Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id

E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 3 (2024) | September

OPEN ACCESS

https://doi.org/10.35870/jimik.v5i3.878

Implementasi Sanitasi Data Pada Sistem Operasi Android dengan Metode *Gutmann*

Muhammad Yus Aditya Trisnawan [1], Yos Richard Beeh [2*]

[1,2*] Program Studi Teknik Informatika, Fakultas Teknologi Informasi, Universitas Kristen Satya Wacana, Kota Salatiga, Provinsi Jawa Tengah, Indonesia.

Email: yuzadityatrisnawan@yahoo.co.id [1], yrb.fti.uksw@gmail.com [2*]

Histori Artikel:
Dikirim 25 Mei 2024; *Diterima dalam bentuk revisi* 13 Juni 2024; *Diterima* 20 Juni 2024; *Diterbitkan* 10 September 2024.

Abstrak

Keamanan Data dan Privasi merupakan hal yang sangat penting bagi semua pengguna perangkat mobile khususnya Android. Namun penghapusan data secara umum masih dapat dipulihkan kembali menggunakan analisis forensik pada perangkat keras media penyimpanan dengan perangkat lunak pemulihan data. Untuk itu, perlu aplikasi penghapusan data yang mampu menghapus secara aman. Ada banyak metode yang tersedia untuk penghapusan aman dengan menerapkan jumlah operan (overwriting) yang berbeda-beda. Penelitian ini membuktikan bahwa kebanyakan metode-metode yang tersedia tidak dapat menghapus secara penuh dan dapat dipulihkan sehingga dibuatlah aplikasi penghapusan data dengan metode yang paling mutakhir yakni metode Gutmann. Metode ini menggunakan 35 jumlah operan untuk menimpa file. Aplikasi dibuat dengan terlebih dahulu melakukan analisis kebutuhan sistem lalu diimplementasikan ke dalam pemrograman mobile. Lalu dilakukan uji perbandingan dengan aplikasi lain yang sudah ada. Hasil menunjukkan bahwa aplikasi yang dibuat dengan metode Gutmann lebih kecil 33% kemungkinan ditemukannya jejak file dibanding aplikasi pembanding yang terbaik.

Kata Kunci: Forensik Digital; Keamanan Data; Metode Gutmann; Penghancuran Data.

Abstract

Data security and privacy are essential for all mobile device users, especially Android. However, deleted data can generally still be recovered using forensic analysis of the storage media hardware with data recovery software. For this reason, you need a data deletion application capable of deleting safely. Many methods are available for safe deletion, such as applying different numbers of passes (overwriting). This research proves that most available methods cannot be deleted entirely and can be recovered, so a data deletion application was created using the most up-to-date method, the Gutmann Method. This method uses 35 number of operands to overwrite the file. Applications are made by analyzing system requirements and implementing them into mobile programming. Then, a comparison test is carried out with other existing applications. The results show that applications created using the Gutmann method are 33% less likely to find file traces than the best comparison application.

Keyword: Digital Forensics; Data Security; Gutmann Method; Data Shredder.

Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id
E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 3 (2024) | September
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i3.878

1. Pendahuluan

Smartphone dengan sistem operasi Android telah menjadi perangkat esensial dalam kehidupan sehari-hari, dengan kemampuan yang hampir setara dengan komputer pribadi. Perangkat ini memungkinkan pengguna menyimpan data dan *file* pribadi dalam jumlah besar, termasuk informasi rahasia (Kuswara *et al*., 2022). Masalah muncul ketika perangkat Android berpindah tangan atau berganti pemilik, karena data yang telah dihapus oleh pemilik sebelumnya masih dapat dipulihkan dan diakses oleh pemilik baru melalui analisis forensik (Kuswara *et al*., 2022).

Salah satu metode analisis forensik pada perangkat *mobile* adalah penggunaan perangkat lunak pemulihan data. Pemulihan data dapat dilakukan karena keberadaan *file slack* pada media penyimpanan. *File slack* adalah bagian dari sistem *file* yang tidak benar-benar dihapus, menyebabkan adanya ruang "terbuang" yang dapat diidentifikasi dan ditemukan melalui investigasi forensik (Mulazzani *et al*., 2013). Oleh karena itu, diperlukan aplikasi berbasis Android yang dapat menghapus data secara aman, dikenal sebagai aplikasi penghancuran data. Dalam dunia perangkat lunak, terdapat banyak aplikasi penghancuran data (*file shredder*) untuk media penyimpanan hard drive dengan berbagai metode penghapusan. Namun, tidak semua aplikasi dapat melakukan penghancuran data secara sempurna, sehingga data masih dapat dipulihkan menggunakan perangkat lunak pemulihan. Pengembangan aplikasi yang lebih efisien dan kuat dalam melakukan shredding diperlukan agar informasi yang dihapus tidak dapat dipulihkan, atau setidaknya menjadi sulit untuk dipulihkan secara utuh (Nahar *et al*., 2018).

Metode Gutmann adalah algoritma penimpaan *file* yang dirancang untuk membuat pemulihan data sesulit mungkin. Metode ini menggunakan 35 pass untuk menimpa *file* dengan pola yang berbeda-beda, menggunakan karakter acak untuk beberapa pass pertama dan terakhir, dan pola penimpaan rumit untuk pass tengah (Nahar *et al*., 2018).

Rumusan masalah dalam penelitian ini adalah bagaimana membangun dan mengembangkan aplikasi penghapusan data menggunakan metode Gutmann dengan tujuan memperkecil kemungkinan *file* pada sistem operasi Android dapat dipulihkan atau ditemukan kembali oleh perangkat lunak pemulihan data. Penelitian tentang metode penghapusan data telah menunjukkan pergeseran dari pendekatan konvensional menuju metode yang lebih aman dan kompleks. Nur Iman *et al.* (2017) mengungkapkan bahwa metode penghapusan data konvensional seringkali tidak efektif karena data yang dihapus masih bisa dipulihkan menggunakan perangkat lunak forensik. Temuan ini didukung oleh Marcellino *et al.* (2023) yang melakukan analisis forensik pemulihan digital pada *Smartphone* menggunakan metode National Institute of Justice (NIJ) dan berhasil memulihkan *file* yang dihapus secara konvensional dengan perangkat lunak pemulihan data.

Penelitian oleh Setiawan *et al.* (2018) membandingkan metode DoD dan Gutmann, menemukan bahwa metode Gutmann lebih unggul dalam mencegah pemulihan data. Penelitian ini berfokus pada implementasi skenario tingkat keamanan data dengan metode Gutmann. Dalam penelitian Nahar *et al.* (2018), metode Gutmann diusulkan untuk memastikan pemulihan data dapat dibuat sesulit mungkin bagi penyerang, menawarkan 35 pass dengan pola penimpaan rumit. Penelitian ini bertujuan untuk membangun aplikasi penghapusan data yang lebih aman menggunakan metode Gutmann dan membandingkannya dengan aplikasi penghancuran data lainnya pada sistem operasi Android.

Tabel 1. Metode Penimpaan Gutmann

Jumlah Penimpaan	Pola Bit Penimpaan	
	Notasi Heksadesimal	Notasi Biner
1-4	Acak	Acak
5	55 55 55	010101010101010101010101
6	AA AA AA	101010101010101010101010

Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id
E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 3 (2024) | September
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i3.878

7-9	92 49 24 49 24 92 24 92 49	100100100100100100100100 010010010010010010010010 001001001001001001001001
10	00 00 00	000000000000000000000000
11	11 11 11	000100010001000100010001
12	22 22 22	001000100010001000100010
13-25	33 33 33 – FF FF FF	Pola Bit dari 10 sampai 25 bertambah dari 0x01 (00 00 00 – FF FF FF)
26-28	92 49 24 49 24 92 24 92 49	Sama dengan *pass* 7-9
29-31	6D B6 DB B6 DB 6D DB 6D B6	011011011011011011011011 101101101101101101101101 110110110110110110110110
32-35	Acak	Acak

Penelitian berjudul Analisis Efektivitas Ingest Module Dalam Mendeteksi dan Memulihkan Bukti Digital Yang Disamarkan Terenkripsi membuktikan bahwa metode Ingest Module sangat efektif dalam menemukan jejak digital pada data yang sudah terhapus berupa *file slack*. Sebagian besar *file slack* merupakan sisa penghapusan data yang dilakukan dengan jumlah overwriting yang rendah, sehingga masih menyisakan *file* yang terenkripsi (Permadi *et al.*, 2023). Oleh karena itu, diperlukan metode penghapusan data dengan jumlah overwriting atau penimpaan yang lebih besar untuk memastikan keamanan data yang lebih tinggi. Pernyataan ini didukung oleh penelitian yang berjudul Analisis Perbandingan Keamanan Teknik Penghapusan Data pada Hardisk dengan Metode DoD 5220.22 dan Gutmann. Penelitian ini membandingkan dua metode penghapusan data dan menemukan bahwa metode Gutmann lebih aman dibandingkan metode DoD 5220.22. Hasil dari penelitian ini menekankan pentingnya implementasi aplikasi penghapusan data dengan metode Gutmann untuk meningkatkan keamanan data (Anhar *et al.*, 2014).

Penerapan data *shredder* tidak hanya berlaku pada harddisk komputer, tetapi juga dapat diterapkan pada media penyimpanan yang digunakan pada sistem operasi Android. Hal ini didukung oleh penelitian berjudul "Implementasi Teknik Penghapusan Data Dengan Metode DoD 5220.22M Pada Sistem Operasi Android" oleh Reza Khalifa *et al.* (n.d.). Penelitian ini telah mengembangkan aplikasi Android untuk menghapus data secara aman menggunakan metode DoD 5220.22M. Penelitian ini menunjukkan bahwa variabel tingkat keamanan data pada aplikasi penghapusan dapat dilihat dari kemampuannya dalam menghancurkan data sehingga tidak dapat dipulihkan kembali. Banyaknya *file slack* yang ditemukan selama proses pemulihan data menunjukkan pentingnya metode penghapusan yang lebih kuat seperti metode Gutmann.

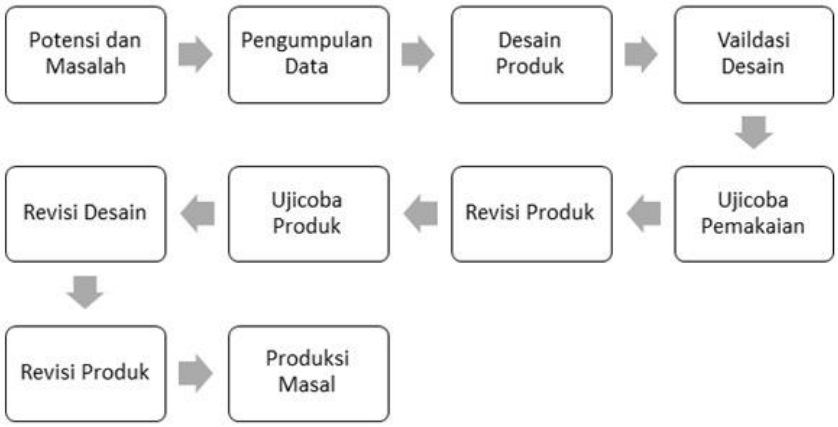
Dalam penelitian tersebut, ditemukan banyak *file* sampah (*file slack*) selama proses pemulihan data. *File slack* adalah ruang di antara akhir sebuah berkas dan akhir dari gugus terakhir yang digunakan oleh berkas tersebut. Ruang ini merupakan area yang tidak akan digunakan lagi untuk menyimpan informasi dan sering terdiri dari teks-teks sampah yang memungkinkan investigasi lebih lanjut (Khalifa *et al.*, n.d.). Oleh karena itu, variabel yang dapat diukur dalam membandingkan metode penghapusan data meliputi; 1) Ada tidaknya nama *file* dan ukuran *file* yang akurat sama seperti saat masih utuh, dan 2) Adanya nama *file* meskipun *file* tersebut tidak dapat dipulihkan lagi (*file slack*).

Berdasarkan penelitian-penelitian yang sudah diuraikan, terdapat kesamaan pada konsep yang dibawa oleh penelitian ini, yaitu analisis forensik, perbandingan metode penghapusan data, penghapusan data pada sistem operasi Android, dan implementasi metode Gutmann. Namun, yang membedakan penelitian ini dengan sebelumnya adalah fokus pada perancangan aplikasi dengan metode Gutmann yang diimplementasikan untuk sistem operasi Android dan membandingkannya dengan aplikasi penghapusan data Android lainnya.

Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id
E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 3 (2024) | September
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i3.878

2. Metode Penelitian

Penelitian ini menggunakan metode *Research and Development* (R&D) yang alurnya dapat dilihat pada Gambar 1 (Sugiyono, 2019). Metode penelitian R&D bertujuan untuk menemukan, mengembangkan, dan memvalidasi suatu produk. Salah satu produk yang dihasilkan dari metode ini bisa berupa perangkat lunak (Haryati, 2012).



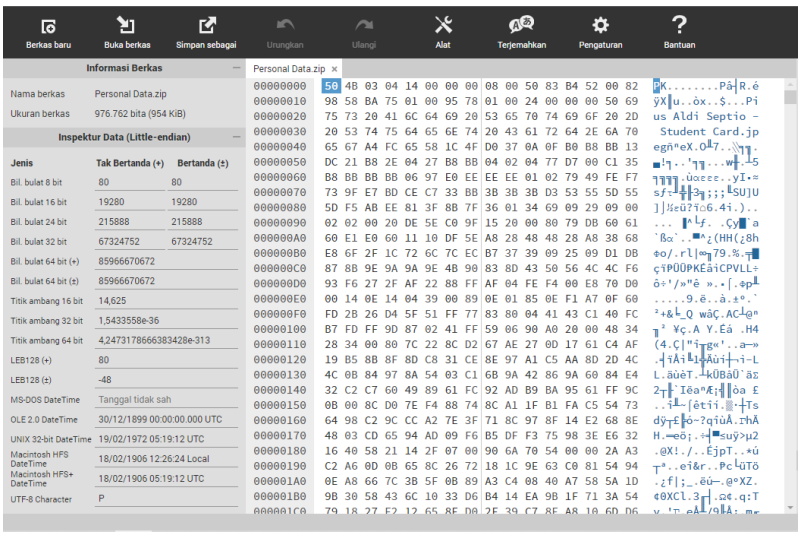
Gambar 1. Metode Penelitian R&D (Sugiyono, 2019)

Prosedur pertama dalam penelitian ini adalah menemukan potensi dan masalah. Potensi dan masalah didapat dari studi literatur dan percobaan yang membuktikan bahwa penghapusan biasa maupun penghapusan aman dengan metode lain pada sistem operasi Android masih bisa dipulihkan dengan perangkat lunak pemulihan. Pada tahap pengumpulan data, didapatkan variabel-variabel dari hasil percobaan penghapusan dan pemulihan untuk membandingkan berapa besar kemungkinan *file-file* tersebut masih bisa dipulihkan. Setelah mengetahui potensi dan masalah serta dilakukannya pengumpulan data, maka dilakukan desain produk yang berupa aplikasi yaitu pembuatan UML berupa *use case* dan *class diagram* untuk perancangan aplikasi.

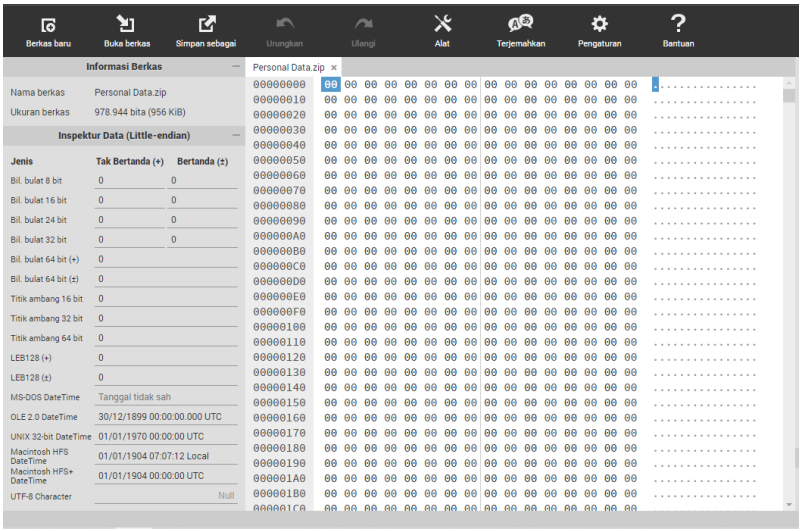
Pada validasi desain, dilakukan implementasi UML pada pembuatan aplikasi yang berbasis Android. Aplikasi yang dibuat yaitu aplikasi untuk menghapus data yang diharapkan dapat menutupi kekurangan dari aplikasi yang sudah diambil sebagai contoh dalam hal keamanan data. Setelah pembuatan aplikasi selesai, maka dilakukan uji coba pemakaian apakah aplikasi sudah mampu melakukan penghapusan aman, jika belum akan dilakukan revisi dalam pembuatan.

Uji Coba pemakaian yakni dengan melakukan uji *overwriting* sebuah *file* dengan byte random, byte 1 dan kemudian byte 0 yang membuktikan bahwa aplikasi yang dibuat sudah melakukan fungsi penghapusan dengan benar. Pada gambar 2, terlihat *hex* sebuah *file* sebelum dilakukan uji coba penghapusan lalu pada gambar 3, terlihat perbedaan *hex* setelah dilakukan uji coba penghapusan. Pengecekan *hex file* ini menggunakan fasilitas hex editor daring yang dapat diakses di https://hexed.it.

Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id
E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 3 (2024) | September
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i3.878



Gambar 2. Hex *File* Sebelum Uji Coba Penghapusan



Gambar 3. Hex *File* Setelah Uji Coba Penghapusan

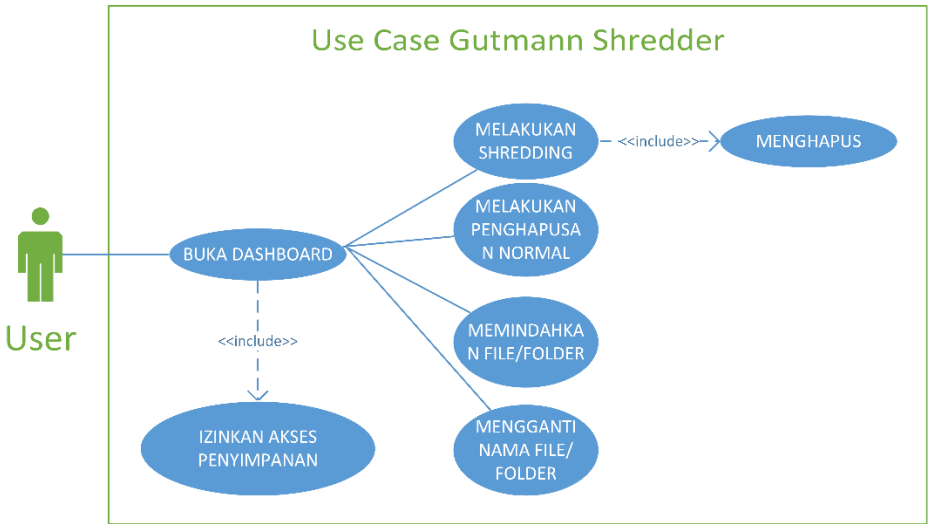
Pada uji coba aplikasi, akan dilakukan serangkaian pengujian keamanan data untuk mengetahui hasil dari aplikasi yang sudah dibuat dari segi keamanannya, lalu dibandingkan dengan aplikasi yang telah dijadikan contoh sehingga dapat diketahui kekurangannya atau ditemukannya *bug*. Apabila ditemukan kekurangan saat dibandingkan atau terdapat *bug* semisal terjadi *force close* atau *error* saat menghapus *file* akan dilakukan tahap revisi desain dan produk yakni perbaikan pada aplikasi. Jika tidak, maka aplikasi sudah siap digunakan dan akan dirilis ke publik.

3. Hasil dan Pembahasan

3.1 Analisis Kebutuhan

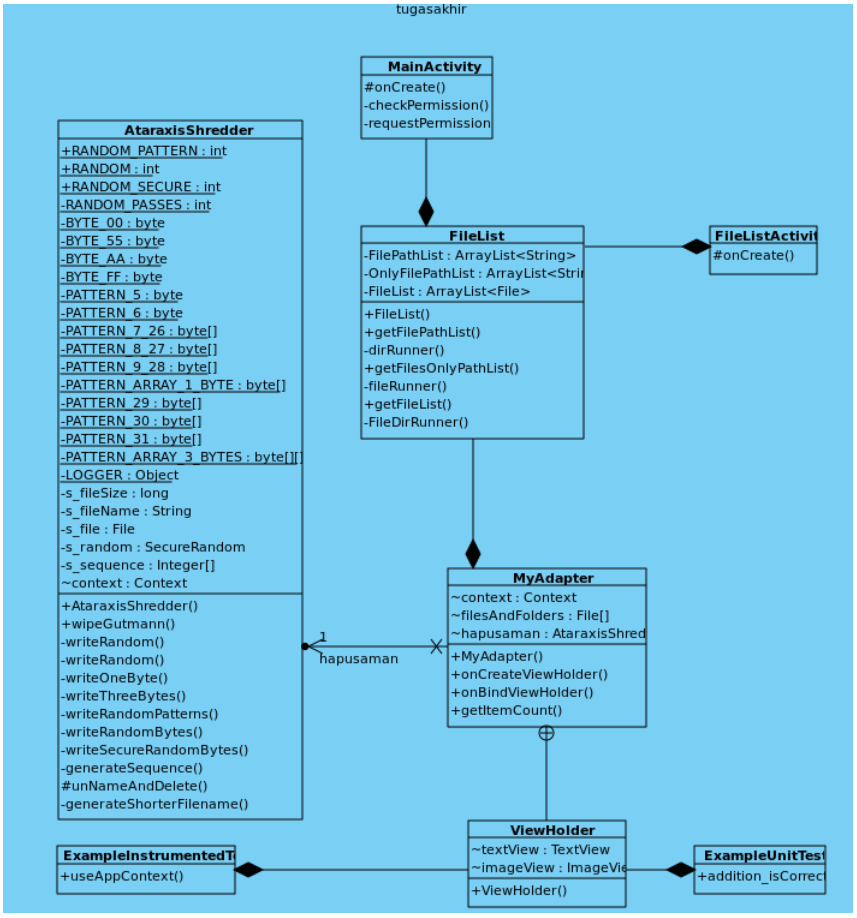
Use case diagram adalah gambaran interaksi antara pengguna sistem dan sistem yang sedang digunakan. Pihak yang terlibat dalam sistem ini adalah seorang *user* yang akan menjalankan aplikasi Gutmann *Shredder* (Kurniawan, 2018). Gambar 4 merupakan *use case* yang digunakan dalam perancangan aplikasi *Shredder* yang dibuat.

Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id
E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 3 (2024) | September
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i3.878



Gambar 4. *Use Case* Gutmann *Shredder*

Class Diagram dibuat untuk menggambarkan *class* dan fungsi pada sistem yang dibuat dan dapat membantu sebagai alat perencanaan untuk pengembangan suatu sistem. Gambar 5 merupakan *Class Diagram* dari aplikasi Gutmann *Shredder*.

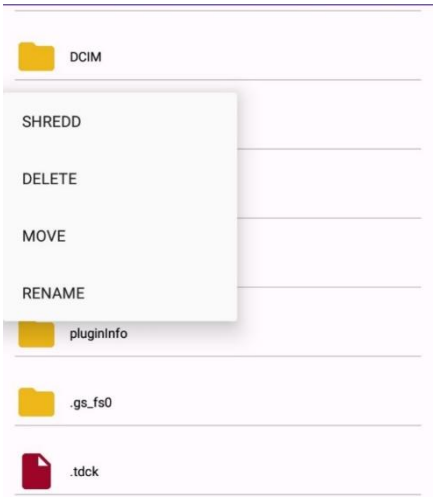


Gambar 5. *Class Diagram* Gutmann *Shredder*

Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id
E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 3 (2024) | September
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i3.878

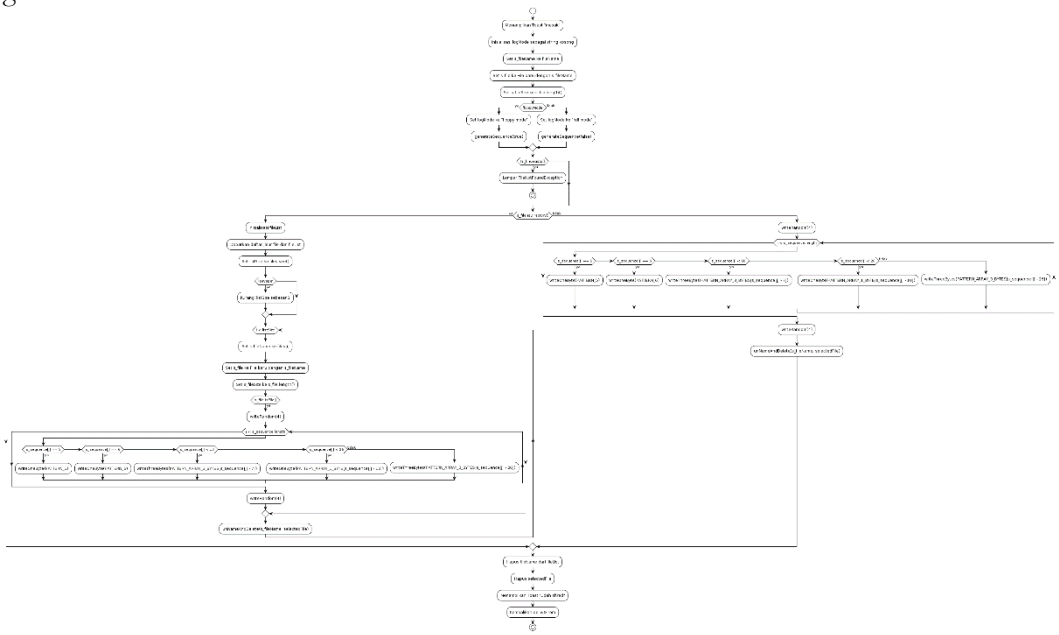
3.2 Implementasi Sistem

File List berfungsi untuk menampilkan isi *storage* berupa daftar folder dan *file*. *File List* menampilkan fitur "SHREDD" untuk melakukan penghancuran data seperti terlihat pada gambar 6 berikut.



Gambar 6. *File List* dan Program Utama

Fungsi Gutmann *Shredder* dijelaskan dalam diagram alurnya pada gambar 7 berikut ini dan kode program 1.



Gambar 7. *Flowchart* Gutmann *Shredder*

Kode Program 1. Kode Program Gutmann *Shredder*

```
public final boolean wipeGutmann(String fileName, File selectedFile,boolean
leaveDir, boolean floppyMode ,Context context)
            throws IOException {
            Toast.makeText(context.getApplicationContext(),          "masuk",
Toast.LENGTH_SHORT).show();
            String logMode = "";
            s_fileName = fileName;
            s_file = new File(s_fileName);
```

Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id
E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 3 (2024) | September
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i3.878

```
              s_fileSize = s_file.length();

              if (floppyMode) {
                     logMode = "floppy mode";
                     generateSequence(true);
              } else {
                     generateSequence(false);
                     logMode = "full mode";
              }


              if (!s_file.exists())
                     throw new FileNotFoundException("File to delete could not
been found: " + s_fileName);

              FileList fileList = null;
              if (s_file.isDirectory()) {
                     fileList = new FileList();
                     final            List<String>            files            =
fileList.getFilePathList(fileName);
                     int listSize = files.size();
                     if (leaveDir)
                            listSize--;
                     for (int i = 0; i < listSize; i++) {
                            s_fileName = files.get(i);
                            s_file = null;
                            s_file = new File(s_fileName);
                            s_fileSize = s_file.length();
                            if (s_file.isFile()) {
                                   writeRandom(4);
                                   for (int j = 0; j < s_sequence.length; j++) {
                                          if (s_sequence[j] == 5) {
       writeOneByte(PATTERN_5);
                                          } else if (s_sequence[j] == 6) {
       writeOneByte(PATTERN_6);
                                          } else if (s_sequence[j] < 10) {
       writeThreeBytes(PATTERN_ARRAY_3_BYTES[s_sequence[j] - 7]);
                                          } else if (s_sequence[j] < 26) {
       writeOneByte(PATTERN_ARRAY_1_BYTE[s_sequence[j] - 10]);
                                          } else {
       writeThreeBytes(PATTERN_ARRAY_3_BYTES[s_sequence[j] - 26]);
                                          }
                                   }
                                   writeRandom(4);
                            }
                            unNameAndDelete(s_fileName, selectedFile);
                     }

              } else
              {
                     writeRandom(4);
                     for (int j = 0; j < s_sequence.length; j++) {
                            if (s_sequence[j] == 5)
                                   writeOneByte(PATTERN_5);
                            else if (s_sequence[j] == 6)
                                   writeOneByte(PATTERN_6);
                            else if (s_sequence[j] < 10)
       writeThreeBytes(PATTERN_ARRAY_3_BYTES[s_sequence[j] - 7]);
                            else if (s_sequence[j] < 26)
       writeOneByte(PATTERN_ARRAY_1_BYTE[s_sequence[j] - 10]);
                            else
       writeThreeBytes(PATTERN_ARRAY_3_BYTES[s_sequence[j] - 26]);
```

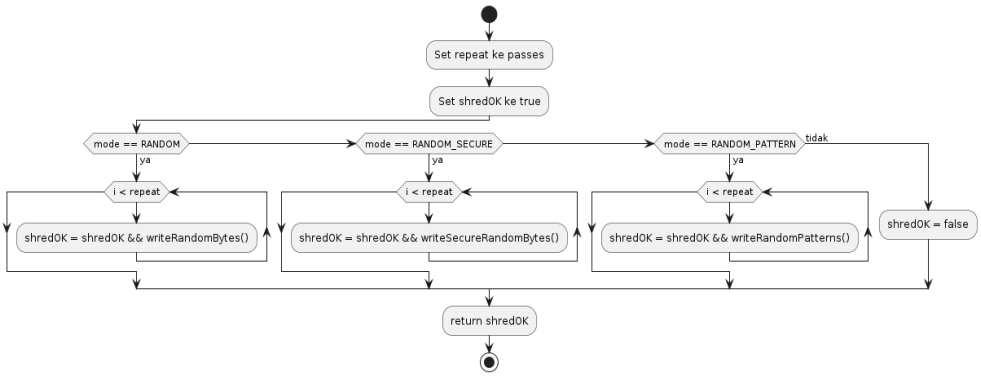
Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id
E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 3 (2024) | September
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i3.878

```
                        }
                        writeRandom(4);
                        unNameAndDelete(s_fileName, selectedFile);
                }
                boolean rem = fileList.getFilePathList(fileName).remove(fileName);
                boolean del = selectedFile.delete();

                Toast.makeText(context.getApplicationContext(),    "udah    shred",
Toast.LENGTH_SHORT).show();
                return del&&rem;
        }
```

Method ini pertama-tama melakukan inisialisasi berbagai variabel yang diperlukan pada baris 5-7, seperti nama *file*, ukuran *file*, dan pada baris 9-15 adalah penentuan mode (mode penuh atau mode floppy). Kemudian pada baris 18-20, akan memeriksa apakah *file* yang dimaksud ada. Pada baris 21-51, jika yang dimaksud adalah sebuah direktori, maka akan menghapus semua *file* di dalamnya menggunakan algoritma Gutmann. Jika *file* tidak ditemukan, maka akan dilemparkan '*File*NotFoundException' pada baris 53-70. Setelah itu, direktori tersebut akan dihapus. Jika yang dimaksud adalah sebuah *file* tunggal, maka *file* tersebut akan dihapus langsung menggunakan algoritma Gutmann.

Algoritma Gutmann yang digunakan dalam *method* ini menggunakan *method* pembantu yakni *method* 'writeOneByte', 'writeThreeBytes', 'writeRandomBytes', dan 'writeRandomPatterns' yang menggunakan pola untuk Algoritma Gutmann yakni 'PATTERN 5', 'PATTERN 6', 'PATTERN ARRAY 1 BYTE', dan 'PATTERN ARRAY 3 BYTES' yang masing-masing berisi proses *rewriting* pada kode biner yang berbeda-beda sesuai peruntukannya. Sedangkan pada method 'writeRandomBytes' dan 'writeRandomPatterns' digunakan untuk menuliskan data acak yang dipanggil melalui method 'writeRandom'. Method 'writeRandom' dijelaskan pada diagram alir gambar 8 dan kode program 2 berikut.



Gambar 8. *Flowchart* fungsi writeRandom

Kode Program 2. Kode Program writeRandom

```
private final boolean writeRandom(int mode, int passes) throws IOException
        {
                final int repeat = passes;
                boolean shredOK = true;

                if (mode == RANDOM)
                {
                        for (int i = 0; i < repeat; i++)
                        {
                                shredOK = shredOK && writeRandomBytes();
                        }
                }
                else if (mode == RANDOM_SECURE)
```

Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id
E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 3 (2024) | September
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i3.878

```
            {
                for (int i = 0; i < repeat; i++)
                {
                    shredOK = shredOK && writeSecureRandomBytes();
                }
            }
            else if (mode == RANDOM_PATTERN)
            {
                for (int i = 0; i < repeat; i++)
                {
                    shredOK = shredOK && writeRandomPatterns();
                }
            }
            else
            {
                shredOK = false;
            }
            return shredOK;
        }
```

Pada *method* 'writeRandom (int passes)', adalah *method overload* di mana ia memanggil *method* 'writeRandom(int mode, int passes)' dengan mode 'RANDOM'. *Method* ini digunakan untuk menulis data secara acak ke *file*. Baris 6-26 berfungsi untuk mengulangi operasi penghapusan sejumlah kali sesuai dengan nilai 'passes' yang diberikan dengan menggunakan metode 'writeRandomBytes' dan metode 'writeRandomPatterns' untuk menulis data acak ke *file*. Ini bergantung pada nilai kembalian dari kedua metode tersebut untuk menentukan apakah operasi penghapusan berhasil. Pada pengujian aplikasi sampel, akan diambil 25 sampel *file* untuk dihancurkan (*shredder*) menggunakan 3 *tools* sampel yang didapatkan dari Google Play Store dimana *tools* tersebut memiliki metode *shredder* yang berbeda-beda. Pemilihan aplikasi berdasarkan rating, kepopuleran, dan banyaknya unduhan di tiap aplikasi tersebut. Aplikasi penghapusan data yang terpilih yaitu:

1) Secure Wipe Out
2) Secure Wipe Out memiliki rating 5,0 dan 10 ribu lebih unduhan. Aplikasi ini mengklaim memiliki metode NATO Standard (7 Pass) dalam melakukan penghapusan.
3) Shreddit – Data Eraser
4) Shreddit memiliki rating 4,4 dan 1 juta lebih unduhan. Aplikasi ini menggunakan metode DoD 5220.22 M (3 pass) dalam melakukan penghapusan.
5) i*Shredder* – Storage Cleaner
6) i*Shredder* memiliki rating 4,2 dan 1 juta lebih unduhan. Aplikasi ini menggunakan metode BSI TL-03423 (8 pass) dalam melakukan penghapusan.

Untuk pengujian *recovery*, telah dipilih perangkat lunak yang digunakan untuk memulihkan data berdasarkan kepopuleran pengguna, yaitu:

1) Recuva
   Recuva adalah perangkat lunak gratis yang juga memiliki fitur Pro milik Piriform Software Ltd. Recuva bisa didapatkan dari situs resmi Piriform.
2) EaseUS Data Recovery Wizard
   EaseUS Data Recovery Wizard adalah perangkat lunak gratis yang juga memiliki fitur peningkatan untuk menikmati fitur lebih lengkap. Perangkat lunak ini bisa didapatkan dari situs resmi pemiliknya yaitu EaseUS.
3) Stellar Data Recovery
   Stellar Data Recovery adalah perangkat lunak pemulihan data dengan masa percobaan gratis yang dimiliki oleh Wondershare dan bisa didapatkan dari situs resminya.

Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id
E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 3 (2024) | September
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i3.878

Dari 25 sampel *file*, akan diuji untuk dihancurkan oleh tiga aplikasi *file shredder* yang sudah dipilih lalu dipulihkan oleh tiga perangkat lunak pemulihan data tersebut. Setelah dilakukan uji performansi oleh tiga aplikasi *file shredder*, dapat diketahui masing-masing kelemahan aplikasi tersebut. Tahap pengujian untuk penghapusan adalah dengan mengkategorikan *file* berdasarkan *cluster* karena tiap *cluster* diletakkan pada media penyimpanan yang berbeda untuk dihapus *file* secara satu-persatu. *Cluster* dikelompokkan berdasarkan ukuran dan tipe *file* agar nanti dapat disimpulkan faktor apa yang memengaruhi dapat atau tidaknya *file* bisa dipulihkan. Tipe-tipe *file* yang dipilih adalah tipe yang paling umum diketahui dan digunakan oleh pengguna Android, yakni audio, dokumen, gambar, dan video dengan lima ukuran yang berbeda. Ukuran *file* disesuaikan dengan ukuran jenis *file* pada umumnya sehingga ada total 25 *file* yang akan diuji seperti yang dijelaskan melalui tabel 2 berikut:

Tabel 2. Tipe dan Ukuran *File* Untuk Uji Coba Pemakaian

No.	Tipe	Ekstensi	Ukuran				
			1	2	3	4	5
1.	Audio	.wav	500 KB	1 MB	5 MB	10 MB	25 MB
2.	Dokumen	.txt	1 KB	5 KB	10 KB	100 KB	1 MB
		.docx	500 KB	1 MB	5 MB	10 MB	25 MB
3.	Gambar	.jpg	100 KB	500 KB	1 MB	5 MB	10 MB
4.	Video	.mp4	10 MB	50 MB	100 MB	500 MB	1 GB

Dari 25 *file* yang akan diuji akan dibagi menjadi 10 cluster, yakni cluster 1-5 diuji penghapusan berdasarkan ekstensi, yakni Audio, Dokumen, Gambar, dan Video. Lalu cluster 6-10 diuji penghapusan berdasarkan ukuran, yakni yakni sesuai kolom ukuran 1, 2, 3, 4, dan 5. Hasil uji coba pemakaian akan dijelaskan melalui simbol berikut:

√ = *File* tidak dapat dipulihkan dan ditemukan
□ = *File* tidak dapat dipulihkan namun dapat ditemukan (*file slack*)
X = *File* dapat ditemukan dan dipulihkan
! = Aplikasi *error* atau *force closed*

Setelah dilakukan pengujian aplikasi, didapatkan kesimpulan keamanan data bahwa ketiga aplikasi sampel masih belum mampu menghapus aman secara total karena masih terdapat beberapa *file* yang bisa dipulihkan baik secara total maupun sebagian dengan penjelasan seperti pada tabel 3-5 berikut ini.

Tabel 3. Data Hasil Uji Keamanan Data Aplikasi Secure Wipe Out

No.	Tipe	Ekstensi	Ukuran					Hasil Cluster Ekstensi
			1	2	3	4	5	
1.	Audio	.wav	500 KB	1 MB	5 MB	10 MB	25 MB	□
2.	Dokumen	.txt	1 KB	5 KB	10 KB	100 KB	1 MB	□
		.docx	500 KB	1 MB	5 MB	10 MB	25 MB	□
3.	Gambar	.jpg	100 KB	500 KB	1 MB	5 MB	10 MB	□
4.	Video	.mp4	10 MB	50 MB	100 MB	300 MB	500 MB	□
Hasil Cluster Ukuran			□	□	□	□	□	

Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id
E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 3 (2024) | September
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i3.878

Tabel 4. Data Hasil Uji Keamanan Data Aplikasi Shreddit – Data Eraser

No.	Tipe	Ekstensi	Ukuran					Hasil Cluster Ekstensi
			1	2	3	4	5	
1.	Audio	.wav	500 KB	1 MB	5 MB	10 MB	25 MB	√
2.	Dokumen	.txt	1 KB	5 KB	10 KB	100 KB	1 MB	√
		.docx	500 KB	1 MB	5 MB	10 MB	25 MB	□
3.	Gambar	.jpg	100 KB	500 KB	1 MB	5 MB	10 MB	√
4.	Video	.mp4	10 MB	50 MB	100 MB	300 MB	500 MB	□
Hasil Cluster Ukuran			√	√	□	□	□	

Tabel 5. Data Hasil Uji Keamanan Data Aplikasi i*Shredder* – Storage Cleaner

No.	Tipe	Ekstensi	Ukuran					Hasil Cluster Ekstensi
			1	2	3	4	5	
1.	Audio	.wav	500 KB	1 MB	5 MB	10 MB	25 MB	√
2.	Dokumen	.txt	1 KB	5 KB	10 KB	100 KB	1 MB	√
		.docx	500 KB	1 MB	5 MB	10 MB	25 MB	√
3.	Gambar	.jpg	100 KB	500 KB	1 MB	5 MB	10 MB	√
4.	Video	.mp4	10 MB	50 MB	100 MB	300 MB	500 MB	□
Hasil Cluster Ukuran			√	√	□	□	□	

Terlihat hasil pengujian pada aplikasi Secure Wipe Out, semua data masih bisa ditemukan, namun tidak ada yang bisa dipulihkan. Lalu pada aplikasi Shreddit – Data Eraser, penghapusan *cluster* 1-2 dan 6-8 semua data masih bisa ditemukan dan tidak bisa dipulihkan, sisanya masih dapat dipulihkan secara total saat menjalankan penghapusan *cluster* 3-5 dan 9-10. Pada aplikasi i*Shredder* – Storage Cleaner terlihat bahwa *cluster* 1-4 dan 6-7 aplikasi sudah mampu melakukan penghapusan dengan aman karena tidak dapat dipulihkan maupun ditemukan, namun pada *cluster* 8, *file* masih bisa ditemukan, dan *cluster* 9-10 aplikasi *crash* atau *hang*. Setelah dilakukan serangkaian pengujian penghapusan berdasarkan 9 cluster yang masing-masing dipulihkan menggunakan tiga buah perangkat lunak yang telah disebutkan, didapatkan hasilnya sebagai berikut:

Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id
E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 3 (2024) | September
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i3.878

Tabel 6. Hasil Uji Coba Pemakaian

No.	Tipe	Ekstensi	Ukuran					Hasil Cluster Ekstensi
			1	2	3	4	5	
1.	Audio	.wav	500 KB	1 MB	5 MB	10 MB	25 MB	√
2.	Dokumen	.txt	1 KB	5 KB	10 KB	100 KB	1 MB	√
		.docx	500 KB	1 MB	5 MB	10 MB	25 MB	√
3.	Gambar	.jpg	100 KB	500 KB	1 MB	5 MB	10 MB	√
4.	Video	.mp4	10 MB	50 MB	100 MB	300 MB	500 MB	□
Hasil Cluster Ukuran			√	√	√	□	□	

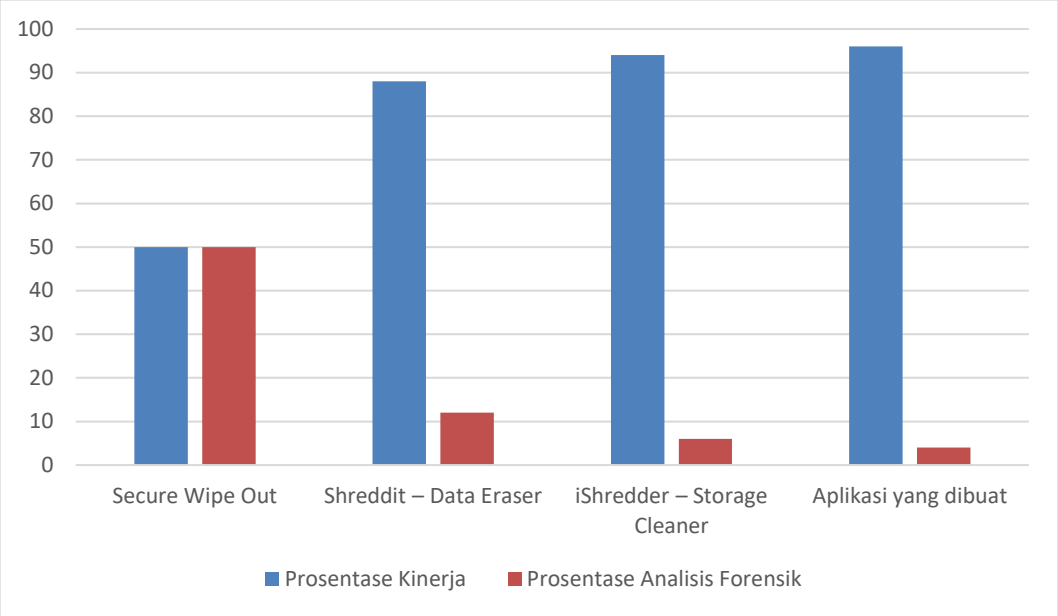
Pada tabel 6, didapatkan hasil bahwa penghapusan aman dengan aplikasi yang dibuat, masih bisa ditemukan nama dan ekstensi *file* saat dilakukan pemulihan namun tidak dapat dikembalikan (*file slack*) yakni pada *cluster* 5, 9, dan 10.

Tabel 7. Statistik Perbandingan Keamanan Data

No.	Nama Aplikasi	*File* bisa dipulihkan	*File* bisa ditemukan	Prosentase analisis forensik	Prosentase kinerja aplikasi	Peringkat
1.	Secure Wipe Out	0	25	50%	50%	4
2.	Shreddit – Data Eraser	0	6	12%	88%	3
3.	i*Shredder* – Storage Cleaner	0	3	6%	94%	2
4.	Aplikasi yang dibuat	0	2	4%	96%	1

Dari tabel 7 menunjukkan statistik yang membandingkan tingkat keamanan data aplikasi sampel dengan aplikasi yang dibuat. Prosentase analisis dihitung dari gabungan jumlah *file* yang bisa dipulihkan dan ditemukan dari total 25 *file* yang diuji maka apabila 25 *file* bisa dipulihkan, akan terdapat 25 *file* juga yang masuk kolom *file* ditemukan sehingga menjadi 100% keberhasilan. Sedangkan prosentase kinerja aplikasi adalah kemampuan aplikasi dalam menghapus *file* secara aman yang dihitung dari 100% dikurangi prosentase analisis forensik. Terlihat bahwa aplikasi Secure Wipe Out memiliki tingkat keamanan terendah karena ada 25 *file* yang bisa ditemukan (*file slack*), diikuti oleh aplikasi Shreddit – Data Eraser yang terdapat 6 *file slack*. Sedangkan aplikasi yang dibuat memiliki tingkat keamanan data tertinggi yakni hanya 2 *file* yang dapat ditemukan. Tidak ada satupun *file* yang bisa dipulihkan baik dari 3 aplikasi sampel maupun aplikasi yang dibuat. Berikut hasil statistik yang dibuat dalam bentuk grafik.

Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id
E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 3 (2024) | September
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i3.878



Gambar 9. Grafik Perbandingan Kinerja Aplikasi yang Dibuat dengan Aplikasi Sampel

4. Kesimpulan

Berdasarkan hasil penelitian, dapat disimpulkan bahwa membangun dan mengembangkan aplikasi penghapusan data dengan metode Gutmann dilakukan dengan cara menganalisis kebutuhan sistem terlebih dahulu berupa UML Diagram lalu diimplementasikan dengan merancang *file manager* biasa untuk mengelola *file* Android kemudian ditambahkan fitur *Shredding*. Fitur *Shredding* inilah yang memiliki fungsi program untuk melakukan penghapusan aman. Aplikasi yang dibuat juga terbukti lebih aman dalam menghapus *file* daripada aplikasi-aplikasi berbasis Android yang lain karena dalam pengujian aplikasi sampel, *file* yang dihapus masih bisa ditemukan dan dapat dipulihkan lebih banyak daripada pengujian penghapusan yang dilakukan pada aplikasi yang dibuat sehingga sesuai dengan tujuan penelitian yakni memperkecil kemungkinan *file* pada sistem operasi Android dipulihkan atau ditemukan oleh perangkat lunak pemulihan data. Penelitian lanjutan sangat direkomendasikan mengenai penghapusan data dengan metode Gutmann ini terutama implementasinya terhadap versi Android yang lebih baru dikarenakan Android versi lebih dari 10 hanya mendukung penggunaan Mediastore API dalam mengizinkan aplikasi pihak ketiga dalam mengelola dan memodifikasi isi *file* perangkat *mobile*.

5. Ucapan Terima Kasih

Ucapan terima kasih pertama-tama ditujukan kepada Ibu penulis sebagai orang tua tunggal yang telah membantu tidak hanya berupa biaya untuk menyelesaikan penelitian ini namun juga senantiasa mendampingi hingga penelitian ini selesai. Terima kasih juga ditujukan kepada yayasan beasiswa Van Deventer Maas Indonesia sebagai pihak yang sudah memberikan dana bantuan selama proses perkuliahan sehingga penelitian ini bisa terwujud. Tidak lupa pula teman-teman dan semua pihak yang sudah menjadi *support system* sehingga penulis bisa bersemangat saat membuat penelitian ini

E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 3 (2024) | September
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i3.878

6. Daftar Pustaka

Al Anhar, A., Satrya, G. B., & Yulianto, F. A. (2014). Analisis perbandingan keamanan teknik penghapusan data pada hardisk dengan metode DoD 5220.22 dan Gutmann. *eProceedings of Engineering*, 1(1), 607–613. Retrieved from http://libraryeproceeding.telkomuniversity.ac.id/index.php/engineering/article/view/2830/4521

Haryati, S. (2012). Research and development (R&D) sebagai salah satu model penelitian dalam pendidikan. *Majalah Ilmiah Dinamika*, 37(1), 13.

Iman, N., Susanto, A., & Inggi, R. (2020). Analisa perkembangan digital forensik dalam penyelidikan cybercrime di Indonesia (Systematic Review). *Jurnal Telekomunikasi dan Komputer*, 9(3), 186. https://doi.org/10.22441/incomtech.v9i3.7210

Khalifa, H. R., Yulianto, F. A., & MJadied, E. (n.d.). Implementasi teknik penghapusan data dengan metode DoD 5220.22M pada sistem operasi Android.

Kurniawan, T. A. (2018). Pemodelan use case (UML): Evaluasi terhadap beberapa kesalahan dalam praktik. *Jurnal Teknologi Informasi dan Ilmu Komputer*, 5(1), 77–86. https://doi.org/10.25126/jtiik.201851610.

Kuswara, B. I. H., Pratama, A. R., & Ramadhani, E. (2022). Studi komparasi metode disk overwrite dan factory reset sebagai teknik anti forensik di perangkat Android. *JATISI (Jurnal Teknik Informatika dan Sistem Informasi)*, 9(2), 1151–1161. https://doi.org/10.35957/jatisi.v9i2.1955

Marcellino, S., Seta, H. B., & Widi, I. W. (2023). Analisis forensik digital recovery data smartphone pada kasus penghapusan berkas menggunakan metode National Institute of Justice (NIJ). *Informatik: Jurnal Ilmu Komputer*, 19(2), 141–156. https://doi.org/10.52958/iftk.v19i2.4676

Mulazzani, M., Neuner, S., Kieseberg, P., Huber, M., Schrittwieser, S., & Weippl, E. (2013). Quantifying Windows file slack size and stability. *IFIP Advances in Information and Communication Technology*, 410, 183–193. https://doi.org/10.1007/978-3-642-41148-9_13

Nahar, N. F. A. M., Rahman, N. H. A., & Mohammad, K. M. (2018). E-raser: File shredder application with content replacement by using random words function. *International Journal on Informatics Visualization*, 2(4-2), 313–317. https://doi.org/10.30630/joiv.2.4-2.175

Permadi, A. B., Rizki, F., Sutiyo, A., & Saputra, M. R. E. (2023). Analisis efektivitas ingest module dalam mendeteksi dan memulihkan bukti digital yang disamarkan terenkripsi. *Jurnal Forensik Digital*, 1(1), 81–87.

Setiawan, N. A. (2017). Analisis perbandingan penghapusan data digital dengan menerapkan metode DoD 5220.22M dan metode Gutmann. Retrieved from http://repository.unpas.ac.id/id/eprint/31437

Sugiyono, P. D. (2019). *Metode penelitian dan pengembangan (Research and Development/R&D)* (4th ed.). Bandung: Alfabeta.