skenira jedan ili više particija. Dobra stvar kod ovog alata jeste to što dok se u pozadini vrši skeniranje fajl sistema, grafički prikaz se konstantno ažurira kad god se obradi novi folder ili fajl. Kada je skeniranje gotovo dobijemo grafički prikaz fajl sistema kako je prikazano na slici 12.

Svaki pravougaonik na slici predstavlja neki folder ili fajl. Dobra stvar je u tome što je moguće ići u dubinu. Tako da dupli klik na bilo koji pravougaonik koji predstavlja folder otvara grafički prikaz za taj folder i tako je moguće ići do kraja u dubinu. SpaceSniffer nema mogućnost da otvori fajl ili folder, ali zato desni klik na bilo koji pravougaonik otvara context menu za taj fajl odnosno folder, tako da je moguće iz aplikacije upravljati njima. Takođe SpaceSniffer ima opciju da u grafički prikaz prikaže i slobodan prostor kao i nealociran prostor.

Za razliku od WinDirStat alata SpaceSniffer je dosta brži i ima bolji i kvalitetniji grafički prikaz, ali količina informacija koju nam pruža je dosta manja od WinDirStat alata. Ukoliko nam nije važno da dobijemo sve informacije o fajlovima i folderima, već samo da vidimo koji fajlovi zauzimaju najviše prostora onda je SpaceSniffer najbolji izbor.



Slika 12. Izgled SpaceSniffer aplikacije

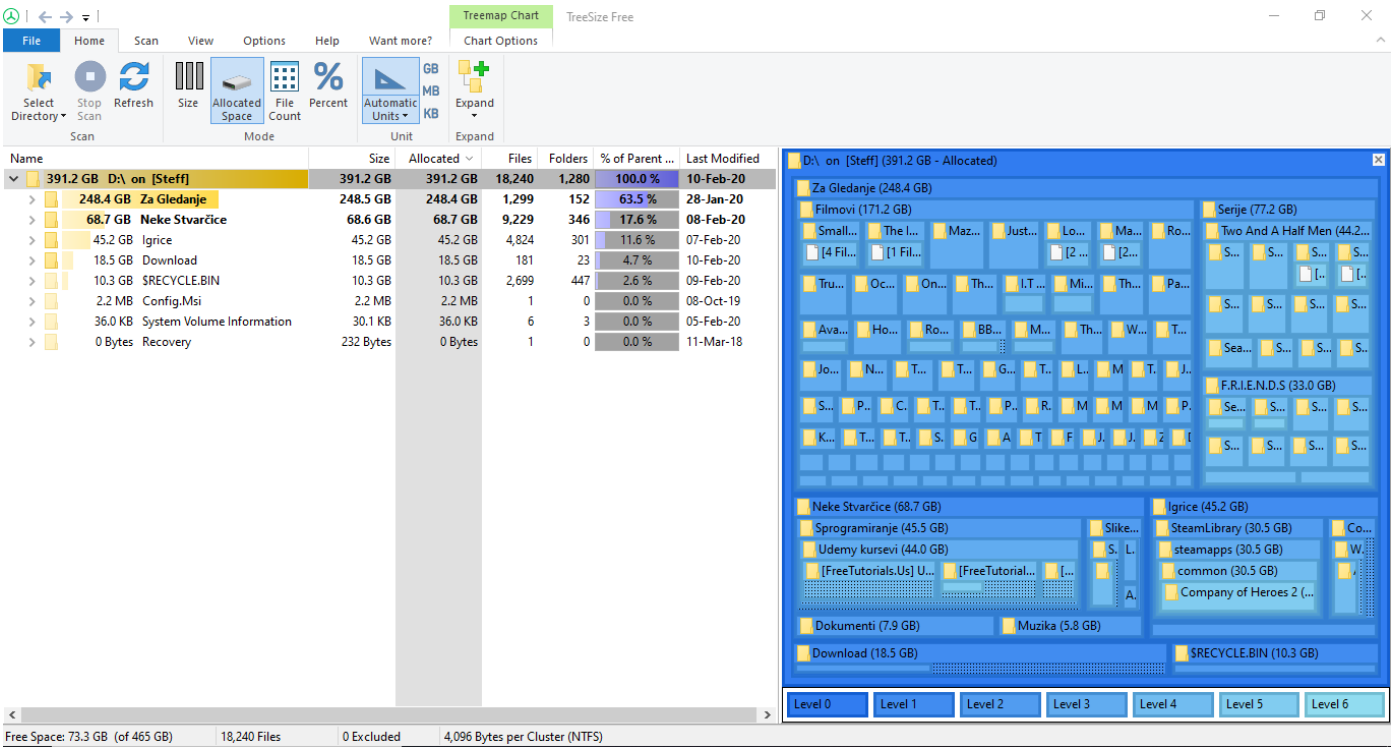
3.3. TreeSize

TreeSize je jedini od ovih alata koji ima i svoju plaćenu verziju. TreeSize je alat koji se preporučuje kao zamena za WinDirStat, mada ima mnogo više funkcija od njega. TreeSize je najmlađi alat od svih navedenih i iz tog razloga ima najbolji i najmoderniji interfejs od svih.

Kao i svi do sad i TreeSize na početku nudi opciju biranja particije, ali je moguće i izbor samo konkretnog foldera. Skeniranje kod TreeSize-a je veoma brzo, jer kako navodi kompanija koja je vlasnik ovog alata oni za skeniranje koriste MFT tabelu, a ako nema pristup MFT tabeli onda se skeniranje vrši u dva paralelna treda za free verziju i do tridesetdva treda kod plaćene verzije TreeSiza-a.

Kada je skeniranje završeno na ekranu je vidljiv samo jedan panel i to hijararhijski prikaz po folderima i informacije o njima. Prikazane su informacije su iste kao i kod WinDirStat alata: veličini svakog foldera i fajla, broj fajlova i podfoldera, procenat koji zauzimaju i kada su zadnji put modifikovani. U okviru Options taba moguće je filtrirati i pretraživati fajlove i foldere kao i uključiti dodatne opcije za prikaz liste. Po default-u grafički prikaz nije uključen i neophodno je uključiti ga u View tabu. Na grafičkom prikazu prikazani sus vi folderi pojedinačno. Za razliku od grafičkog prikaza kod WinDirStat i SpaceSniffer alata manje je čitljiviji i pregledan. Dupli klik na bilo koji folder otvara detaljan prikaz tog foldera i proširuje listu do tog foldera. Klikom na bilo koji folder u listi otvara njegov grafički prikaz u drugom panelu. Takođe desni klik na bilo koji od foldera u listi i u grafičkom prikazu otvara windows context meni tako da je moguće upravljati fajlom ili folderom preko njega. Dobra stvar kod ovog grafičkog prikaza jeste opcija koja omogućava da se folderi i fajlovi prebacuju jedan u drugi jednostavnim prevlačenjem. Ova opcija kao i ostale vezane za grafički prikaz nalaze se u Chart Options tabu. Takođe tu se nalaze funkcije kojim se može podešavati grafički prikaz, kao što je na primer broj detalja, 3D prikaz, pozicija prikaza, itd...

Kada uporedimo TreeSize sa ostalim alatima možemo da zaključimo da je u prednosti u odnosu na njih zbog mnogo više funkcija i to kod besplatne verzije, plaćena verzija uključuje još mnogo drugih funkcija. Iako grafički prikaz nije baš najbolji opet je dosta dobar u kombinaciji sa svim infomacijama koje nam nudi ovaj alat. Tako da ovaj alat je sigurno naslednik WinDirStat alata.



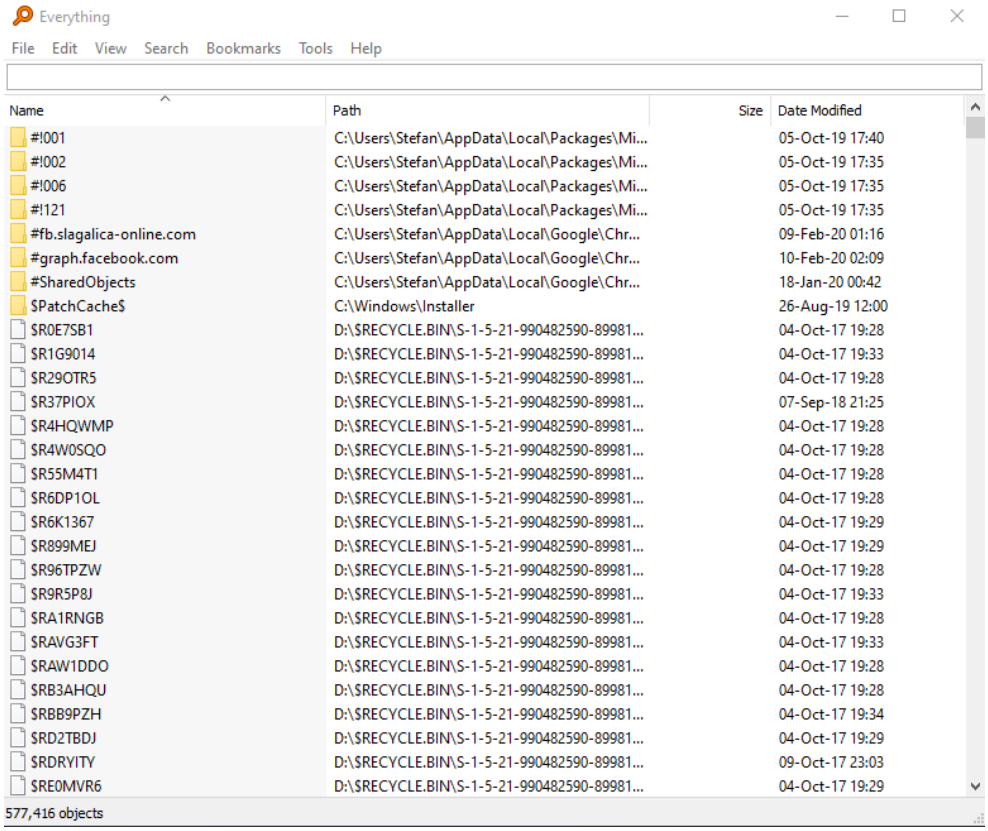
Slika 13. Izgled TreeSize aplikacije

3.4. Everything

Everything je alat koji za razliku od ostalih pomenutih alata se najviše koristi za pronalaženje konkretnog fajla ili foldera na fajl sistemu. Interfejs Everything aplikacije je jednostavan i ne prikazuje nikakve dodatne informacije osim imena fajla, putanje i vreme poslednje promene.

Prilikom pokretanja aplikacije potrebno je malo vremena da bi se učitali svi podaci, ali pošto Everything skenira sve particije ovo vreme je zanemarljivo. Nakon skeniranja dobijamo listu svih fajlova i foldera na fajl sistemu uključujući i one sakrivene. Pretraživanje se vrši u okviru text boxa koji se nalazi na vrhu aplikacije. U opcijama moguće je uključiti pretraživanje koristeći regularne izraze. Takođe moguće je filtriranje i pretraživanje fajlova i foldera po mnogim kriterijumima koji se mogu naći u Search tabu. Takođe moguće je i obeležiti neke foldere tako da prilikom sledećeg otavaranja aplikacije možemo brže pronaći.

Iako se Everything dosta razlikuje od drugih navedenih alata i ima samo jednu svrhu a to je pronalaženje foldera ili fajla ipak je vredan da se uvrsti u alate za analizu fajl sistema. Ako bi kombinovali upotrebu Everything-a i nekog od prethondih alata dobijamo veoma dobre alate za pretraživanje i analizu fajl sistema.



Slika 14. Izgled Everything aplikacije

4. FileForensiq

U ovom delu rada biće detaljno opisana implementacija aplikacije za analizu fajl sistema FileForensiq.

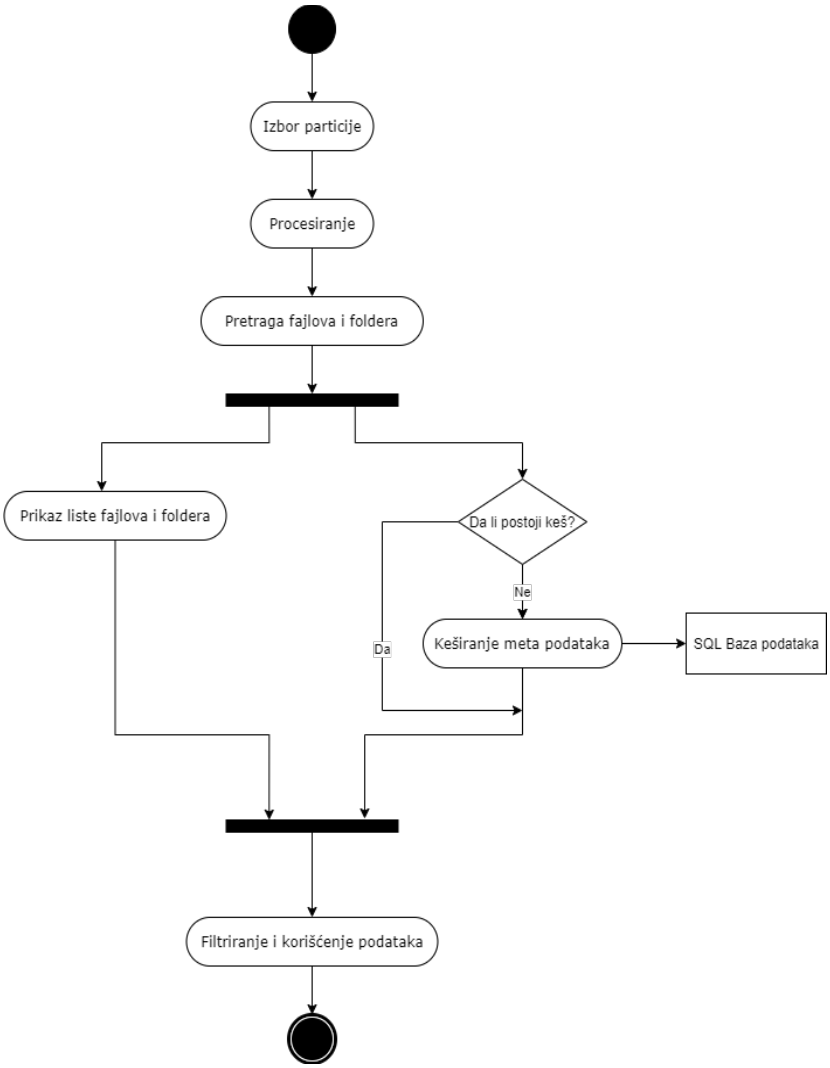
FileForensiq je alat za analizu fajl sistema koji je razvijen po ugledu na WinDirStat alat koji je opisan u prethodnom delu rada. Koristi se za prikaz svih fajlova i foldera koji se nalaze na fajl sistemu, uz to postoji opcija filtriranja po raznim kriterijumima. Ali jedna funkcionalnost koja razlikuje ovaj alat od drugih jeste mogućnost keširanja podataka odnosno čuvanja meta podataka fajlova i foldera. Uz keširanje moguće je pratiti kako su se menjali fajlovi i folderi kroz neki period. Takođe uz keširanje moguće je i videti koji su fajlovi obrisani i kada. Keširanje podataka se vrši na nivou dana a ne na nivou otvaranja aplikacije. Sve ove funkcionalnosti biće prikazane u nastavku.

4.1. Strutkura aplikacije

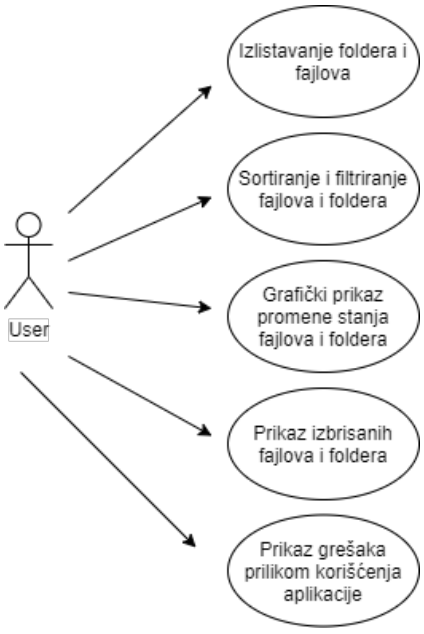
Aplikacija je kreirana u C# i Windows Formama. Baza koja se koristi je Microsoft SQL, koja se koristi za keširanje meta podataka fajlova i foldera.

Na slici 15 prikazan je UML dijagram aktivnosti aplikacije. Neki tok aktivnosti nako pokretanja aplikacije jeste da se prvo izabere particija koja će se skenirati nakon čega se u pozadini procesiraju svi fajlovi i folderi particije i kada se svi podaci prikupe oni se prikazuju u listi dok se u pozadini proverava da li postoji keš za taj dan i ako ne postoji podaci se keširaju u SQL bazu, a ako postoji keširanje se preskače. Nakon toga aplikacija se koristi za pretraživanje i filtriranje podataka.

Na slici 16 prikazan je Use Case UML dijagram aplikacije. Na njemu su prikazane sve funkcije korisnika prilikom korišćenja aplikacije. U te funkcije spadaju: izlistavanje fajlova i foldera fajl sistema, njihova pretraga, sortiranje i filtriranje, grafički prikaz promena nekog fajla ili foldera, prikaz svih obrisanih fajlova i foldera, kao i prikaz grešaka koje su nastale prilikom korišćenja aplikacije.



Slika 15. UML Dijagram Aktivnosti aplikacije



Slika 16. Use case UML dijagram

4.2. Keširanje

Keširanje je i najbitnija funkcija koja razlikuje ovu aplikaciju od drugih. Ideja keširanje jeste da se nekako sačuvaju svi bitni meta podaci fajlova i foldera kako bi se kasnije koristili za analizu promena za neki dati period. Problem koji se javio jeste način na koji sačuvati te podatke i gde.

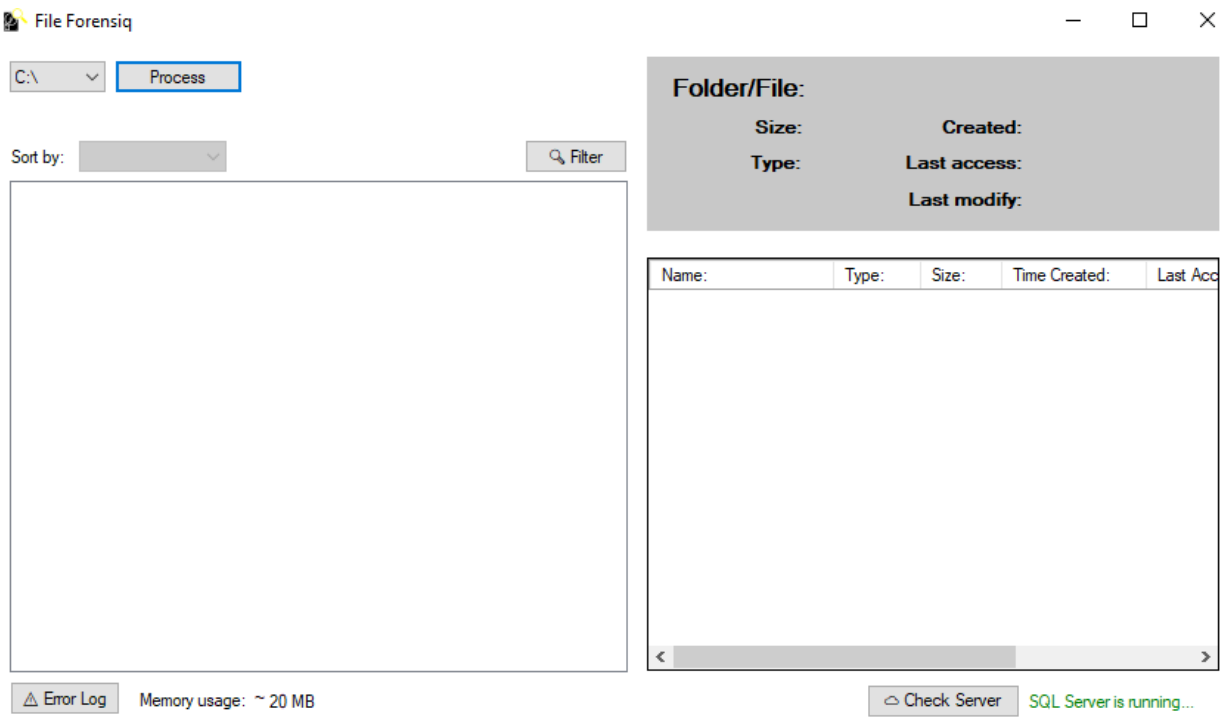
Prva ideja bila je da se ti podaci čuvaju u okviru nekog txt fajla ili csv fajla. Iako ovo rešenje doprinosi tome da ti podaci ne zauzimaju mnogo memorije, problem je to što je procesiranje txt i csv fajla da bi našli podatke o konkretnom fajlu ili folderu veoma teško, a pošto su performanse bitne ova ideja je odbačena. Druga ideja je bila da se iskoristi Redis baza podataka zbog njenih performansi, ali i tu se naravno javio problem zbog toga što kod besplatne verzije redisa postoji ograničenje u čitanju i upisu u periodu od sat vremena na 6000. A pošto naš fajl sistem ima i po 400000 fajlova ni ova ideja nije moguća. I na kraju izbor je pao na Microsoft SQL bazu. Dobra stvar kod SQL baze jeste to što je upis i čitanje podataka brzo i lako, ali loša stvar je to što zauzima dosta memorije.

Keširanje se vrši na nivou dana, a ne svaki put kada se upali aplikacija. Ideja iza keširanja je da se prodje kroz ceo fajl sistem i za svaki fajl i folder sačuvaju podaci kao što ime (koje je ustvari cela putanja do fajla ili foldera), ekstenzija, veličina, broj fajlova (ako je folder), vreme kreiranja, vreme poslednje modifikacije i vreme poslednjeg pristupa i ti podaci sačuvaju u DataTable objekat. Pre nego što se podaci upišu kreira se nova tabela u bazi sa trenutnim datumom i onda se bulk funkcijom svi podaci odjednom upisuju u bazu čime smanjujemo broj pristupa bazi. Što je veći broj podataka koji se upisuju u tabelu tako i ta tabela zauzima više prostora, ali svaka tabela se briše posle 30 dana. Ali kasnije je pretraga podataka i čitanje iz baze mnogo brže i lakše.

	Id	Name	Extension	Size	NumberOfFiles	CreationTime	LastAccessTime	LastModificationTime
1	1	D:/$RECYCLE.BIN/S-1-5-21-295004308-3827068042-42...	ini	129	0	2019-10-05 17:36:38.540	2020-02-07 20:21:06.820	2019-10-05 17:36:38.540
2	2	D:/$RECYCLE.BIN/S-1-5-21-295004308-3827068042-42...	folder	129	1	2019-10-05 17:36:38.540	2020-02-07 17:19:18.373	2020-02-07 17:19:18.337
3	3	D:/$RECYCLE.BIN	folder	129	1	2015-08-31 10:37:41.107	2020-02-07 17:05:03.300	2019-11-23 13:54:18.430
4	4	D:/Download/Brooklyn.Nine-Nine.S07E01.HDTV.x264-S...	mkv	173153961	0	2020-02-07 12:42:45.213	2020-02-07 15:41:22.853	2020-02-07 15:41:21.843
5	5	D:/Download/Brooklyn.Nine-Nine.S07E01.HDTV.x264-S...	nfo	4506	0	2020-02-07 15:41:14.083	2020-02-07 15:41:22.903	2020-02-07 15:41:14.083
6	6	D:/Download/Brooklyn.Nine-Nine.S07E01.HDTV.x264-S...	txt	30	0	2020-02-07 15:41:14.010	2020-02-07 15:41:14.010	2020-02-07 15:41:14.010
7	7	D:/Download/Brooklyn.Nine-Nine.S07E01.HDTV.x264-S...	exe	99	0	2020-02-07 15:41:14.010	2020-02-07 15:41:14.080	2020-02-07 15:41:14.080
8	8	D:/Download/Brooklyn.Nine-Nine.S07E01.HDTV.x264-S...	folder	173158596	4	2020-02-07 12:42:45.213	2020-02-07 17:00:00.923	2020-02-07 15:41:14.083
9	9	D:/Download/Brooklyn.Nine-Nine.S07E02.HDTV.x264-KI...	mkv	148506330	0	2020-02-07 12:43:00.647	2020-02-07 15:41:12.980	2020-02-07 15:41:11.563
10	10	D:/Download/Brooklyn.Nine-Nine.S07E02.HDTV.x264-KI...	nfo	10872	0	2020-02-07 15:41:08.737	2020-02-07 15:41:13.093	2020-02-07 15:41:08.737
11	11	D:/Download/Brooklyn.Nine-Nine.S07E02.HDTV.x264-KI...	txt	30	0	2020-02-07 15:41:08.723	2020-02-07 15:41:08.723	2020-02-07 15:41:08.723

Slika 17. Izgled tabele u SQL bazi

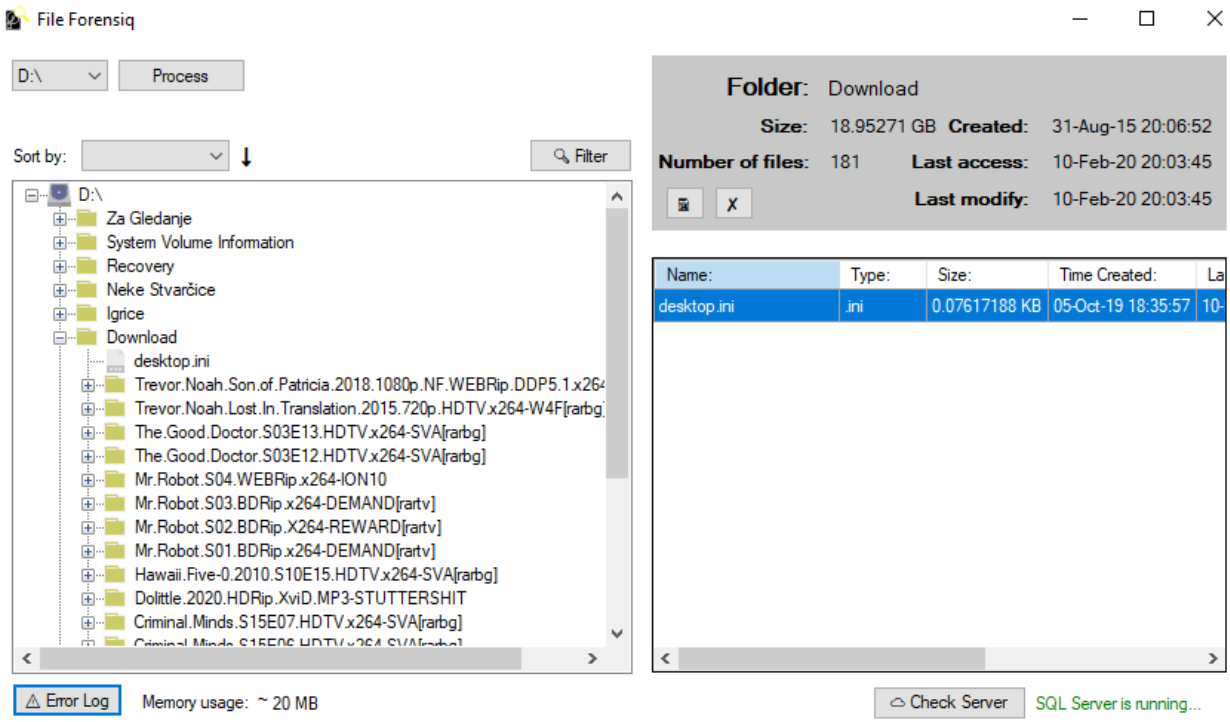
4.3. Izgled i Funkcionalnosti aplikacije



Slika 18. Izgled početne forme FileForensiq aplikacije

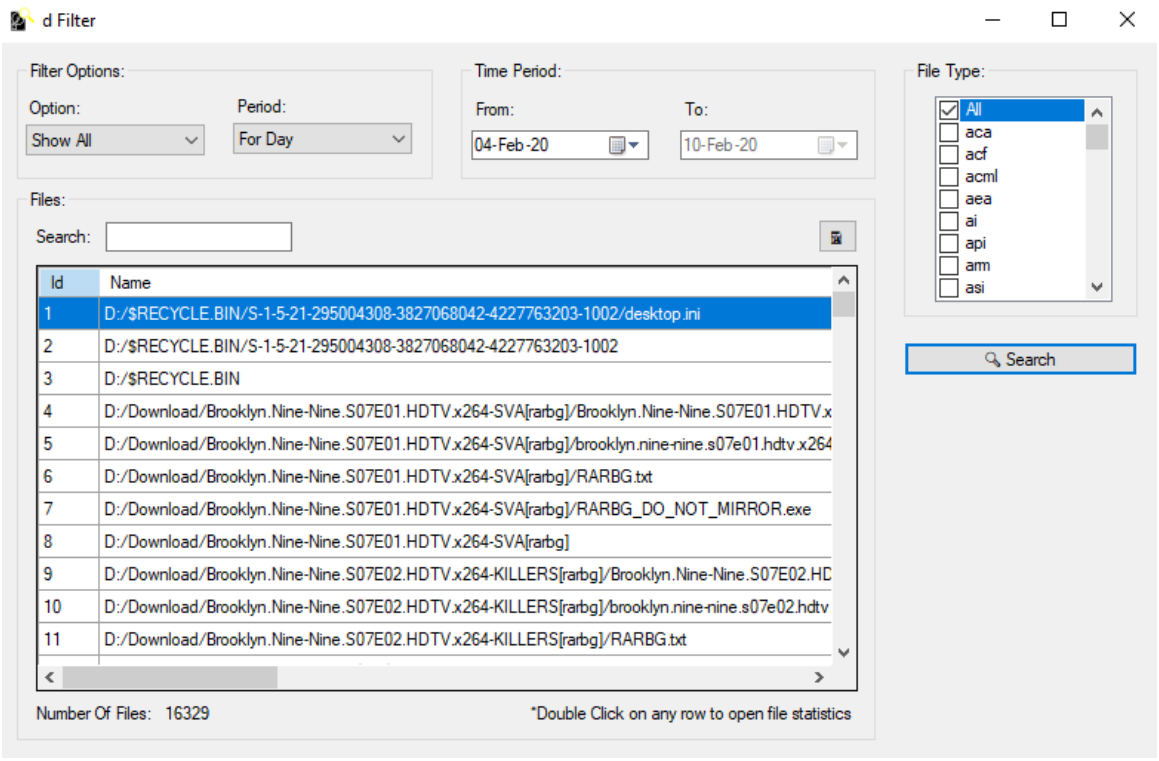
Na slici 18 prikazan je izgled početne forme FileForensiq aplikacije tj. izgled aplikacije kada se pokrene. Forma je podeljena u dva dela. Jedan deo je namenjen za izlistavanje svih fajlova i foldera, a drugi deo za detaljniji prikaz istih. U gornjem levom uglu nalazi se combobox za izbor particije i nakon čega se vrši procesiranje pritiskom na dugme Process. Pretraživanje fajlova i foldera se vrši korišćenjem kalsa DirectoryInfo i FileInfo i funkcija EnumerateDirectories i EnumerateFiles koje nudi .Net frejmvork. Na slici 19 prikazan je izgled aplikacije kada su izlistani fajlovi i folderi i kada je odabran jedan od foldera ili fajlova. Zbog ograničenja po broju čvorova koje može da ima treeview kontrola u windows formama, kada želimo hijerarhijskom prikazu da idemo u dubinu foldera, sadržaj foldera se asinhrono dodaje u kontrolu korišćenjem eventa before expand koji se aktivira pre nego što se čvor expanduje. Time se dobija na brzini i manjoj količini utrošene ram memorije jer se ne učitavaju bespotrebni podaci. Jednim klikom na neki od čvorova popunjava se drugi deo koji je namenjen za detaljniji prikaz foldera ili fajla. Takođe duplim klikom na neki čvor se vrši njegovo otvaranje ako je folder ili pokretanje ako je fajl. U drugom delu su prikazani detaljni podaci izabranog foldera ili fajla. Ukoliko je upitanju folder onda su prikazani

njegovo ime, veličina, broj fajlova, datum kreiranja, datum poslednje promene i datum poslednjeg pristupa, a jedina razlika kada je fajl upitanju jeste što je prikazan tip odnosno ekstenzija tog fajla. Takođe kada se izabere folder postoje dodatna dva dugmeta od kojih je jedan namenjen za otvaranje nove forme za prikaz izbrisanih fajlova i foldera, a drugo dugme za statistički prikaz informacija o folderu u nekom period. A ako je izabran fajl onda je vidljivo samo dugme za statistikčki prikaz. Tabelarni prikaz koji se nalazi ispod ovih informacija namenjen je za izlistavanje svih fajlova u okviru foldera koji se izabere i njihov detaljniji prikaz, a ako se izabere fajl onda je ova tabela prazna.



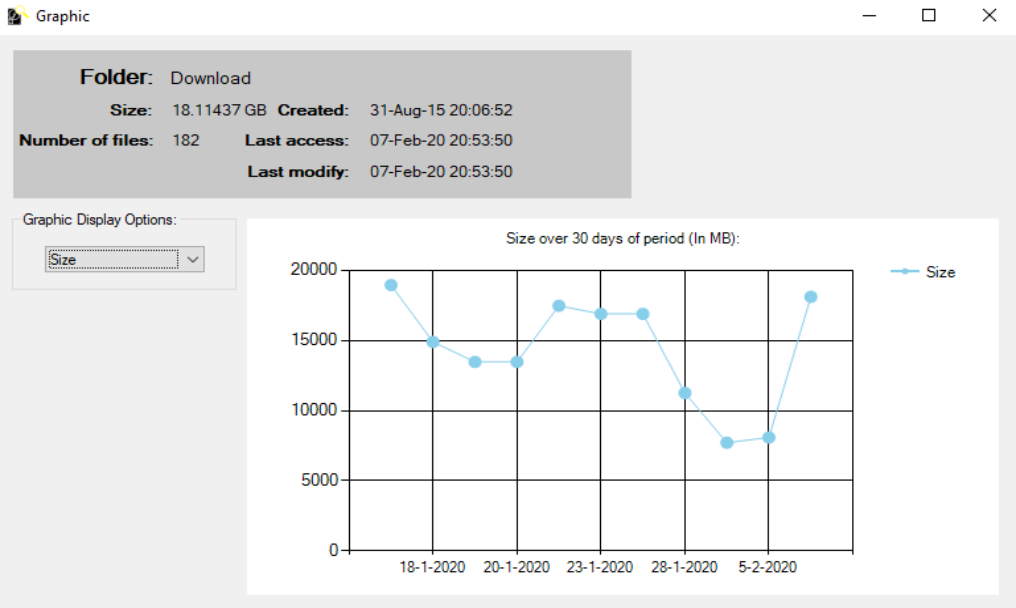
Slika 19. Izgled početne strane nakon skeniranja particije

Takođe na početnoj formi vidljivi su i informacije o količini rama koje koristi aplikacija kao i status SQL servera koji nam je potreban za funkciju keširanja. Takođe vidimo i dva dugmeta Filter i Error Log koji otvaraju nove forme za filtriranje podataka odnosno za prikaz grešaka. Error forma je najprostija forma (slika 22.). Ona sadrži prikaz grešaka koje su se javile prilikom korišćenja aplikacije. Prikazano je tačno vreme nastanka greške kao i detaljan opis greške.



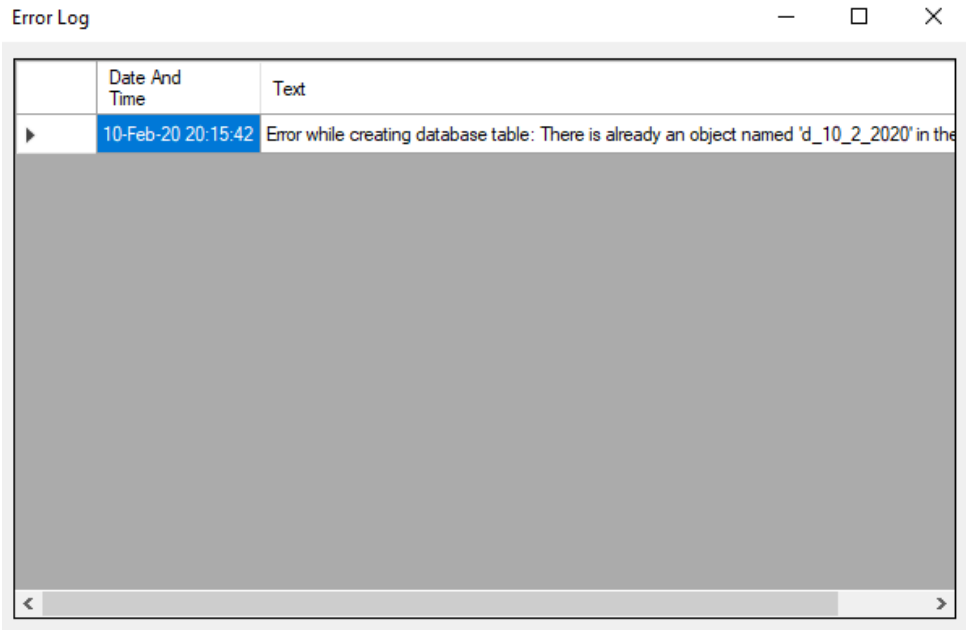
Slika 20. Izgled filter forme

Filter forma je možda i najvažnija forma od svih, jer nam omogućava pretragu i filtriranje podataka po raznim kriterijumima i za određeni period. Opcije koje su moguće za filtriranje su: prikaz svih fajlova i foldera, prikaz izbrisanih fajlova, prikaz po vremenu kreiranja, vremenu modifikacije i vremenu pristupa. Poslednje tri opcije zahtevaju da se odabere period za koji se vrši pretraživanje dok prve dve opcije mogu ali ne zahtevaju odabir perioda i u tom slučaju prikazuju sve ili obrisane fajlove i foldere. Opcije za periode su: za dan, za konkretan period i za period od 30 dana koliko keš traje u bazi. Posle toga se iz date time pickera bira datum za koji želimo da izvršimo pretragu. Kada se izabere opcija i period onda je moguće izabrati i ekstenziju fajla ili izabrati prikaz samo foldera. Ukoliko se ne izabere onda se prikazuju svi fajlovi i folderi. Kada se izvrši pretraga svi pronađeni fajlovi i folderi koji odgovaraju zadatim opcijama su prikazani u tabeli. Dobijene rezutate moguće je pretražiti korišćenjem text box-a iznad tabele. Ispod tabele nalazi se broj rezultata pretrage. Takođe dupli klik na bilo koju vrstu u tabeli otvara novu formu koja prikazuje detaljnije informacije o folderu ili fajlu kao i grafički prikaz promene nekih meta podataka u period od 30 dana.



Slika 21. Izgled forme za statistiku

Forma za statistiku nekog fajla ili foldera prikazuje detaljnije informacije kao i na početnoj formi. Pored toga prikazan je grafikon koji prikazuje promenu fajla ili foldera po nekoliko kriterijuma i to veličini, broju fajlova, vreme modifikacije i vreme pristupa. X osa se odnosi za datum kada je keš napravljen a Y osa se odnosi na vrednost. Svaka tačka na grafikonu sadrži tačnu vrednost. Svi ovi podaci se prikupljaju iz svih tabela koje su dostupne u bazi.



Slika 22. Izgled error forme

Literatura

- **B. Carrier,** File System Forensic Analysis, 2005
- https://en.wikipedia.org/wiki/File_system
- https://www.computerhope.com/jargon/f/filesyst.htm
- https://www.howtogeek.com/196051/htg-explains-what-is-a-file-system-and-why-are-there-so-many-of-them/
- https://www.howtogeek.com/113012/10-best-free-tools-to-analyze-hard-drive-space-on-your-windows-pc/