Pendeteksian Malware Dengan Pendekatan Grayscale Image

Ivan Limosi–6181801032

1 Data Skripsi

Pembimbing utama/tunggal: **Chandra Wijaya, S.T., M.T.**
Pembimbing pendamping: **-**
Kode Topik : **CHW5201**
Topik ini sudah dikerjakan selama : **1** semester
Pengambilan pertama kali topik ini pada : Semester **52 - Genap 21/22**
Pengambilan pertama kali topik ini di kuliah : **Skripsi 1**
Tipe Laporan : **B** - Dokumen untuk reviewer pada presentasi dan **review Skripsi 1**

2 Latar Belakang

Pengguna komputer di Indonesia maupun di dunia sekarang sudah sangat banyak, dikarenakan ada nya pandemi Covid-19 saat ini hampir seluruh pekerjaan membutuhkan sebuah komputer atau laptop seperti guru, dosen, mahasiswa, pekerja kantoran dan masih banyak lagi. Perangkat komputer yang digunakan oleh masyarakat sekarang pasti semua memiliki yang namanya Sistem Operasi atau biasa disingkat OS(*Operating System*). Ada beberapa contoh OS yang cukup terkenal seperti *Windows* dan Linux atau Ubuntu. Sistem operasi yang saat ini paling sering digunakan oleh masyarakat adalah *Windows*. Pada sistem operasi *Windows* banyak sekali aplikasi dan *tools* yang dapat digunakan untuk berbagai macam kegiatan yang ada di komputer tersebut. Tapi tidak semua *file* yang dimiliki oleh komputer itu merupakan *file* yang tidak merusak atau *file* yang baik. Ada beberapa jenis *file* yang dapat mengganggu atau bahkan merusak komputer kita sendiri. Jenis *file* ini biasa lebih lazim disebut sebagai *Malware*.

Malicious Software atau yang biasa disebut *Malware* adalah sebuah perangkat lunak yang jika aktif akan merusak kinerja perangkat keras atau komputer kita, mengambil data-data penting yang ada di komputer kita, mendapatkan akses ke komputer kita, dapat menampilkan iklan-iklan yang mengganggu di komputer kita, dapat memberatkan kinerja prosesor walaupun tidak banyak aplikasi yang terbuka dan yang paling umum terjadi adalah adanya degradasi performa pada komputer kita. Pada umumnya *Malware* ini akan muncul dengan beberapa bentuk berkas-berkas yang memiliki jenis seperti *executable*, *Scripts* dan konten yang aktif seperti *Flash* atau bisa juga berasal dari *Flashdisk* yang mengandung *file malware* yang kita masukkan ke dalam komputer kita.

Malware memiliki banyak jenis, berikut adalah beberapa contoh dari jenis-jenis *malware* yang sudah diketahui yaitu Virus, Worm, Trojan Horse, Spyware, Scareware, Adware, Botnet, dan Ransomware. Setiap jenis dari *malware* tersebut memiliki karakteristik dan cara merusak komputer pengguna yang berbeda-beda, misalnya pada umumnya *malware* bertipe virus akan merusak komputer dengan cara memasukkan kode dirinya kedalam program yang ada didalam komputer tersebut dan akan melakukan hal yang sama jika ada seseorang yang membuka program tersebut. Pada pengerjaan skripsi ini akan digunakan repository *malware* yang bernama theZoo. TheZoo adalah sebuah proyek yang ditujukan untuk membuat kesempatan untuk pengguna agar dapat membuat analisis tentang *malware* yang masih aktif dan dapat diakses oleh publik. Dikarenakan pengumpulan data *malware* dapat dibilang susah, theZoo menyediakan hampir seluruh versi dari *malware* jenis apapun yang masih ada dan sudah dikumpulkan dalam sebuah repository github agar dapat diakses dengan aman.

Pada saat ini ada beberapa cara untuk mendeteksi keberadaan *file* tersebut di komputer anda, salah satunya adalah dengan menggunakan pendekatan *Grayscale Image.Grayscale Image* adalah sebuah pendekatan yang digunakan untuk memvisualisasikan sebuah *file malware* yang nanti akan diklasifikasikan. Setiap *byte* dari sebuah *file malware* akan direpresentasikan dalam bentuk angka 0 sampai 255 yang berarti 0 adalah hitam dan 255 adalah putih, lalu gambar tersebut akan digunakan sebagai pembanding dari gambar *file* pendahulunya, jika terdapat kesamaan terhadap kedua gambar tersebut berarti *file malware* yang baru ini dibuat berdasarkan kode pada *malware* yang sebelumnya. Hal tersebut dilakukan agar mendeteksi *file malware* lebih sulit dilakukan.

Untuk melakukan pembandingan *Grayscale Image* tersebut dapat dilakukan dengan menggunakan teknik *entrophy graph similarity.*Teknik *entrophy graph similarity* adalah teknik yang menggunakan angka entropi sebagai variabel yang digunakan untuk menentukan apakah kedua buah gambar tersebut mirip atau sama.Jadi jika angka-angka yang ada cenderung mirip atau sama maka nilai dari entropi akan tinggi, sebaliknya jika angka-angka yang ada itu berbeda-beda atau cenderung tidak seragam maka nilai dari entropi akan rendah.

Pada pembuatan skripsi ini bahasa pemrograman yang digunakan untuk membangun aplikasi tersebut adalah python. Bahasa pemrograman python merupakan salah satu bahasa pemrograman yang paling populer untuk pendeteksian *malware* dan analisisnya. Penggunaan python sebagai bahasa pemrograman pada skripsi ini dikarenakan python lebih mudah untuk dipelajari, diaplikasikan dan lebih fleksibel dibandingkan bahasa pemrograman yang lain karena dapat dijalankan pada berbagai *Operating System* seperti Windows, Linux dan lainnya.

3 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan di atas, rumusan masalah pada skripsi ini adalah sebagai berikut :

1. Bagaimana cara kerja pendekatan *Grayscale Image* untuk memvisualisasikan jenis dari *malware* tersebut?
2. Bagaimana cara membuat aplikasi untuk mendeteksi *malware* menggunakan pendekatan *Grayscale Image*?
3. Bagaimana cara menampilkan visualisasi dari perbandingan kemiripan suatu *malware* dengan keluarga tertentu dengan menggunakan teknik *entrophy graph similarity*?

4 Tujuan

Berdasarkan rumusan masalah yang telah ada, berikut adalah tujuan dari pembuatan skripsi ini :

1. Mencari cara kerja pendekatan *Grayscale Image* untuk memvisualisasikan jenis dari *malware* tersebut.
2. Membangun aplikasi untuk mendeteksi *malware* menggunakan pendekatan *Grayscale Image.*
3. Menampilkan visualisasi dari perbandingan kemiripan suatu *malware* menggunakan teknik *entrophy graph similarity.*

5 Detail Perkembangan Pengerjaan Skripsi

Detail bagian pekerjaan skripsi sesuai dengan rencan kerja/laporan perkembangan terkahir :

1. **Mempelajari bahasa pemrograman python.**
   Status : Ada sejak rencana kerja skripsi.
   Hasil : Python adalah sebuah bahasa pemrograman yang sangat terkenal dikarenakan dapat digunakan hampir di seluruh platform yang ada. Python ditemukan pada tahun 1991 oleh Guido Van Rossum dan di dikembangkan oleh *Python Software Foundation*. Pada pengerjaan pengembangan perangkat lunak untuk mendeteksi *malware* ini akan digunakan bahasa pemrograman python dikarenakan lebih mudah untuk dibaca dan juga dapat dibuat menjadi lebih pendek dan singkat. Selain itu python juga memiliki kelebihan dimana program yang dibuat dalam bahasa python dapat digunakan di berbagai macam OS atau *Operating System* seperti Windows, Linux dan lain-lain.

   Pada pengerjaan perangkat lunak ini akan digunakan sebuah *framework* untuk GUI yang bernama Tkinter.Tkinter adalah sebuah *library* pada python yang dikhususkan sebagai pembuat GUI atau *Graphical User Interfaces* agar pengguna dapat lebih mudah untuk mengetahui apa sebenarnya yang dapat dilakukan oleh aplikasi setelah jadi.Tkinter sering digunakan untuk menjadi GUI *library* dikarenakan termasuk salah satu yang paling simpel dan fleksibel dibandignkan dengan *library* yang lain dan juga Tkinter sudah termasuk dalam python versi 3 jadi tidak perlu untuk menginstall ulang *library* tersebut.

   Tkinter pada umumnya akan lebih sering menggunakan *widgets* sebagai atribut yang akan muncul di layar pengguna. *Widgets* adalah sebuah element dari GUI yang memiliki banyak sekali fungsi dan cara untuk menggunakan mereka.Berikut adalah contoh *widgets* yang ada di dalam Tkinter GUI library.

   - **Frame** : Frame adalah sebuah garis untuk pembatas windows aplikasi yang akan memiliki ukuran yang sudah pasti. Pada Tkinter windows, frame sangat dibutuhkan agar aplikasi memiliki struktur dan lebih mudah untuk dikenali.
   - **Button** : Button adalah sebuah *widget* standar yang sering digunakan pada Tkinter. Sebuah button biasanya digunakan untuk sarana user berinteraksi dengan User Interface yang ada. Setelah button itu di klik maka akan ada sebuah aksi yang dijalankan oleh program.
   - **Entry** : Entry adalah sebuah *widget* standar yang digunakan untuk menerima masukan dari user melalui user interface yang ada. Entry ini pada umumnya berbentuk seperti sebuah box dimana user dapat memasukan tulisan kedalamnya.
   - **Check Button** : Check Button adalah sebuah *widget* yang digunakan untuk merepresentasikan sebuah pilihan yang sudah pasti atau sudah ditentukan sebelumnya. User dapat memilih lebih dari satu buah pilihan jika menggunakan *widget* ini.
   - **Radio Button** : Radio Button juga mirip dengan Check Button yaitu sebuah *widget* untuk merepresentasikan sebuah pilihan yang sudah ditentukan, tapi jika dengan menggunakan radio button user hanya dapat memilih satu buah pilihan.
   - **Label** : Label adalah sebuah *widget* yang digunakan untuk menandai sesuatu dengan sebuah tulisan.
   - **Menu** : Menu pada Tkinter adalah sebuah *widget* yang digunakan untuk membuat beberapa macam menu seperti menu yang ada di paling atas halaman sebelum title bar.
   - **Combo Box** : Combo Box adalah sebuah *widget* yang dapat menampilkan sebuah box yang dapat diklik agar memunculkan drop-down list. Data yang ditampilkan di Combo Box akan memiliki maksimal sehingga jika sudah mencapai paling bawah dari tampilan maka akan ada slider di sebelah kanan untuk menandakan bahwa masih ada isi yang lain.

- **List Box** : Mirip dengan Combo Box, List Box adalah sebuah *widget* untuk menampilkan beberapa pilihan untuk user, tapi jika List Box data akan ditampilkan secara keseluruhan tidak ada maksimal.
- **Menu Button** : Menu Button adalah kombinasi dari Button dan Menu yang digunakan untuk menampilkan beberapa opsi untuk dipilih oleh pengguna.
- **Canvas** : Canvas adalah sebuah *widget* yang digunakan untuk menggambar grafik atau plot di dalam aplikasi yang dikembangkan. Canvas juga dapat digunakan untuk memasukkan gambar ataupun menggambar dari awal.
- **Scale** : Scale adalah sebuah *widget* yang digunakan untuk menampilkan sebuah horizontal slider untuk user memasukan input value dari range yang sudah ditentukan.
- **Scrollbar** : Scrollbar adalah sebuah *widget* yang digunakan untuk berpindah posisi menggunakan slider yang berada di sebelah kanan dari windows.
- **Text** : Text adalah sebuah *widget* yang digunakan untuk mendapatkan input dari user yang isinya lebih dari 1 baris dan juga memiliki banyak fitur lainnya seperti dapat memasukkan gambar, tags, dan lain-lain.
- **File Dialogs** : File Dialogs adalah salah satu *widget* yang cukup unik di Tkinter. *Widget* ini digunakan untuk user menyimpam atau memilih file yang berada di komputer pengguna menggunakan sebuah dialog.
- **Toplevel** : Toplevel adalah sebuah *widget* yang digunakan untuk mengeluarkan sebuah windows Tkinter baru.

Selain beberapa *widget* diatas ada juga beberapa special extension dari Tkinter GUI *library* pada python yang sering disebut sebagi Tkinter ttk module. Ttk module ini sebenarnya merupakan kependekan dari Themed Tkinter yang menyediakan berbagai macam styling dan theming support untuk Tkinter. Berikut adalah contoh *widgets* yang hanya ada pada Themed Tkinter library.

- **Notebook** : Notebook adalah *widget* yang mengatur semua windows atau tabs yang sedang terbuka pada aplikasi dan juga dapat mengganti windows yang sedang ditampilkan pada aplikasi.
- **Progress Bar** : Progress Bar adalah *widget* yang dapat menunjukan progress atau loading process dengan menggunakan animasi.
- **Separator** : Separator adalah *widget* untuk memisahkan berbagai macam *widgets* menggunakan separator line.
- **Treeview** : Treeview adalah *widget* yang digunakan untuk mengelompokkan beberapa item ke dalam sebuah hirarki yang berbentuk seperti pohon. Setiap item pada hirarki tersebut dapat memiliki label, gambar, dan list data isinya.
- **Sizegrip** : Sizegrip adalah *widget* yang digunakan untuk merubah ukuran windows dari aplikasi. Biasanya akan terdapat sebuah handle kecil di ujung kanan bawah layar aplikasi.

2. **Mempelajari lebih dalam pemahaman tentang *malware*.**
   Status : Ada sejak rencana kerja skripsi.
   Hasil : *Malicious Software* atau yang biasa disebut *malware* adalah segala aplikasi atau *file* yang ketika dijalankan akan melakukan hal yang mengganggu kinerja dari komputer pengguna seperti pengguna akan merasakan *lag* yang dialami komputernya, bahkan jika *malware* ini sudah masuk sangat dalam ke komputer pengguna maka *malware* ini dapat mendapatkan dan mencuri data-data yang penting yang ada didalam komputer pengguna. Pada umumnya *malware* yang sudah mendapat akses ke dalam data yang ada didalam komputer pengguna akan mengenkripsi data yang ada didalam komputer pengguna dan meminta bayaran atau ganti rugi agar kita dapat mendapatkan data itu kembali.Untuk

mendapatkan data dari *malware* diperlukan *malware repository*. *malware repository* yang digunakan pada pembuatan skripsi ini adalah theZoo. TheZoo adalah sebuah live *malware repository* yang dapat diakses oleh publik dengan mudah dan aman. Berikut adalah beberapa keluarga *malware* yang sering ditemukan dalam komputer dan *malware repository* pada umumnya yaitu :

- Virus
  Virus adalah salah satu jenis *malware* yang pada umumnya akan menyerang komputer dari satu ke komputer yang lainnya juga, biasanya jenis *malware* ini akan menyusup melalui *file* yang berada di komputer pengguna dan jika *file* tersebut dikirimkan ke perangkat lain maka virus akan keluar untuk menyerang perangkat tujuannya tersebut.
- Worm
  Worm adalah jenis *malware* yang jika dijalankan akan mereplika dirinya sendiri. Program ini biasanya menggunakan jaringan untuk menyerang perangkat-perangkat lain yang tersambung dalam jaringan yang sama. Perbedaan dari virus dan worm adalah jika virus biasanya akan lebih cenderung ke merusak atau mengubah *file* yang ada di perangkat korban, sedangkan worm biasanya akan mengganggu jaringan karena akan memakai jaringan sebagai sarana untuk mengirimkan dirinya ke perangkat lain yang ada di jaringan.
- Trojan Horse
  Trojan Horse atau yang biasa disebut sebagai Trojan adalah sebuah aplikasi yang jika dilihat secara kasat mata adalah aplikasi yang berguna bagi pengguna sehingga aplikasi tersebut akan sering diunduh dan diinstal pada komputer-komputer banyak pengguna , tapi sebenarnya di dalam aplikasi tersebut terdapat virus yang dapat merusak komputer tempat dimana aplikasi tersebut diunduh dan diinstal. *malware* jenis Trojan Horse ini terinspirasi dari teknik perang yang digunakan oleh Romawi saat menyerang kota troya yang dimana pasukan Romawi memberikan sebuah hadiah yang berbentuk patung kuda kayu yang dari luar terlihat seperti hanya patung, padahal didalam patung tersebut ada beberapa pasukan dari Romawi yang tersembunyi untuk menyerang kota troya tersebut.
- Spyware
  Spyware adalah sebuah aplikasi yang jika diinstal secara diam-diam dapat menyebabkan komputer pengguna dapat dipantau dari jauh oleh oknum yang tidak bertanggung jawab. Spyware ini memang dirancang dan dibuat untuk mengakses dan memonitor aktivitas yang dilakukan korban mulai dari sejarah pencarian di internet, username dan kata sandi, hingga dapat juga membuka kamera yang terpasang kedalam komputer korban untuk mengetahui wajah dan tempat dimana korban menggunakan perangkat tersebut. Spyware ini biasa ditemukan dalam sebuah software yang biasanya berkedok *free trial* atau aplikasi bajakan yang banyak beredar di internet. Ada beberapa jenis dari spyware ini contohnya Keyloggers, Password Stealers, dan Banking Trojans.
- Scareware
  Scareware adalah salah satu jenis *malware* yang sering ditemukan jika pengguna menginstall aplikasi-aplikasi bajakan yang tersebar di internet. Tanda-tanda munculnya scareware salah satunya adalah dari munculnya pemberitahuan untuk menginstall sebuah aplikasi dikarenakan komputer pengguna sedang terkena virus dan akan mengalami kerusakan jika tidak menginstall aplikasi yang disarankan. Hal seperti itu sering terjadi jika kita sering membuka halaman-halaman web yang tidak secure atau tidak memiliki pengamanan yang ketat.Jika aplikasi yang diberikan itu diinstall di komputer pengguna maka aplikasi tersebut kemungkinan besar dapat menyimpan informasi-informasi penting yang ada didalam perangkat korban seperti email, password, atau lebih buruknya adalah nomor pin kartu kredit pengguna.

- Adware
  Adware adalah sebuah jenis *malware* yang tidak secara langsung menyerang komputer pengguna melainkan akan memunculkan iklan-iklan pada komputer pengguna. Contoh umum untuk adware adalah jika kita membuka website dan muncul *pop-up* iklan maka itu adalah adware atau jika pada komputer kita tiba-tiba muncul iklan di kanan bawah walaupun tidak menggunakan aplikasi apapun. Pada umumnnya adware hanya ditujukan untuk mengirimkan iklan kepada pengguna komputer, tetapi tidak jarang juga beberapa adware digabungkan dengan sebuah spyware yang dapat melacak aktivitas pengguna pada komputer.

- Botnet
  Botnet adalah sebuah program *malware* yang dibuat untuk menjalankan sebuah operasi secara otomatis. Pada umumnya serangan botnet ada beberapa macam seperti DDoS attacks, spambot, dan lain-lain. Biasanya botnet akan dijalankan oleh beberapa komputer dan dapat di kontrol oleh pembuatnya.

- Ransomware
  Ransomware adalah salah satu jenis *malware* yang jika dijalankan akan merusak kinerja komputer pengguna dengan berbagai macam cara. Pada umumnya program ransomware akan menampilkan sebuah pesan atau gambar yang meminta pengguna komputer untuk membayar agar dapat menggunakan komputernya kembali. Jika pengguna tidak membayar maka komputer tersebut akan tetap tidak dapat dijalankan.Berikut adalah contoh tampilan dari program ransomware.

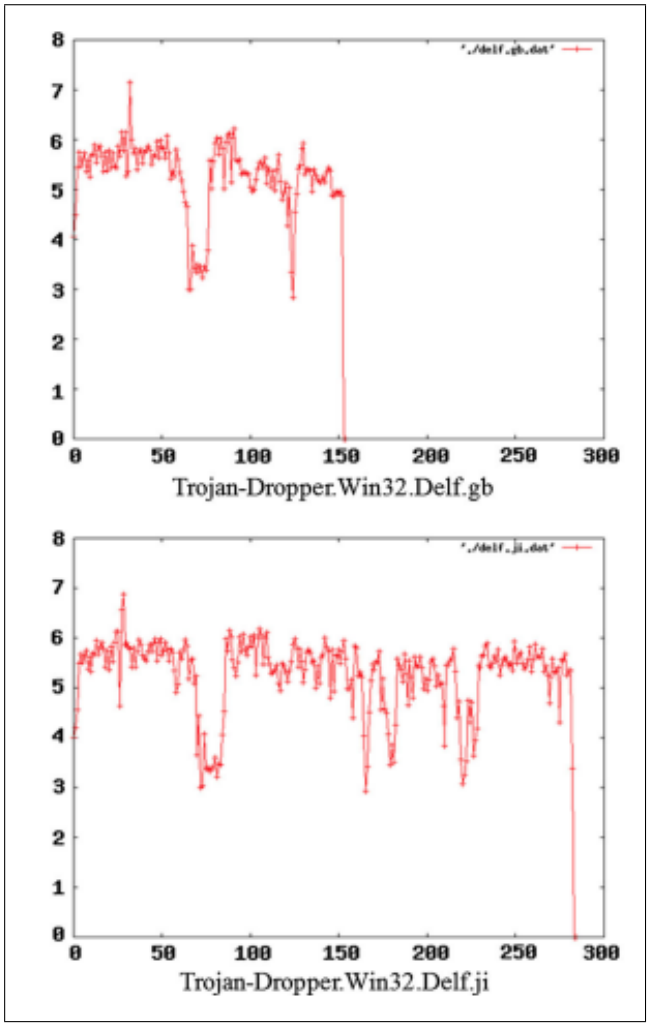
Gambar 1: Contoh Pesan Yang Dikirimkan Ransomware.

Seiring berjalannya waktu ransomware juga semakin berkembang dalam kinerjanya, dengan menggunakan banyak variasi cara untuk mengisolasi sebuah komputer.Ancaman paling terkenal saat ini adalah seperti yang ada pada gambar di atas. Komputer pengguna akan menampilkan sebuah pesan yang mengatasnamakan pemerintah atau aparat hukum yang biasanya beralasan bahwa pengguna telah mengakses situs-situs illegal yang dilarang oleh pemerintah, dan oleh karena itu maka pengguna harus membayarkan biaya tilang yang ditetapkan oleh penyerang tersebut. Biasanya pada pesan-pesan tersebut banyak yang menggunakan logo resmi dari aparat hukum agar pengguna komputer semakin percaya bahwa pesan itu memang benar dari pemerintah.

3. **Mempelajari teknik *entropy graph similarity*.**
   Status : Ada sejak rencana kerja skripsi.
   Hasil : Nilai entropi adalah sebuah nilai yang mengindikasi kan sebanyak apa atau sesering apa sebuah nilai muncul, misalkan jika semua nilai yang muncul bernilai sama maka nilai entropi akan menjadi besar dikarenakan angka tersebut sering muncul, namun jika angka yang keluar berbeda-beda maka nilai entropi pun akan menjadi kecil. Nilai entropy ini dapat digunakan untuk membandingkan dua buah bitmap image. Berikut adalah rumus untuk menghitung nilai entropi.

$$\text{Entropy} = -\sum_{i=0}^{255} p_i \times \log_2 p_i \tag{1}$$

   Entropy graph adalah sebuah grafik yang merepresentasikan nilai entropi dari sebuah gambar. grafik tersebut digunakan untuk melihat kemiripan dari dua buah gambar yang berbeda. Berikut adalah contoh gambaran *entropy graph* dari *malware* berjenis trojan.



Gambar 2: Contoh *Entropy Graph malware* Jenis Trojan.

Untuk menghitung kemiripan dari dua buah grafik membutuhkan beberapa value berikut yaitu k1 dan k2

- **k1** : k1 adalah sebuah value yang dihitung menggunakan penjumlahan dari perbedaan nilai entropi di titik yang sama dalam sumbu x. k1 menunjukan kesamaan dari bentuk general dari dua buah *entropy graph* tersebut. Waktu k1 dihitung , perhitungan pertama yang harus dilakukan adalah menyamakan tinggi dari dua buah *entropy graph* tersebut atau batas atas sumbu x nya tidak boleh lebih dari sebuah titik. Berikut adalah rumus untuk menghitung nilai k1.

$$k_1[H,\ L] = exp\left(-\frac{s[H,\ L]}{\bar{s}} \right) \tag{2}$$

S adalah variabel yang digunakan untuk menandakan jumlah dari jarak antara dua buah titik di grafik pada titik yang sama, dan $\bar{S}$ adalah luas area dari *entropy graph* tersebut. Berikut adalah rumus untuk variabel S dan $\bar{S}$.

$$s[H, L] = \int_x |H(x) - L(x)|dx \tag{3}$$

$$\bar{s} = \text{ Average } (H) \times \text{ Length of } y \text{ axis} \tag{4}$$

H dan L yang dimaksud dalam rumus yang sudah dicantumkan diatas adalah menandakan dua buah grafik yaitu grafik H dan L. Setelah mengakumulasi dan menghitung seluruh hasil dari penjumlahan, hasil tersebut akan di bagi dengan $\bar{S}$ dan k1 akan dihitung dengan fungsi exp(). Jika dua buah *entropy graph* yang dibandingkan sama persis, maka tidak akan ada jarak untuk dimasukan ke rumus, atau dengan kata lain hasil dari S adalah 0, dan hasil akhir k1 akan menjadi 1. Dikarenakan range dari s[H, L] adalah dari 0 hingga infiniti maka jika *input value* dari s[H, L] adalah 0 maka k1 akan menjadi semakin dekat dengan 1, tapi jika *input value* semakin meningkat maka hasil k1 akan semakin mendekati 0.

- **k2** : k2 adalah sebuah value yang dihitung menggunakan jarak antara *local maxima* dari dua buah *entropy graph* yang akan dibandingkan.berikut adalah rumus dari k2.

$$k_2[H, L] = \sum_i u_i[H]c_i[H, L] \tag{5}$$

Dimana

$$u_i = \frac{\left|H^{(2)}\left(x_i\right)\right| \times l_i}{\sum_{i=1}^{n(H)} \left|H^{(2)}\left(x_i\right)\right| \times l_i} \tag{6}$$

$$c_{i,j} = c_{i,j}^x \times c_{i,j}^y \tag{7}$$

$$\begin{aligned} c_{i,j}^x &= -\exp\left(\frac{\Delta x_{i,j}}{\delta x}\right)^2 \\ \Delta x_{i,j} &= x_i - \bar{x}_j \end{aligned} \tag{8}$$

$$\begin{aligned} c_{i,j}^y &= -\exp\left(\frac{\Delta y_{i,j}}{\delta y}\right)^2 \\ \Delta y_{i,j} &= H\left(x_i\right) - L\left(\bar{x}_j\right) \end{aligned} \tag{9}$$

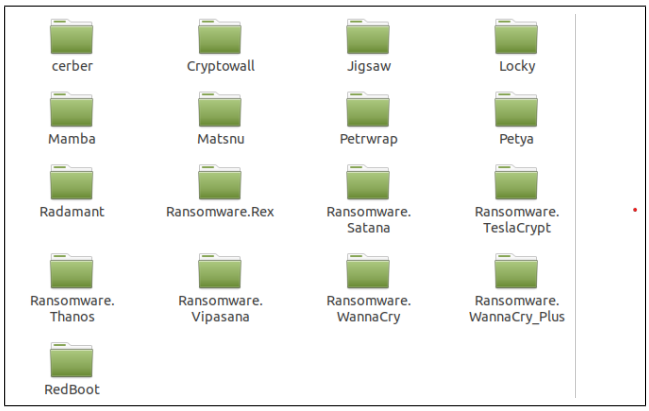
u_i adalah perbandingan dari *local maxima*, dimana c_i dihitung dengan menggunakan jarak diantara *local maxima* ke-i dan *local maxima* terdekat kepada i. l_i adalah sebuah jarak diantara sebuah

garis lurus yang dibentuk oleh dua buah *local maxima* dimana nilainya akan berkisar antara 0 sampai dengan nilai *local maxima* tersebut.

Setelah k1 dan k2 sudah berhasil dihitung maka dua buah variabel tersebut akan dijumlahkan untuk menghitung nilai akhir dari kesamaan dua buah *entropy graph* yang ada.Perbandingan kemiripan dari k1 dan k2 akan ditentukan dengan cara menghitung jumlah dari k1 dan k2 pada perhitungan kemiripan dari dua buah *entropy graph* tersebut. Untuk menghitung kemiripan dari dua buah *entropy graph* tersebut akan digunakan dua buah variabel t1 dan t2 yang jika dijumlahkan akan menjadi 1. Berikut adalah rumus untuk menghitung kemiripan dari dua buah *entropy graph*.

$$S = t_1 \times k_1 + t_2 \times k_2 \tag{10}$$

4. **Mengumpulkan data set untuk berbagai macam jenis *malware*.**
   Status : Ada sejak rencana kerja skripsi.
   Hasil : Pengumpulan data set yang dilakukan pada skripsi ini dilakukan agar pengguna aplikasi nantinya dapat melihat apa saja jenis *malware* yang dapat di klasifikasikan didalam aplikasi yang akan dikembangkan tersebut. Sejauh ini telah dapat dikumpulkan hampir seluruh *dataset malware* yang memiliki jenis ransomware, figur berikut adalah contoh hasil *dataset* dari *malware* bertipe ransomware yang telah didapatkan :



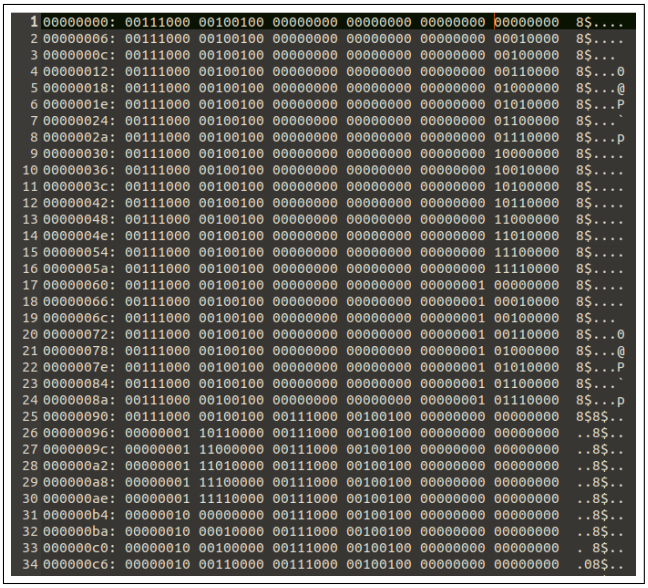
Gambar 3: *Dataset malware* bertipe ransomware.

Semua jenis dari *malware* tersebut akan didekripsi dan dimasukkan kedalam bank *malware* yang akan ada dalam aplikasi yang akan dikembangkan. Untuk saat ini *malware* yang digunakan untuk percobaan hanya dari kelas ransomware saja.

5. **Melakukan pengujian terhadap data set yang sudah didapatkan.**
   Status : Ada sejak rencana kerja skripsi.
   Hasil : Setelah berhasil mengumpulkan seluruh *dataset* yang dibutuhkan, seluruh data yang didapatkan tersebut akan diekstrak dan akan dicoba untuk diubah menjadi sebuah *file* biner yang disimpan dalam sebuah *file* txt.Pengujian ini dilakukan untuk mengetahui bahwa apakah seluruh jenis *malware* dapat dikonversi menjadi sebuah *file* biner yang nantinya akan di visualisasikan menjadi sebuah grayscale image. Untuk mengubah *file executables* dari *malware* tersebut menjadi sebuah *file* biner dibutuhkan sebuah *tools* lagi yang dapat digunakan dalam linux yaitu xxd. Xxd adalah sebuah *library* yang berada di linux yang dapat membuat *hex dump* dari sebuah *file* atau masukan biasa. Selain membuat *hex dump* xxd juga dapat mengkonversi sebuah *file* kembali menjadi *file* biner. Xxd memiliki beberapa parameter penting yang dapat digunakan. Berikut adalah beberapa parameter penting yang digunakan dalam pengujian *dataset malware* yang sudah didapatkan.

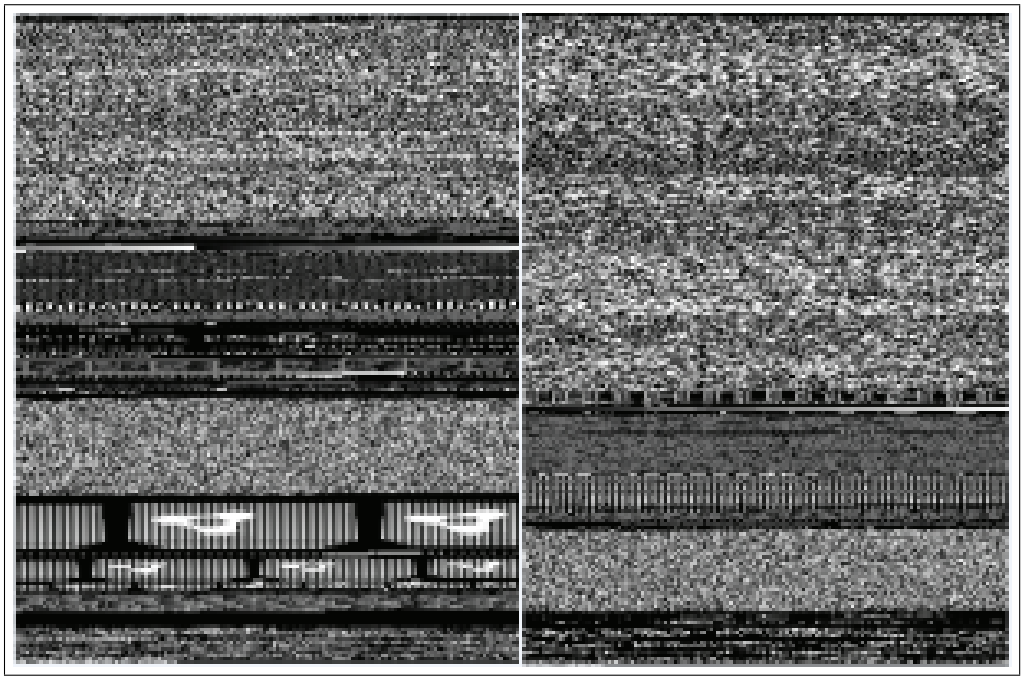
- **-b** / **-bits** : parameter ini digunakan untuk mengeluarkan hasil menjadi sebuah *binary dump* melainkan hexadecimal. Setiap baris dari biner tersebut akan diikuti oleh sebuah representasi bilangan ascii.
- **-r** / **-revert** : parameter ini digunakan untuk mengubah sebuah hexdump menjadi *binary files*.

Berikut adalah salah satu contoh *file executables* dari *malware* keluarga cerber yang sudah berhasil diconvert menjadi sebuah *binary file* menggunakan xxd.

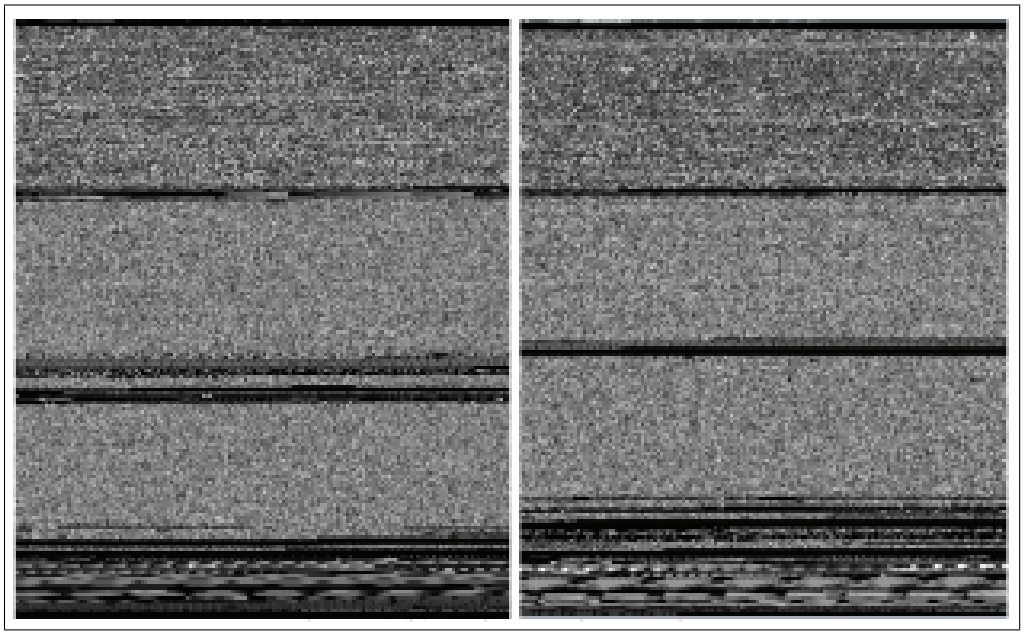


Gambar 4: *Dataset malware* Keluarga Cerber Dalam Biner.

6. **Mempelajari cara menganalisis sebuah *Grayscale Image***
   Status : Ada sejak rencana kerja skripsi.
   Hasil : Setelah seluruh *dataset* dikonversi menjadi *file* biner menggunakan xxd, *file* tersebut akan diubah menjadi sebuah *grayscale image*. *Grayscale* adalah sebuah nilai yang berkisar antara 0 sampai dengan 255 yang tiap poinnya menandakan warna dari sebuah pixel. 0 akan merepresentasikan warna hitam dan 1 akan merepresentasikan warna putih. Berikut adalah contoh *file executables* yang sudah diubah menjadi *grayscale image* dari *file executables* biasa dan juga *file malware*.



Gambar 5: Contoh *Grayscale Image* untuk *File Executables* Pada Umumnya.

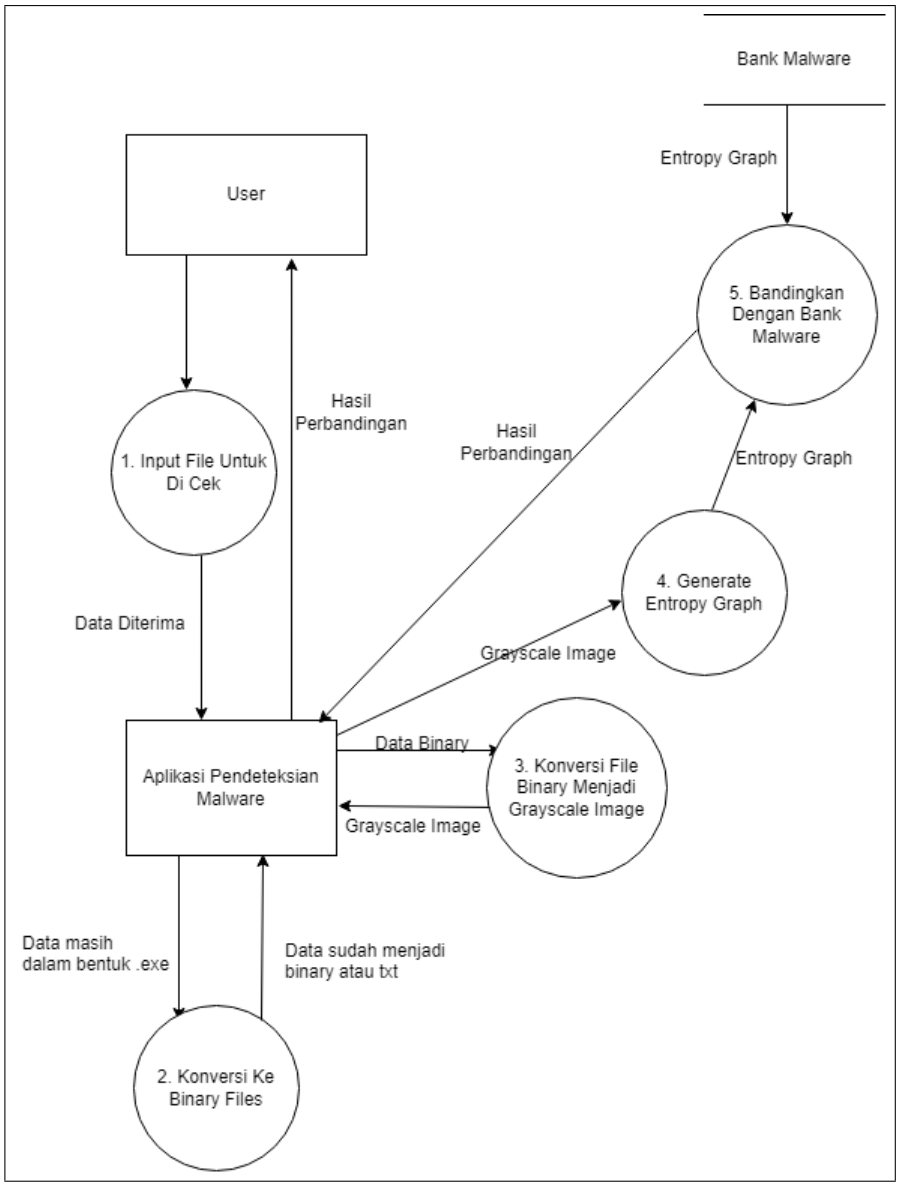


Gambar 6: Contoh *Grayscale Image* untuk *Malware* Trojan Keluarga C2lop.

Dikarenakan perbedaan ukuran dari *file malware* yang beredar maka ukuran dari *grayscale image* juga akan menjadi berbeda. Jika ukuran dari dua buah *grayscale image* tersebut berbeda maka dua buah gambar tersebut tidak akan bisa untuk dibandingkan. Oleh karena itu agar setiap pixel dari kedua buah gambar tersebut dapat dibandingkan maka ukuran dari grayscale image harus diubah sedemikian rupa agar menjadi sama dan tiap pixel akan dapat dibandingkan.

7. **Mempelajari cara untuk menggunakan *file malware* tanpa merusak komputer.**
   Status : Ada sejak rencana kerja skripsi.
   Hasil : Sebuah *file* .exe yang ada didalam komputer belum tentu aman seluruhnya. Oleh karena itu *file* tersebut harus di cek dulu apakah aman untuk dibuka langsung atau ternyata berisi *malware* yang dapat merusak komputer. Jika pengguna tahu bahwa *file* tersebut adalah *malware* , pengguna dapat melakukan *sandboxing*. *Sandboxing* adalah sebuah cara untuk mempelajari, menganalisis dan mengobservasi sebuah kode pada sebuah *environment* pada jaringan yang aman. *Sandboxing* di desain untuk mengantisipasi kerusakan yang terjadi dalam jaringan dan komputer yang dapat diakibatkan oleh kode yang berbahaya. Salah satu cara untuk *sandboxing* adalah menggunakan VM atau *Virtual Machine* untuk membuka *file* yang berbahaya tersebut untuk dianalisis dan dijalankan. Pada pengerjaan pengembangan perangkat lunak pada skripsi ini *Virtual Machine* yang digunakan adalah VM DEVASC yang berasal dari oracle yang memiliki *operating system* Ubuntu.

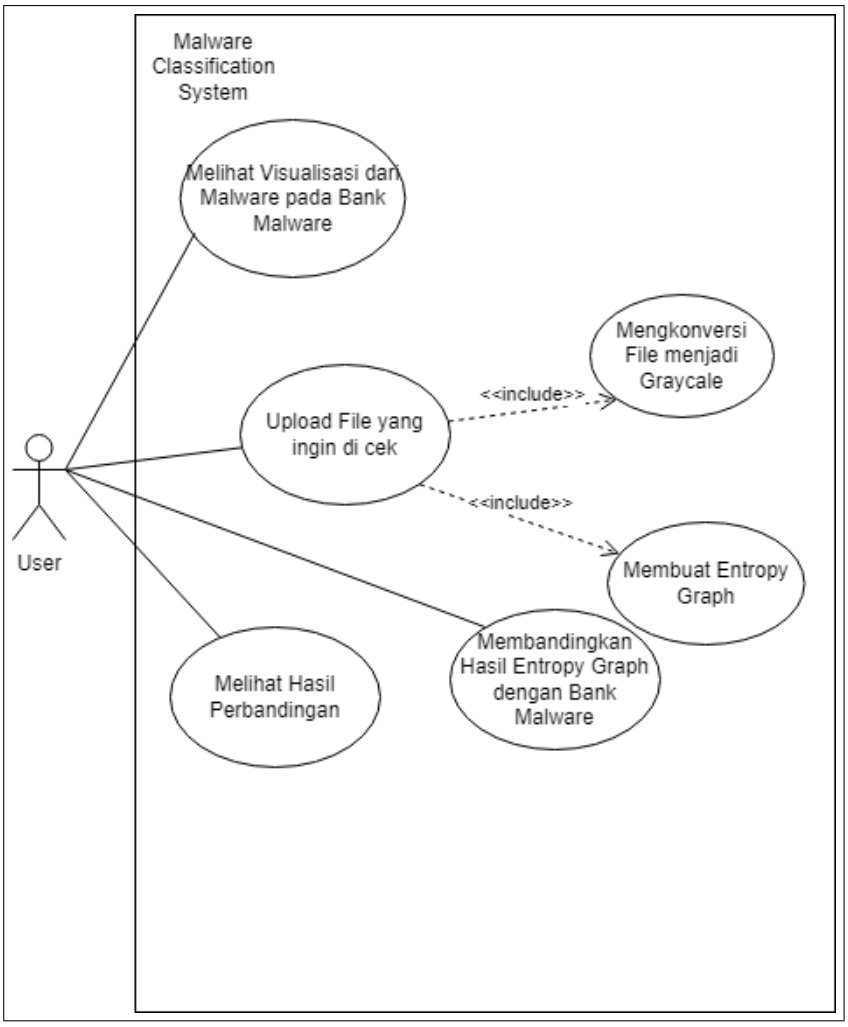
8. **Menganalisis kebutuhan perangkat lunak.**
   Status : Ada sejak rencana kerja skripsi.
   Hasil : Analisis tentang kebutuhan perangkat lunak telah dilakukan dengan cara menggunakan beberapa diagram untuk menggambarkan tentang bagaimana cara aplikasi berjalan dan juga bagaimana user akan menggunakan aplikasi tersebut. Berikut adalah gambaran tentang *data flow diagram* dan juga beserta penjelasannya.



Gambar 7: *Data Flow Diagram.*

Pertama-tama user akan mengirimkan *file* yang ingin di cek apakah *file* tersebut adalah *malware* atau bukan. Lalu aplikasi akan mengkonversi *file* input dari user untuk menjadi sebuah *binary files* yang nantinya *binary files* tersebut akan diubah lagi menjadi sebuah *grayscale image*. Setelah menjadi *grayscale image*, akan dibuat sebuah *entropy graph* yang berdasarkan dari *grayscale image* tersebut dan akan dibandingkan dengan data yang berada pada bank *malware*.Setelah perbandingan sudah selesai maka hasil akan dikirimkan lagi ke aplikasi agar dapat dikirimkan ke user.

Selain *data flow diagram* ada juga *use case diagram* yang dapat membantu analisis kebutuhan perangkat lunak yang akan dikembangkan nantinya. Berikut adalah gambar dari *use case diagram* dan juga beserta penjelasannya.



Gambar 8: *Use Case Diagram.*

Mirip dengan *data flow diagram* user akan dapat melakukan dua buah aktivitas yaitu mengupload *file input* yang ingin di cek apakah *file* tersebut adalah *malware* dan juga user dapat melihat semua *malware* yang terdaftar pada bank *malware*.Setelah user mengirimkan *input file* maka aplikasi langsung akan mengkonversi *file* tersebut menjadi sebuah *file grayscale image* dan juga dikonversi lagi menjadi sebuah *entropy graph*. Setelah dikonversi menjadi *entropy graph*, aplikasi akan membandingkan dengan *file entropy graph* yang ada pada bank *malware*. Setelah itu maka hasil akan langsung dikirimkan kepada user.

9. **Menulis dokumen skripsi bagian bab 1, 2, dan 3.**
   Status : Ada sejak rencana kerja skripsi.
   Hasil : Hingga saat ini dokumen skripsi sudah dikerjakan untuk beberapa bagian, seperti pada bagian bab pendahuluan, landasan teori dan analisis. Pada bagian bab analisis sudah ada beberapa diagram yang dibuat seperti *Use Case Diagram* dan *data flow diagram*. Keseluruhan dokumen yang sudah disebutkan diatas masih dapat diubah, dilakukan penambahan atau pengurangan di hari yang akan datang.

10. **Melakukan implementasi pembuatan sebuah *entropy graph* dari *grayscale image***
    Status : Baru ditambahkan semester ini.
    Hasil : Akan dikerjakan di skripsi 2.

11. **Melakukan implementasi pengubahan *file executable* menjadi *grayscale image***
    Status : baru ditambahkan semester ini.
    Hasil : Akan dikerjakan di skripsi 2.

12. **Merancang antarmuka dan struktur kode.**
    Status : Ada sejak rencana kerja skripsi.
    Hasil : Akan dikerjakan di skripsi 2.

13. **Menguji perangkat lunak yang sudah dibuat.**
    Status : Ada sejak rencana kerja skripsi.
    Hasil : Akan dikerjakan di skripsi 2.

14. **Menarik kesimpulan dari pengujian perangkat lunak.**
    Status : Ada sejak rencana kerja skripsi.
    Hasil : Akan dikerjakan di skripsi 2.

15. **Menyelesaikan keseluruhan dokumen skripsi**
    Status : Ada sejak rencana kerja skripsi.
    Hasil : Akan dikerjakan di skripsi 2.

6 Pencapaian Rencana Kerja

Langkah-langkah kerja yang berhasil diselesaikan dalam Skripsi 1 ini adalah sebagai berikut:

1. Mempelajari bahasa pemrograman python.
2. Mempelajari lebih dalam pemahaman tentang *malware*.
3. Mempelajari teknik *entrophy graph similarity*.
4. Mempelajari cara menganalisis sebuah *Grayscale Image*.
5. Mengumpulkan data set untuk berbagai macam jenis *malware*.
6. Melakukan pengujian terhadap data set yang sudah didapatkan.
7. Mempelajari cara untuk menggunakan *file malware* tanpa merusak komputer.
8. Menganalisis kebutuhan perangkat lunak.
9. Menulis dokumen skripsi bagian bab 1, 2, dan 3.

Bandung, 02/04/2022

Ivan Limosi

Menyetujui,

Nama: Chandra Wijaya, S.T., M.T.
Pembimbing Tunggal