Penerapan Kriptografi dalam Pengamanan *Data at Rest* Level *Filesystem* melalui Program *Encrypted File Explorer* untuk Penggunaan Sehari-hari

Dimas Faidh Muzaki - 13520156
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail : 13520156@std.stei.itb.ac.id

***Abstract*—Kebocoran data merupakan salah satu ancaman keamanan yang paling ditakuti karena data dapat bersifat rahasia dan krusial. Kriptografi merupakan salah satu ilmu yang dapat menjawab ketakutan tersebut. Menggunakan kriptografi, kita dapat mengamankan data yang tersimpan dengan cara mengenkripsinya. Data yang tersimpan dalam bentuk terenkripsi tidak akan dapat dipakai ataupun disebar oleh penyusup karena makna yang telah disembunyikan. Di sisi lain, pengaksesan oleh pihak yang berhak akan memerlukan program khusus untuk menggunakan data secara normal.**

Keywords—AES; Data Leak; File System;

I. PENDAHULUAN

Kriptografi, seni dan ilmu yang bertujuan untuk mengamankan informasi, telah memainkan peran yang sangat penting dalam melindungi data sensitif sepanjang sejarah. Kriptografi memberikan sarana untuk melindungi kerahasiaan, integritas, dan otentisitas data, memastikan bahwa data tetap tidak terbaca dan tidak dapat diubah oleh pihak yang tidak berwenang.

Keamanan data merupakan hal yang sangat penting dalam era digital ini. Dengan pertumbuhan yang pesat dalam volume dan nilai data yang dipertukarkan dan disimpan oleh organisasi dan individu, perlindungan terhadap kebocoran, manipulasi, atau akses yang tidak sah terhadap data menjadi suatu keharusan. Resiko kerugian dari pelanggaran keamanan data dapat diminimalisir salah satunya dengan cara pengenkripsian data.

Ransomware adalah salah satu ancaman siber yang paling merusak dan memprihatinkan dalam dunia digital saat ini. Ini merupakan jenis perangkat lunak berbahaya yang secara diam-diam memasuki sistem komputer seseorang dan mengenkripsi data yang berharga. Pelaku kemudian meminta pembayaran tebusan dalam bentuk mata uang digital, sering kali Bitcoin, sebagai imbalan untuk mengembalikan akses ke data yang terenkripsi. Tak berhenti disitu, pelaku pun memiliki kuasa untuk menyebarkan data tersebut. Hal ini dapat menjadi masalah apabila data bersifat sangat rahasia.

Akhir-akhir ini, banyak perbincangan tentang insiden ransomware yang menimpa sebuah bank BUMN. Kabarnya, data bank tersebut akhirnya tersebar secara publik melalui dark web setelah tidak tercapainya kesepakatan dalam hal pembayaran tebusan. Seperti yang kita ketahui, ransomware pada umumnya menyerang "*data at rest"* yang rentan terhadap serangan remote access. Oleh sebab itu, penulis mengusulkan sebuah ide inovatif mengenai penyimpanan data yang selalu terenkripsi. Dengan pendekatan ini, akses terhadap data akan memerlukan penggunaan *explorer* khusus dan kunci yang sesuai. Hal ini bertujuan agar data yang berhasil diambil oleh ransomware menjadi tidak berguna saat tersebar.

II. DASAR TEORI

A. Enkripsi Data

Enkripsi data merupakan salah satu teknik untuk memenuhi aspek keamanan yang ditawarkan oleh kriptografi, yaitu *confidentiality* (Kerahasiaan) yang merupakan aspek keamanan yang menjaga pesan/data agar tidak dapat dibaca oleh pihak yang tak berwenang. Enkripsi melibatkan konsep dan algoritma yang digunakan untuk melindungi data dengan mengubahnya menjadi bentuk yang tidak terbaca, atau "terenkripsi", yang hanya dapat diurai kembali menjadi bentuk aslinya oleh pihak yang memiliki "kunci enkripsi" yang sesuai.

Kunci enkripsi adalah nilai rahasia atau parameter yang digunakan dalam algoritma enkripsi untuk mengubah data menjadi bentuk terenkripsi. Kunci enkripsi harus disimpan secara aman dan hanya dapat diakses oleh pihak yang berwenang. Kriptografi menurut jenis kuncinya dibedakan menjadi simetris dan asimetris.

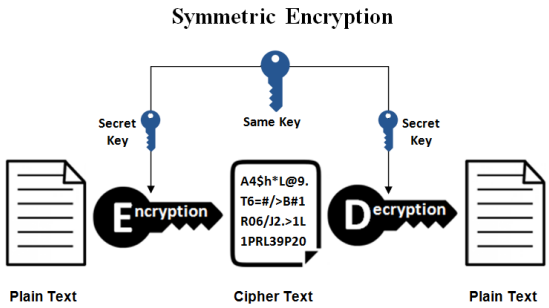


Fig. 1. Symmetric Encryption, sumber: SSL2BUY

Enkripsi simetris melibatkan penggunaan satu kunci yang sama untuk melakukan proses enkripsi dan dekripsi. Algoritma enkripsi simetris yang umum digunakan termasuk Advanced Encryption Standard (AES) dan Data Encryption Standard (DES). Keuntungan dari enkripsi simetris adalah kecepatan yang tinggi dan efisiensi dalam mengenkripsi dan mendekripsi data. Namun, tantangan utamanya adalah bagaimana mengamankan pertukaran kunci enkripsi yang sama di antara pihak-pihak yang berkomunikasi.

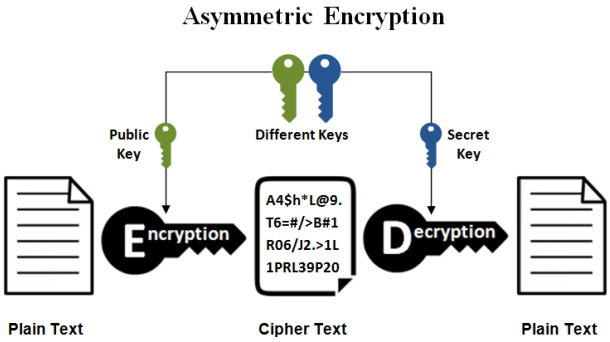


Fig. 2. Asymmetric Encryption, sumber: SSL2BUY

Enkripsi asimetris, atau dikenal juga sebagai enkripsi kunci publik, menggunakan sepasang kunci yang berbeda untuk melakukan proses enkripsi dan dekripsi. Kunci publik digunakan untuk mengenkripsi data, sementara kunci pribadi yang hanya diketahui oleh penerima digunakan untuk mendekripsi data tersebut. Keuntungan utama dari enkripsi asimetris adalah bahwa tidak diperlukan pertukaran kunci rahasia, sehingga menjadikannya lebih aman dalam konteks komunikasi yang tidak terpercaya. Algoritma enkripsi asimetris yang umum digunakan termasuk RSA (Rivest-Shamir-Adleman) dan ElGamal.

B. *Three States of Data*

Menurut [1], *Three states of data* merupakan cara untuk mengategorikan data yang terstruktur maupun tidak terstruktur. Ketiga *Three states of data* yang dimaksud adalah *data at rest*, *data in motion*, dan *data in use*. Berikut penjelasan singkat untuk setiap definisi:

1) Data at rest:

Data at rest mengacu pada data yang tersimpan dan tidak aktif dalam media penyimpanan fisik atau digital, seperti basis data, hard drive, atau arsip.

2) Data in motion:

Data in motion merujuk pada data yang sedang aktif ditransmisikan atau dipindahkan dari satu lokasi ke lokasi lain melalui jaringan. Ini termasuk data yang dikirim melalui email, ditransfer antara server, atau diakses melalui koneksi remote. Data dalam perjalanan rentan terhadap penyadapan atau akses tidak sah. Untuk memastikan keamanan data, sering kali digunakan protokol enkripsi, jaringan pribadi virtual (VPN), protokol transfer file aman (SFTP), atau saluran komunikasi yang aman lainnya.

3) Data in use:

Data in use mengacu pada data yang sedang aktif diproses, diakses, atau dimanipulasi oleh aplikasi, sistem, atau pengguna. Keadaan ini terjadi saat data dimuat ke dalam memori atau cache dan digunakan untuk operasi komputasi atau kegiatan lainnya. Data dalam penggunaan lebih rentan terhadap ancaman keamanan, seperti akses tidak sah, modifikasi tidak sah, atau perangkat lunak jahat.

C. *Perlindungan Data at Rest*

Perlindungan *Data at Rest* bertujuan untuk mengamankan data tidak aktif yang disimpan pada perangkat atau jaringan. Meskipun *Data at Rest* dianggap lebih tidak rentan daripada *Data in motion*, serangan sering kali menyasar *data at rest* karena dianggap lebih berharga daripada data yang sering bergerak. Salah satu teknik perlindungan yang paling konvensional untuk menjaga *data at rest* adalah dengan antivirus dan firewall. Namun, cara-cara tersebut tidak menjamin keamanan dari *phising* atau serangan *social engineering* yang dapat membuka celah serangan untuk mengakses data. Selain itu, ada juga *insider threats* yang tidak lepas menghantui keamanan *data at rest*.

Dengan begitu, salah satu cara terbaik dan termudah untuk memproteksinya adalah dengan mengimplementasikan pengenkripsian. Menurut [2], ada empat level dimana pengenkripsian *data at rest* dapat diimplementasikan. Empat level tersebut terdiri atas:

1) Application-level encryption

Pada level ini, aplikasi/perangkat lunak yang melakukan modifikasi atau memproduksi data akan menerapkan enkripsi untuk setiap data yang digunakan

2) Database encryption

Sebagian atau seluruh basis data dienkripsi untuk menjaga keamanannya

3) File system encryption

Pada tipe ini, pengenkripsian dilakukan hanya untuk file system terpilih atau bahkan beberapa *folder* di dalam file system. Siapa saja dapat mengaktifkan perangkat namun pengaksesan data pada *file system* yang terproteksi akan membutuhkan sandi atau kunci tertentu

4) Full disk encryption

Level ini merupakan level tertinggi dengan pengenkripsian dilakukan seluruhnya pada *hard drive*. Hal ini membuat data data disimpan dalam bentuk yang tidak jelas.

D. Ransomware

Berdasarkan [3], *ransomware* merupakan salah satu jenis *malware* yang dapat menghalangi akses ke perangkat dan data yang tersimpan di dalamnya, umumnya dengan mengenkripsi file-file yang dimiliki. Penyerang kemudian meminta tebusan untuk kita dapat memiliki akses kembali. Tidak jarang, penyerang mengancam akan menyebarkan data yang dicurinya.

E. AES (Advanced Encryption Standard)

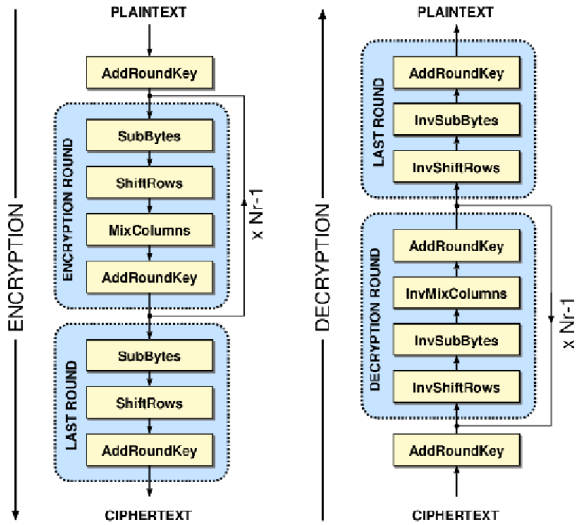


Fig. 3. The basic AES-128 architecture, sumber: ResearchGate

Menurut [4] Algoritma Enkripsi AES (Advanced Encryption Standard) (juga dikenal sebagai algoritma Rijndael) adalah algoritma blok simetris dengan ukuran blok/chunk sebesar 128 bit. Algoritma ini mengubah blok-blok individu ini menggunakan kunci berukuran 128, 192, dan 256 bit. Setelah mengenkripsi blok-blok ini, algoritma menggabungkannya menjadi ciphertext.

Algoritma ini didasarkan pada jaringan substitusi-permutasi, yang juga dikenal sebagai jaringan SP. Algoritma ini terdiri dari serangkaian operasi, yang terdiri atas subtitusi dan permutasi

III. RANCANGAN DAN IMPLEMENTASI

Seperti yang telah disebutkan sebelumnya, *ransomware* mencuri dan mengunci data pada suatu perangkat dengan menanamkan sebuah perangkat lunak. Data-data yang dicuri kemungkinan besar akan digunakan sebagai bahan ancaman untuk disebar. Data-data yang bersifat rahasia seperti data pengguna, dokumen keuangan, dokumen pernjanjian, akan sangat menimbulkan kerugian apabila tersebar. Oleh sebab itu, penulis menawarkan sebuah solusi berupa program untuk mengenkripsi *data at rest* pada level *file system.* Program akan digunakan layaknya *file explorer* namun memiliki kemampuan untuk mengenkripsi dan mendekripsi fail-fail yang ada. Dengan demikian, pengaksesan filesystem tanpa program dan kunci yang benar hanya akan menampilkan data-data tidak berarti. Harapannya, hal ini dapat mencegah resiko kebocoran data oleh program berbahaya *ransomware* karena data yang tercuri tidak memiliki makna karena terenkripsi.

A. Rancangan Solusi

Berikut adalah rancangan fungsionalitas program yang hendak dibuat:

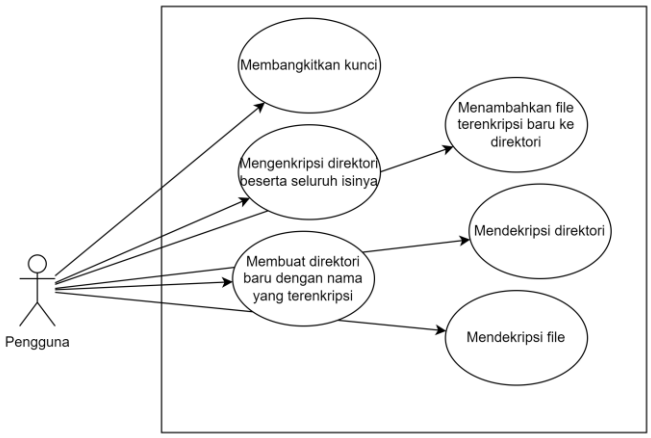


Fig. 4. Use Case Diagram, sumber: dokumen penulis

Fungsionalitas program digambarkan pada Fig 3. Pengguna dapat membangkitkan kunci yang akan digunakan untuk proses enkripsi dan dekripsi. Selanjutnya pengguna dapat memilih sebuah *folder* untuk dienkripsi beserta seluruh anak folder dan dokumen yang terdapat di dalamnya. Dengan begitu, pengaksesan *folder* tersebut tanpa program akan menampilkan nama-nama atau dokumen-dokumen tidak bermakna. Lalu, layaknya program *file explorer*, pengguna dapat membuat folder baru di path tertentu, pengguna akan memasukkan nama folder lalu program akan membuat folder dengan nama yang sudah terenkripsi. Selain itu pengguna dapat menambahkan file baru ke direktori, file akan secara otomatis dienkripsi oleh program. Program memiliki fitur untuk mendekripsi kembali suatu direktori beserta seluruh isinya jika pengguna ingin berhenti menggunakan program. Selanjutnya program dapat mendekripsi file tertentu agar file dapat diakses oleh pengguna.

B. Batasan dan Asumsi

Berikut ini adalah batasan dan asumsi yang digunakan dalam proses implementasi pembangunan program:

1) Program dibangun menggunakan bahasa pemrograman Python dengan pertimbangan kemudahan pengembangan dan bantuan pustaka yang mumpuni untuk melakukan proses enkripsi, dekripsi, dan pembangkitan kunci

2) Proses enkripsi dan dekripsi dilakukan menggunakan bantuan pustaka cryptography.fernet yang menggunakan algoritma AES (algoritma Rjindael) yang merupakan *symmetric block cipher*. Jenis yang digunakan adalah CBC dengan panjang kunci 128-bit serta menggunakan PKCS7 *padding*.

3) Pengguna mengamankan kunci yang dibangkitkan sehingga tidak dapat diketahui penyerang.

4) Program yang diimplementasikan pada penulisan makalah ini bersifat prototipe sehingga tidak menutup kemungkinan bahwa *interface* belum terlalu baik.

C. Implementasi

Program *file explorer* dibangun adalah program berbasis CLI *(command line interface)* dengan menggunakan bahasa pemrograman Python. Interaksi atau aksi yang dapat dilakukan oleh pengguna dibuat mirip *filesystem commands* – seperti cd, mkdir, dan ls—dengan beberapa tambahan beberapa perintah baru untuk operasi enkripsi-dekripsi. Walaupun sebatas CLI, penulis berharap program dapat menggambarkan secara jelas ide yang ingin digagas pada makalah ini. Detail terkait perintah yang dapat digunakan pengguna untuk pengaksesan dan pengamanan data dapat dilihat pada tabel di bawah ini.

TABLE I. TABEL DAFTAR PERINTAH

Perintah	Fungsionalitas
cwd	Menampilkan *current working directory* untuk mengetahui *path* dimana semua perintah dijalankan.
secret	Untuk mengaktifkan mode rahasia. Dengan mode rahasia, program dapat menampilkan *files* terenkripsi dan *folders* dengan nama yang terenkripsi selayaknya *file system* biasa pada kondisi *file explorer* biasa hanya menampilkan berkas dan direktori tidak bermakna.
ls	Menampilkan semua fail dan direktori yang terdapat pada *current working directory*.
mkdir <directory-name>	Membuat direktori baru di *file system* pada *cwd*. Nama direktori akan otomatis terenkripsi apabila mode rahasia aktif.
cd [..] [<directory-name>]	Memindahkan *cwd* ke direktori yang dimasukkan ke directory-name. *cd ..* akan memindahkan *cwd* ke *parent directory* dari *cwd.*
encdir [.] [<directory-name>]	Mengenkripsi isi dari sebuah *folder* bernama *directory-name* di *cwd*. Pengenkripsian dilakukan secara mengakar menggunakan rekursif sehingga seluruh folder terdapat di dalamnya akan terenkripsi beserta dengan *file-file* yang dikandungnya. Encdir . sama dengan encdir *cwd.*
decdir [.] [<directory-name>]	Kebalikan dari encdir, decdir berfungsi untuk mendekripsi isi *folder*. Pendekripsian juga dilakukan secara mengakar
encfile <file-name>	Mengenkripsi satu file dengan nama file sesuai argumen. Dengan begitu, file akan berubah jadi tidak bermakna secara nama dan isi binary jika diakses menggunakan *file explorer* biasa.
decfile <file-name>	Mendekripsi kembali file yang terenkripsi untuk dapat digunakan kembali oleh pengguna.

Selain perintah-perintah di atas, pengguna dapat menentukan *current working directory* pertama saat memulai program. Selain itu, pengguna dapat memasukkan kunci simetri yang diasumsikan sudah disimpan oleh pengguna. Apabila belum memiliki kunci, pengguna dapat memilih untuk membangkitkan kunci simetri untuk digunakan dalam proses enkripsi maupun dekripsi.

IV. HASIL DAN PEMBAHASAN

A. Inisialisasi Pengujian

Pada tahap pengujian, program diasumsikan akan menggunakan *path* "D:\Test Makalah\" untuk dijadikan *cwd* pertama. Adapun isi dari direktori tersebut dengan struktur sebagai berikut.

```
Test Makalah/
├── Data Transaksi/
│   ├── 2022/
│   │   ├── Januari_2022.csv
│   │   ├── Februari_2022.csv
│   │   ├── Maret_2022.csv
│   │   ├── ...
│   │   └── Desember_2022.csv
│   ├── 2023/
│   │   ├── Januari_2023.csv
│   │   ├── Februari_2023.csv
│   │   ├── Maret_2023.csv
│   │   └── April_2023.csv
│   └── Summary Transaksi 2022.pdf
├── Data_pelanggan.csv
├── Draft Proposal Marketing.doc
├── MOU distributor.pdf
├── Paycheck Pegawai.xlsx
├── POS-Setup.exe
├── Rekap_Keuangan.xlsx
└── Sosialisasi UMKM.pptx
```

Selain itu, program diasumsikan akan digunakan untuk pertama kali sehingga pengguna belum memiliki kunci simetri sama sekali. Dengan begitu, kita perlu memasang *cwd* pertama dan membangkitkan kunci sebelum menggunakan program. Proses keduanya dilakukan sebagai berikut.

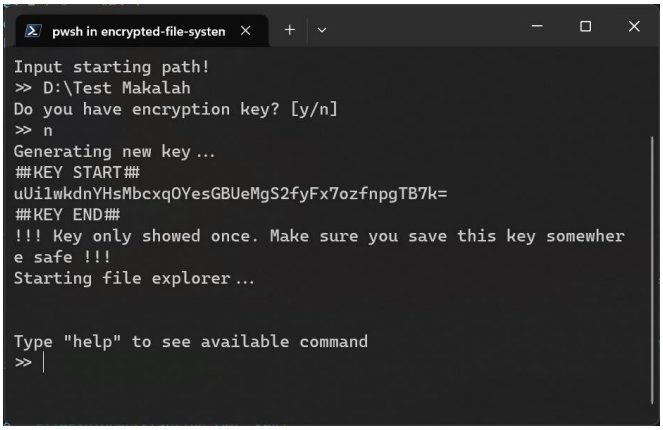


Fig. 5. Penetapan *cwd* pertama dan pembangkitan kunci, sumber: dokumen penulis

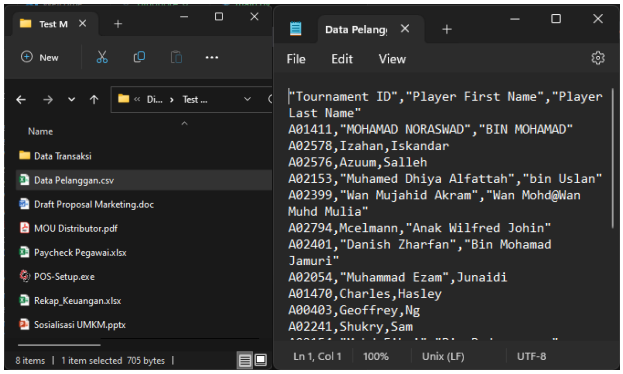
Pada titik ini diasumsikan pengguna mencatat dan menyimpan kuncinya dengan aman karena kunci tidak dapat ditampilkan kembali kemudian hari.

B. Pengujian dan Analisis

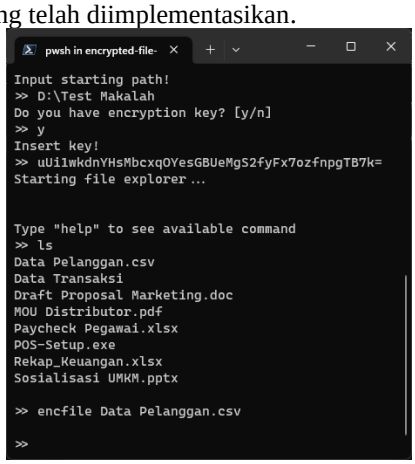
Setelah mendapatkan kunci enkripsi, maka proses pengamanan data sudah bisa dilakukan. Terdapat x kasus uji yang akan dilakukan terhadap program yang sudah diimplementasikan sebelumnya.

1) Kasus 1: Pengamanan File

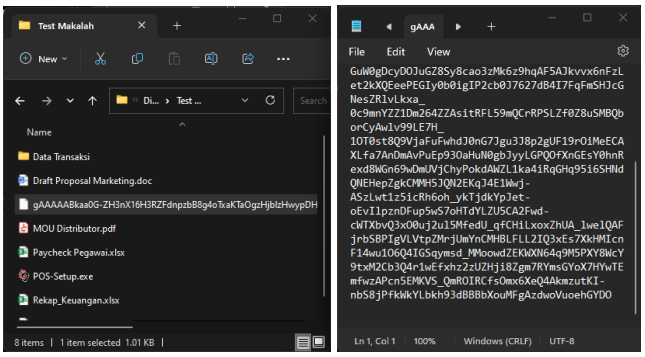
Kasus ini merupakan kasus normal yang akan terjadi apabila pengguna ingin mengenkripsi data yang disimpannya. Sebagai contoh, kita akan mengenkripsi sebuah file "Data Pelanggan.csv". Berikut ini adalah tangkapan layar dari isi file sebelum dienkripsi serta isi dari file explorer biasa.



Selanjutnya kita akan mengenkripsi file tersebut menggunakan program yang telah diimplementasikan.

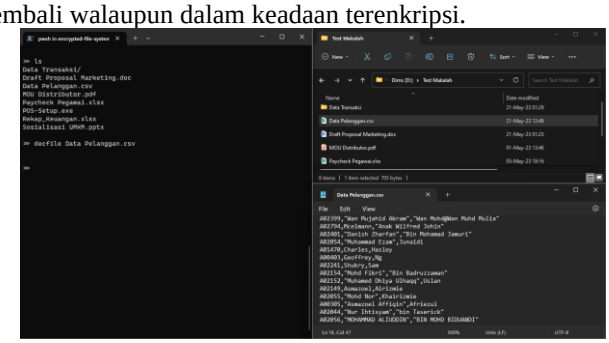


Setelah program selesai mengenkripsi, kita dapat memeriksanya dengan menjalankan ls ataupun membuka file explorer biasa.



Dua gambar di atas memperlihatkan bahwa file "Data Pelanggan.csv" telah terenkripsi sepenuhnya dari isi file beserta dengan nama file. Dengan begitu, keberadaan data dan isi data tidak akan diketahui oleh pihak yang tidak berwenang. Sebagai pihak yang berwenang dan memiliki kunci yang benar, kita masih dapat melihat keberadaan file "Data Pelanggan.csv". Hal tersebut dapat dilakukan dengan mengaktifkan mode secret pada program *encrypted file explorer*.

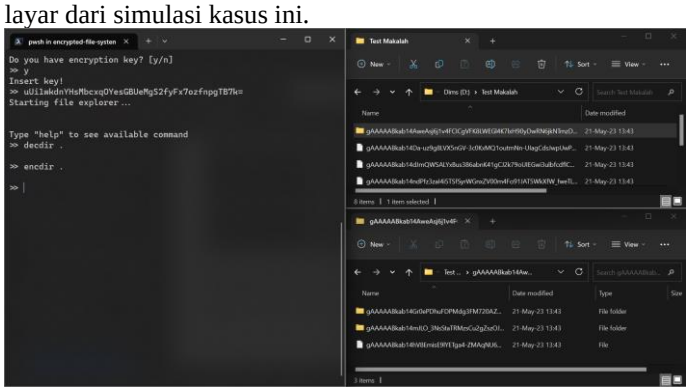
Gambar di atas menunjukkan perbedaan keluaran dari perintan ls pada saat sebelum dan sesudah secret mode diaktifkan. Sebelum secret mode diaktifkan, file yang tertampil tidak berbeda dengan file explorer biasa. Namun dengan secret mode yang aktif, file Data Pelanggan.csv dapat terlihat kembali walaupun dalam keadaan terenkripsi.



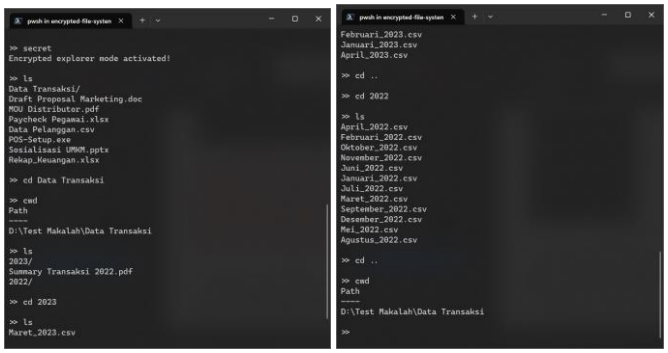
Pendekripsian dapat dilakukan dengan menjalankan perintah decfile Data Pelanggan.csv. Setelah didekripsi, file asli "Data Pelanggan.csv" akan terlihat di file explorer biasa beserta dengan isi file.

2) Kasus 2: Pengamanan Direktori

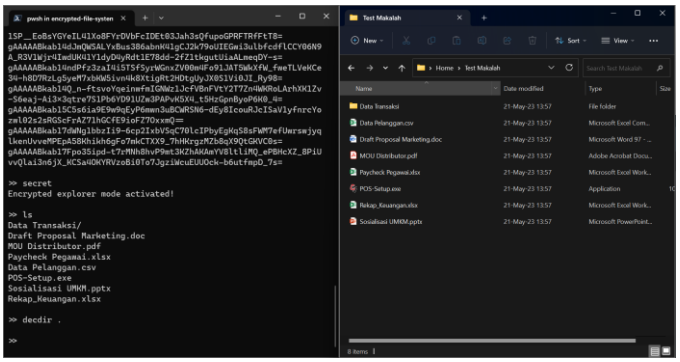
Untuk kasus sebelumnya, pengenkripsian dilakukan untuk satu file saja. Pada kasus ini, kita ingin mengamankan banyak file dan struktur folder dengan satu perintah saja. Kita akan menggunakan perintah “encdir .”. Dengan perintah tersebut, program akan mengenkripsi seluruh file yang ada di folder “Test Makalah” serta dengan folder-folder yang terdapat di dalamnya seperti file-file pada folder “Data Transaksi/2022” maupun “Data Transaksi/2023”. Berikut adalah tangkapan layar dari simulasi kasus ini.



Sebelumnya, *cwd* dari program adalah D:\Test Makalah. Pengeksekusian “encdir .” akan menghasilkan perubahan isi folder menjadi ke bentuk yang ditampilkan pada sisi kanan gambar. Nama dan isi dari file serta folder dienkripsi seluruhnya seluruhnya sehingga pengaksesan biasa hanya akan memberikan hasil tak bermakna. Walaupun begitu, kita masih dapat melakukan kegiatan *file exploring* normal dengan menggunakan program yang telah dibuat. Untuk melakukannya, kita perlu mengaktifkan mode secret terlebih dahulu.



Gambar di atas menunjukkan bahwa kegiatan seperti memindahkan *cwd*, melihat isi folder, dan menampilkan *cwd* masih dapat dilakukan secara normal dengan menggunakan program. Pada saat ingin menggunakan data, pengguna dapat memasukkan perintah untuk mendekripsi kembali data-data yang ada. Hal tersebut dapat dilakukan satu-satu untuk setiap file seperti pada kasus sebelumnya atau menggunakan perintah “decdir .” untuk pendekripsian secara rekursif.



Perintah “decdir .” pada direktori D:\Test Makalah akan mengembalikan kondisi folder seperti semula dengan file-file yang dapat dibuka kembali.

C. Analisis Secara Keseluruhan

Berdasarkan hasil pengujian yang ada, penulis menarik beberapa poin penting terkait pengamanan *data at rest* untuk *file system* dengan menggunakan algoritma AES serta eksplorasi *file system* terdedikasi untuk data terenkripsi. Beberapa poin penting tersebut adalah sebagai berikut.

- Algoritma AES dinilai cocok untuk *use case* ini karena memiliki kecepatan yang tinggi. Pencobaan enkripsi dan dekripsi file sebesar 100MB membutuhkan waktu tidak lebih dari 2 detik.
- Data berhasil disembunyikan maknanya. Enkripsi berhasil menyembunyikan makna file maupun folder. Pengembalian makna dengan dekripsi pun dapat dilakukan apabila memiliki program dengan kunci yang sesuai. Dengan begitu, pengaksesan oleh pihak yang tidak berwenang hanya akan memberikan hasil yang tidak bermakna
- Perintah dasar *file system* masih dapat dilakukan untuk data-data terenkripsi. Dengan menggunakan kunci yang benar, pengguna masih dapat melaksanakan perintah dasar seperti membuka folder dan membuat direktori selayaknya data tidak terenkripsi.

V. KESIMPULAN DAN SARAN.

Pengenkripsian *data at rest* dapat mengurangi efek negatif dari pengaksesan oleh penyusup karena maknanya yang tersembunyikan. Walaupun begitu pengguna yang berhak masih dapat mengakses seperti biasa jika memiliki kunci yang benar. Pada makalah ini, telah dibuktikan bahwa hal tersebut dapat dilakukan pada level *filesystem* dengan menggunakan program explorer khusus. Hal ini merupakan salah satu contoh penggunaan kriptografi pada kehidupan sehari-hari untuk perihal pengamanan data.

UCAPAN TERIMAKASIH

Penulis menguckapkan terima kasih banyak kepada bapak Dr. Rinaldi Munir, S.T, M.T. selaku dosen pengajar IF4020 Kriptografi yang telah memberikan bimbingan dan ilmu

terkait kriptografi beserta pengaplikasiannya di kehidupan sehari-hari. Sungguh ilmu yang diajarkan sangat bermanfaat dan berguna untuk bidang keinformatikaan.

REFERENSI

[1] Fitzgibbons, L. (2019, February 27). States of Digital Data. Data Management. https://www.techtarget.com/searchdatamanagement/reference/states-of-digital-data#:~:text=Three%20states%20of%20data%20is,life%20cycle%20of%20a%20computer.

[2] Velimirovic, A. (2023, March 7). *Data Encryption at rest explained*. phoenixNAP Blog. https://phoenixnap.com/blog/encryption-at-rest

[3] A guide to ransomware. NCSC. (n.d.). https://www.ncsc.gov.uk/ransomware/home#:~:text=What%20is%20ransomware%3F-,Ransomware%20is%20a%20type%20of%20malware%20which%20prevents%20you%20from,ransom%20in%20exchange%20for%20decryption.

[4] Jena, B. K. (2023, February 9). *What is AES encryption and how does it work? - simplilearn*. Simplilearn.com. https://www.simplilearn.com/tutorials/cryptography-tutorial/aes-encryption#:~:text=The%20AES%20Encryption%20algorithm%20(also,together%20to%20form%20the%20ciphertext.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi

Jakarta, 21 Mei 2023

Dimas Faidh Muzaki