JURNAL ILMU KOMPUTER
KHARISMA .TECH

Jurnal KHARISMA Tech | ISSN:1907-2317 | e-ISSN: 2810-0344
https://jurnal.kharisma.ac.id/kharismatech/
published by: Pusat Penelitian STMIK KHARISMA Makassar

Volume: 17 no. 01 – Maret 2022- hlm. 58-72

ENKRIPSI DATA PADA GAME SMARTPHONE SIMULASI HIDUP BERSIH DAN SEHAT MENGGUNAKAN UCS TRANSFORMATION FORMAT 8

Oleh:

Geraldi Radityo Wiwono[1*], Marlina[2], Izmy Alwiah Musdar[3]
[1,2,3]Teknik Informatika, STMIK Kharisma Makassar
e-mail: [1]geraldiradityo_18@kharisma.ac.id, [2]marlina@kharisma.ac.id,
[3]izmyalwiah@kharisma.ac.id

Abstrak: *Penelitian ini bertujuan untuk mengenkripsikan data simpanan pengguna pada Game Smartphone Simulasi Perilaku Hidup Bersih dan Sehat dengan UCS Transformation Format 8 menggunakan bahasa C# di platform Unity dan Visual Studio 2019. Metode yang digunakan dalam penelitian ini adalah metode eksperimen untuk enkripsi dan dekripsi simpanan data pengguna yang berupa nama(string), skor(float), total skor(float), nyawa(float), dan kategori yang terbuka(float). Hasil dari penelitian ini menunjukkan bahwa kode enkripsi dan dekripsi yang dibuat telah berjalan dengan optimal. Simpanan data telah berhasil dienkripsi sehingga data tidak dapat dilihat dan dibaca oleh pihak yang tidak berwenang. Ukuran simpanan data yang telah dienkripsi bertambah sekitar 105-110 bytes.*

Kata kunci: *kriptografi,UTF-8, game, smartphone, android, enkripsi, dekripsi*

Abstract: *This study aims to encrypt user saved data on a Smartphone Game Simulasi Hidup Bersih dan Sehat with UCS Transformation Format 8 using C# language on Unity and Visual Studio 2019 platforms. The method used in this study is an experimental method for encryption and decryption of user stored data. in the form of name (string), score (float), total score (float), lives (float), and unlocked categories (float). The results of this study indicate that the created encryption and decryption codes have run optimally. The stored data has been successfully encrypted so that the data cannot be seen and read by unauthorized parties. The size of the encrypted data store increased by about 105-110 bytes.*

Keywords: *kriptography, UTF-8, game, smartphone, android, encryption, decryption*

1. PENDAHULUAN

Setiap aplikasi mobile, rumit maupun sederhana membutuhkan keamanan minimal untuk dapat digunakan user dan developer dengan aman. Keamanan tersebut dapat dicapai dengan mengenkripsi data. Menurut National Institute of Standards and Technology (NIST), enkripsi data adalah transformasi kriptografik data (disebut "plaintext") ke dalam bentuk (disebut "ciphertext") yang menyembunyikan makna asli data untuk mencegahnya diketahui atau digunakan. Sistem enkripsi mengacak data sensitif menggunakan kalkulasi matematis untuk mengubah data menjadi kode. Data asli hanya dapat diungkapkan dengan kunci yang benar, sehingga tetap aman dari semua orang kecuali pihak yang berwenang [1].

[*] Corresponding author : Geraldi Radityo Wiwono (geraldiradityo_18@kharisma.ac.id)

Diterima : *Februari, Tahun* ***Disetujui :*** *Februari, 2022* ***Dipublikasikan :*** *Mei, 2022*

Dalam penelitian ini, peneliti akan menggunakan UTF-8 untuk mengenkripsi data simpanan pengguna di game Simulasi Edukasi Perilaku Hidup Bersih dan Sehat. Game berupa kuis-kuis seputar keseharan umum dan perilaku sehari-hari yang benar sesuai dengan himbauan Kementrian Kesehatan tentang Perilaku Hifup Bersih dan Sehat (PHBS) [2]. Pengguna akan memulai akan memulai game dengan 3 kategori kuis awal yang dapat dimainkan. Seiring pengguna memainkan dan menyelesaikan quis-quis yang ada, akan terbuka kategori-kategori lainnya sampai sebanyak 12 kategori yang dapat dibuka sesuai dengan skor tertinggi dan skor total pengguna. Data simpanan kategori dan skor pengguna inilah yang dienkripsi dengan UTF-8 agar pengguna lebih sulit untuk membaca dan mengubah nilai-nilainya. Data simpanan pengguna yang akan dienkripsi adalah nama pengguna berformat string, skor terakhir pengguna berformat float, skor tertinggi pengguna berformat float, dan jumlah kategori yang sudah terbuka berformat float.

UTF-8 adalah singkatan dari UCS Transformation Format 8 (UTF-8). UCS adalah singkatan dari Universal Coded Character Set atau singkatnya Unicode. Unicode adalah pengkodean karakter universal, dikelola oleh Unicode Consortium [7]. Agar komputer dapat menyimpan teks dan angka yang dapat dipahami manusia, diperlukan suatu kode yang mengubah karakter menjadi angka. Standar Unicode mendefinisikan kode poin menggunakan pengkodean karakter. Alasan mengapa pengkodean karakter sangat penting adalah agar setiap perangkat dapat menampilkan informasi yang sama. Standar pengkodean ini memberikan dasar untuk pemrosesan, penyimpanan, dan pertukaran data teks dalam bahasa apa pun di semua perangkat lunak modern dan protokol teknologi informasi [9]. Unicode mencakup semua karakter untuk semua sistem penulisan dunia, modern dan kuno. Ini juga mencakup simbol teknis, tanda baca, dan banyak karakter lain yang digunakan dalam menulis teks. Standar Unicode dimaksudkan untuk mendukung kebutuhan semua jenis pengguna, baik dalam bisnis atau akademis, menggunakan skrip mainstream atau minoritas [7]. Tetapi ada beberapa masalah dengan ini, misalnya [9]:

1. Untuk mengkodekan 100000 karakter, diperlukan sekitar 32 digit biner. Unicode menggunakan ASCII untuk bahasa Inggris, jadi A masih 65. Namun, dikodekan dalam 32 bit, representasi biner untuk huruf A akan menjadi 00000000000000000000000001000001. Ini memakan banyak ruang [9].
2. Banyak komputer lama menafsirkan delapan nol berturut-turut (nol) sebagai akhir dari serangkaian karakter. Jadi komputer ini tidak akan mengirim karakter apa pun yang muncul setelah delapan angka nol berturut-turut (komputer tidak akan mengirim A jika direpresentasikan sebagai 00000000000000000000000000000001) [9].

Metode penyandian Unicode UTF-8 memecahkan masalah ini. Hingga karakter nomor 128, nilai American Standard Code for Information Interchange (ASCII) biasa digunakan (jadi misalnya A adalah 01000001). ASCII adalah 128 huruf, angka, dan tanda baca standar dalam penulisan bahasa inggris [10]. Untuk karakter apa pun di luar 128, UTF-8 memisahkan kode menjadi dua byte dan menambahkan '110' ke awal byte pertama untuk

ditampilkan bahwa itu adalah byte awal, dan '10' ke awal byte kedua untuk menunjukkan bahwa itu mengikuti byte pertama [9].

Jadi, untuk setiap karakter di luar angka 128, penyandian memiliki dua byte:

```
[110xxxxx] [10xxxxxx]
```

Dan cukup dengan mengisi biner untuk nomor di antaranya:

```
[11000101] [10000101] (itu adalah angka 325 → 00101000101)
```

Ini berfungsi untuk 2048 karakter pertama. Untuk karakter di luar itu, satu lagi '1' ditambahkan di awal byte pertama dan byte ketiga juga digunakan:

```
[1110xxxx] [10xxxxxx] [10xxxxxx]
```

Ini memberi 16 spasi untuk kode biner. Dengan cara ini, UTF-8 naik hingga empat byte:

```
[11110xxx] [10xxxxxx] [10xxxxxx] [10xxxxxx]
```

Dengan cara ini, UTF-8 menghindari masalah yang disebutkan di atas dan memungkinkan untuk memecahkan kode karakter dari bentuk biner secara mundur (*backward compatible*) [8]. Dikarenakan hal-hal tersebut, UTF-8 adalah encoding yang paling banyak digunakan di web dengan persentase sebesar 92,4% [12]. Diharapkan setelah game telah di-enkripsi dengan pengkodean UTF-8, data-data game seperti akun, simpanan data, kode aplikasi dan data-data pengguna seperti nama user, email, password akan terenkripsi dengan baik, sehingga tidak dapat diubah, dicuri, maupun dirusakkan oleh pengguna yang tidak berwenang.

2. METODE PENELITIAN

2.1 Tahapan Penelitian

Tahapan penelitian yang digunakan dalam penelitian ini adalah sebagai berikut.

a. Perumusan Masalah

Bagaimana meng-enkripsi simpanan data pengguna game smartphone Simulasi Edukasi Hidup Bersih dan Sehat menggunakan UCS Transformation Format dilakukan.

b. Hipotesis Masalah

Data game smartphone Simulasi Edukasi Hidup Bersih dan Sehat dapat di-enkripsi dengan metode UCS Transformation Format 8 menggunakan C# di unity yang kemudian akan ditambahkan beberapa baris kode.

c. Tujuan Penelitian

Meng-enkripsi simpanan data pengguna game smartphone Simulasi Edukasi Hidup bersih dan Sehat dengan UCS Transformation Format 8.

d. Pengumpulan Data

Dalam penelitian ini, proses pengumpulan data dilakukan dengan melakukan studi literatur, di mana penulis mempelajari berbagai bahan baik dalam bentuk buku, jurnal, prosiding, website, makalah yang berkaitan dengan apa yang penulis teliti.

e. Pengolahan Data

Metode yang digunakan dalam mengolah data adalah metode experimen. Data yang telah dikumpulkan dari hasil dokumen akan disusun dan diolah berdasarkan kecocokan dan kesamaan data dengan penelitian yang dilakukan penulis.

f. Penarikan Kesimpulan

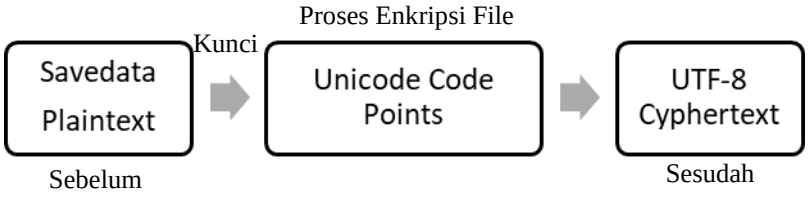
Penarikan kesimpulan dilakukan dengan menguji enkripsi yang telah dibuat dan diterapkan di game. Pengujian akan dilakukan sebelum dan sesudah diterapkannya enkripsi dari sisi pengembang dan dari sisi pengguna. Pengembang akan langsung mengakses hasil enkripsi menggunakan platform pengembangan game dan pengguna akan mengakses hasil enkripsi melalui Windows.

2.2 Deskripsi Singkat Penelitian

Pada penelitian ini akan dibuat kelas baru yang berisi dua fungsi untuk Game Simulasi Perilaku Hidup Bersih dan Sehat. Fungsi tersebut adalah fungsi untuk melakukan proses enkripsi dan dekripsi data. Setelah dibuatnya kedua fungsi tersebut, fungsi akan dipanggil dan diaplikasikan di data simpanan pengguna.

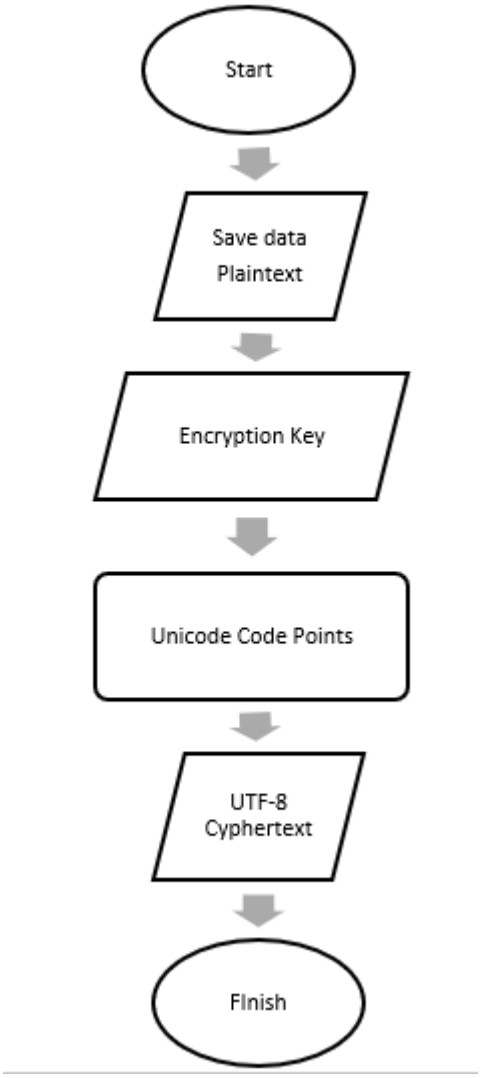
2.3 Proses Enkripsi File

Pada tahap ini merancang fungsi untuk mengenkripsi data dengan UTF-8. Dibawah ini merupakan diagram blok untuk proses enkripsi file.



Gambar 1. Diagram Blok Proses Enkripsi File

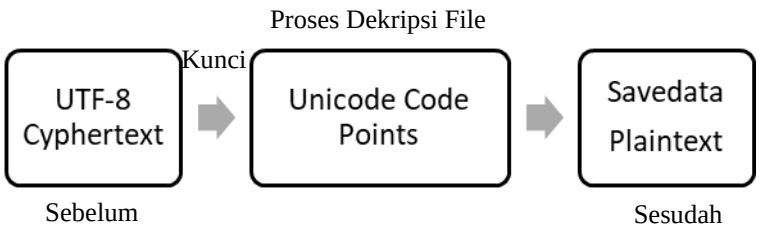
Gambar 2 memperlihatkan flowchart proses enkripsi file secara keseluruhan. Untuk melakukan proses enkripsi file hal pertama yang dilakukan adalah membaca plainteks berupa data save yang telah disimpan oleh game.



Gambar 2. Flowchart Proses Enkripsi File

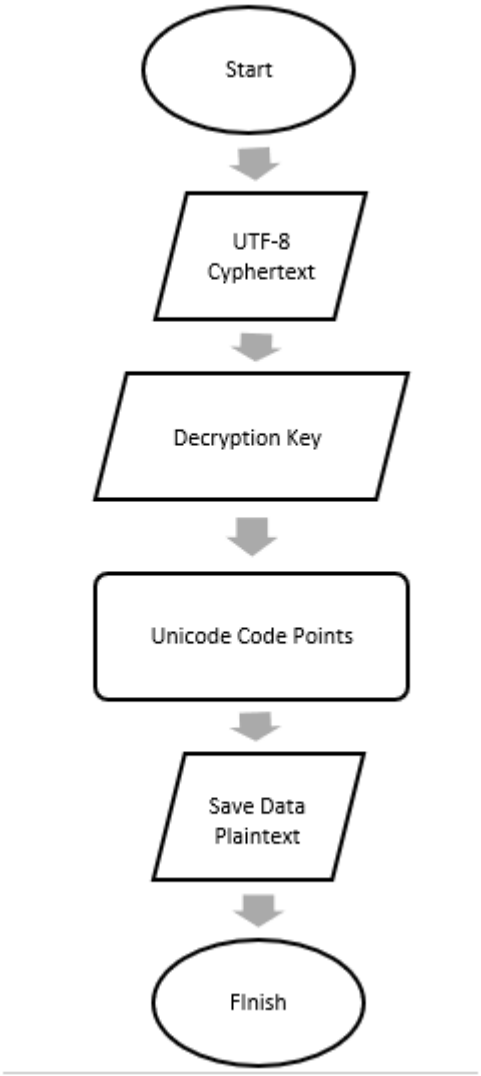
2.4 Proses Dekripsi File

Pada tahap ini merancang fungsi untuk dekripsi data UTF-8 ke data semula. Gambar 3 merupakan diagram blok untuk proses dekripsi file.



Gambar 3. Diagram Blok Proses Dekripsi File

Gambar 4 memperlihatkan flowchart proses dekripsi file secara keseluruhan. Untuk melakukan proses dekripsi file hal pertama yang dilakukan adalah membaca cipertext berupa data save terenkripsi yang telah disimpan oleh game.



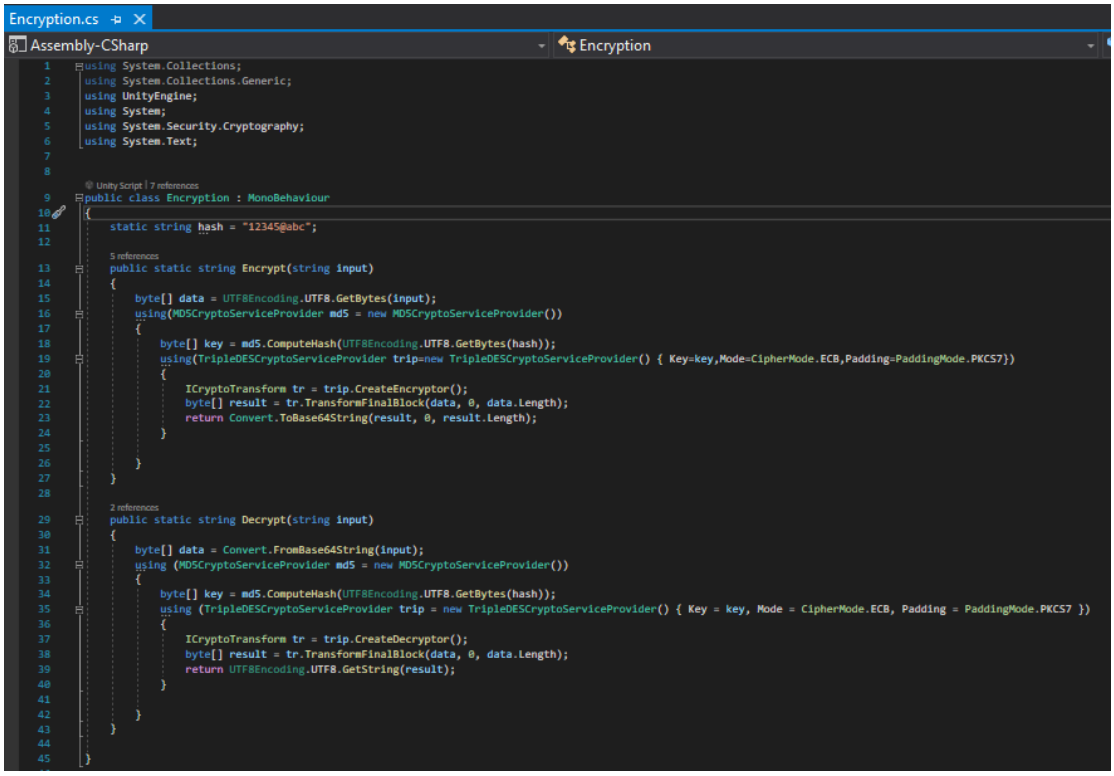
Gambar 4. Flowchart Proses Dekripsi File

2.5 Perancangan Kode Program

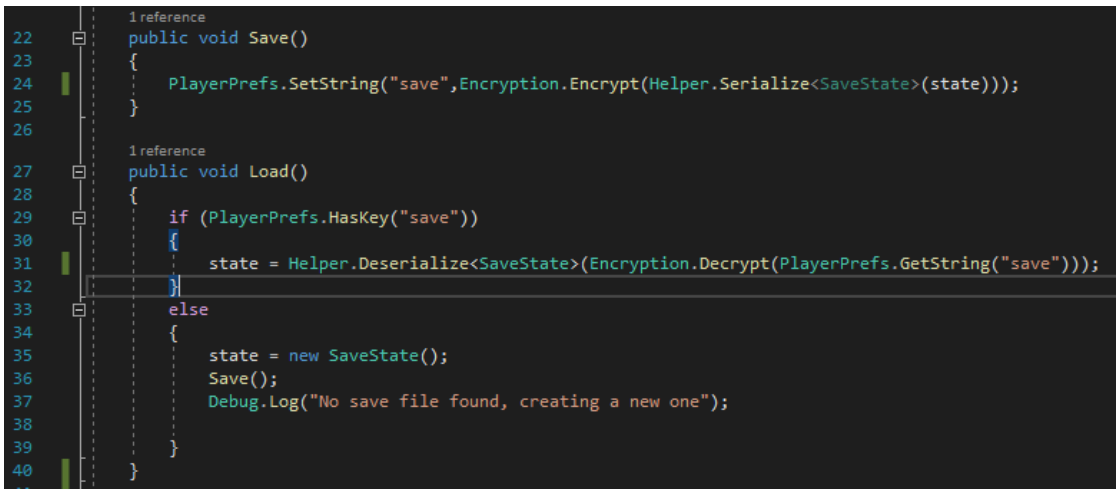
Pada bagian ini, akan dirancang kode program yang akan digunakan untuk enkripsi dan dekripsi data game pengguna.

Pada Gambar 5, adalah sisipan kode program cara merancang dan membuat enkripsi dan dekripsi di Unity menggunakan Bahasa pemograman C# dibuka di Visual Studio 2019. Berikut adalah Langkah-langkahnya:

1. Membuat Script Baru di Assets -> Create -> C# Script.
2. Menambahkan Library yang dibutuhkan.
3. Membuat kunci.
4. Membuat Kelas enkripsi dan isinya.
5. Membuat kelas dekripsi dan isinya.



Gambar 5. Koding kelas enkripsi dan dekripsi menggunakan C#



Gambar 6. Kode Penerapan Enkripsi dan Dekripsi di Save dan Load menggunakan C#

Pada Gambar 6, adalah sisipan kode program untuk memanggil dan menerapkan enkripsi dan dekripsi pada simpanan data pengguna di Unity menggunakan Bahasa pemograman C# dibuka di Visual Studio 2019.

3. HASIL DAN PEMBAHASAN

3.1 Pengujian Kelas Enkripsi dan Dekripsi

a. Pengujian Terhadap string

Pada bagian ini dilakukan pengujian untuk mengenkripsi string dan setelah proses enkripsi string selesai dilakukan akan dilihat hasilnya kemudian dilakukan pengujian dekripsi

untuk mengembalikan string seperti semula. Pengujian akan dilakukan di kelas Test yang dibuka di Visual Studio 2019 dan consolenya akan dilihat di Unity.

1. Pengujian Enkripsi

Pengujian dilakukan dengan memasukkan perintah Debug.Log untuk memunculkan perintah di console, dan ditambahkan kelas enkripsi yang telah dibuat yaitu Encyption.Encrypt. Tujuan pengujian untuk memastikan apakah kelas enkripsi yang dibuat telah bekerja. Pengujian dilakukan dengan format dan panjang string yang berbeda-beda. Pengujian dilakukan sebanyak lima kali. Hasil pengujian dapat dilihat di Tabel 1.

Tabel 1: Pengujian Enkripsi terhadap string menggunakan C# di Visual Studio 2019

String	Hasil Enkripsi
STMIK	/vIWA59gLeg=
STMIK Kharisma	rE1u5nSOImk32iKKB7HylA==
STMIK Kharisma Makassar	rE1u5nSOImn+WtT+1INH45kzdy5lCdEZ
STMIK Kharisma Makassar 2019	rE1u5nSOImn+WtT+1INH4wxzf9qiwgjEVq3bMGONQVg=
STMIK Kharisma Makassar 2019 Teknik Informatika @@@	rE1u5nSOImn+WtT+1INH4wxzf9qiwgjEnUEZwOmu8omPso+axZBeQiG71xdo6KKw7PHy82Kp7ZI=

2. Pengujian Dekripsi

Pengujian dilakukan dengan memasukkan perintah Debug.Log untuk memunculkan perintah di console, dan ditambahkan kelas dekripsi yang telah dibuat yaitu Encyption.Encrypt. Tujuan pengujian untuk memastikan apakah kelas dekripsi yang dibuat telah bekerja. Pengujian dilakukan dengan format enkripsi UTF-8 dalam string yang merupakan hasil dari pengujian enkripsi. Pengujian dilakukan sebanyak lima kali. Hasil pengujian dapat dilihat di Tabel 2.

Tabel 2: Pengujian kelas dekripsi terhadap hasil enkripsi menggunakan C# di Visual Studio 2019

UTF-8	Hasil Dekripsi
/vIWA59gLeg=	STMIK
rE1u5nSOImk32iKKB7HylA==	STMIK Kharisma
rE1u5nSOImn+WtT+1INH45kzdy5lCdEZ	STMIK Kharisma Makassar
rE1u5nSOImn+WtT+1INH4wxzf9qiwgjEVq3bMGONQVg=	STMIK Kharisma Makassar 2019
rE1u5nSOImn+WtT+1INH4wxzf9qiwgjEnUEZwOmu8omPso+axZBeQiG71xdo6KKw7PHy82Kp7ZI=	STMIK Kharisma Makassar 2019 Teknik Informatika @@@

b. Pengujian Terhadap Angka dan Desimal

Pada bagian ini dilakukan pengujian untuk mengenkripsi angka dan desimal dan setelah proses enkripsi string selesai dilakukan akan dilihat hasilnya kemudian dilakukan pengujian dekripsi untuk mengembalikan angka dan desimal seperti semula. Pengujian akan dilakukan di kelas Test yang dibuka di Visual Studio 2019 dan consolenya akan dilihat di Unity.

1. Pengujian Enkripsi

Pengujian dilakukan dengan memasukkan perintah Debug.Log untuk memunculkan perintah di console, dan ditambahkan kelas enkripsi yang telah dibuat yaitu Encyption.Encrypt. Tujuan pengujian untuk memastikan apakah kelas enkripsi yang dibuat telah bekerja. Pengujian dilakukan dengan angka dan desimal yang berbeda-beda. Pengujian dilakukan sebanyak lima kali. Hasil pengujian dapat dilihat di Tabel 3.

Tabel 3. Pengujian kelas enkripsi terhadap angka dan desimal menggunakan C# di Visual Studio 2019

UTF-8	Hasil Dekripsi
axUaBlBVbRI=	0
8HV3vl0ZF9o=	1000
Mafp3LezSFA=	1234
BSXQnKeJG6Y=	1,234
FBFKuKKBigE=	12,34

2. Pengujian Dekripsi

Pengujian dilakukan dengan memasukkan perintah Debug.Log untuk memunculkan perintah di console, dan ditambahkan kelas dekripsi yang telah dibuat yaitu Encyption.Encrypt. Tujuan pengujian untuk memastikan apakah kelas dekripsi yang dibuat telah bekerja. Pengujian dilakukan dengan format enkripsi UTF-8 dalam string yang merupakan hasil dari pengujian enkripsi. Pengujian dilakukan sebanyak lima kali. Hasil pengujian dapat dilihat di Tabel 4

.

Tabel 4. Pengujian kelas dekripsi terhadap angka dan desimal menggunakan C# di Visual Studio 2019

Angka dan Decimal	Hasil Enkripsi
0	axUaBlBVbRI=
1000	8HV3vl0ZF9o=
1234	Mafp3LezSFA=
1,234	BSXQnKeJG6Y=
12,34	FBFKuKKBigE=

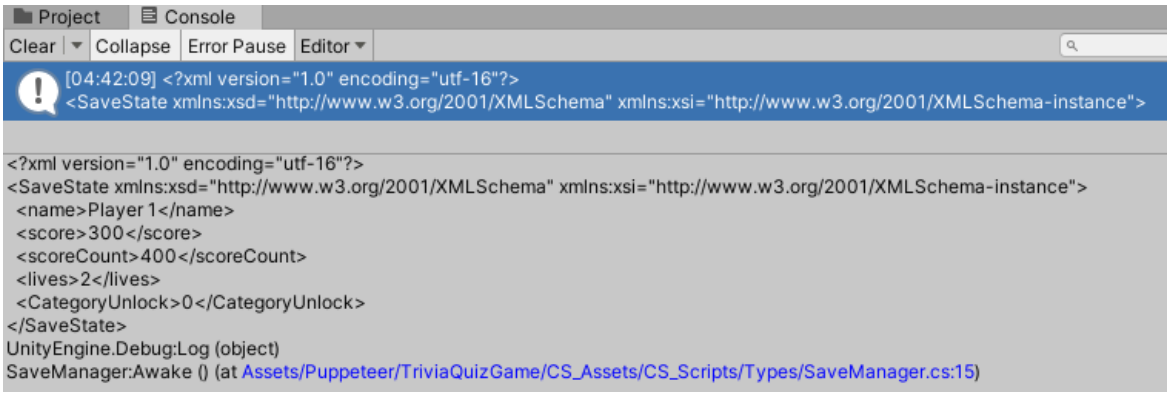
c. Pengujian terhadap simpanan data pengguna

Koding enkripsi dan dekripsi akan diuji dan diterapkan di aplikasi game PHBS. Penerapan akan dilakukan di platform Windows 10, menggunakan Unity2D dan Visual Studio 2019 untuk debugging. Penguji akan memperlihatkan hasil enkripsi simpanan data pengguna sebelum dan sesudah dienkripsi. Hasil enkripsi dan dekripsi akan diperlihatkan dari dua hak akses yang berbeda yaitu pengembang dan pengguna.

1. Pengujian Enkripsi Simpanan Data Pengguna dari Sisi Pengembang

Pengembang dapat mengakses langsung data pengguna saat mengembangkan game di Unity dengan menggunakan perintah Debug untuk menampilkannya di console.

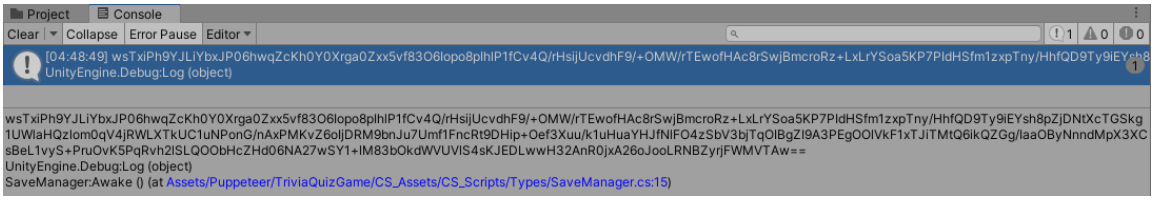
a. Sebelum Dienkripsi



Gambar 7. Tampilan console simpanan data pengguna untuk pengembang sebelum dienkripsi

Pada Gambar 7, dapat dilihat bahwa data pengguna belum terenkripsi, sehingga data simpanan pengguna seperti nama, skor, skor tertinggi, nyawa, dan kategori yang terbuka dapat dilihat secara langsung.

b. Setelah Dienkripsi



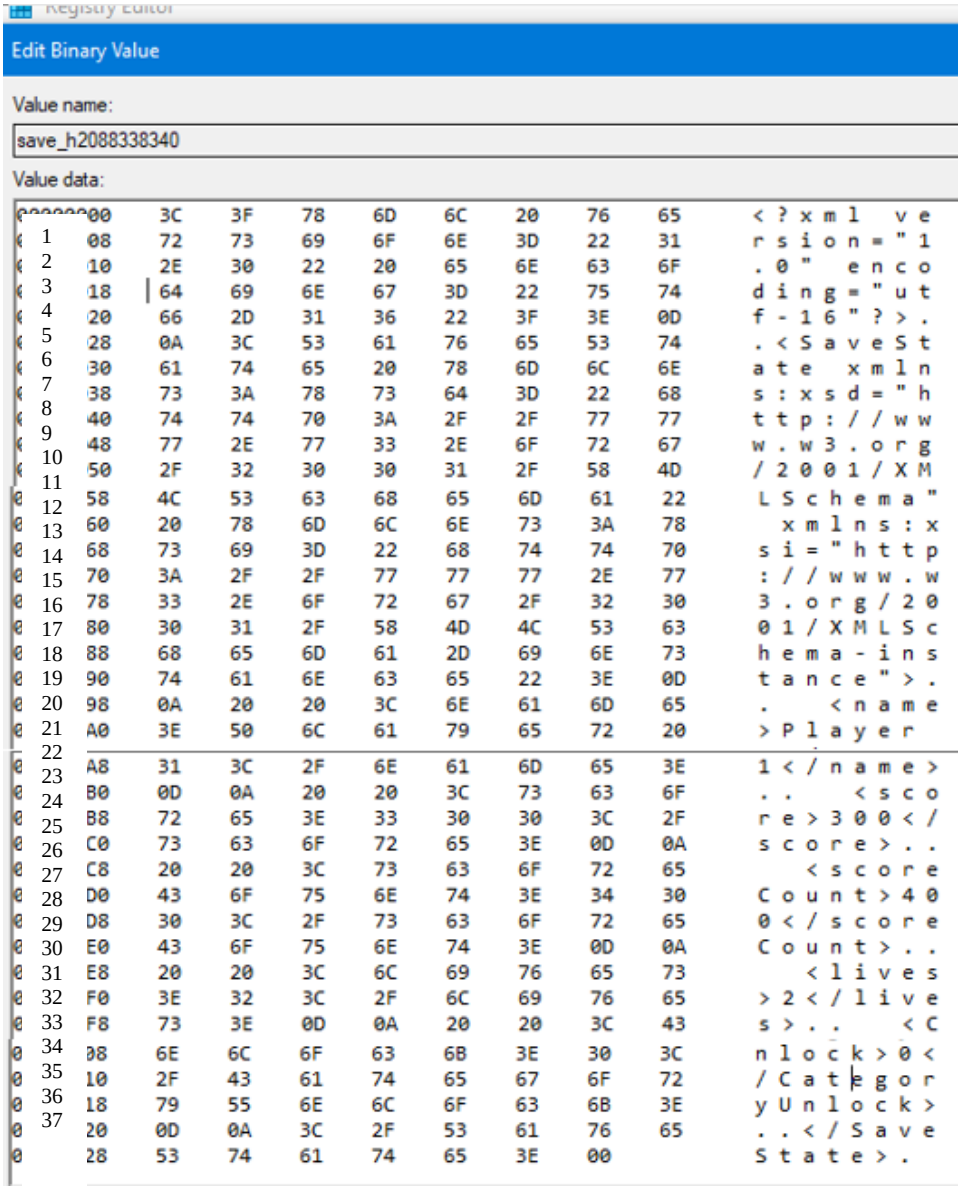
Gambar 8. Tampilan console simpanan data pengguna untuk pengembang setelah dienkripsi

Pada Gambar 8, dapat dilihat bahwa data simpanan pengguna sudah terenkripsi dengan UTF-8, sehingga semua simpanan data tidak dapat terbaca dengan bahasa awam karena sudah dalam bentuk enkripsi UTF-8.

2. Pengujian Enkripsi Simpanan Data Pengguna dari Sisi Pengguna

Pengguna dapat mengakses data pengguna dalam game di Computer\HKEY_CURRENT_USER\SOFTWARE*Nama Perusahaan**Nama Game*.

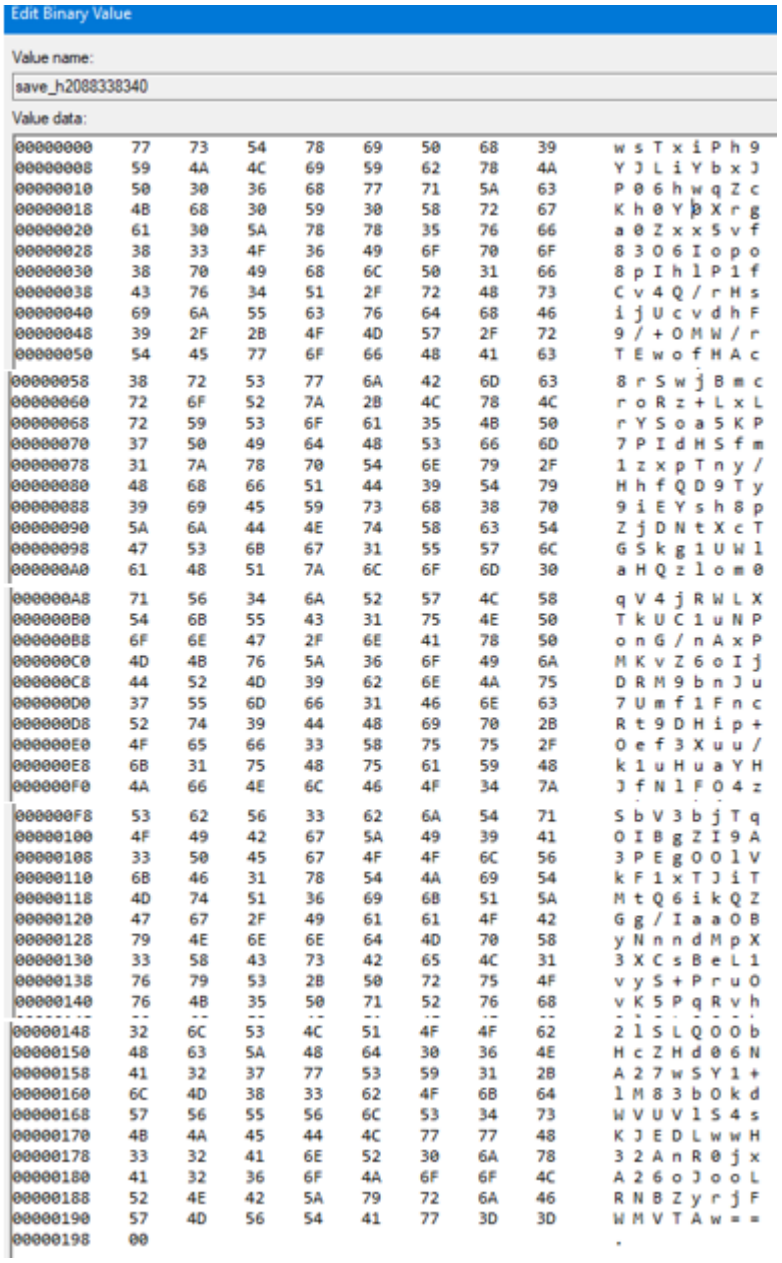
a. Sebelum Dienkripsi



Gambar 9. Tampilan simpanan data pengguna untuk pengguna sebelum dienkripsi

Pada Gambar 9, dapat dilihat bahwa data simpanan pengguna belum terenkripsi sehingga semua simpanan data dapat terbaca dalam bahasa xml. Pengguna yang mempunyai pengetahuan tentang scripting akan dengan mudah mengubah nilai data pengguna dalam game.

b. Setelah Dienkripsi



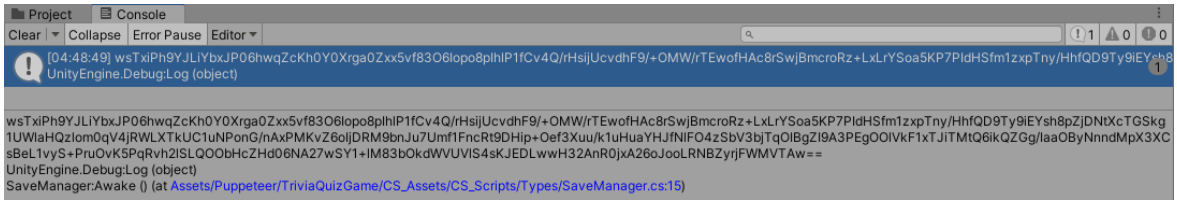
Gambar 10. Tampilan simpanan data pengguna untuk pengguna setelah dienkripsi

Pada Gambar 10, dapat dilihat data pengguna yang telah dienkripsi yang diakses oleh pengguna. Nilai data pengguna di bagian kanan menjadi sulit untuk dipecahkan. Akan dibutuhkan kemampuan dan usaha yang jauh lebih tinggi untuk mengubah nilai simpanan data pengguna yang telah terenkripsi oleh UTF-8.

3. Pengujian Dekripsi Simpanan Data Pengguna dari Sisi Pengembang

Pengembang dapat mendekripsi data pengguna yang telah dienkripsi saat mengembangkan game di Unity dengan menggunakan perintah Debug untuk menampilkannya di console.

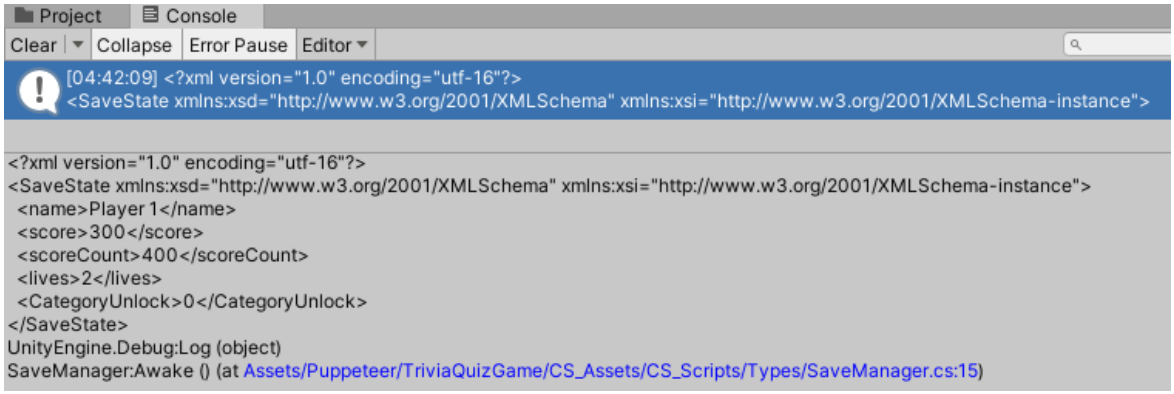
a. Sebelum Didekripsi



Gambar 11. Tampilan console simpanan data pengguna untuk pengembang setelah dienkripsi

Pada Gambar 11, dapat dilihat bahwa data simpanan pengguna terenkripsi dengan UTF-8, sehingga semua simpanan data tidak dapat terbaca dengan bahasa awam karena dalam bentuk enkripsi UTF-8.

b. Setelah Didekripsi



Gambar 12. Tampilan console simpanan data pengguna untuk pengembang sebelum dienkripsi

Pada Gambar 12, dapat dilihat bahwa data pengguna telah terdekripsi sehingga data simpanan pengguna seperti nama, skor, skor tertinggi, nyawa, dan kategori yang terbuka dapat dilihat secara langsung.

4. Pengujian Ukuran Data Sebelum dan Sesudah Enkripsi

Tabel 5: Pengujian ukuran data sebelum dan sesudah enkripsi terhadap simpanan data pengguna menggunakan C# di Visual Studio 2019

Simpanan Data Pengguna					Ukuran Data (Bytes)	
Nama	Skor	Total Skor	Jiwa	Kategory	Sebelum Dienkripsi	Sesudah Dienkripsi
Player 1	0	0	3	3	299	409
Player 1	100	100	3	3	303	409
Player 1	200	300	3	3	303	409
Player 1	300	600	2	3	303	409
Player 1	400	1000	2	4	304	409
Player 1	500	1500	3	4	304	409
Player 1	600	2100	3	5	304	409

Player 1	700	2800	1	5	304	409
Player 1	800	3600	2	6	304	409
Player 1	900	4500	1	7	304	409

Pada Tabel 5, merupakan hasil pengujian ukuran data simpanan data pengguna yang berupa nama (string), skor (float), total skor (float), jiwa (float), dan kategori (float). Pengujian dilakukan sebanyak 10 kali dengan nilai yang berbeda-beda. Hasil pengujian mempelihatkan ukuran data sebelum dan sesudah dienkripsi.

4. KESIMPULAN

Hasil enkripsi game smartphone Simulasi Perilaku Hidup Bersih dan Sehat dengan UTF-8 mengandung semua simpanan data pengguna yang telah dimainkan dan dites oleh peguji. Simpanan data yang telah dienkripsi adalah nama pengguna, skor terakhir, skor tertinggi, sisa nyawa, dan kategori yang sudah terbuka.

Data simpanan yang telah di-enkripsi sudah merupakan format UTF-8 sehingga lebih sulit untuk dilihat maupun diubah oleh pihak yang tidak berwenang. Bila ada mempunyai niat untuk melihat dan mengubah data yang telah di-enkripsi, dibutuhkan usaha dan kemampuan ekstra dan memakan waktu. Oleh karena itu, data simpanan aplikasi game smartphone Simulasi Perilaku Hidup Bersih dan Sehat sekarang mempunyai pertahanan standar dari kejahatan cyber. Semoga penelitian ini dapat menjadi referensi untuk bagian kriptografi di Indonesia.

DAFTAR PUSTAKA

[1] NIST Special Publication 1800-21 Mobile Device Security: Corporate-Owned Personally-Enabled (COPE) Includes Executive Summary (A); Approach, Architecture, and Security Characteristics (B); and How-To Guides (C) Final. September 2020.

[2] Kementerian Kesehatan RI Indonesia. Kementerian Kesehatan RI. Sekretariat Jenderal Peraturan Menteri Kesehatan Republik Indonesia nomor: 2269/MENKES/PER/XI/2011 Pedoman Pembinaan Perilaku Hidup Bersih dan Sehat (PHBS),-- Jakarta: Kementerian Kesehatan RI. 2011.

[3] Erickson, S. (2021). Character Encoding: A Primer for Data Curators. Journal of EScience Librarianship, 10.

[4] Kirana, C., & Sugianto, E. (2019). Penerapan Algoritma AES dan Konversi SMS ke dalam Bahasa KHEK pada Aplikasi Enkripsi Berbasis Mobile Application. *Khazanah Informatika: Jurnal Ilmu Komputer Dan Informatika*, *5*(1), 68–77.

[5] Prameshwari, A., & Sastra, N. P. (2018). Implementasi Algoritma Advanced Encryption Standard (AES) 128 Untuk Enkripsi dan Dekripsi File Dokumen. *Eksplora Informatika*, *8*(1).

[6] Widodo, B. E., & Purnomo, A. S. (2020). IMPLEMENTASI ADVANCED ENCRYPTION STANDARD PADA ENKRIPSI DAN DEKRIPSI DOKUMEN RAHASIA DITINTELKAM POLDA DIY. *Jurnal Teknik Informatika (Jutif)*, *1*(2), 69–77. https://doi.org/10.20884/1.jutif.2020.1.2.21

[7] Unicode Consortium. 2021. “What Is Unicode?”. https://home.unicode.org/basic-info/faq/, diakses pada 23 Januari 2022 12:02 AM.

[8] Cloudflare, Inc. 2022. “How Encryption Works”. https://www.cloudflare.com/learning/ssl/what-is-encryption/, diakses pada 24 Januari 2022 4:55 AM.

[9] Paul Leahy. 2019. “An Explanation of Unicode Character Encoding”. https://www.thoughtco.com/what-is-unicode-2034272, diakses pada 24 Januari 2022 5:20 AM.

[10] American National Standard. 2007. Coded Character Sets - 7-Bit American National Standard Code for Information Interchange (7-Bit ASCII).

[11] Unity Technology. 2022. Unity 3D. In *Unity Technology.*

[12] Santos, E. A. (2019). OCR Evaluation Tools for the 21st Century. *Proceedings of the Workshop on Computational Methods for Endangered Languages*.