Sistem Validasi Berkas Berbasis *Cloud Storage* Menggunakan *Secure Hash Algorithm*

Sayed Khaidir Ali[1], Hendrawaty[2], Azhar[3]
[1,3]*Jurusan Teknologi Informasi dan Komputer, Politeknik Negeri Lhokseumawe,*
Jln. B.Aceh Medan Km. 280 Buketrata 24301 INDONESIA

[1] said_khaidirali@yahoo.com
[2] waty.hendra@yahoo.com
[3] tgkazhar@yahoo.com

Abstrak*— Suatu berkas dapat dikatakan asli jika tidak mengalami perubahan. Pada layanan *cloud storage*, berkas dapat dikelola oleh penyedia layanan maupun pihak ke-tiga (Third Party Auditor), sehingga berkas tersebut tidak dapat dijamin keasliannya. Oleh karena itu pada penelitian ini dirancang dan dibangun sistem validasi berkas yang nantinya dapat diterapkan pada layanan *cloud storage*. Algoritma yang digunakan untuk menjaga keutuhan pada berkas adalah *Secure Hash Algorithm* 256-bit (*SHA-256).* dengan sistem ini diharapakan data/berkas yang di *upload* dan di *download* oleh para pengguna pada media penyimpanan berkas dapat dibuktikan keutuhannya. Dari pengujian dilakukan sebanyak 20 buah berkas berupa 10 buah berkas yang tidak mengalami modifikasi dan 10 buah berkas yang mengalami modifikasi pada berbagai ekstensi seperti **.doc, *.mp3, *.jpg, *.rar, *.txt,* maka system dinyatakan 100% berhasil tanpa mengalami kesalahan dalam melakukan validasi data maupun kegagalan *user interfaces.

Kata Kunci*: Cloud Storage, Data Integrity, Secure Hash Algorithm,* Validasi Berkas

Abstract*— A file can be said to be original if it does not change. In cloud storage services, files can be managed by the service provider or third party (Third Party Auditor), so the file can't be guaranteed authenticity. Therefore, this research is designed and built a file validation system that can be applied to cloud storage services. The algorithm used to maintain the integrity of the file is a 256-bit Secure Hash Algorithm (SHA-256). With this system expected data/files uploaded and downloaded by the users on the cloud data storage can be proved its integrity. From the test conducted as many as 20 pieces of files in the form of 10 pieces of unmodified files and 10 pieces of files that have modifications in various extensions such as * .doc, * .mp3, * .jpg, * .rar, * .txt, then the system is declared 100 % Succeed without experiencing errors in data validation or failure of user interfaces.

Keyword: Cloud Storage, Data Integrity, File Validation, Secure Hash Algorithm.

I. PENDAHULUAN

Cloud storage adalah sebuah konsep baru yang muncul bersamaan dengan teknologi *cloud computing*. *Cloud storage* dapat diartikan sebagai suatu layanan yang disediakan oleh *cloud service provider* sebagai sumber daya penyimpanan yang dapat digunakan oleh setiap pengguna melalui jaringan internet [1].

Pada layanan *cloud storage* terdapat kekurangan yaitu masalah keutuhan data, dimana data yang disimpan pengguna pada layanan *cloud storage* akan dikelola atau diketahui oleh penyedia layanan.

Tujuan dari penelitian ini adalah membuat sebuah system validasi berkas dengan memanfaatkan algoritma *sha-256* untuk menjamin bahwa berkas yang disimpan pada layanan cloud storage masih asli. Sehingga hasil penelitian ini nantinya dapat membantu pihak-pihak penyedia layanan penyimpanan berkas dalam memberikan keamanan pada layanan yang mereka berikan.

Sebelum tersedia teknologi *cloud computing*, data dan perangkat lunak harus disimpan dan diproses pada komputer yang sama. Tapi saat ini, *cloud computing* memungkinkan pemisahan fungsional antara sumber daya yang digunakan dan komputer pengguna, yang membuat segalanya lebih fleksibel dan mudah. Prinsip di balik *cloud* adalah bahwa setiap komputer yang terhubung ke Internet terhubung ke aplikasi, daya komputasi dan file yang sama. Pengguna dapat menyimpan dan mengakses file pribadi, seperti gambar, musik, video dan bookmark, atau melakukan pengolah kata di server jauh daripada membawa secara fisik sekitar media penyimpanan seperti pemutar MP3 atau DVD. Ini menjadi solusi untuk meningkatnya biaya penyimpanan dan pemeliharaan untuk perusahaan IT dan memungkinkan pelanggan meningkatkan kemampuan dan kapasitas mesin mereka saat dalam perjalanan.[2]

Cloud computing mempunyai 3 jenis layanan yaitu *SaaS* (*Software as a service*), *PaaS* (*Platform as a Service*) dan *IaaS* (*Infrastructure as a Service*) [3].
SaaS (Software as a Service) adalah Layanan yang menyediakan aplikasi jadi / siap pakai kepada *End user*. Ciri dari layanan ini adalah *user* tidak perlu membuat

aplikasi, tidak perlu menyiapkan tempat dan juga infrastruktur. Contoh SaaS adalah gmail, ymail, facebook, twitter, dropbox. Atau yang berbayar seperti salesforce, office365, dan sebagainya. [3]

PaaS (Platform as a Service) adalah Layanan yang menyewakan "tempat" untuk menjalankan aplikasi dari *user*. Tempat yang dimaksud seperti sistem operasi, *database*, *framework*, dan sebagainya yang merupakan wadah untuk berjalannya aplikasi. Ciri dari layanan ini adalah *user* tidak perlu melakukan *maintenance* dan tidak perlu menyiapkan infrastruktur. Sehingga *user* dapat tetap fokus membangun aplikasinya. Contoh *Pass* adalah *Windows Azure*, *Amazon Web Service*, *GoogleApp Engine*. [3]

*IaaS (Infrastructure as a Service) adalah l*ayanan yang menyewakan infrastruktur IT kepada *user* yang ingin membangun layanan *cloud*. Infrastruktur disini bersifat fisik, bisa berupa memory, penyimpanan, *server*, jaringan, dan sebagainya. Hal-hal seperti membuat aplikasi dan konfigurasinya diserahkan kepada *user*. *Cloud provider* hanya menyediakan infrastruktur berdasarkan *request* dari *user*. Ciri layanan ini adalah jika *user* ingin mengupgrade memory atau menambah *server*, *user* tinggal menghubungi *provider* kemudian *provider* akan menyediakan sesuai dengan permintaan. Contoh *IaaS* adalah *Amazon EC2, Rackspace cloud*. [3]

Suatu proses penyimpanan data di server cloud jarak jauh (remote) dapat dikatakan *cloud storage*. penyimpanan berbasis cloud storage lebih baik dari teknologi penyimpanan pada umumnya karena alasan sebagai berikut [4]:

- Pengguna tidak perlu melakukan instalasi perangkat penyimpanan fisik.
- Keperluan perawatan terhadap penyimpanan, seperti *backup*, ataupun penambahan kapasitas penyimpanan merupakan tanggung jawab penyedia layanan,
- Pengguna hanya perlu membayar jumlah kapasitas penyimpanan yang digunakan.

II. METODOLOGI PENELITIAN

Pada bulan Agustus 1991, *NIST* (*The National Institute of Standard and Technology*) mengumumkan bakuan (*standard*) untuk tanda-tangan *digital* yang dinamakan *Digital Signature Standard* (DSS). *DSS* terdiri dari dua komponen, yang pertama adalah algoritma tanda-tangan digital yang disebut *Digital Signature Algorithm (DSA)*, dan yang kedua adalah fungsi *hash standard* yang disebut *Secure Hash Algorithm (SHA). SHA* adalah fungsi *hash* satu-arah yang dibuat oleh *NIST* dan digunakan bersama *DSS* (*Digital Signature Standard*). Oleh *NSA*, *SHA* dinyatakan sebagai *standard* fungsi *hash* satu-arah. *SHA* didasarkan pada *MD4* yang dibuat oleh *Ronald L. Rivest* dari *MIT* [5].

Langkah-langkah pembangkitan nilai *hash* untuk algoritma SHA-256 sebagai berikut:

- *Message Conversion*: Pada tahap pertama, pesan masukan dikonversi kedalam bentuk bilangan biner.
- *Message Padding*: Pada tahap kedua, pesan yang sudah dikonversi kedalam bentuk binary disisipkan dengan angka 1 dan ditambahkan bit-bit pengganjal yakni angka 0 hingga panjang pesan tersebut kongruen dengan 448 modulo 512. Panjang pesan yang asli kemudian ditambahkan sebagai angka biner 64 bit. Setelah itu maka panjang pesan sekarang menjadi kelipatan 512 bit.
- *Message Parsing*: Pesan yang sudah dipadding tadi kemudian dibagi menjadi N buah blok 512 bit: M(1), M(2), ..., $M^{(N)}$.
- *Message Expansion*: Masing-masing blok 512-bit tadi lalu dipecah menjadi 16 buat *word* 32-bit: $M_0^{(i)}$, $M_1^{(i)}$, ..., $M_{15}^{(i)}$ yang mana nantinya diperluas menjadi 64 *word* yang diberi label W_0, W_1, ..., W_{63} dengan aturan tertentu yang sudah ditentukan sebelumnya oleh standar SHA-2.
- *Message Compression*: Masing-masing dari 64 *word* yang diberi label W_0, W_1, ..., W_{63} tadi kemudian diproses dengan algoritma fungsi *hash* SHA-256. Dalam proses tersebut, inti utama dari algoritma SHA-256 adalah membuat 8 variabel yang diberikan nilai untuk nilai awal dari $H_0^{(0)}$-$H_7^{(0)}$ di awal masing-masing fungsi *hash*.
- *Hash Function*: Menyiapkan fungsi-fungsi logika matematis yang menjadi standar perhitungan nilai *hash* untuk algoritma SHA-256. Setiap fungsi tersebut dijabarkan seperti berikut: [5]

$$Ch(x,y,z) = (x \wedge y) \oplus (\neg x \wedge z) \quad (1)$$

$$Maj(x,y,z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z) \quad (2)$$

$$\sum_0^{\{256\}}(x) = ROTR^2(x) \oplus ROTR^{13}(x) \oplus ROTR^{22}(x) \quad (3)$$

$$\sum_1^{\{256\}}(x) = ROTR^6(x) \oplus ROTR^{11}(x) \oplus ROTR^{25}(x) \quad (4)$$

$$\sigma_0^{\{256\}}(x) = ROTR^7(x) \oplus ROTR^{18}(x) \oplus SHR^3(x) \quad (5)$$

$$\sigma_1^{\{256\}}(x) = ROTR^{17}(x) \oplus ROTR^{19}(x) \oplus SHR^{10}(x) \quad (6)$$

- *Initial Hash Value*: Menyiapkan 8 variabel awal (*initial*) sebelum memulai proses perhitungan nilai *hash*. Nilai-nilai awal tersebut adalah sebagai berikut:

Tabel 1. *Initial Hash Value of SHA-256*

Label	Value
A=$H_0^{(0)}$	6a09e667
B=$H_1^{(0)}$	bb67ae85
C=$H_2^{(0)}$	3c6ef372
D=$H_3^{(0)}$	a54ff53a
E=$H_4^{(0)}$	510e527f
F=$H_5^{(0)}$	9b05688c
G=$H_6^{(0)}$	1f83d9ab
H=$H_7^{(0)}$	5be0cd19

- *Define SHA-256 Constant*: menyiapkan koefisien SHA-256 yang sudah ditetapkan pada standar SHA-2. Koefisien-koefisien tersebut adalah 64 buah bilangan hexadecimal yang didapatkan dari akar kubik dari nilai pecahan 32-bit dari 64 bilangan prima pertama.
- *Message Schedule*: Menyiapkan *message schedule* dari 16 buah word hasil parsing menggunakan rumus berikut: [5]

$$W_t = \begin{cases} M_t^{(i)} & 0 \le t \le 15 \\ \sigma_1^{\{256\}}(W_{t-2}) + W_{t-7} + \sigma_0^{\{256\}}(W_{t-15}) + W_{t-16} & 16 \le t \le 63 \end{cases} \quad (7)$$

- *Hash Calculation*: Algoritma ini melakukan perhitungan sebanyak 64 kali putaran untuk setiap perhitungan blok. Delapan variabel yang diberi label A, B, C, ..., H tadi nilainya terus berganti selama perputaran sebanyak 64 kali putaran sebagai berikut:

For t=0 to 63:
{
$T_1 = h + \sum_1^{\{256\}}(e) + Ch(e,f,g) + K_t^{\{256\}} + W_t$
$T_2 = \sum_0^{\{256\}}(a) + Maj(a,b,c)$
$h = g$
$g = f$
$f = e$
$e = d + T_1$
$d = c$
$c = b$
$b = a$
$a = T_1 + T_2$
}

Gambar 1 Perhitungan *Hash*

- Setelah perputaran sebanyak 64 kali tadi, nilai *hash* $H^{(i)}$ kemudian dihitung sebagai berikut:

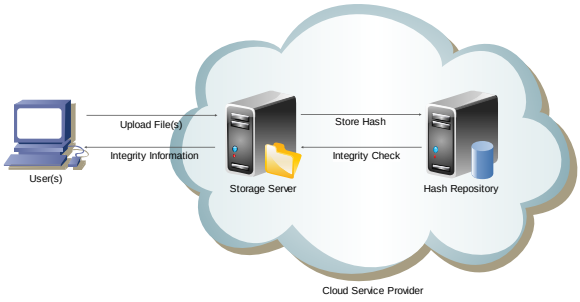
$$H_0^{(i)} = a + H_0^{(i-1)}$$
$$H_1^{(i)} = b + H_1^{(i-1)}$$
$$H_2^{(i)} = c + H_2^{(i-1)}$$
$$H_3^{(i)} = d + H_3^{(i-1)}$$
$$H_4^{(i)} = e + H_4^{(i-1)}$$
$$H_5^{(i)} = f + H_5^{(i-1)}$$
$$H_6^{(i)} = g + H_6^{(i-1)}$$
$$H_7^{(i)} = h + H_7^{(i-1)}$$

Gambar 2. Penjumlahan dengan *Initial Hash Value*

- Selanjutnya hasil akhir SHA-256 didapat dari penggabungan delapan variabel yang tadi sudah dikomputasi.

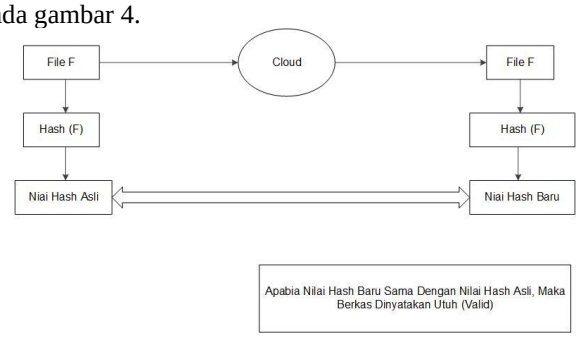
A. Ilustrasi Sistem Validasi Berkas

Sistem validasi berkas merupakan suatu proses melakukan validasi terhadap berkas yang tersimpan pada layanan *cloud storage*. validasi berkas dilakukan untuk membuktikan keutuhan berkas yang telah di upload oleh user kedalam layanan *cloud storage* dengan cara membandingkan nilai hash yang terdapat pada *hash repository* dengan nilai hash yang dibangkitkan ulang dari file tersebut.Ilustrasi sistem validasi berkas dapat dilihat pada gambar 3.



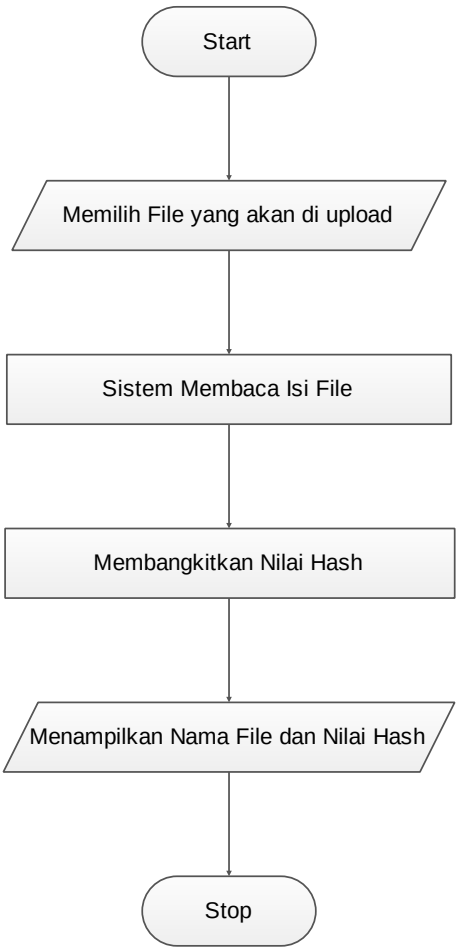
Gambar 3. Ilustrasi Sistem Validasi Berkas

Blok diagram system validasi berkas dapat dilihat pada gambar 4.



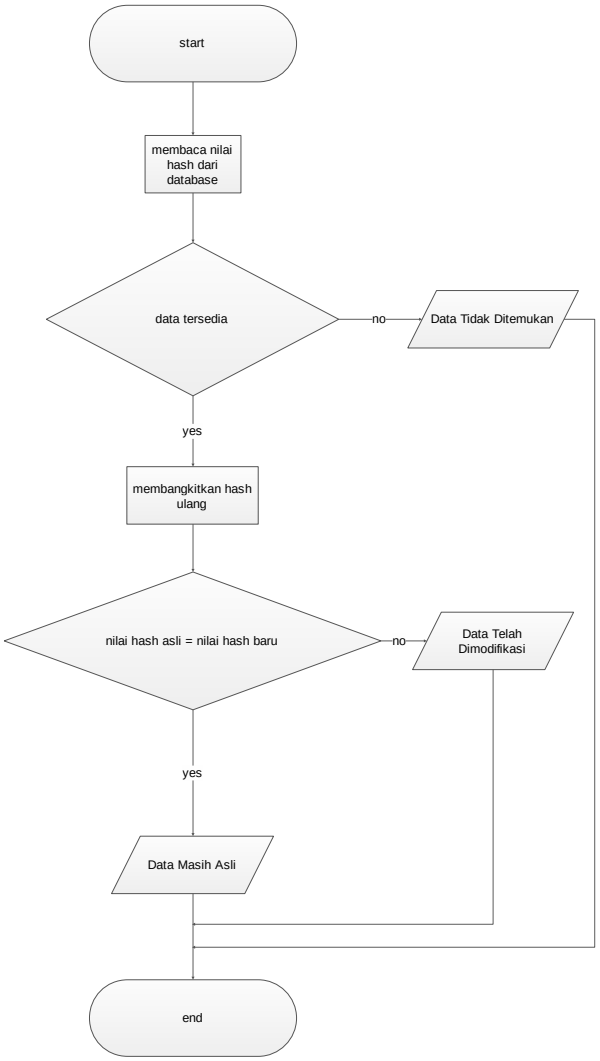
Gambar 4. Blok Diagram Validasi Berkas

User melakukan *upload* dengan cara memilih file yang akan di *upload*, lalu men-drag file tersebut pada kolom *upload* yang terdapat pada form *upload*. Proses pembangkitan *hash* pada saat melakukan *upload* diawali dengan membaca content dari file tersebut dan di simpan dalam bentuk kumpulan string (array of string). Lalu content tersebut dikonversi kedalam bentuk biner (binary string), lalu sistem melakukan proses pembangkitan nilai *hash* berdasarkan content yang sudah di ubah ke bentuk biner. flowchart proses upload dapat dilihat pada gambar 5.



Gambar 5. Flowchart proses upload

Flowchart system validasi berkas dapat dilihat pada gambar 6.



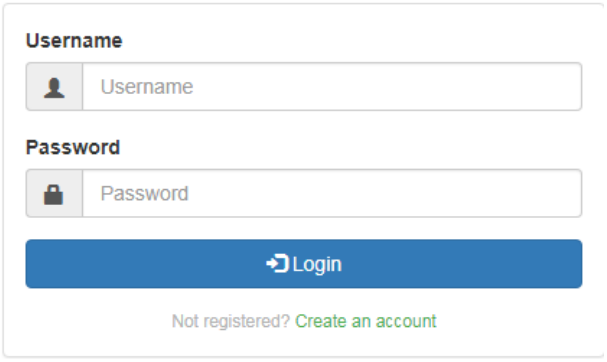
Gambar 6. Flowchart Sistem Validasi Berkas

III. Hasil dan Pembahasan

Pada tahap ini melakukan pengujian dan memaparkan hasil dan pembahasan dari aplikasi yang telah di bangun, yang terdiri dari hasil pengujian *user interface*, *hasil* pengujian database, dan hasil pengujian algoritma *SHA* (fungsi *hash).*

A. Tampilan Halaman Login

Pada saat *user* membuka aplikasi, *user* harus melakukan login terlebih dahulu agar dapat mengakses ke halaman utama aplikasi *cloud storage*. Tampilan login dapat dilihat pada Gambar 7.



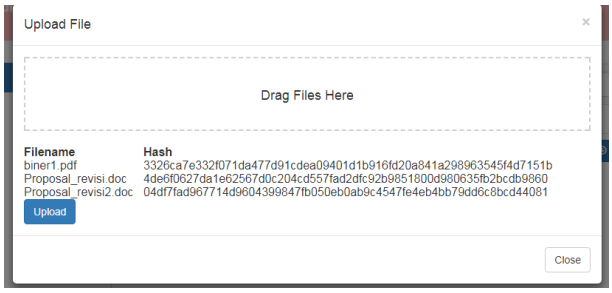
Gambar 7. Halaman Login

Dapat dilihat pada gambar 4.1 bahwa jika *user* ingin masuk ke dalam aplikasi, *user* harus melaukan login terlebih dahulu. *Username* dan *password* yang dimasukkan harus seusia dengan data yang ada di database. Sehingga jika *user* belum melakukan registrasi, maka *username* dan *password user* tersebut belum terdaftar di dalam database. Sehingga *user* tersebut tidak bisa masuk ke dalam *cloud* tersebut untuk melakukan berbagai proses sesuai dengan keinginan *user*.

B. Tampilan Halaman My Files

Halaman ini berfungsi untuk menampilkan data-data yang telah di *upload* oleh pengguna layanan. Halaman ini muncul setelah *user* melakukan *login* ke dalam aplikasi. Pengguna dapat meng*upload* data pada pada tombol *upload file* dengan cara melakukan *drag and drop* pada kolom yang di sediakan seperti ditunjukkan pada gambar 9. Berikut adalah tampilan halaman *My Files.*

Gambar 8. Halaman *My Files*



Gambar 9. Form Upload

C. Pengujian Validasi Berkas

Pada bagian ini akan membahas hasil pengujian validasi terhadap beberapa jenis *file* dengan menggunakan aplikasi yang cocok untuk mengubah isi *file* tersebut seperti *audacity* untuk modifikasi *file .mp3*, *notepad++* untuk modifikasi *file text*, dan sejenisnya untuk membuktikan keberhasilan algoritma yang diterapkan untuk melakukan validasi terhadap *file* yang asli maupun yang sudah di modifikasi.

Pengujian *file* dilakukan dengan meng*upload file* ke *server* cloud, lalu melakukan validasi terhadap *file* tersebut sehingga dapat dilihat hasil dari validasi. Kemudian melakukan modifikasi terhadap *file* tersebut menggunakan aplikasi yang sesuai dengan type *file.*

Berikut adalah hasil pengujian validasi terhadap file dokumen dengan extensi file type .doc.

1. Pengantar

   Filename: *Proposal_revisi.doc*

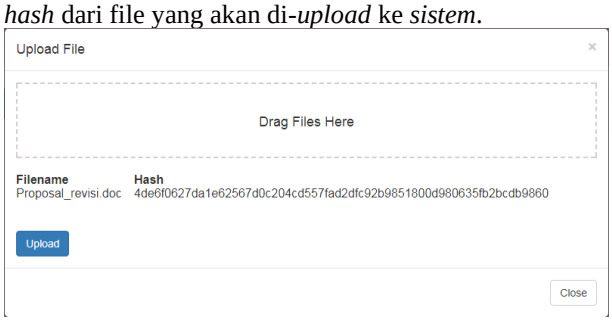
   File location: D:/datauji

   Application: *Microsoft Word 2007*

   File yang telah di-*upload*, akan dimodifikasi menggunakan *Microsoft Word 2007*, lalu melakukan validasi ulang terhadap *file* untuk memastikan bahwa *file* telah dimodifikasi. Ini bertujuan untuk mengetahui apabila terjadi perubahan atau modifikasi terhadap *file*, maka sistem akan memberi informasi kepada penggunanya bahwa *file* tidak valid.

2. Pengujian

   Berikut adalah uji coba *upload* dan melihat nilai *hash* dari file yang akan di-*upload* ke *sistem*.



Gambar 10. Pengujian *upload file* type .doc

File yang telah di *upload* terlihat pada gambar 10. telah dibangkitkan nilai *hash* menggunakan *SHA-256.*

3. Hasil

   Berikut adalah hasil validasi file yang masih asli dan juga file yang telah dimodifikasi. Pengujian tersebut bertujuan untuk memastikan keberhasilan algoritma yang diterapkan pada layanan *cloud storage*. Pada gambar 11 menunjukkan hasil nilai *hash* yang sama dengan nilai *hash* dari file yang asli.

Gambar 11. Hasil Validasi file yang asli

Jika file milik pengguna yang tersedia di cloud telah mengalami modifikasi maka nilai *hash* yang akan di bangkitkan menjadi berbeda dengan nilai *hash* aslinya, seperti pada gambar 12.

Gambar 12. Hasil Validasi file yang telah dimodifikasi

IV. KESIMPULAN

Dari penelitian yang telah selesai dilaksanakan tentang aplikasi sistem validasi berkas pada layanan menggunakan *secure hash alrogithm*, maka dapat disimpulkan bahwa:

1. User interface pada aplikasi ini seperti Halaman Login, Halaman Register, Halaman My Files, Halaman Profiles, Form *Upload*, Form Validasi, Form Ubah Profil, dan Form Ubah Password sudah dapat berfungsi sebagaimana yang diharapkan.
2. Proses *upload* dan *download* file dengan format .doc, .rar, .jpg, .txt pada aplikasi ini sudah berhasil.
3. Proses pembangkitan nilai *hash* dari file.doc, .rar, .jpg, .txt telah berhasil dilakukan, baik itu pada saat proses *upload* file dan pada saat *download* file.
4. Proses validasi untuk melihat apakah file (.doc, .rar, .jpg, .txt) yang telah tersimpan mengalami perubahan karena adanya modifikasi telah berhasil dengan tingkat keberhasilan 100%

1. terpendek terhadap kata yang di-*input*-kan walaupun kata yang diinputkan tidak terdapat didalam *database*.
2. *Output* sistem yang dihasilkan tergantung juga dengan jumlah dan variasi kosa kata yang terdapat pada *database*. Semakin banyak jumlah dan variasi kata pada *database*, maka *output* sistem yang dihasilkan akan semakin akurat.

REFERENSI

[1] J. Wu, J. Fu, Z. Lin and J. Zhang, "A Survey on Cloud Storage," *JOURNAL OF COMPUTERS, VOL. 6, NO. 8,* pp. 1764-1771, 2011.

[2] N. Alsulami, E. Alharbi and M. Mostafa Monowar, "A Survey on Approaches of Data Confidentiality and Integrity Models in Cloud Computing System," *Journal of Emerging Trends in Computing and Information Sciences Vol. 6 No. 3,* pp. 188-197, 2015.

[3] S. Haxhijaha, G. Bajrami and F. Prekazi, "Data Integrity Check using Hash Functions in Cloud environment," Faculty of Computer Science and Engineering, University for Business and Tecnology, Austria, 2012.

[4] A. G. Tammam, D. B. Kurniawan, H. Arifunas, M. I. Aziz, A. P. D and R. D. Cahyono, Fungsi Hash dan Algoritma SHA-256, Kediri: Universitas Nusantara PGRI Kediri, 2016.

[5] S. Eswaran and D. S. Abburu, "Identifying Data Integrity in the Cloud Storage," *International Journal of Computer Science Issues,* pp. 403-408, 2013.