Majalah Ilmiah Teknologi Elektro, Vol. 19, No. 2, Juli - Desember 2020
DOI: https://doi.org/10.24843/MITE.2020.v19i02.P08

Perancangan Sistem Penyedia File Sharing dengan Enkripsi URL menggunakan Algoritma Rijndael

Adi Bhaskara[1], Dewa Made Wiharta[2], Oka Saputra[3]

[Submission: 05-11-2020, Accepted: 03-12-2020]

***Abstract*—Information is growing very rapidly makes information widely spread and can be disseminated easily and quickly. Generally, a website uses website-based programming with GET method where GET method has a weakness, system data can be seen in access directory. It may cause data in directory to be retrieved by unauthorized people. Based on that problem, researchers offered a solution by creating a file sharing system to share data safely. On system there are applications for uploaders, applications for download for downloaders and admins who regulate number of both application users. File Sharing system has been implemented can storage file directories on website by encrypting files with Rijndael Algorithm. Object in this research using URLs. Testing security level of URL folder that is taken without encryption algorithm Rijndael studied found that directory visible and it can be downloaded files without using an application and without registration. In addition, testing length of URL address shows that increase in length of URL does not affect length of time for encryption and decryption process because in addition to length of URL address, many factors affect, such as traffic on network or server usage load, etc. so that it gets fluctuating results.**

***Keywords*— GET method, encryption, URL, Rijndael algorithm, authentication, download, upload, ciphertext**

***Intisari*— Informasi yang berkembang sangat pesat membuat Informasi tersebar secara luas dan dapat disebarkan secara mudah dan cepat. Umumnya sebuah *website* menggunakan pemrograman berbasis *website* dengan metode GET dimana metode GET memiliki kelemahan data sistem dapat dilihat di direktori akses. Hal ini menyebabkan data yang ada pada direktori tersebut dapat diambil oleh orang yang tidak berwenang. Berdasarkan masalah tersebut peneliti menawarkan solusi dengan membuat sistem file sharing layanan untuk membagikan data dan melakukan unduh data secara aman. Pada sistem terdapat aplikasi untuk unggah untuk *uploader*, aplikasi untuk unduh untuk *downloader* dan admin yang bertugas mengatur jumlah kedua pengguna aplikasi. Sistem penyedia File Sharing yang telah diimplementasikan dapat mengamankan direktori file pada website dengan melakukan enkripsi pada direktori file yang dalam penelitian ini yaitu URL dengan menggunakan Algoritma Rijndael. Pengujian tingkat keamanan URL folder yang dibagikan tanpa enkripsi algoritma Rijndael didapatkan bahwa terlihat direktori folder yang dapat dilakukan unduh file tanpa menggunakan aplikasi dan tanpa registrasi. Selain itu, pengujian panjang alamat URL didapatkan hasil dengan kenaikan panjang alamat URL tidak mempengaruhi lama waktu proses enkripsi dan dekripsi karena selain panjang alamat URL banyak faktor yang mempengaruhi seperti trafik pada jaringan atau beban penggunaan server, dan lain-lain sehingga mendapatkan hasil yang fluktuatif.**

***Kata Kunci*— metode GET, enkripsi, URL, algoritma Rijndael, autentikasi, *download, upload,* cipherteks**

I. PENDAHULUAN

Pertukaran informasi pada era digital ini memungkinkan semua orang dapat saling bertukar informasi baik berupa data maupun teks dengan cepat dan mudah melalui suatu website. Umumnya sebuah website menggunakan pemrograman berbasis *website* dengan metode GET. metode GET memiliki kelemahan yaitu direktori akses terlihat pada *address bar browser* [1]. Apabila direktori file pada website diketahui maka seluruh data yang ada pada direktori tersebut dapat diambil oleh orang yang tidak berwenang. Pengambilan data oleh orang yang tidak berwenang sangat merugikan terutama bagi website penyedia data berbayar.

Salah satu mekanisme untuk meningkatkan keamanan adalah dengan menggunakan teknologi enkripsi pada kriptografi. Kriptografi merupakan teknik pengamanan data yang berguna melindungi kerahasiaan dan keaslian data, serta informasi atau data [2]. Enkripsi diterapkan pada URL sehingga teks pada URL tidak dapat dimengerti orang. URL *(Uniform Resource Locator)* saat ini merupakan salah satu media untuk membagikan data dengan mudah dan cepat terutama pada website-website *file hosting*.

Penggunaan algoritma simetris dipilih pada penelitian ini karena dibandingkan menggunakan asimetris key dengan kunci yang berbeda untuk enkripsi dan dekripsi, mereka lebih memilih simetris key karena lebih simpel [3]. Selain itu, proses enkripsi algoritma asimetris 1000 kali lebih lambat dibandingkan enkripsi karena memerlukan tenaga komputasional yang lebih dibanding enkripsi algoritma simetris [3]. Terkait penelitian-penelitian tersebut, peneliti melakukan hal yang sama yaitu memilih menggunakan kunci simetris pada sistem file sharing. Namun, algoritma simetris memiliki kelemahan pada saat membagikan kunci simetris dari pemberi ke penerima [4]. Solusi mengamankan pendistribusian kunci yang bersifat privat pada kunci simetris pada penelitian ini di lakukan dengan melakukan enkripsi oleh sistem dengan kunci yang disimpan pada *database*.

Algoritma kriptografi yang digunakan untuk proses enkripsi pada penelitian ini yaitu Algoritma Rijndael. Alasan pada penelitian ini menggunakan Algoritma Rijndael yaitu keseluruhan kunci dari Rijndael cukup untuk melindungi informasi yang ada didalamnya. Belum ada yang bisa membuktikan keberhasilan serangan melawan Algoritma

[1]*Mahasiswa, Program Magister Teknik Elektro Fakultas Teknik Universitas Udayana, Jl. PB. Sudirman Denpasar,Bali, Indonesia (Tlp: 082247597291; e-mail: adibhaskara39@gmail.com)*

[2, 3] *Dosen, Jurusan Teknik Elektro dan Komputer Fakultas Teknik Universitas Udayana, Jalan Kampus Bukit Jimbaran 80361 INDONESIA; (e-mail: wiharta@unud.ac.id, okasaputra@unud.ac.id)*

p-ISSN:1693 – 2951; e-ISSN: 2503-2372

Rijndael [5]. Penelitian lain juga menjelaskan algoritma AES lebih baik dibanding algoritma DES dalam waktu compile [6]. Berdasarkan waktu yang dibutuhkan untuk memecahkan algoritma kriptografi, semakin lama waktu yang dibutuhkan untuk memecahkan algoritma tersebut maka semakin baik keamanan algoritma tersebut. Sesuai hasil percobaan sebanyak 4 kali pada teks yang berbeda, algoritma Algoritma Rijndael membutuhkan waktu yang paling lama untuk dipecahkan dibanding algoritma DES, E-DES, T-DES, RSA, dan Blowfish [7]. Selain itu, Algoritma AES memiliki performa yang lebih baik dari segi *throughput*, waktu enkripsi dan waktu dekripsi dibanding algoritma DES, RSA,dan Blowfish [4]. Bukan hanya itu, Performa AES lebih unggul dibanding algoritma DES, 3DES (112 bit key), 3DES (168 bit key), dan Blowfish sehingga cocok diterapkan pada MANET [8].

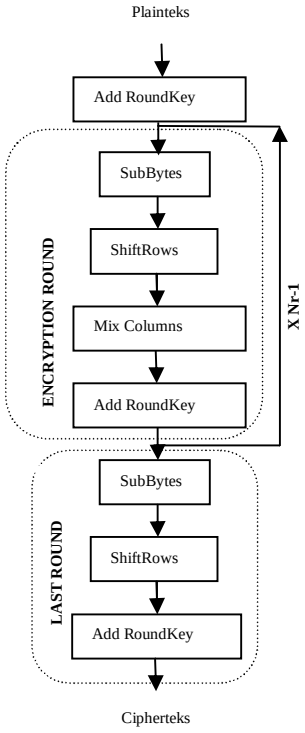
Penelitian ini mengusulkan suatu metode keamanan file dengan enkripsi URL yang diimplementasikan pada sistem *file sharing*. Berdasarkan pada penelitian-penelitian yang terkait yaitu penelitian-penelitian sebelumnya yaitu hanya melakukan enkripsi Rijndael pada URL [9], [10] sedangkan pada penelitian ini dirancang suatu sistem penyedia *file sharing* dengan enkripsi URL menggunakan algoritma Rijndael. Penelitian yang sejenis sebelumnya juga yaitu Perancangan Sistem Aplikasi *Download Manage*r Dengan Enkripsi Pada URL Menggunakan Algoritma Blowfish tidak spesifik menjelaskan sistem *file sharing* dimana terdapat aplikasi *upload*, aplikasi *download*, dan website untuk mengatur pengguna pada kedua aplikasi dan hanya menekankan pada aplikasi *download*. Selain itu, kebaharuan terdapat pada sistem *file sharing* yang diimplementasikan menggunakan algoritma Rijndael dimana penelitian sebelumnya proses enkripsi URL menggunakan Algoritma Blowfish [11]. Pada sistem file sharing terdapat 3 bagian utama yaitu aplikasi untuk *upload file*, aplikasi untuk *download file*, dan halaman admin. Aplikasi *upload* digunakan oleh penyedia file untuk mengunggah file yang ingin dibagikan. Orang yang memiliki kewenangan untuk mendapatkan data yang dibagikan menggunakan aplikasi *download* untuk mengunduh file. Sedangkan admin bertugas untuk mengatur jumlah pengguna sebagai *uploader* dan *downloader* file. Pengidentifikasian antara *uploader* dan *downloader* dilakukan dengan proses autentikasi pada masing-masing aplikasi.

II. TINJAUAN PUSTAKA

Pada penelitian digunakan tipe sistem *file sharing centralized*. Penelitian ini menggunakan sistem *file sharing centralized* dibanding *decentralized* karena *decentralized* memiliki banyak kelemahan keamanan contohnya *file sharing* berbasis *cloud* sedangkan *centralized* dimana sistem dikontrol secara terpusat. Berdasarkan penelitian yang ada *cloud computing* banyak memberikan user kenyamanan *service*, tetapi disisi lain banyak masalah kritis pada keamanan dan reliabilitas sistem *cloud*. Kasus yang sering terjadi yaitu pengambilan data dan kebocoran *user data* [12]. selain itu, sistem *file sharing* dengan menggunakan *cloud server* memiliki kelemahan yaitu seseorang dengan akses ke server bisa memodifikasi data tanpa ijin klien [13]. Penelitian ini juga menggunakan teknik keamanan kriptografi dimana kriptografi melindungi konten selama transmisi tanpa pengecekan tambahan setelah di dekripsi [14]. Selain itu kriptografi menyediakan banyak keamanan layanan, kenyamanan, integritas, dan autentikasi [15]. Implementasi algoritma enkripsi yang digunakan pada penelitian ini yaitu algoritma Rijndael dengan kunci 128 bit untuk mengamankan URL dari kelemahan metode GET. algoritma Rijndael sangat kompetitif dengan algoritma Blowfish dilihat dari waktu proses enkripsi dan dekripsi [7]. Namun pada penelitian ini diplih algoritma Rijndael karena evaluasi performa yang dilakukan menggunakan *tool Pycrypto package* pada bahasa pemrograman phyton mendapatkan hasil algoritma Rijndael lebih cepat 200 sampai 300 ms dibandingkan Blowfish [16].

Sebelum masuk ke proses enkripsi yang dijelaskan pada gambar sekian, Hal pertama yang dilakukan yaitu melakukan proses key schedule untuk mendapatkan round key yang digunakan pada proses add round key. Penjelasan operasi key schedule yaitu sebagai berikut :

- Operasi *Rotate*, yaitu operasi perputaran 8 bit pada 32 bit dari kunci.
- Operasi *SubBytes*, pada operasi ini 8 bit dari *subkey* disubstitusikan dengan nilai dari S-Box.
- Operasi Rcon, operasi ini dapat diterjemahkan sebagai operasi pangkat 2 nilai tertentu dari *user*. Operasi ini menggunakan nilai-nilai dalam *Galois field*. Nilai-nilai dari Rcon kemudian akan di-XOR dengan hasil operasi *SubBytes*.
- Operasi XOR dengan $w[\,i - Nk\,]$ yaitu word yang berada pada Nk sebelumnya.



Gambar 1. Alur Proses Enkripsi Algoritma Rijndael

Selanjutnya proses dilanjutkan sesuai dengan gambar 1 yaitu proses *AddRoundKey*. Pada proses ini *subkey* digabungkan dengan *state*. Proses penggabungan ini menggunakan operasi XOR untuk setiap *byte* dari *subkey* dengan *byte* yang bersangkutan dari *state*. Untuk setiap tahap, subkey dibangkitkan dari kunci utama dengan menggunakan proses *key schedule*. Setiap *subkey* berukuran sama dengan state yang bersangkutan. Pada Proses *SubBytes* adalah operasi yang akan melakukan substitusi tidak linier dengan cara mengganti setiap *byte state* dengan *byte* pada sebuah tabel yang dinamakan tabel *SBox* yang terdapat pada gambar 2

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
00	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
10	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
20	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
30	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
40	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
50	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
60	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
70	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
80	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
90	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a0	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b0	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c0	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d0	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e0	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f0	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Gambar 2. S Box

Proses *Shift Rows* akan beroperasi pada tiap baris dari tabel *state*. Proses ini akan bekerja dengan cara memutar *byte-byte* pada 3 baris terakhir (baris 1, 2, dan 3) dengan jumlah perputaran yang berbeda-beda. Baris 1 akan diputar sebanyak 1 kali, baris 2 akan diputar sebanyak 2 kali, dan baris 3 akan diputar sebanyak 3 kali. Sedangkan baris 0 tidak akan diputar

Proses *Mix Columns* akan beroperasi pada tiap kolom dari tabel *state*. Operasi ini menggabungkan 4 *bytes* dari setiap kolom tabel *state* dan menggunakan transformasi linier. Operasi *Mix Columns* memperlakukan setiap kolom sebagai *polynomial* 4 suku dalam *Galois field* dan kemudian dikalikan dengan c(x) mod (x^4+1) , dimana c(x) = 3x^3+ x^2+x+2. Kebalikkan dari *polynomial* ini adalah c(x) = 11x^3+ 13x^2+9x+14. Operasi *Mix Columns* juga dapat dipandang sebagai perkalian *matrix*. Langkah *Mix Columns* dapat ditunjukkan dengan mengalikan 4 bilangan di dalam *Galois field* oleh *matrix* berikut ini.

$$r_0 = 2a_0 + a_3 + a_2 + 3a_1$$
$$r_1 = 2a_1 + a_0 + a_3 + 3a_2$$
$$r_2 = 2a_2 + a_1 + a_0 + 3a_3$$
$$r_3 = 2a_3 + a_2 + a_1 + 3a_0$$

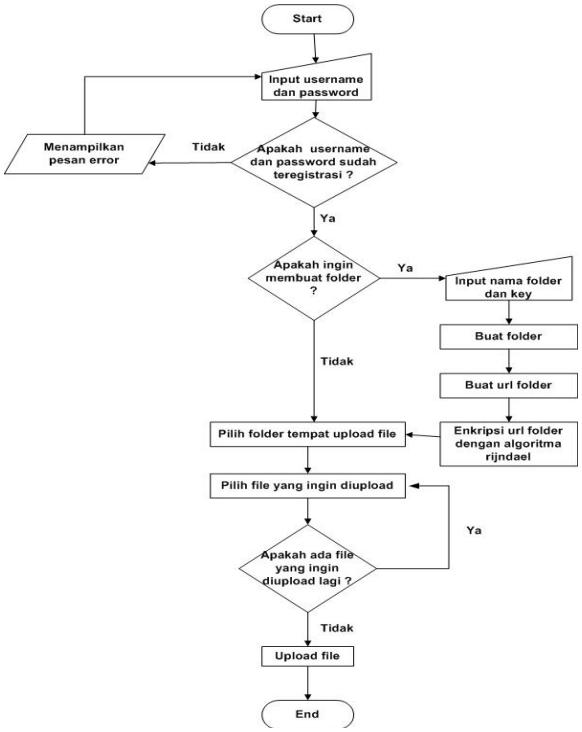
Operasi penjumlahan di atas dilakukan dengan operasi XOR, sedangkan operasi perkalian dilakukan dalam *Galois field*. Proses dekripsi memiliki 10 *round* kebalikan dari proses enkripsi. round satu sampai sembilan, terdapat *Addroundkey*, *Invshiftrow*, *InvByteSub*, dan *InvMixColumn*, sementara round kesepuluh tidak terdapat *InvMixColumn*. cipherteks yang didapatkan via *link* pada *website* akan melalui proses 10 *round* proses dekripsi untuk mendapatkan plainteks yang berupa alamat URL kembali.

A. *Alur Kerja Proses Upload File*

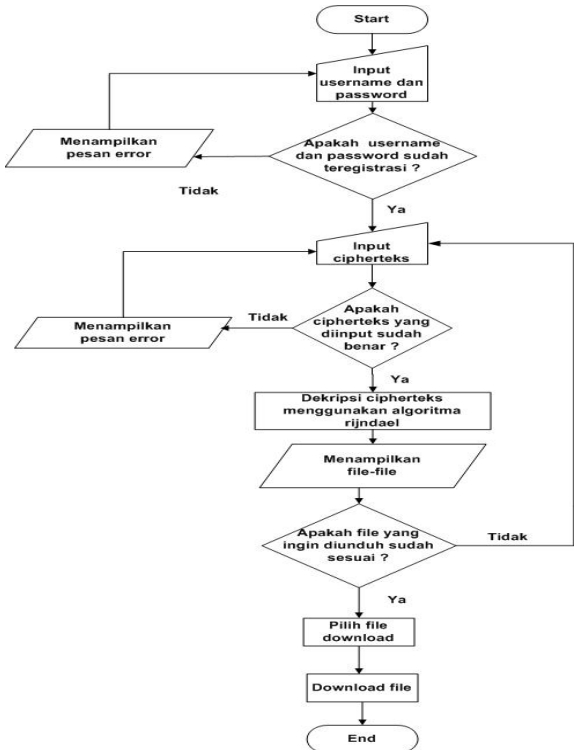
Berdasarkan gambar 3. hal pertama yang dilakukan pada aplikasi *upload* yaitu melakukan proses autentikasi untuk memastikan bahwa *user upload* yang terdaftar saja yang dapat mengakses aplikasi. Setelah masuk aplikasi, pada pembuatan folder diaplikasi terdapat kotak tempat menginputkan nama folder dan *key*. *Key* yang digunakan untuk enkripsi menggunakan algoritma Rijndael yaitu kunci simetris. *Key* pada implementasi aplikasi *download manager* diinputkan oleh *user upload*, disimpan pada *database* dan pembuatan folder pada aplikasi menghasilkan *URL folder*. *URL folder* tersebut dienkripsi menggunakan algoritma Rijndael dengan *key* yang telah diinputkan saat pembuatan folder. Jika file ingin ditempatkan pada folder yang sudah ada maka *upload file* pada folder tempat *upload file* yang diinginkan.

B. *Alur Kerja Proses Download File*

Sesuai dengan gambar 4. langkah awal yang dilalui untuk bisa mengakses aplikasi *download* yaitu memasukkan *username* dan *password* yang terdaftar pada *server*. Setelah masuk aplikasi, hal pertama yang dilakukan dalam mengunduh file dengan URL yaitu melakukan *copy URL* yang terenkripsi pada aplikasi *download*. Lalu URL dilakukan proses dekripsi dengan kunci yang diambil pada *database* dengan melakukan pengecekan kunci mana yang cocok dengan URL yang masih terenkripsi. Hasil dari dekripsi URL ditampilkan berupa data yang ada pada URL tersebut.



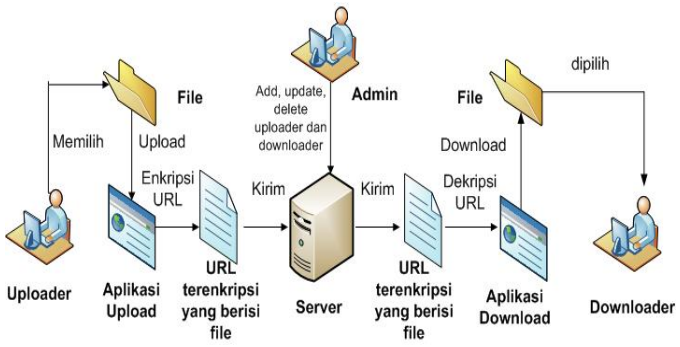
Gambar 3. Alur Kerja pada Aplikasi *Upload*



Gambar 4. Alur Kerja pada Aplikasi *Download*

C. *Perancangan Sistem*

Pada gambar 5. akan dibangun suatu sistem berupa *file sharing* yang mengamankan file yang dibagikan kepada orang yang memiliki hak akses terhadap file tersebut. Pada file ini telah dilakukan enkripsi dengan menggunakan algoritma Rijndael sehingga dengan aplikasi *download* ini saja yang dapat melakukan unduh file yang dibagikan. Pada sistem ini terdapat juga *uploader* yang bertugas membuat dan menyediakan file terenkripsi tersebut. file yang dibagikan berupa file yang terenkripsi yang nantinya akan didekripsi aplikasi *download*. Aplikasi *download* dan *upload* memiliki sistem autentikasi dimana hanya *user* yang memiliki akun saja yang dapat mengakses aplikasi ini. Selain itu, terdapat *website* yang dapat menambah, melakukan *update*, dan melakukan *delete user* sebagai *uploader* maupun *downloader*.



Gambar 5 Alur Kerja Sistem

III. HASIL DAN PEMBAHASAN

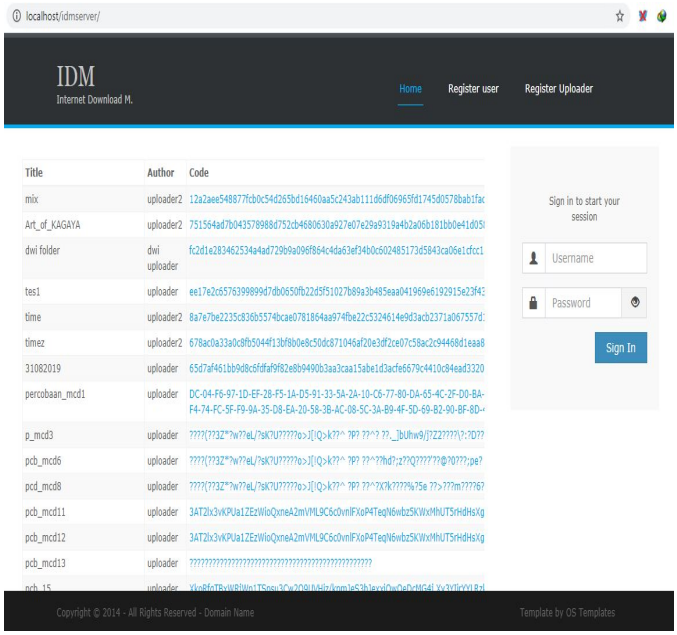
Pada penelitian ini menggunakan arsitektur *client-server* di mana merupakan sistem terdistribusi yang melibatkan *request services client* dari komponen *server* melalui koneksi jaringan. Kesatuan dari *service client* dan *server* disebut juga sebagai *web service* yang dapat diakses melalui pemanggilan method oleh suatu aplikasi dengan protokol standar web[17]. *Web service* pada penelitian ini digunakan *Web Service* REST. REST bisa diselesaikan melalui protokol HTTP dan menggunakan *cache* yang dapat menambah kecepatan response, performa, dan efisiensi yang mana seluruhnya lebih baik dibandingkan protokol SOAP [18]. *Web service* REST yang digunakan pada penelitian ini, pertama melakukan *request* sebagai *client* dan diproses oleh server.

Metode *client-server* yang digunakan pada sistem *file sharing* pada penelitian ini yaitu *long polling*. *long polling* merupakan metode yang dimana *client* mengirimkan *request* ke server dan server menahan *live link* setelah menerima *request* [19]. apabila terdapat *message* baru, server akan mengembalikan informasi response ke *client* dan menutup koneksi. selama *disconnection*, informasi baru dari user akan di *cache* oleh server. *client* mengirim *request* baru ke server setelah proses response dari server.

Aplikasi *Uploader* dan *downloader* dibuat berbasis desktop dengan bahasa C# dengan *framework* Visual Studio 2012 dan untuk website menggunakan PHP dengan *framework* Code Igniter.

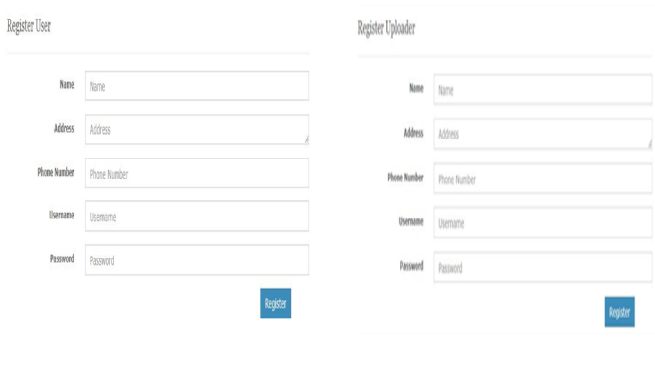
A. Website

Halaman awal website berisi *data sharing* dari *uploader* yang didesain hanya bisa diunduh menggunakan aplikasi *downloader* yang diimplementasi dalam penelitian ini. data sharing berupa file yang didalamnya terdapat cipherteks. Pada gambar 6. terdapat juga menu untuk *registrasi user download*, *registrasi uploader*, dan halaman *login admin.*



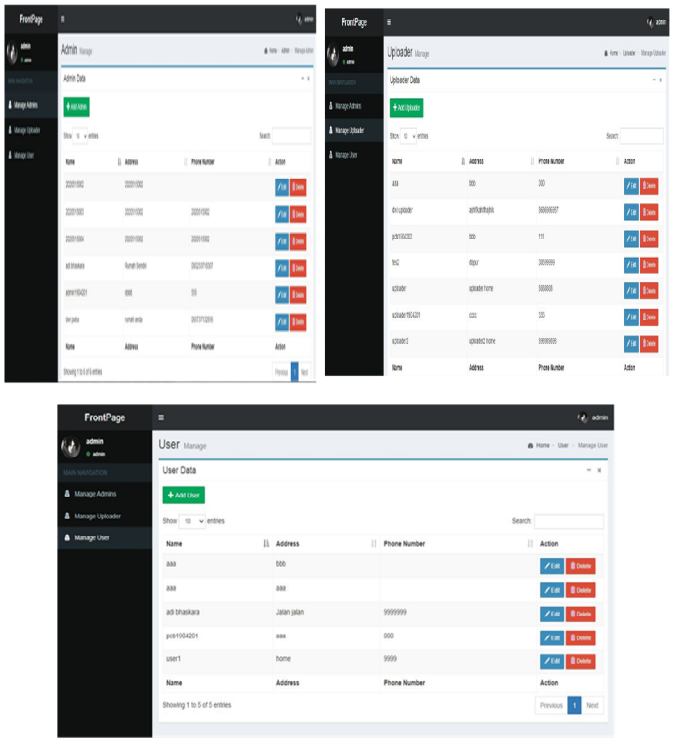
Gambar 6. Tampilan Awal Website

Pada gambar 7 terdapat halaman *registrasi user download* dan halaman *registrasi uploader* terdapat kolom inputan yang sama yaitu nama, alamat, nomor telepon, *username* dan *password*. Setelah seluruh inputan diisi maka inputan disimpan pada *database* dengan menekan tombol *register*.



Gambar 7. Halaman registrasi user *download* dan halaman registrasi *uploader*

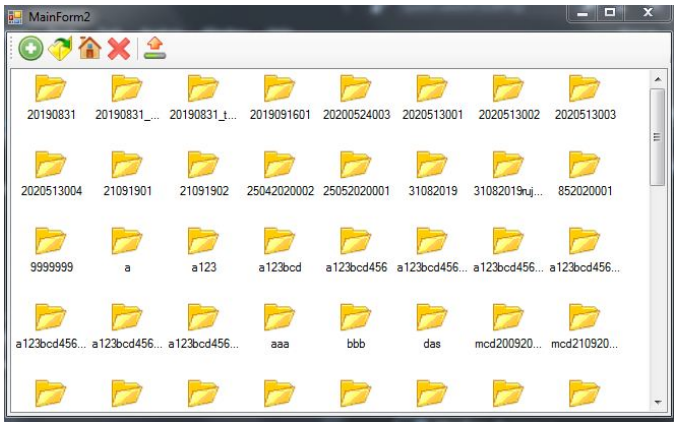
Halaman admin tampil apabila user sudah masuk sebagai admin pada halaman login dengan menginputkan *username* dan *password* admin. Admin pada sistem ini mempunyai fungsi untuk Menambahkan akun user dan *Uploader*, mengatur akun user baik mengedit informasi user dan *Uploader* atau menghapus akun user dan *Uploader*. Selain itu admin juga dapat menambahkan admin baru dan mengedit maupun menghapus akun admin tersebut. Setelah melakukan login maka terdapat 3 menu halaman yaitu *manage admin*, *manage uploader*, dan *manage user* seperti gambar 8. Pada masing-masing menu tersebut dapat dilakukan tambah, edit, maupun *delete* akun.



Gambar 8. *manage Admins, manage uploader*, dan *manage user download*

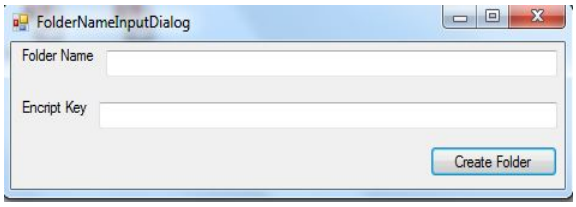
B. Aplikasi *Uploader*

Algoritma yang digunakan dalam proses enkripsi adalah algoritma Rijndael dengan panjang kunci 128 bit.



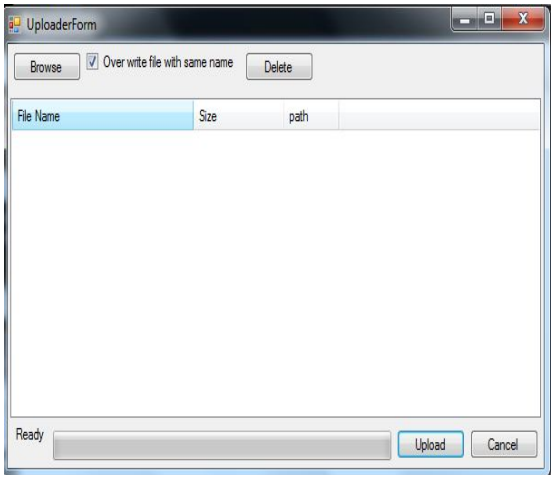
Gambar 9 Tampilan Utama *Form Uploader*

Pada gambar 10. merupakan tampilan *form add folder*. Untuk membuat folder, diinputkan nama folder dan *key* untuk enkripsi. *key* enkripsi dan folder akan disimpan didatabase yang nantinya akan digunakan untuk proses dekripsi. Setelah folder dibuat secara otomatis URL terenkripsi ditampilkan pada halaman website.



Gambar 10. Tampilan *form Add Folder*

Pada gambar 11. merupakan tampilan pada fitur *add upload* file yang menampilkan *Uploader form* yang berisi tombol *browse* untuk memilih data yang mana yang akan di unggah. *checkbox* berfungsi untuk *overwrite file* dengan nama file yang sama. Tombol *delete* digunakan untuk menghapus file yang tidak jadi untuk di unggah. Pada proses pemilihan file yang di unggah dapat dilakukan secara simultan untuk beberapa file.Tombol *Upload* digunakan untuk mengeksekusi agar file yang dipilih langsung di unggah. Pada *progress bar* akan menampilkan *progress* dari proses *upload file*.

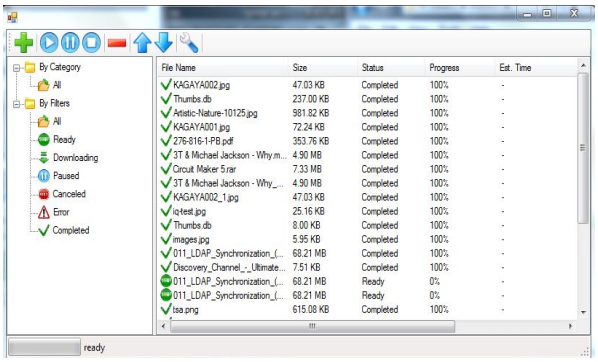


Gambar 11 Tampilan *Form Browse*

p-ISSN:1693 – 2951; e-ISSN: 2503-2372

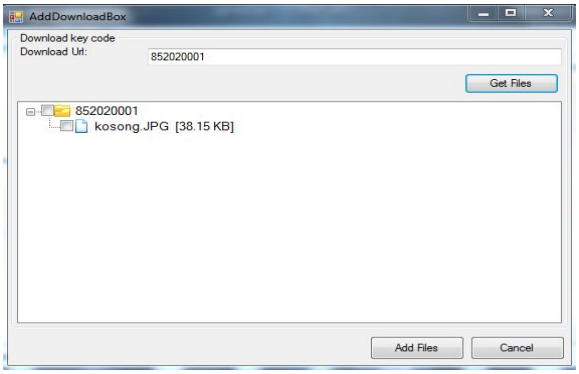
C. Aplikasi Downloader

Form *AddDownloadBox* berfungsi sebagai tempat URL di dekripsi dan ditampilkan pada *listview.* Form pada gambar 12. ini memiliki tombol *Paste From ClipBoard* untuk paste URL yang sebelumnya sudah dilakukan *copy clipboard* pada website terlebih dulu. Tombol *Get files* untuk menampilkan hasil dekripsi URL pada *textbox* dengan URL yang sudah ada pada *textbox*



Gambar 12 Tampilan Utama Aplikasi *Download*

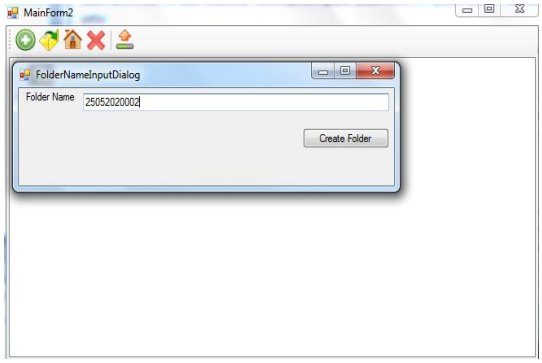
Pada gambar 13. merupakan tampilan form AddDownloadBox yang sudah dilakukan proses dekripsi inputan nama file yang berisi cipherteks pada kolom *download* URL. Apabila file yang diinginkan sudah dipilih tekan tombol *add files* maka file yang akan di unduh ditampilkan pada *listview* ditampilan utama yang berstatus *ready.*



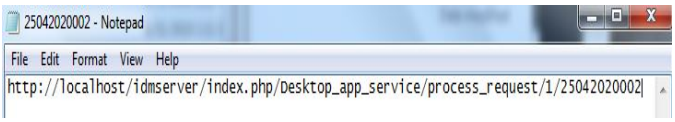
Gambar 13 Tampilan *Form AddDownloadBox* setelah proses dekripsi

D. Pengujian Tingkat Keamanan

Berdasarkan gambar 14. dan gambar 15. sekian memperlihatkan proses aplikasi *uploader* membuat folder sharing. Hasil folder sharing yang dibuat dengan URL yang dibagikan tanpa enkripsi sehingga memperlihatkan direktori folder yang bisa dilakukan *download file* tanpa menggunakan aplikasi dan tanpa autentikasi. Hal tersebut membuat keamanan file di dalam folder yang hanya dibagikan kepada orang yang berwenang sangat rentan di unduh oleh orang yang tidak berwenang dan disebarluaskan. Namun, Seiring dengan kecepatan *processing* yang secara terus menerus bertambah ukuran *key* harus ditambah untuk keamanan data. Selain itu, jumlah sistem yang digunakan untuk penyerangan dengan *brute force* juga meningkat [20]. Enkripsi dengan menggunakan *key* yang lebih panjang lebih susah dipecahkan terutama dengan serangan *brute force* sehingga penelitian lebih lanjut diharapkan dapat menggunakan algoritma rijndael 256 bit untuk meminimalisir penyerangan dengan metode *brute force.*



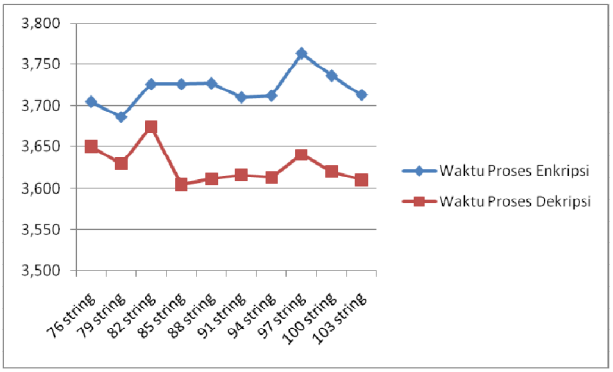
Gambar 14. Pembuatan Folder Tanpa menggunakan algoritma enkripsi



Gambar 15. Tampilan URL dalam file tanpa proses enkripsi

E. Pengujian Panjang Alamat URL

Pengujian panjang alamat URL untuk mengukur pengaruh waktu proses enkripsi dan dekripsi URL menggunakan algoritma Rijndael. Proses enkripsi URL ada pada aplikasi *uploader* sedangkan proses dekripsi pada aplikasi *downloader*. Pengujian dilakukan sebanyak 10 alamat URL dengan panjang alamat yang bervariasi yaitu dilakukan proses enkripsi dan proses dekripsi menggunakan algoritma Rijndael. Pengujian panjang alamat URL



Gambar 16. Grafik waktu proses enkripsi dan dekripsi URL

Hasil yang dirata-ratakan secara total didapatkan hasil rata-rata proses enkripsi 3,720 detik dan proses dekripsi yaitu 3,626 detik. 10 URL folder yang digunakan pada proses enkripsi dan dekripsi mempunyai rentang lebih banyak 3 *string* secara berurutan dari urutan *string* sebelumnya.

Penelitian terkait menyebutkan panjang teks plainteks mempengaruhi terhadap waktu proses enkripsi dan dekripsi dengan kenaikan pada *response time* dengan adanya peningkatan pada panjang teks dengan panjang kunci setiap teks yang sama [21]. Namun, pada hasil pengujian pada gambar 16. menunjukkan panjang alamat URL tidak

mempengaruhi lama waktu proses enkripsi dan dekripsi pada program. Merujuk kembali pada penelitian terkait disebutkan bahwa, pada observasi sebenarnya *response time* terkadang mengalami hasil yang fluktuasi dimana saat menjalankan tes dua kali dengan algoritma enkripsi pada *web browser* yang sama menggunakan panjang teks yang sama. Hal tersebut dikarenakan trafik pada jaringan atau beban penggunaan server [21]. Berdasarkan penelitian yang telah dilakukan terdapat faktor lain yang mempengaruhi waktu proses enkripsi dan dekripsi selain panjang alamat URL sehingga mendapatkan hasil yang fluktuatif pada hasil pengujian.

F. Pengujian Tingkat Keberhasilan Proses enkripsi dan dekripsi URL

Berdasarkan hasil pengujian tingkat keberhasilan proses enkripsi dan dekripsi file yang berisi URL menggunakan Rijndael kesimpulan valid didapatkan dari masing-masing pengujian yang dilakukan masing-masing sebanyak 50 dari 50 kasus uji sehingga untuk menghitung tingkat keberhasilan proses enkripsi dan dekripsi menggunakan rumus sebagai berikut :

$$\text{Persentase keberhasilan} = \frac{\text{Jumlah data valid}}{\text{Total data}} \times 100\%$$

$$\text{Persentase keberhasilan} = \frac{50}{50} \times 100\%$$

$$\text{Persentase keberhasilan} = 100\%$$

Hasil yang didapatkan pada perhitungan di atas memperlihatkan tingkat keberhasilan proses enkripsi pada aplikasi untuk *uploader* dan dekripsi pada aplikasi untuk *downloader* menggunakan algoritma Rijndael yaitu 100 %.

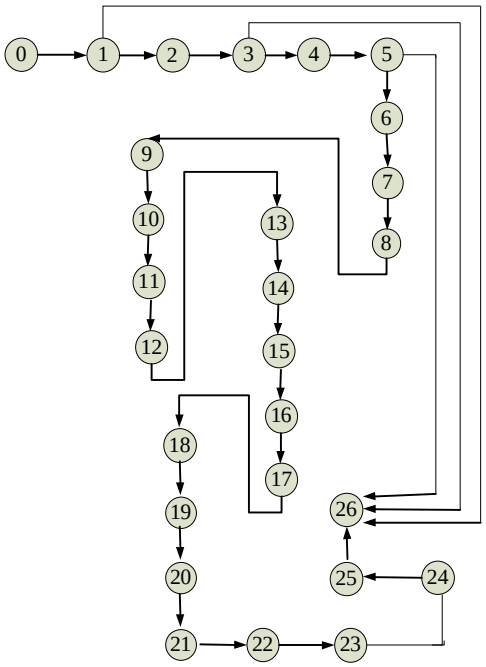
G. Pengujian *White Box*

Pengujian *white box* yang dilakukan berdasarkan perhitungan *cyclomatic complexity* yang dikutip pada buku McCabe software yang berjudul *"Structured Testing: A Testing Methodology Using the Cyclomatic Complexity Metric"*[22]. Tujuan dari *White Box testing* adalah untuk memverifikasi pernyataan pernyataan sistem pada aplikasi *uploader* dan aplikasi *downloader*. Dengan kata lain, pengujian ini untuk memastikan bahwa semua perintah dan kondisi aplikasi dieksekusi secara minimal. Hasil pengujian ini didapatkan nilai berupa perhitungan *cyclomatic complexity* yang dimana dibandingkan dengan tabel hubungan cyclomatic *complexity*. Rumus perhitungan *cyclomatic complexity* yaitu V(G)=E-N+2 yang dimana E adalah jumlah *edge* pada *flowgraph* dan N berupa jumlah node pada *flowgraph*[23]. tabel hubungan *cyclomatic complexity* dijelaskan pada tabel I[24].

TABEL I
HUBUNGAN CYCLOMATIC COMPLEXITY

No	Cyclomatic Complexity	Evaluasi Resiko
1	1-10	Fungsi yang simpel, tanpa banyak resiko kesalahan
2	11-20	Lebih kompleks, resiko kesalahan sedang
3	21-50	Kompleks, resiko kesalahan fungsi tinggi
4	Lebih dari 50	Program tidak dapat dites (sangat berisiko terjadi kesalahan)

Nilai yang diperoleh dari hasil perhitungan *cyclomatic complexity* pada *flowgraph* proses enkripsi rijndael gambar 17. pada aplikasi *uploader* adalah 4. Jika diperhatikan dalam tabel hubungan *cyclomatic complexity* dan resiko, nilai yang diperoleh terdapat dalam rentang 1-10 yaitu merupakan sebuah proses sederhana tanpa banyak resiko. Sehingga dengan rentang 1-10 memiliki hubungan resiko terhadap kemungkinan rawan kesalahan kecil.



Gambar 17. Flowgraph proses enkripsi pada aplikasi *uploader*

$$V(G) = E - N + 2$$
$$V(G) = 29 - 27 + 2$$
$$V(G) = 4$$

Jalur pengujian :
Jalur 1 : 0-1-26
Jalur 2 : 0-1-2-3-26
Jalur 3 : 0-1-2-3-4-5-26
Jalur 4 : 0-1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24- 25-26

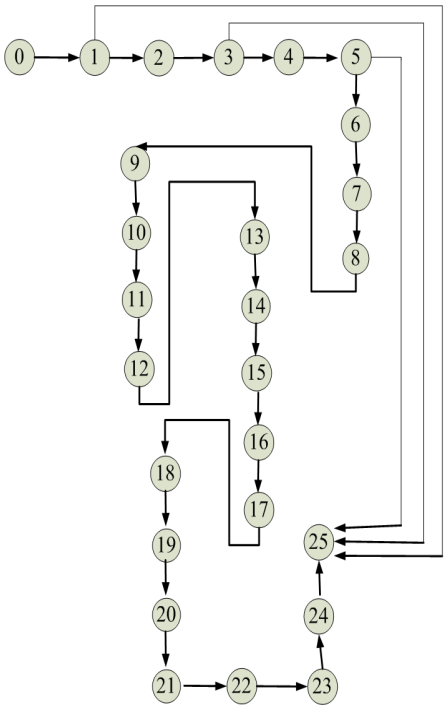
Nilai yang diperoleh dari hasil perhitungan *cyclomatic complexity* pada flowgraph proses dekripsi rijndael gambar 18. pada aplikasi *downloader* adalah 4. Jika diperhatikan dalam

p-ISSN:1693 – 2951; e-ISSN: 2503-2372

tabel hubungan *cyclomatic complexity* dan resiko, nilai yang diperoleh terdapat dalam rentang 1-10 yaitu merupakan sebuah proses sederhana tanpa banyak resiko. Sehingga dengan rentang 1-10 memiliki hubungan resiko terhadap kemungkinan rawan kesalahan kecil.

$V(G) = E - N + 2$
$V(G) = 28 - 26 + 2$
$V(G) = 4$

Jalur pengujian :
Jalur 1 : 0-1-25
Jalur 2 : 0-1-2-3-25
Jalur 3 : 0-1-2-3-4-5-25
Jalur 4 : 0-1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25



Gambar 18. flowgraph proses *download* pada aplikasi *downloader*

Hasil dari pengujian *white box* pada proses enkripsi dan dekripsi didapatkan nilai cyclomatic complexity rendah saat eksekusi karena teknik keamanan yang digunakan efisien dalam segi waktu sesuai dengan literatur yang ada mengenai algoritma rijndael. Diperkuat dengan hipotesis pada penelitian terkait yang menyatakan *control path* menentukan kompleksitas dari *rule based* program. pada umumnya, semakin tinggi struktur kompleksitas sebuah program, semakin sulit program tersebut di mengerti, di *test, debug* dan dilakukan *maintain* [25].

Jadi berdasarkan hipotesis tersebut pengujian *white box* yang dilakukan didapatkan nilai kompleksitas dengan rentang rendah karena algoritma rijndael sudah memiliki struktur kompleksitas yang rendah sehingga mudah dimengerti, mudah di test, di debug dan memiliki tingkat error yang rendah sehingga mudah dilakukan maintain.

H. Pengujian *Black Box*

Pengujian *Black Box* yang dilakukan pada aplikasi untuk *Uploader*, aplikasi untuk user *download* dan pada website didapatkan hasil dari 30 kasus uji aplikasi untuk *Uploader* didapatkan 30 nilai kesimpulan valid. Sedangkan dari 47 kasus uji aplikasi untuk user didapatkan 47 nilai kesimpulan valid dan 25 kasus uji pada website didapatkan 25 nilai kesimpulan valid. Hasil input dari aplikasi untuk user, aplikasi untuk *uploader* dan *website* sesuai dengan output yang diharapkan.

IV. KESIMPULAN

Kesimpulan yang dapat diambil penelitian yang telah dilakukan yaitu sebagai berikut :

1. Sistem penyedia *File Sharing* yang telah diimplementasikan dapat mengamankan direktori file pada website dengan melakukan enkripsi pada direktori file yang dalam penelitian ini yaitu URL dengan menggunakan Algoritma Rijndael. Bukan hanya itu, enkripsi yang dilakukan pada URL menggunakan Algoritma Rijndael yang dimana *key* dari URL terenkripsi tersimpan pada *database* sehingga orang yang tidak berkepentingan tidak dapat menyebarkan URL asli. Selain itu terdapat autentikasi pada masing-masing aplikasi file sharing yaitu aplikasi *uploader* dan aplikasi *downloader* sehingga hanya orang yang berwenang yang dapat menggunakan aplikasi.
2. Berdasarkan pengujian tingkat keamanan URL folder yang dibagikan tanpa enkripsi algoritma Rijndael didapatkan bahwa terlihat direktori folder yang dapat dilakukan *download file* tanpa menggunakan aplikasi dan tanpa registrasi. Hal tersebut memiliki resiko data dapat didownload oleh orang yang tidak berwenang dan URL dapat disebarluaskan.
3. Pengujian Panjang Alamat URL didapatkan hasil dengan kenaikan panjang alamat URL tidak mempengaruhi lama waktu proses enkripsi dan dekripsi karena selain panjang alamat URL banyak faktor yang mempengaruhi seperti trafik pada jaringan atau beban penggunaan server, dan lain-lain sehingga mendapatkan hasil yang fluktuatif.

REFERENSI

[1] P. K. Lestari, I K. G. Suhartana, I G. A. Wibawa, " Perbandingan Algoritma Enkripsi AES dan BASE64 dalam Pengamanan METHOD GET pada URL Website ", Jurnal Manajemen dan Teknologi Informasi Vol.8 No.2, pp. 66-71, 2018.

[2] I. B. A. Peling, N. P. Sastra, " Enhanced Audio Steganografi dengan Algoritma Advanced Encryption Standard untuk Pengamanan Data pada File Audio ", Majalah Ilmiah Teknologi Elektro Vol.17 No.1, pp. 66-71, 2018.

[3] V. V. Deotare, D. V. Padole, A. S. Wakode, "Area and Power Analysis of AES using Hardware and Software Co-Design" IEEE Global Conderence on Wireless Computing and Networking (GCWCN), 978-1-4799-6298-3, pp. 194-297, 2014.

[4] M. Panda, "Performance Analysis of Encryption Algorithms for Security", IEEE International conference on Signal Processing, Communication, Power and Embedded System (SCOPES), 978-1-5090-4620-1, pp. 278-284, 2016.

[5] M. A. Saleh, N. M. Hisham, E. Hisham, H. Hashim, " An Analysis and Comparison for Popular Video Encryption Algorithms ", IEEE

Information and Communication Technologies 978-1-4799-8969, pp 90-94, 2015.
[6] R. Indrayani, Subektiningsih, P. Ferdiansyah, D. A. Satria, "Effectiveness comparison of the AES and 3DES cryptography methods on email text messages", IEEE International Conference on Information and Communications Technology (ICOIACT), 978-1-7281-1655-6, pp. 66-69, 2019.
[7] O. G. Abood, M. A. Elsaad, and S. K. Guirguis, " Investigation of Cryptography Algorithms used for Security and Privacy Protection in Smart Grid ", Nineteenth International Middle East Power System Conference (MEPCON), pp. 644-649, 2017.
[8] M. Umaparvathi, D. K. Varughese, "Evaluation of Symmetric Encryption Algorithms fir MANETs", IEEE Information and Communication Technologies 978-1-4244-5967-4, 2010.
[9] A. Rahardian, " Implementasi Uniform Resource Locator Encryption pada Website Berbasis Algoritma BASE64 Studi Kasus pada Pimpinan Wilayah Aisyiyah Jawa Tengah ".
[10] A. Subari, S. Manan, " Implementasi Aeschipper Class untuk Enkripsi URL di Sistem Informasi Akademik Fakultas Teknik Universitas Diponegoro ", Jurnal Sistem Komputer Vol.4 No.2, pp. 63-69, 2014.
[11] I M. A. Bhaskara, I. K. A. Mogi, I P. G. H., 2016 " Perancangan Sistem Aplikasi Download Manager dengan Enkripsi pada URL menggunakan Algoritma Blowfish ".
[12] Z. Xinyi, Z. Ru, W. Fangyu, L. Jianyi, Y. Yuangang, "Research on the Privacy-preserving Retrieval over Ciphertext on Cloud", IEEE 6th International Conference on Information Communication and Management, 978-1-5090-3495-6, pp. 100-104, 2016.
[13] G. Wang, C. Liu, Y. Dong, "SafeBox: A Scheme for Searching and Sharing Encrypted Data in Cloud Applications", IEEE International Conference on Security, Pattern Analysis and Cybernetics, 978-1-5386-3016-7, pp. 648-653, 2017.
[14] M. Tanha, S. D. S. Torshizi, M. T. Abdullah, F. Hashim, "An Overview of Attacks against Digital Watermarking and their Respective Countermeasure", IEEE Information and Communication Technologies 978-1-4673-1677-4, pp. 265-270, 2012.
[15] L. A. Tawalbeh, O. Banimelhem, M. Al-Batati, "A Novel High Quality High Capacity Image Hiding Scheme Based on Image Compression and an Optical Pixel Adjustment Process", Information Security Journal: A Global Perspective, 21:256-268, pp. 256-268, 2012.
[16] J. Raigoza, K. Jituri, "Evaluating Performance of Symmetric Encryption Algorithms ", International Conference on Computational Science and Computational Intelligence, pp.1378-1381, 2016.
[17] Yogiswara, Wijono, and H. S. Dahlan, "Kinerja Web Service pada Proses\ Integrasi Data". Jurnal EECCIS 1(1), pp.73-78. 2014.
[18] B. Costa, P. F. Pires, F. C. Delicato, " Evaluating a Representational State Transfer (REST) Architecture", IEEE/IFIP Conference on Software Architecture 978-1-4799-3412-6, pp. 105-114, 2014.
[19] Y. Song, "Research on Web Instant Messaging Using REST Web Service", IEEE Information and Communication Technologies 978-1-4244-6359-6, pp. 497-500, 2010.
[20] B. Lakshmi, T. N. Prabakar, E. Kirubakaran, "Real time cryptography with dual key encryption", IEEE Proceedings of the 2008 International Conference on Computing, Communication and Networking (ICCCN), 978-1-4244-3595-1, 2008.
[21] S. Z.. S. Idrus, S. A. Aljunid, S. M.. Asi, S. Sudin, R. B. Ahmad, "Performance Analysis of Encryption Algorithms Text Length Size on Web Browsers", IJCSNS International Journal of Computer Science and Network Security, Vol.8 No.1, pp. 20-25, 2008.
[22] A. H. Watson, T. J. McCabe, *McCabe Software Structured Testing: A Testing Methodology Using the Cyclomatic Complexity Metric*, Gaithersburg: National Institute of Standards and Technology Special Publication 500-235, 1996.
[23] N. S. Fitriasari, S. A. Fitriani, R.A. Sukamto, "Comparison of Weighted Product Method and Technique for Order Preference by Similarity to Ideal Solution Method: Complexity and Accuracy", IEEE 3rd International Conference on Science in Information Technology (ICSITech) 978-1-5090-5864-8, pp. 453-458, 2017.
[24] J. L. Anderson Jr., " Using Software Tools and Metrics to Produce Better Quality Test Software" IEEE Autotestcon 0-7803-8449-0, pp. 293-297, 2004.
[25] A. M. K. Cheng,, "Measuring the Structural Complexity of OPS5 Rule-Based Programs", IEEE Information and Communication Technologies 0730-3157, pp. 522-527, 1996.

{Halaman ini sengaja di kosongkan }