# Pengamanan Sistem Menggunakan *One Time Password* Dengan Pembangkit Password Hash SHA-256 dan *Pseudo Random Number Generator* (PRNG) *Linear Congruential Generator* (LCG) di Perangkat Berbasis Android

**Dolly Virgian Shaka Yudha Sakti[1], Nazori Agani[2], Mardi Hardjianto[3]**
[1,2,3]*Magister Ilmu Komputer*
*Universitas Budi Luhur, Jakarta, 12260*
*Telp : (021) 5853753 ext 253, Fax : (021)5853752*
[1]*dollyvirgian.shaka@budiluhur.ac.id,* [2]*nazori @budiluhur.ac.id,* [3]*mardi.hardjianto@budiluhur.ac.id*

***Abstract***

*The prevention of access to web-based system can be done with authentication, i.e. the use of passwords, but the use of static passwords still allows the password theft by hackers. Security system with One Time Password (OTP) method can be one of the solutions to these problems. OTP password can only be used one time and with a short time limit, if not used immediately, then the password will expire and cannot be used again. OTP applications use the user ID, the mobile phone number is registered, and the access time as a variable to generate a password using SHA-256 hash function and selected six characters at random using one of the methods on the Pseudo Random Number Generator (PRNG) that is a Linear Congruential Generator (LCG). With increasing system using One Time Password, then the risk of password theft by unauthorized persons can be minimised.*

***Keywords:*** *One Time Password, SHA-256, Pseudo Random Number Generator, Linear Congruential Generator*

***Abstrak***

*Pencegahan akses sistem berbasis web dapat dilakukan dengan otentikasi, yaitu menggunakan password, tetapi penggunaan password statis tetap memungkinkan terjadinya pencurian password oleh hacker. Pengamanan Sistem dengan metode One Time Password (OTP) dapat menjadi salah satu solusi permasalahan tersebut. Password OTP hanya dapat digunakan satu kali dan dengan batas waktu yang singkat, jika tidak segera digunakan, maka password akan kadaluarsa dan tidak dapat digunakan lagi. Aplikasi OTP menggunakan ID pengguna, nomor handphone yang teregistrasi, dan waktu akses sebagai variable untuk men-generate password menggunakan fungsi hash SHA-256 dan dipilih enam karakter secara acak menggunakan salah satu metode yang ada pada Pseudo Random Number Generator (PRNG) yaitu Linear Congruential Generator (LCG). Dengan peningkatan sistem menggunakan One Time Password, maka resiko pencurian password oleh orang yang tidak berwenang dapat diminimalisir.*

***Kata Kunci :*** *One Time Password, SHA-256, Pseudo Random Number Generator, Linear Congruential Generator*

## I. PENDAHULUAN

Kemudahan dalam mengakses data dan informasi menjadi salah satu alasan kuat yang membuat perusahaan menggunakan sistem informasi berbasis web[1]. Sistem informasi berbasis web berjalan di atas jaringan komputer yang saling terhubung. Hal ini yang membuat seluruh data dan informasi dapat diakses dengan mudah oleh siapa saja. Tidak hanya dapat diakses oleh orang yang berkepentingan, tetapi juga oleh orang yang ingin mencuri data dan informasi dari sistem tersebut [2]. Untuk itu komputer harus dapat mengetahui, pengguna yang akan menggunakan sistem adalah pihak yang berkepentingan. Pengguna harus mengidentifikasikan dirinya ke komputer, dan komputer harus memastikan apakah identifikasi tersebut otentik atau tidak.

Salah satu metode yang banyak digunakan untuk otentikasi adalah dengan menggunakan password. Akan tetapi terkadang pengguna kurang waspada terhadap password yang dimiliki sehingga terjadi pencurian password. Misalnya pemilihan password yang sudah umum digunakan seperti tanggal lahir, hal ini menyebabkan orang lain akan dapat dengan mudah menebak password yang dibuat. Pada kasus lain pengguna menuliskan passwordnya pada media tertulis yang kemudian secara sengaja atau tidak sengaja dapat dibaca pihak lain[3].

Terkadang ditemui pula masalah non teknis berupa penggunaan password lebih dari satu orang[4]. Selain itu banyak pengguna yang sulit dalam menghafalkan password yang dimiliki karena terlalu banyak akun dan password yang pernah dibuat. Baik akun untuk email, sosial media, dan lain lain.

Beberapa solusi telah diajukan dan dikembangkan, diantaranya adalah penggunaan token[3]. Token akan menghasilkan *password*

yang hanya bisa digunakan sekali, pada banyak penelitian hal ini disebut *One Time Password* atau OPT. Pembangkitan *password* OTP dilakukan dengan beberapa cara, ada yang menggunakan SMS dan ada juga yang menggunakan aplikasi Mobile. Pada penelitian yang dilakukan oleh Santoso pembangkitan password dilakukan oleh sistem lalu dikirimkan ke pengguna melalui SMS[5]. *Password* yang diterima melalui SMS digunakan untuk otentikasi dan akan dicocokkan dengna yang ada di sistem. Sedangkan pada penelitian yang dilakukan oleh Mustofa, terdapat *generator* yang akan menerima masukan kode rahasia *user* dan *chalange* dari *server* kemudian menghasilkan OTP[6].

Pada kebanyakan penelitian termasuk yang dilakukan oleh Acharya, Santoso dan Mustofa, pembangkitan password OTP diambil dari beberapa karakter awal atau awal hasil algoritma fungsi Hash. Agar lebih kuat, pengambilan karakter dari fungsi Hash perlu diacak. Pengambilan karakter acak dapat dilakukan dengan *Pseudo Random Number Generator* atau PRNG.

## II. STUDI LITERATUR

Penelitian ini mengacu pada beberapa penulisan terkait penelitian yang telah dilakukan sebelumnya, yaitu sebagai berikut:

1. Penelitian yang dilakukan oleh Shally dan Gagangget Singh Aujla dengan judul penelitian "*A Review of One Time Password Mobile Verification*" menguraikan tentang peninjauan proses otentikasi pada *One Time Password* via SMS. *One Time Password* menggunakan aplikasi mobile dan proses otentikasi password statis. Dalam penelitian tersebut tidak dijelaskan secara jelas tentang hasil perbandingan antara tiga metode otentikasi yang dibahas, tetapi dapat diketahui bahwa penggunaan otentikasi One Time Password via SMS memiliki kelemahan yaitu dapat terjadinya intersepsi nirkabel atau Trojan ponsel[7].

2. Penelitian yang dilakukan oleh Khaled Alghathbar dan Hanan A. Mahmoud dengan judul penelitian "*Noisy Password Scheme: A New One Time Password Sistem*" membandingkan teknik otentikasi password statis dan teknik otentikasi berbasis gambar. Menurut mereka penggunaan password statis sangat rentan dicuri, sedangkan teknik otentikasi password berbasis gambar menyulitkan server karena perlu menyediakan penyimpanan gambar untuk masing-masing pengguna. Lalu mereka mengusulkan pembuatan skema One Time Password dengan algoritma yang mereka sebut dengan *noisy password technique*[8].

3. Sagar Acharya bersama dengan Apporva Polawar dan P. Y. Pawar dalam penelitian mereka yang berjudul "*Two Factor Authentication Using Smartphone Generated One Time Password*" mengusulkan dibuatnya sebuah aplikasi yang diinstall pada perangkat mobile untuk meng-generate password OTP secara offline sebagai pengganti perangkat token. Mereka juga membahas bagaimana OTP dibuat berdasarkan beberapa parameter yaitu, username, password, tanggal lahir pengguna, ditambahkan dengan tanggal dan waktu akses password lalu dimasukkan ke dalam fungsi Hash SHA-1 yang akan mengembalikan nilai 20 byte karakter ascii. Dari nilai yang dikembalikan fungsi hash SHA-1 diambil sepuluh karakter untuk digunakan sebagai One Time Password[3].

4. Berdasarkan penelitian yang dilakukan oleh R. Selva Bhuvaneshwari dan P. Anuja yang berjudul "*Secured Password Management Technique Using One-Time Password Protocol in Smartphone*" diusulkan sebuah protocol atau aturan One Time Password pada otentikasi sebuah sistem. Mereka mengusulkan pembuatan aplikasi untuk menghasilkan OTP pada perangkat mobile android. Pembentukan password OTP dibuat menggunakan fungsi Hash MD5. Aplikasi berbasis mobile android yang digunakan untuk membuat OTP juga diamankan menggunakan Long-Time Password untuk memastikan pengguna perangkat mobile tersebut adalah benar pemilik dari perangkat[9].

5. Penelitian berjudul "*An Approach for User Authentication One Time Password (Numeric And Graphical) Scheme*" yang dibuat oleh Brajesh Kumar Kushwaha membahas tentang skema pembuatan password satu kali pakai (OTP) dengan memadukan antar gambar dan karakter. Penggunaan gambar dalam penginputan password akan membantu pengguna dalam mengingat password yang telah dibuat. Tetapi di dalam penelitian ini tidak begitu jelas skema pembuatan atau generate password OTP yang dilakukan[10].

6. Analisa perbandingan antara desain model otentikasi password plain text dengan OTP diuraikan oleh Adang Suhendra, Anne Yulianti, Bisma Junatas dan Vega Valentine dalam penelitian yang berjudul "*Modified Authentication using One Time Password to Support Web Services Security*". Dengan semakin banyaknya penggunaan sistem berbasis web, semakin meningkat pula masalah keamanan yang perlu ditangani. Pada penelitian ini peneliti mencoba untuk memberikan informasi tentang pengamanan yang terjadi di layanan web. Penelitian ini juga mengusulkan desain otentikasi One Time Password (OTP) sebagai teknik untuk meningkatkan keamanan[11].

7. R. Mohan dan N. Partheeban melakukan penelitian berjudul "*Secure Multimodal Mobile Authentication Using One Time Password*". Mereka menyatakan bahwa kebanyakan sistem saat ini bergantung pada password statis untuk memverifikasi identitas pengguna. Pengguna cenderung menggunakan password yang mudah ditebak, password yang sama di beberapa akun, serta menulis dan menyimpannya pada mesin mereka. Hal ini menjadi masalah ketika ada orang lain mencuri password mereka. Dalam penelitian ini diusulkan pengunaan OTP (One Time Password) untuk otentikasi. Password yang dihasilkan OTP hanya berlaku selama 3 menit, jika tidak segera digunakan, maka masa berlakunya akan habis. Jika percobaan otentikasi dilakukan lebih dari 8 kali salah berturut turut, pengguna akan dikunci untuk sementara waktu[12].

8. Penelitian mengenai pengamanan login web menggunakan One Time Password dilakukan oleh Kartika Imam Santoso dengan judul "*Dua Faktor Pengamanan Login Web Menggunakan Otentikasi One Time Password Dengan Hash SHA*". Pengamanan login untuk mengakses halaman web, berupa pengamanan menggunakan OTP yang dibangkitkan dengan Hash SHA menghasilkan sebuah kode lewat SMS untuk otentikasi. Aplikasi OTP menggunakan NIM, No Telp, dan waktu akses untuk masukan fungsi Hash SHA yang kemudian menghasilkan 40 digit bilangan hexadecimal. Selanjutnya diambil enam digit dari bilangan tersebut. Enam digit bilangan hexadecimal tersebut dikirim sebagai OTP dengan layanan aplikasi Gammu berupa SMS. Waktu aktif untuk pengamanan login dengan OTP berbasis SMS ini berlaku selama tiga menit. Pembatasan tersebut untuk mempersempit waktu hacker untuk menyadap dan menyusup[5].

## III. STUDI PUSTAKA

### A. One Time Password

Password atau kata sandi dapat digunakan untuk layanan otentikasi, yaitu layanan yang berhubungan dengan identifikasi, baik mengidentifikasi kebenaran pihak-pihak yang berkomunikasi (*user authentication* atau *entity authentication*) maupun mengidentifikasi kebenaran sumber pesan. Dua pihak yang saling berkomunikasi harus dapat mengotentikasi satu sama lain sehingga ia dapat memastikan sumber pesan. Otentikasi sumber pesan secara implisit juga memberikan kepastian integritas data, sebab jika pesan telah dimodifikasi berarti sumber pesan sudah tidak benar.

*One Time Password* (OTP) adalah sebuah password yang hanya berlaku untuk sesi login tunggal atau transaksi tunggal. Berbeda dengan penggunaan password statis, OTP tidak menggunakan password yang sama untuk setiap login atau transaksi, sehingga jika pihak yang tidak berkepentingan berhasil merekam password OTP yang sudah digunakan maka dia tidak akan dapat menyalahgunakan password tersebut karena sudah tidak berlaku lagi. Untuk dapat membuat sebuah password OTP, digunakan salah satu metode kriptografi, yaitu fungsi hash, dan untuk pemilihan karakternya dipilih secara acak dengan *Pseudo Random Number Generator*.

### B. Fungsi *Hash*

Fungsi *hash* adalah fungsi yang menerima masukan string yang panjangnya sembarang dan mengkonversinya menjadi string keluaran yang panjangnya tetap (*fixed*). Fungsi hash dapat menerima masukkan string apa saja. Jika string menyatakan pesan (*message*), maka sembarang pesan M berukuran bebas dikompresi oleh fungsi *hash* H melalui persamaan:

**h=H(M)**

Keluaran fungsi hash disebut juga nilai hash (*hash-value*) atau pesan-ringkas (*message digest*). Pesan ringkas dinyatakan dalam kode heksadesimal yang panjangnya 128 bit. Satu karakter heksadesimal = 4 bit. Dua pesan yang berbeda akan selalu menghasilkan nilai hash yang berbeda pula. Sifat-sifat fungsi hash:

1) Fungsi H dapat diterapkan pada blok data berukuran berapa saja.
2) H menghasilkan nilai (h) dengan panjang tetap.
3) H(x) mudah dihitung untuk setiap nilai x yang diberikan.
4) Untuk setiap h yang diberikan, tidak mungkin menemukan x sedemikian sehingga H(x)=h. Itulah sebabnya fungsi H dikatakan fungsi hash satu arah.


**Gbr 1. Fungsi Hash Satu Arah**

### C. Algoritma Hash SHA-256

SHA-256 dirancang oleh The National Institute of Standards and Technology (NIST) pada tahun 2002. SHA-256 menghasilkan message digest dengan panjang 256 bit. SHA-256 aman karena didesain sedemikian rupa sehingga tidak memungkinkan mendapat pesan yang berhubungan dengan message digest, atau untuk menemukan dua pesan yang berbeda yang menghasilkan message digest yang sama. Proses untuk menghasilkan message digest pada algoritma ini adalah meliputi 5 tahapan:

1) *Message Padding*

Input pesan pada algoritma SHA-256 akan dibagi menjadi blok-blok yang masing-masing panjangnya adalah 512 bit. Akibat pembagian ini, maka jumlah blok terakhir akan lebih kecil atau sama dengan 512 bit. Blok terakhir tersebut akan mengalami *message padding*.

2) Penambahan panjang bit

Setelah proses *message padding*, jumlah bit pada blok terakhir adalah 448 bit. Representasikan M ke dalam bilangan biner untuk bit terakhir, agar total panjang blok terkahir 512 bit.

3) Inisialisasi nilai hash awal

Pada SHA-256 untuk menyimpan nilai inisialisasi awal dan nilai output sementara digunakan buffer $H_0$, $H_1$, $H_2$, $H_3$, $H_4$, $H_5$, $H_6$, $H_7$. Di sisi lain, untuk penyimpanan proses sementara digunakan buffer a, b, c, d, e, f, g, h. Nilai $H_0$, $H_1$, $H_2$, $H_3$, $H_4$, $H_5$, $H_6$, $H_7$ untuk inisialisasi awal dalam notasi heksadesimal.

4) Pemprosesan

Pemrosesan merupakan bagian inti yang terdiri atas 1 round yang mempunyai 64 operasi. Untuk memproses setiap satu blok pesan 512 bit diperlukan 64 operasi. Setiap blok pesan $M^{(1)}$, $M^{(2)}$, …, $M^{(N)}$ dengan N adalah jumlah blok pesan.

5) Output

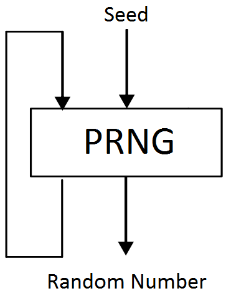
Output diperoleh setelah semua blok $M^{(N)}$ 512 bit diproses. Setelah semua langkah pemrosesan dilakukan sejumlah N kali, maka akan didapat 256 bit *message digest* untuk pesan M yaitu:

$$H_0^{(N)} \| H_1^{(N)} \| H_2^{(N)} \| H_3^{(N)} \| H_4^{(N)} \| H_5^{(N)} \| H_6^{(N)} \| H_7^{(N)}$$

D. *Pseudo Random Number Generator* (PRNG)

*Random Number Generator* (RNG) atau Pembangkit Bilangan Acak adalah suatu metode yang dirancang untuk menghasilkan suatu urutan nilai yang tidak dapat ditebak polanya dengan mudah, sehingga urutan nilai tersebut dapat dianggap sebagai suatu keadaan acak (random). Bilangan acak yang dihasilkan oleh komputer sekalipun tidak benar-benar acak dan kebanyakan bilangan acak yang diterapkan dalam kriptografi juga tidak benar-benar acak, tetapi hanya berupa acak semu. Dengan kata lain ilangan acak yang dihasilkan itu dapat ditebak susunan atau urutan nilainya. Dalam kriptografi, bilangan acak sering dibangkitkan dengan menggunakan pembangkit bilangan acak semu atau *Pseudo Random Number Generator* (PRNG).

Pseudo Random Number Generator (PRNG) atau Pembangkit Bilangan Acak Semu merupakan suatu algoritma yang menghasilkan suatu urutan nilai yang setiap hasil nilainya bergantung pada nilai yang dihasilkan sebelumnya. Output dari PRNG tidak betul-betul acak, hanya terlihat acak. Hal ini didukung oleh penelitian yang pernah dilakukan sebelumnya, yang menarik kesimpulan dari penelitian beberapa algoritma PRNG. Tidak ada algoritma yang benar benar dapat menghasilkan bilangan acak secara sempurna tanpa ada perulangan selama pembangkit yang digunakan adalah komputer yang memiliki sifat deterministik[13].




**Gbr 2. Pseudo Random Number Generator**

Secara umum, sebuah PRNG didefinisikan sebagai algoritma kriptografi yang digunakan untuk menghasilkan bilangan secara acak. Pengertian acak sendiri adalah bilangan yang dihasilkan dalam setiap perhitungan memiliki nilai yang berbeda. Sebuah PRNG memiliki sebuah kondisi awal X yang rahasia. Saat digunakan, PRNG harus membangkitkan output acak yang tidak dapat diidentifikasi oleh kriptanalis yang tidak tahu dan tidak dapat menebak kondisi awal X. Sebuah PRNG harus mampu mengubah kondisi awalnya dengan memproses input sehingga tidak dapat diprediksi oleh kriptanalis.

E. *Linear Congruential Generator* (LCG)

Salah satu algoritma *pseudo random number* yang tertua dan paling populer adalah *Linear Congruential Generator* (LCG). Algoritma ini diciptakan oleh D. H. Lehmer pada tahun 1951. Teori dari algoritma ini mudah dipahami dan dapat diimplementasikan secara cepat. Hal ini didukung oleh penelitian sebelumnya.yang menyimpulkan hasil analisis, diperoleh bahwa dari segi kecepatan LCG membutuhkan waktu yang paling pendek dalam menghasilkan bilangan acak dibandingkan dengan metode lain.

Algoritma ini memiliki empat parameter angka sebagai berikut[14]:

| | | |
|---|---|---|
| m | modulus | $m > 0$ |
| a | faktor pengali | $0 < a < m$ |
| c | *increment* | $0 \leq c < m$ |
| $X_0$ | Nilai awal (*seed*) | $0 \leq X_0 < m$ |
| $X_n$ | Bilangan acak ke n | |
| $X_{n-1}$ | Bilangan acak setelahnya | |

Random number diperoleh melalui persamaan berulang berikut:

$$\mathbf{X_{n+1} = (a.X_n + c) \bmod m. \quad n \geq 0}$$

Jika m, a, c, X 0 dan bilangan bulat, maka teknik ini akan menghasilkan urutan bilangan bulat dengan setiap bilangan bulat dalam kisaran 0 = Xn <m.

Penentuan nilai awal $X_0$ dan konstanta (a, b, dan m) akan menentukan kualitas bilangan acak yang dihasilkan. Bilangan acak yang baik (pada umumnya) apabila terjadinya perulangan atau munculnya bilangan acak yang sama, dapat terjadi setelah sekian banyak pembangkitan bilangan acak (semakin banyak akan semakin baik) serta tidak bisa diprediksi kapan terjadi perulangannya.
Periode LCG paling besar adalah M bahkan pada kebanyakan kasus periodenya kurang dari M. Maksudnya adalah deret bilangan acak yang dihasilkan tidak lebih banyak dari modulonya. Perhatikan contoh berikut.

Misalkan untuk $X1_n$ A = 5 , B = 13, M = 23 dan $X_0$ = 0 untuk $X2_n$ A = 4 , B = 12, M = 23 dan $X_0$ = 0

$$X1n = (5X1n\text{–}1 + 13) \bmod 23$$
$$X2n = (4X1n\text{–}1 + 12) \bmod 23$$

**Tabel 1. Linear Congruential Generator**

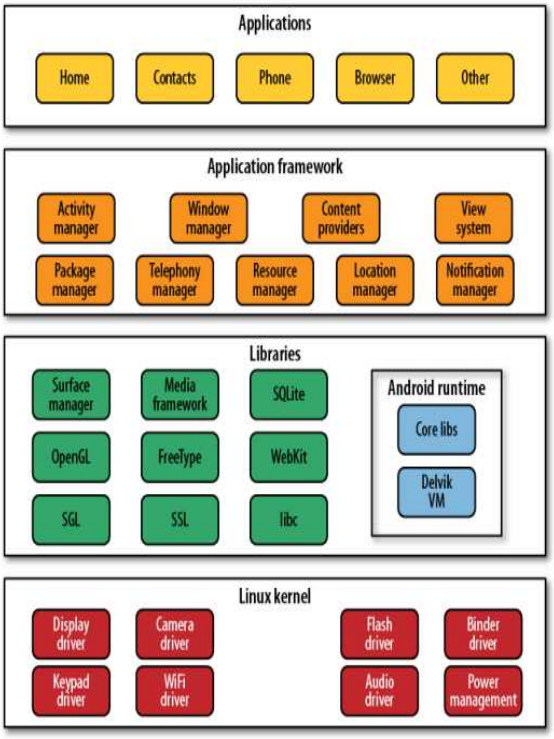
| n | X1n | X2n |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 13 | 12 |
| 2 | 9 | 14 |
| 3 | 12 | 22 |
| 4 | 4 | 8 |
| 5 | 10 | 21 |
| 6 | 17 | 4 |
| 7 | 6 | 5 |
| 8 | 20 | 9 |
| 9 | 21 | 2 |
| 10 | 3 | 20 |
| 11 | 5 | 0 |
| 12 | 15 | 12 |
| 13 | 19 | 14 |
| 14 | 16 | 22 |
| 15 | 1 | 8 |
| 16 | 18 | 21 |
| 17 | 11 | 4 |
| 18 | 22 | 5 |
| 19 | 8 | 9 |
| 20 | 7 | 2 |
| 21 | 2 | 20 |
| 22 | 0 | 0 |
| 23 | 13 | 12 |
| 24 | 9 | 14 |
| 25 | 12 | 22 |

Dari tabel 1 diatas, terlihat bahwa deret bilangan acak yang dihasilkan berulang setelah n tertentu. Untuk $X1_n$ deret berulang pada n yang ke 23, hal ini sama dengan nilai Mnya yang berarti periode untuk $X1_n$ adalah sebesar M atau dengan kata lain $X1_n$ mempunyai periode penuh. Sedangkan pada $X2_n$ kita dapat melihat bahwa periodenya kurang dari M, berulang pada n = 11. Pemilihan nilai **a** sebagai faktor pengali dan **c** sebagai *increment* mempengaruhi deret bilangan acak yang dihasilkan oleh algoritma ini[13].
LCG akan memiliki periode penuh jika memenuhi syarat sebagai berikut[15]:

1) c relatif prima terhadap M
2) a-1 dapat dibagi dengan semua faktor prima dari M
3) a-1 kelipatan 4 jika M kelipatan 4
4) nilai M lebih besar dari max(a,c,X0)
5) a > 0 dan c > 0

F. Sistem Operasi Android

Android adalah sistem operasi untuk telepon seluler berbasis Linux sebagai kernelnya. Perkembangan Android saat ini sangat pesat karena sifat platform Android yang terbuka (Opern Source) bagi para pengembang untuk menciptakan aplikasi mereka sendiri. Awalnya, perusahaan search engine terbesar saat ini, yaitu Google Inc. membeli Android Inc., pendatang baru yang membuat peranti lunak untuk ponsel. Hingga November 2013, Android menggunakan kernel yang berbasis kernel Linux versi 3.x (versi 2.6 pada Android 4.0 Ice Cream Sandwich dan pendahulunya).
Arsitektur kernel Linux pada Android telah diubah oleh Google, berbeda dengan siklus pengembangan kernel Linux biasa. Secara standar, Android tidak memiliki X Window Sistem asli ataupun dukungan set lengkap dari perpustakaan GNU standar. Oleh sebab itu, sulit untuk memporting perpustakaan atau aplikasi Linux pada Android.
Berikut ini gambaran arsitektur dari Sistem Operasi Android:

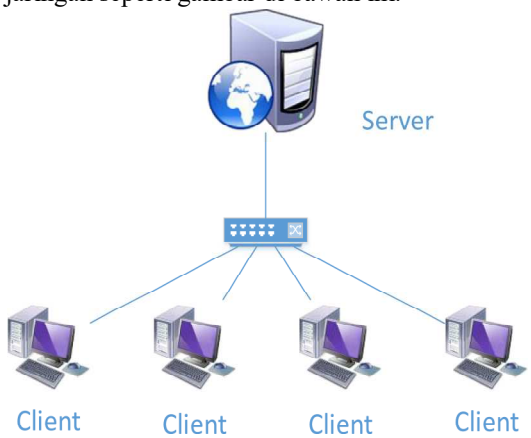



**Gbr 3. Arsitektur Sistem Operasi Android**

Setelah membuat struktur basis pengetahuan dan menentukan teknik inferensi pengetahuan selanjutnya adalah pembuatan sistem pakar. Selanjutnya menguji sistem pakar yang dibuat, agar dapat diketahui apakah sistem pakar telah sesuai dengan kebutuhan.

## IV. RANCANGAN APLIKASI

Sistem berbasis web yang berjalan menggunakan metode client-server. Sistem ditanamkan ke dalam sebuah server yang siap melayani permintaan service dari client melalui sebuah jaringan seperti gambar di bawah ini.

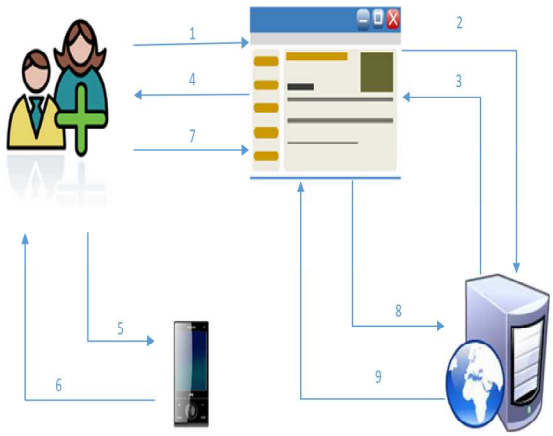



**Gbr 4. Jaringan Client Server**

Dengan jaringan client server seperti Gambar III.2 setiap client yang terhubung dengan server dapat dengan mudah mendapat akses sistem yang berada di server. Metode pengamanan yang dibuat yaitu dengan membuat otentikasi password. Password yang harus dimasukkan adalah hasil generate password dari aplikasi Mobile OTP berdasarkan kode Challange yang diberikan oleh sistem. Kode Challange yang diberikan oleh sistem, dibuat dari ID pengguna, nomor handphone milik pengguna, dan Time stamp (waktu akses) yang dilakukan hash menggunakan fungsi SHA-256 dan dipilih delapan karakter secara acak menggunakan *Pseudo Random Number Generator* (PRNG) dengan metode *Linear Congruential Generator* (LCG).

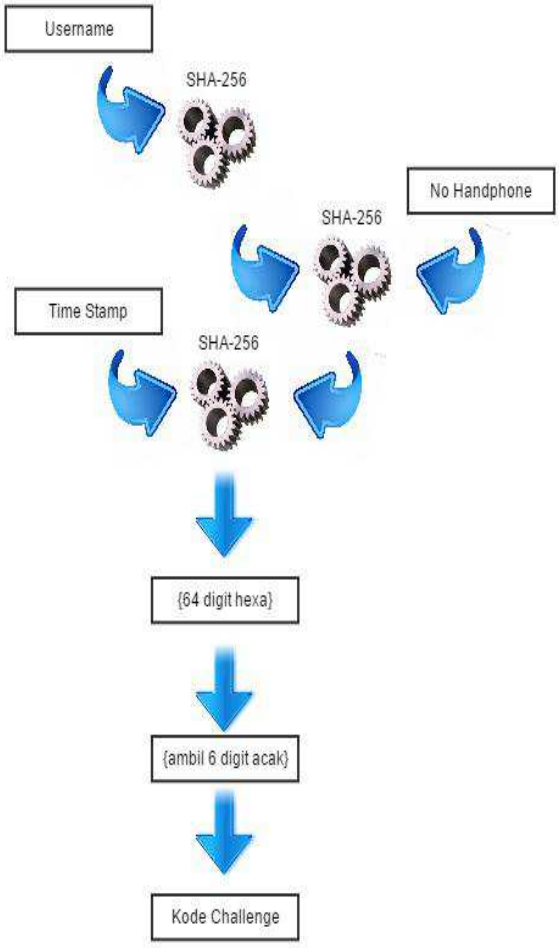
SHA-256 merupakan salah satu dari banyaknya fungsi Hash yang ada, Fikri[16] dan Syafriadi[17] mengatakan bahwa meskipun fungsi SHA-256 membutuhkan waktu sedikit lebih lama dibandingkan dengan fungsi Hash yang lain, tetapi fungsih Hash-256 memiliki tingkat keamanan yang sangat baik karena masih belum ada yang dapat menjebol hasil dari fungsi SHA-256. Panjang password OTP dibuat hanya enam karakter, agar pengguna lebih mudah ketika memasukkan password ke dalam sistem. Pemilihan karakter acak dari hasil fungsi hash menggunakan metode LCG karena LCG merupakan salah satu PRNG yang sederhana dan paling cepat[15].

Setiap pengguna harus meregistrasikan nomor handphone yang dimiliki ke sistem. Aplikasi Mobile OTP yang dikembangkan, memanfaatkan perangkat mobile untuk memberikan hasil generate password yang dilakukan oleh web service pada sistem. Pengguna sistem saat akan login pada sistem akan diminta memasukkan password (token) untuk otentikasi pengguna sistem. Untuk dapat menggunakan perangkat mobile yang dimiliki, pengguna harus memasang Aplikasi Pengamanan Sistem dengan Otentikasi OTP ini ke perangkat mobilenya. Selain itu, pengguna harus mendaftarkan nomor handphone yang dipakai pada perangkat mobile agar ter-register ke sistem. Alur sistem secara umum digambarkan sebagai berikut.




**Gbr 5. Arsitektur Sistem Secara Umum**

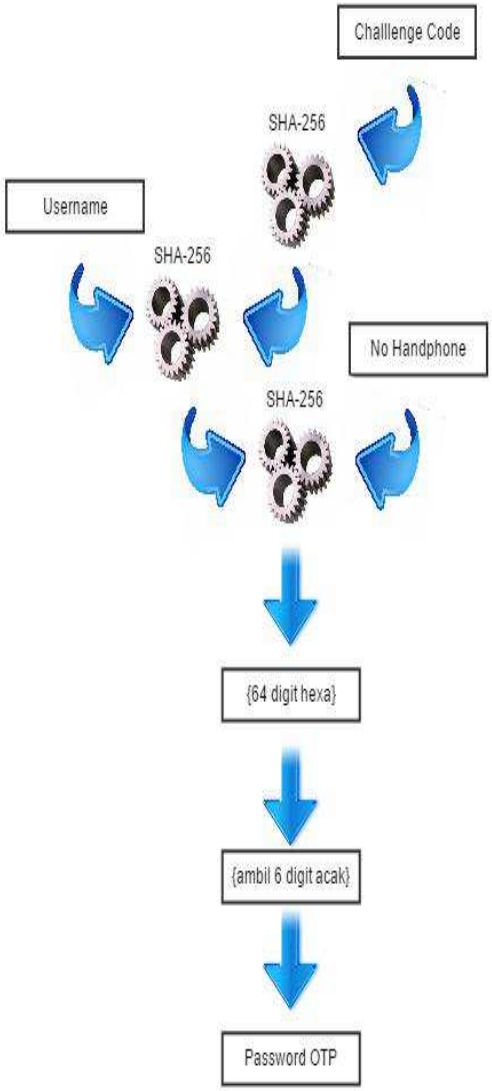
1. *User* melakukan request untuk dapat membuka form transaksi,
2. sistem melakukan request kepada server untuk menampilkan form transaksi,
3. membuat kode Challange, dengan langkah:




**Gbr 6. Membuat Kode Challenge[5]**

a. ambil data *username* kemudian dilakukan hash dengan SHA-256,
b. hasil hash dilakukan hash dengan SHA-256 kembali dengan ditambahkan salt yaitu nomor handphone,
c. hasil hash dilakukan hash SHA-256 sekali lagi dengan ditambahkan salt Time stamp (tanggal dan jam akses) ,
d. dari hasil hash yang terakhir terdiri dari 64 digit yang berisi bilangan hexadesimal, diambil enam digit secara acak menggunakan metode LCG menjadi kode Chalange,
e. kode Challange yang didapat disimpan ke dalam database bersama Time stamp (tanggal dan jam akses) dan diberi flag bahwa kode Challange belum digunakan,
f. tampilkan kode Challange ke sistem

4. kode Challange ditampilkan ke *user*,
5. kode Challange digunakan untuk parameter aplikasi OTP,
6. membuat password OTP dengan langkah:

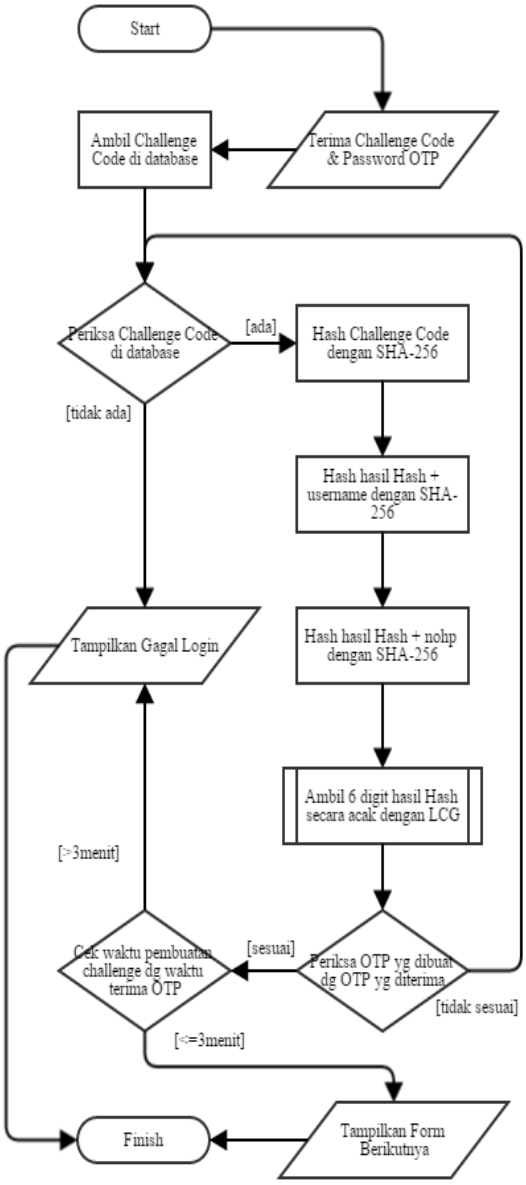



**Gbr 7. Membuat Password OTP[5]**

a. ambil kode Challange kemudian dilakukan hash dengan SHA-256,
b. hasil hash dilakukan hash dengan SHA-256 kembali dengan ditambahkan salt yaitu username,
c. hasil hash dilakukan hash SHA-256 sekali lagi dengan ditambahkan salt no handphone,
d. dari hasil hash yang terakhir terdiri dari 64 digit yang berisi bilangan hexadesimal, diambil enam digit secara acak menggunakan metode LCG menjadi password OTP,
e. password OTP yang didapat disimpan ke dalam database bersama kode Challange dan Time stamp (tanggal dan jam akses),
f. tampilkan password OTP

7. pasword OTP yang didapat dari aplikasi mobile OTP dimasukkan sebagai otentikasi ke sistem

8. password OTP yang dimasukkan akan diperiksa dengan langkah seperti Gambar III.6.
9. periksa password OTP dengan langkah-langkah:




**Gbr 8. Periksa Password OTP**

a. ambil data kode Challange yang ada di database, difilter berdasarkan flag yang belum digunakan,
b. buat password OTP seperti yang dilakukan aplikasi mobile OTP berdasarkan kode Challange yang diambil dari database tadi
c. bandingkan password OTP yang baru dibuat (berdasarkan database) dengan password OTP yang dimasukkan ke sistem tadi
d. jika sesuai, bandingkan lagi waktu pembuatan kode Chalange dengan waktu memasukkan password OTP, jika masih dalam Orange waktu tiga(3) menit maka tampilkan form transaksi yang diminta
e. jika tidak sesuai, tampilkan Gagal Login

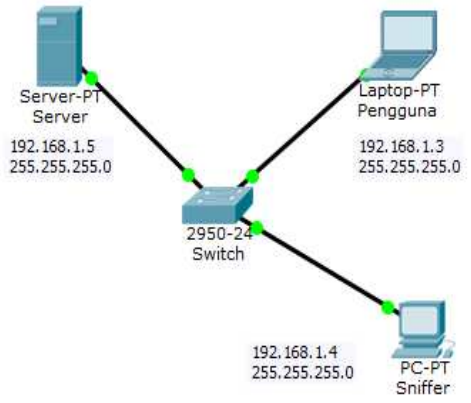
## V. HASIL PENELITIAN

Setelah pembuatan sistem berbasis web serta aplikasi mobile OTP berbasis android selesai maka dilakukan pengujian sistem untuk mengidentifikasi apakah sistem dan aplikasi yang dibangun sudah sesuai dengan analisis kebutuhan sebelumnya. Ada beberapa pengujian yang akan dilakukan, yaitu pengujian menggunakan *Sniffing*. Pengujian *Sniffing* digunakan untuk memastikan sistem telah melakukan pengamanan terhadap otentikasi.
Pengujian kali ini membutuhkan pengguna yang melakukan pengujian, yang juga nantinya diminta untuk mengisi kuesioner. Data pengguna yang melakukan pengujian diantaranya sebagai berikut:

**Tabel 2. Data Pengguna Pengujian**

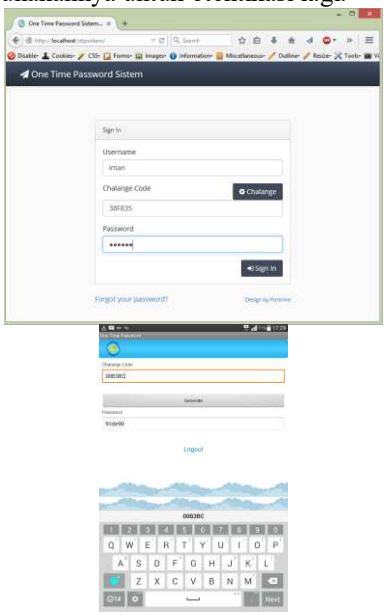
| *Username* | Nama | No Handphone |
|---|---|---|
| **dani** | Dani Anggoro | 087871178926 |
| **ferdi** | Ferdiansyah | 083894263083 |
| **jati** | Sejati Waluyo | 085782069395 |
| **jaya** | Tri Ika Jaya | 02194972556 |
| **lihin** | Law Lihin | 08129743900 |
| **lusi** | Lusi Fajarita | 081385277752 |
| **nofi** | Nofiyani | 02197682003 |
| **Iman** | Iman Permana | 085279445345 |
| **wulan** | Wulandari | 085692293236 |
| **yadi** | Lis Suryadi | 081381595493 |

Pengujian kali ini menggunakan infrastruktur jaringan dengan satu buah komputer *web server*, satu buah laptop sebagai komputer pengguna, satu buah komputer sebagai *sniffer*, satu buah switch, dan tiga buah kabel UTP straight through. Ketiga komputer dihubungkan dengan satu buah switch dan tiga kabel UTP straight through, dan diberikan IPv4.



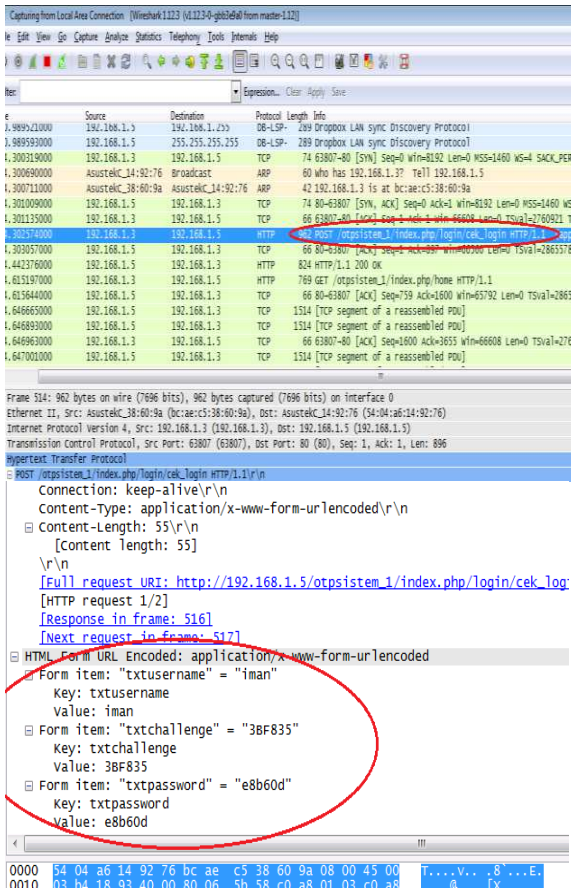

**Gbr 9. Infrastuktur Pengujian Sniffing**

Pengguna membuka sistem berbasis web dan mencoba melakukan otentikasi di komputer pengguna. Sedangkan di saat yang sama *sniffer* melakukan pencurian password dan mencoba menggunakannya untuk otentikasi lagi.



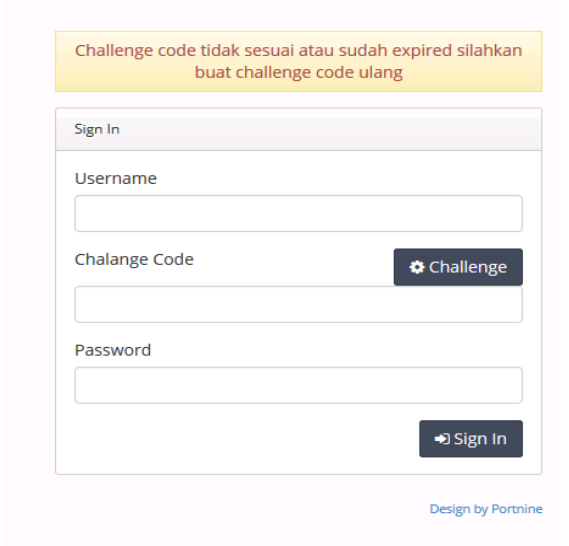

**Gbr 10. Pengguna Melakukan Otentikasi**

Pengguna bernama Iman Permana memasukkan *username* iman. Kemudian mencoba mendapatkan *Challenge code* dengan cara klik tombol Challenge dan memasukkan *challenge code* ke aplikasi OTP berbasis mobile android. Password diperoleh setelah pengguna klik tombol Generate, yang digunakan sebagai otentikasi ke sistem berbasis web. Pengguna berhasil masuk ke sistem. Pada saat yang bersamaan, sinffer menjalankan wireshark dengan memantau paket data yang keluar masuk ip 192.168.1.5 dan ip 192.168.1.3.




**Gbr 11. Sniffer Melihat Username dan Password**

*Sniffer* dapat melihat *username*, *challenge code* dan password yang digunakan oleh pengguna untuk otentikasi ke sistem berbasis web. Dengan menggunakan *username*, *challenge code* dan password yang diperoleh, *sniffer* mencoba membuka sistem berbasis web.




**Gbr 12. Username dan Password tidak biasa digunakan**

Dengan menggunakan *username*, *challenge code* dan password yang sama, *sniffer* tidak bisa masuk ke sistem berbasis web, dan mendapatkan

pesan "*Challenge code* tidak sesuai atau sudah expired, silahkan buat *challenge code* ulang".

## VI. KESIMPULAN

A. Kesimpulan

Berdasarkan analisis dan hasil pengujian yang telah dilakukan pada pengembangan model pengamanan sistem menggunakan *One Time Password* dengan metode pembangkit password Hash SHA-256 dan *Pseudo Random Number Generator* (PRNG) *Linear Congruential Generator* (LCG) di perangat berbasis android, maka dapat ditarik kesimpulan sebagai berikut:

a. Kerahasiaan otentikasi sistem dapat terjaga kerahasiaannya
b. Dengan terjaga kerahasiaan otentikasi sistem, maka resiko pencurian password oleh orang yang tidak berwenang dapat diminimalisirr.

B. Saran

Beberapa saran untuk penelitian lebih lanjut dan penyempurnaan penelitian tentang penelitian ini adalah sebagai berikut.

a. Metode pengambilan karakter secara acak dapat dikembangkan menggunakan beberapa metode yang lebih kompleks seperti Blum Blum Shub.
b. Tampilan dibuat lebih rapi dan teratur, misalnya penambahan icon agar lebih menarik.
c. Aplikasi mobile dikembangkan menggunakan device selain android, misalnya Blackberry, iPhone, atau Windows Phone.
d. Sistem ini dapat diimplementasikan pada banyak hal, dan akan lebih baik jika dilakukan pada layanan yang berhubungan dengan keuangan, misalnya layanan validasi pembayaran kuliah, validasi pembayaran internet, dan sebagainya.

## DAFTAR PUSTAKA


[1] D. Setiawan, *Sistem Keamanan Komputer*. Elex Media Komputindo, 2005.

[2] J. Y. Babys, "Analisis Aspek Keamanan Informasi Jaringan Komputer ( Studi Kasus : STIMIK Kupang )," *Semin. Nas. Inform. 2013*, vol. 2013, no. semnasIF, pp. 7–14, 2013.

[3] S. Acharya, A. Polawar, and P. Y. Pawar, "Two Factor Authentication Using Smartphone Generated One Time Password," vol. 11, no. 2, pp. 85–90, 2013.

[4] M. Syafrizal, "ISO 17799 : Standar Sistem Manajemen Keamanan Informasi," *Semin. Nas. Teknol. 2007 (SNT 2007)*, pp. 1–12, 2007.

[5] K. I. Santoso, "Dua Faktor Pengamanan Login Web Menggunakan Otentikasi One Time Password Dengan Hash SHA," *Semin. Nas. Teknol. Inf. Komun. Terap. 2013*, no. November, pp. 204–210, 2013.

[6] R. P. Mustofa, "Aplikasi Mobile Android 'One Time Password(OTP)' Untuk Meningkatkan Keamanan Otentikasi," 2013.

[7] Shally and G. S. Aujla, "A review of one time password mobile verification," *Int. J. Comput. Sci. Eng. Inf. Technol. Res.*, vol. 4, no. 3, pp. 113–118, 2014.

[8] K. Alghathbar and H. A. Mahmoud, "Noisy Password Scheme : A New One Time Password System," pp. 841–846, 2009.

[9] R. S. Bhuvaneshwari and P. Anuja, "Secured Password Management Technique Using One-Time Password Protocol In Smartphone," *Int. J. Comput. Sci. Mob. Comput.*, vol. 3, no. 3, pp. 976–981, 2014.

[10] B. K. Kushwaha, "An Approach for User Authentication One Time Password ( Numeric And Graphical ) Scheme," *J. Glob. Res. Comput. Sci.*, vol. 3, no. 11, pp. 54–57, 2012.

[11] A. Suhendra, A. Yulianti, B. Junatas, and V. Valentine, "Modified Authentication using One Time Password to Support Web Services Security," *Work. Open Source Open Content*, no. December, pp. 1–3, 2008.

[12] R. Mohan and N. Partheeban, "Secure Multimodal Mobile Authentication Using One Time Password," *Int. J. Recent Technol. Eng.*, vol. 1, no. 1, pp. 131–136, 2012.

[13] S. Gharge, H. Brijwani, M. Pugrani, G. Sukhwani, and D. Udherani, "Percon8 Algorithm for Random Number Generation," *J. Eng. Res. Appl.*, vol. 4, no. 5, pp. 54–60, 2014.

[14] D. E. Knuth, *The Art of Computer Programming*, Second Edi. Canada: Addison-Wesley, 1981.

[15] A. Ramadhan, "Perbandingan Algoritma Linear Congruential Generators , BlumBlumShub , dan MersenneTwister untuk Membangkitkan Bilangan Acak Semu," Institut Teknologi Bandung, 2010.

[16] A. Fikri, "Analisis dan Perbandingan Algoritma Fungsi Hash SHA-2 256 dan Keccak," Institut Teknologi Bandung, Bandung, 2011.

[17] M. Syafriadi, "Analisis Kecepatan dan Keamanan Algoritma Secure Hash Algorithm 256 (SHA-256) Untuk Otentikasi Pesan Teks," Institut Pertanian Bogor, Bogor, 2006.

[18] A. Aryasanti, "Model Pengamanan Berkas Bank Soal Dengan Metode Steganografi LSB dan Kompresi," Universitas Budi Luhur, 2014.