IMPLEMENTASI PENGGUNAAN *RAINBOW TABLE* UNTUK MENCARI KUNCI CHIPER DATA KOMUNIKASI GSM

Haryo Novianto **Henry Rossi Andrian,ST.,MT.** **Moch Fahru Rizal,ST.,MT**

Telkom University haryoooo@students.telkomuniversity.ac.id

Telkom University rossi@tass.telkomuniversity.ac.id

Telkom University mfrizal@tass.telkomuniversity.ac.id

Abstrak

Untuk mengurangi hal kejahatan dalam peretasan jaringan komunikasi GSM, akan dilakukan analisis terhadap kelemahan keamanan data GSM. Diperlukan suatu program yang digunakan khusus untuk mencari kunci chiper (Kc) dari data GSM yaitu Kraken. Untuk membangkitkan kraken, diperlukan 40 data *rainbow table* yang merupakan kumpulan 90% Kc GSM. Kraken akan membaca 40 data tersebut dan menulis ulang dalam bentuk *raw* dan data *raw* inilah yang akan menjadi *master file* dalam mencari Kc data GSM. Analisis ini ditujukan untuk masyarakat pemakai jaringan GSM yang belum pindah ke jaringan komunikasi yang lebih aman. Dari hasil analisis ini, dapat disimpulkan bahwa 40 data *rainbow table* merupakan kunci chiper GSM dan jaringan GSM memiliki kelemahan yaitu pada enkripsi datanya.

Kata kunci: GSM, Kunci Chiper , *Rainbow Table*, Analisis

Abstract

To solve the crime in GSM communication network hacking that aims to obtain GSM communication data, Will be analyzed GSM data security weakness. In this analyzing, It will required a special program called Kraken, this program wil be used to find the chiper key (Kc) from GSM data. To generate kraken, it will required 40 data rainbow table which is a collection of 90% Kc and also adequate hardware. Kraken will read 40 data and rewrite it in raw form and this raw data will being the master file in search for Kc GSM data. This analyzing is aimed at GSM network users who have not moved to a secure communications network. From the results of this analyzing, it can be concluded that 40 data rainbow table is a collection of Chiper Key GSM and GSM network has a weakness that is on the data encryption.

Keywords: GSM, *Chiper Key*, *Rainbow Table*, *Analyzing*

1. Pendahuluan

Pada zaman modern ini jaringan GSM telah menyebar luas di seluruh dunia dan dipakai pada telepon selular mayoritas masyarakat pada zaman ini. Adapun pengertian Global System for Mobile Communication disingkat GSM adalah sebuah teknologi komunikasi selular yang bersifat digital. Teknologi GSM banyak diterapkan pada komunikasi bergerak, khususnya telepon genggam. Teknologi ini memanfaatkan gelombang mikro dan pengiriman sinyal yang dibagi berdasarkan waktu, sehingga sinyal informasi yang dikirim akan sampai pada tujuan. GSM dijadikan standar global untuk komunikasi selular sekaligus sebagai teknologi selular yang paling banyak digunakan orang di seluruh dunia. Saat ini keamanan komunikasi dalam jaringan GSM terbilang cukup aman karena menggunakan standar enkripsi A5/1.

Akan tetapi, menurut [7] yang mengatakan bahwa algoritma A5/1 memiliki kelemahan serius didalam keamanan komunikasi datanya. Oleh karena itu, akan dibuat suatu proyek akhir yang mempunyai tujuan untuk analisis kelemahan dari keamanan algoritma A5/1 yang digunakan oleh jaringan GSM.

Dari suatu Encrypted Burst nantinya akan didapatkan XoR'ed burstnya, burst tersebut akan dicari kunci chiper-nya dengan menggunakan software bernama Kraken. Tersambung dengan Harddisk External 4 Terabyte sebagai tempat *master file*-nya yaitu *Rainbow Table*. Cara ini digunakan untuk melakukan analisis kelemahan keamanan data komunikasi GSM. Dikarenakan data komunikasi merupakan hal yang sangat penting bagi pemakai jaringan GSM.

Berdasarkan latar belakang yang telah diuraikan diatas, maka dirumuskan masalah seperti berikut.

1. Bagaimana cara melakukan *convert rainbow table* ke dalam program kraken?
2. Bagaimana cara konfigurasi kraken untuk mencari kunci chiper data komunikasi GSM?

Adapun tujuan dari proyek akhir ini adalah.

1. Dapat melakukan convert rainbow table ke dalam program kraken
2. Dapat melakukan konfigurasi kraken untuk mencari kunci chiper data komunikasi GSM

Batasan masalah dalam proyek akhir ini sebagai berikut.

1. Tidak membahas lebih jauh tentang algoritma A5/2 dan A5/3.
2. Tidak membahas lebih jauh tentang RTL-SDR.
3. Tidak membahas lebih jauh tentang autentikasi pada GSM.
4. Tidak membahas lebih jauh tentang penangkapan sinyal GSM.

5. Tidak membahas lebih jauh tentang decoding data komunikasi GSM.
6. Hanya membahas tentang keamanan data komunikasi GSM.
7. Hanya membahas pencarian kunci chiper data komunikasi GSM.

2. Tinjauan Pustaka

2.1 GSM

Global System for Mobile Communication (GSM) awalnya berasal dari singkatan *Grupe Special Mobile* yang memiliki pengertian sebuah teknologi komunikasi seluler yang bersifat digital. Teknologi GSM banyak diterapkan pada komunikasi bergerak, khususnya telepon genggam. Teknologi ini memanfaatkan gelombang mikro dan pengiriman sinyal yang dibagi berdasarkan waktu, sehingga sinyal informasi yang dikirim akan sampai pada tujuan. GSM dijadikan standar global untuk komunikasi seluler sekaligus sebagai teknologi yang paling banyak digunakan orang di seluruh dunia.

2.2 Rainbow Table

Rainbow Table adalah *precomputed table* yang digunakan untuk mengembalikan fungsi kriptografi hash. Umumnya digunakan untuk *crack* hash kata sandi. Idenya disini adalah hanya menyimpan bagian dari *precomputing tables* lalu bagian tabel lainnya akan dilakukan komputasi ulang pada saat pencarian.

2.3 Ubuntu 14.04 LTS

Ubuntu Versi 14.04 "Trusty Tahr" merupakan distribusi Linux yang paling populer menggunakan *user interface unity* yang khas dan disesuaikan. Trusty Tahr merupakan edisi dengan dukungan jangka panjang "Long Term Support" (LTS) selama 5 tahun, berupa dukungan keamanan berikut jalur *upgrade* yang lebih mudah dibandingkan rilis versi LTS (12.04) sebelumnya.

2.4 Kraken

Kraken adalah *software open source* yang digunakan untuk mencari Kc dari data komunikasi yang ada pada jaringan GSM. *Software* ini dikenal dengan performanya yang lebih efisien dan lebih cepat dalam mencari kunci chiper yang cocok untuk membuka data komunikasi GSM.

2.5 Kunci Chiper

Kunci Chiper (Kc) adalah kunci yang digunakan untuk enkripsi dan dekripsi. Sistem ini mengharuskan dua pihak yang berkomunikasi menyepakati suatu kunci rahasia yang sama sebelum keduanya saling berkomunikasi.

2.6 Harddisk Drive

Hard drive disingkat HD adalah sebuah komponen perangkat keras yang menyimpan data sekunder dan berisi piringan magnetis. *Hard Disk Drive* diciptakan pertama kali oleh insinyur IBM, Reynold Johnson pada tahun 1956. *Hard Disk Drive* pertama tersebut terdiri dari 50 piringan berukuran 2 kaki (0,6 meter) dengan kecepatan rotasinya mencapai 1.200 rpm (*rotation per minute*) dengan kapasitas penyimpanan 4,4 MB.

2.7 A5/1

A5/1 adalah algoritma enkripsi yang digunakan oleh sekitar 130 juta pelanggan GSM di Eropa untuk melindungi privasi *over-the-air* dari seluler suara dan data komunikasi mereka. Serangan yang terbaik diterbitkan terhadap itu membutuhkan antara 240 dan 245 langkah. Tingkat keamanan membuatnya rentan terhadap serangan berbasis *hardware* oleh organisasi besar, tetapi tidak untuk serangan berbasis *software* pada beberapa sasaran oleh hacker.

2.8 Burst

Burst adalah format informasi yang ditransmisikan selama *satu time slot*. Informasi ditumpangkan pada satu *time slot* melalui *Air Interface* yang biasa disebut Burst (pemecahan). Burst yang normal mengandung paket 57 bit dari data *encrypted* atau *voice*.

2.9 Git

Git adalah perangkat lunak pengontrol versi atau proyek manajemen kode perangkat lunak yang diciptakan oleh Linus Torvalds, yang pada awalnya ditujukan untuk pengembangan kernel Linux.

2.10 XoR

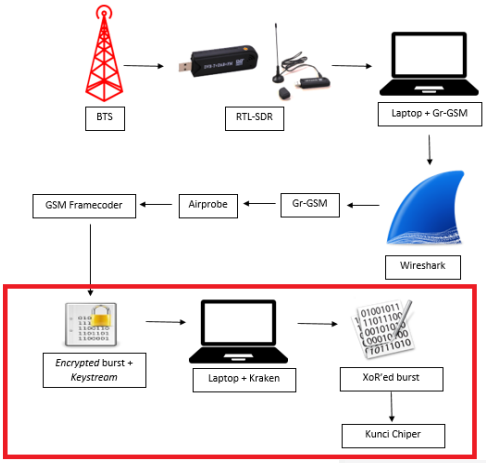
XoR merupakan kepanjangan dari "Exclusive OR" yang mana keluarannya akan berlogika 1 apabila inputannya berbeda, namun apabila semua inputannya sama maka akan memberikan keluarannya 0. Operator XoR sering dijadikan sebagai salah satu komponen dalam pembentukan chiper yang lebih kompleks.

2.11 Keystream

Keystream adalah aliran karakter acak atau *pseudorandom* yang dikombinasikan dengan pesan *plaintext* untuk menghasilkan pesan terenkripsi (*chipertext*). "Karakter" dalam *keystream* bisa berupa bit, byte, angka atau karakter aktual seperti A-Z tergantung pada penggunaan.

3. Analisis dan perancangan

3.1 Gambaran Sistem Produk



Deskripsi alur kerja sistem yang hendak dibangun.

1. Menjalankan *tool* xor.py pada program kraken untuk menambahkan bit *keystream* dengan bit *encrypted* burst
2. Mendapatkan hasil pertambahannya yaitu XoR'ed burst yang nantinya akan disalin ke dalam program kraken
3. Menghubungkan *rainbow table* bentuk *raw* yang telah di *convert* dengan file indexes dengan perintah ./kraken ../indexes. Perintah ini dilakukan agar dapat melakukan *crack* XoR'ed burst
4. Salin XoR'ed burst yang telah didapat dan *crack* burst tersebut dengan perintah *crack* pada program kraken untuk mendapatkan data indexes yang akan menunjukan kunci chiper dari burst tersebut
5. Salin hasil *crack* dari XoR'ed burst tersebut ke dalam *tool* find_kc dan masukkan juga *frame number* dari burst tersebut untuk mendapatkan *potential chiper key*
6. Tambahkan *frame number* urutan sebelumnya dan XoR 'ed burst-nya untuk mengeliminasi Kc yang salah dan mendapatkan Kc yang sesuai.

3.2 Kebutuhan Perangkat Keras dan Perangkat Lunak

Berikut kebutuhan *Hardware* untuk sistem yang digunakan:

No.	*Hardware*	Keterangan	Spesifikasi
1.	Laptop	Untuk menjalankan proses mencari Kc.	Intel® Core™ i5 RAM 4 *Gigabyte Harddisk* 500 *Gigabyte*
2.	*Harddisk External*	Untuk data *Rainbow table*	Berukuran 4TB

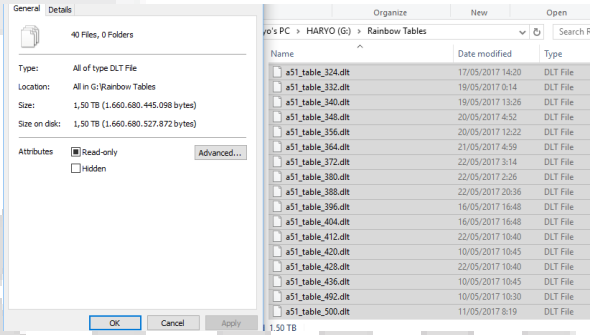
Berikut Kebutuhan *Software* yang digunakan:

No.	*Software*	Keterangan
1.	Kraken	Untuk mencari kunci chiper data komunikasi GSM.
2	Git	Untuk menghubungkan *library* yang dibutuhkan *software.*
3	Gcc	Sebagai *compiler* agar kraken dapat dijalankan di CPU
4.	VMware 12.0	Sebagai mesin virtual.
5.	Ubuntu 14.04 LTS	Sebagai sistem operasi *Virtual Machine.*

4. Implementasi dan Pengujian

4.1 Melakukan unduh *rainbow table*

Mengunduh 40 file *rainbow table* berukuran 1.6 *Terabyte* Hal ini merupakan syarat wajib dalam mengerjakan proyek akhir ini karena 40 tabel ini merupakan data yang merupakan kemungkinan kunci enkripsi yang akan dipakai oleh program kraken dari XoR'ed Burst.



4.2 Instalasi *Software* pada OS Ubuntu

Instalasi yang dilakukan pada OS Ubuntu 14.04 meliputi instalasi vmware, git, gcc dan kraken yang merupakan software yang dibutuhkan pada proyek akhir ini.

1. Instalasi Git.

```
#apt-get install git
```

2. Instalasi Gcc

```
#apt-get install make automake gcc g++
```

3. Instalasi Kraken

```
#git-clone /halaman web software/
```

4.3 Konfigurasi Indexes

1. Konfigurasi pada *file tables.conf.sample* dengan masuk ke dalam direktori indexes yang ada pada kraken dengan perintah berikut pada terminal :

```
# cd kraken /indexes/
```

2. Masuk kedalam *file tables.conf.sample* dengan perintah berikut pada terminal :

```
# nano tables.conf.sample
```

3. Pada *file tables.conf.sample* terdapat contoh alokasi file *rainbow table* pada partisi. Pada gambar dibawah ini terdapat 4 alokasi partisi yaitu sda1, sdb1, sdd1 dan sde1 yang masing-masing partisi akan dialokasikan 10 file hasil *convert rainbow table*

```
root@ubuntu: /home/haryoooo/kraken/indexes
 GNU nano 2.2.6            File: tables.conf.sample

#Devices:  dev/node max_tables
Device: /dev/sda1 10
Device: /dev/sdb1 10
Device: /dev/sdd1 10
Device: /dev/sde1 10

#Tables: dev id(advance) offset
```

4. Alokasi pada partisi baru yaitu /dev/sdb 40. *Save* dengan nama *tables.conf* agar tidak merubah contoh alokasi tabel yang ada pada program kraken. Dalam *tables.conf*, kondisikan sesuai dengan storage partisi yang ada. Pada kasus ini, karena partisi lain terdapat hanya satu yaitu /dev/sdb dan dapat menampung hasil *convert table* secara keseluruhan maka tulis 40.

```
root@ubuntu: /home/haryoooo/kraken/indexes
 GNU nano 2.2.6            File: tables.conf

#Devices:  dev/node max_tables
Device: /dev/sdb 40

#Tables: dev id(advance) offset
```

4.4 *Convert Rainbow table*

1. Melakukan *convert rainbow table* dengan menggunakan *tool* Behemoth.py. *Convert rainbow table* diperlukan karena kraken hanya dapat membaca file dalam bentuk *raw* dan juga *tool* Behemoth.py harus dijalankan dengan mode root privileges. Dengan mengetikkan perintah berikut di terminal :

```
# sudo python Behemoth.py /tempat disimpannya 40 file rainbow table format .dlt/
```

2. Proses *convert rainbow table* akan membuat file indexes dan data indexes untuk mempermudah pencarian kunci chiper oleh program kraken nantinya. Berikut gambar proses *convert rainbow table* pada terminal :

```
root@ubuntu:/home/haryoooo/kraken/indexes# sudo python Behemoth.py /home/haryooo/A5/
Adding table:  /home/haryoooo/A5//a51_table_492.dlt 492
Running "./../TableConvert/TableConvert di /home/haryoooo/A5//a51_table_492.dlt /dev/sdb:0 492.idx"
seek offset: 0i blocks (/dev/sdb)
6196522141i chains written.
61965221411 chains written, 54.096798 bits pr chain.
Adding table:  /home/haryoooo/A5//a51_table_500.dlt 500
Running "./../TableConvert/TableConvert di /home/haryoooo/A5//a51_table_500.dlt /dev/sdb:10229859 500.idx"
seek offset: 10229859i blocks (/dev/sdb)
```

4.5 Pengujian Sistem

Pada tahapan ini dilakukan pengujian *rainbow table*, XoR *keystream* dengan *encrypted* burst, *crack* XoR'ed burst dan pencarian *chiper key* pada mesin virtual. Hal ini dilakukan dengan menggunakan *encrypted* burst contoh dan pengujian dilakukan untuk memastikan sistem yang telah dibuat dapat berjalan dengan baik.

4.5.1 Ujicoba Rainbow Table

Untuk melakukan ujicoba kinerja *rainbow table*, ketik perintah berikut pada terminal :

```
# test
```

```
root@ubuntu: /home/haryoooo/kraken/Kraken
Allocated 41257276 bytes: ../indexes//230.idx
Allocated 41249048 bytes: ../indexes//340.idx
Allocated 41314028 bytes: ../indexes//380.idx
Allocated 41248788 bytes: ../indexes//404.idx
Allocated 41239116 bytes: ../indexes//492.idx
Allocated 41246592 bytes: ../indexes//356.idx
Allocated 41236892 bytes: ../indexes//372.idx
Allocated 41235976 bytes: ../indexes//276.idx
Allocated 41281052 bytes: ../indexes//250.idx
Allocated 41274520 bytes: ../indexes//124.idx
Tables: 388,260,364,140,196,148,238,132,412,116,420,348,436,172,108,180,500,42
188,324,204,292,268,332,156,212,164,220,396,100,230,340,380,404,492,356,372,27
250,124
Commands are: crack test quit

Kraken> test

Cracking 00110111001100000000100000110001100010011011011001101101001111000110
01001001011111101011110000011010100110101
Found 160271036984773819800 @ 12  #0  (table:340)
Found 8050061555739560956x @ 23  #0  (table:372)
crack #0 took 854083 msec
```

4.5.2 Ujicoba XoR *keystream* dengan *Encrypted* Burst

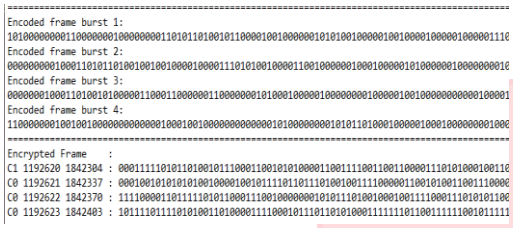
1. Masuk ke dalam direktori Utilities pada program kraken dengan ketikkan perintah berikut pada terminal :

```
# cd Utilities
```

2. Pada percobaan kali ini akan menggunakan *tool* yaitu xor.py dan menggunakan *encrypted* burst dan *keystream* contoh. *Tool* ini berfungsi untuk menambahkan bit pada *keystream* dengan bit pada *encrypted* burst untuk mendapatkan XoR'ed burst, yang akan diberikan kepada kraken untuk di *crack*. Jalankan :

```
# sudo python xor.py /masukkan keystream dan encrypted burst/
```

3. Terdapat *keystream* yang telah didapat dari gsmframecoder dan akan digabungkan dengan *encrypted* Burst ke dalam *tool* xor.py untuk mendapatkan XoR'ed Burst

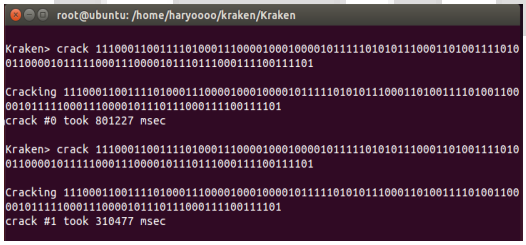


4. Masukkan *keystream* yang telah didapat dan gabungkan dengan *encrypted* burst ke dalam *tool* xor.py untuk mendapatkan XoR'ed burst

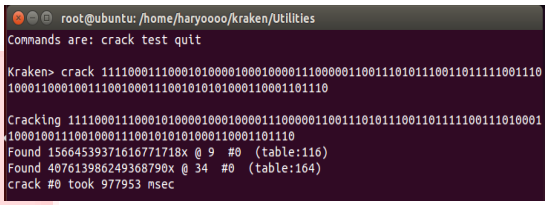
4.5.3 Ujicoba *crack* XoR'ed burst

1. Pengujian crack XoR'ed burst menggunakan data *encrypted* burst yang didapat dari *live decoding*. Untuk melakukan ujicoba dengan burst, ketik perintah berikut ini pada terminal :

```
# crack /masukkan XoR'ed burst/
```



2. Pengujian *crack* XoR'ed burst menggunakan data contoh dan XoR'ed burst berhasil di *crack*, output akan muncul berupa *index* data yang digunakan untuk menunjukan kemungkinan kunci chiper dari burst tersebut



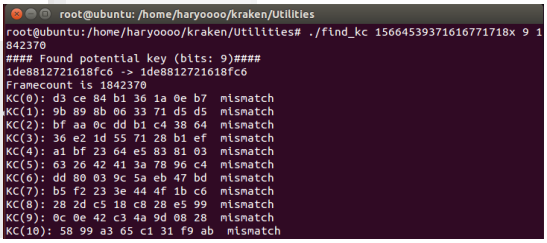
4.5.4 Ujicoba mencari kunci chiper

1. Pada pengujian kali ini menggunkan *tool* find_kc dan masukkan burst dan *frame number* burst yang telah di *crack* untuk mendapatkan potensial Kc-nya, masuk ke direktori dimana *tool* find_kc berada yaitu direktori *Utilities* pada kraken. Untuk menjalankannya ketik perintah berikut pada terminal :

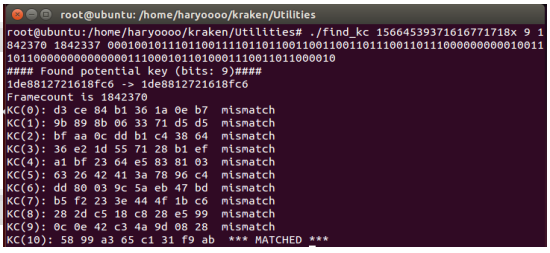
```
# cd Utilities
```

2. Masukkan burst dan *frame number* burst yang telah di *crack* untuk mendapatkan potensial Kc-nya. Untuk menjalankannya ketik perintah berikut pada terminal :

```
# ./find_kc /index data/posisi data index/frame number
```



3. Untuk mendapatkan kunci chiper dari XoR'ed burst yang telah di *crack*, masukkan hasil *crack* burst dan *frame number* beserta burst urutan sebelumnya untuk mengeliminasi Kc yang salah.



5. Penutup

5.1 Kesimpulan

Dari hasil analisis pada proyek akhir ini dapat disimpulkan bahwa :

1. *Convert rainbow table* telah berhasil dilakukan.
2. Konfigurasi kraken untuk mencari kunci chiper data komunikasi GSM telah berhasil dilakukan.

5.2 Saran

Pada jaringan GSM, layanan yang ditawarkan pada jaringan ini sangat memuaskan pengguna jaringan tersebut. Akan tetapi, dari hasil analisis ini kunci chiper jenis sudah tidak aman lagi digunakan untuk meng-enkripsi data pengguna jaringan tersebut. Oleh karena itu, disarankan kepada para pengguna jaringan GSM agar lebih memberi perhatian terhadap keamanan datanya dan mulai berpindah ke jaringan yang mempunyai enkripsi pada data yang lebih baik seperti UMTS (3G) LTE (4G).

Daftar Pustaka

[1] S. M. Redl dan M. K. Weber, “An Introduction to GSM,” Artech House, March 1995.

[2] S. Meyer, “Rainbow Tables,” *Breaking GSM with Rainbow Tables*, p. 2, 2010.

[3] Ubuntu, “14.04,” 2017. [Online]. Available: https://wiki.ubuntu.com/TrustyTahr/ReleaseNotes/14.04.4. [Diakses 24 03 2017].

[4] R. McMillan, “New 'Kraken' GSM-cracking software is released,” 21 07 2010. [Online]. Available: http://www.computerworld.com/article/2519495/mobile-wireless/new--kraken--gsm-cracking-software-is-released.html. [Diakses 04 03 2017].

[5] D. Lestari dan M. Z. Riyanto, “SUATU ALGORITMA KRIPTOGRAFI STREAM CIPHER BERDASARKAN FUNGSI CHAOS,” *Algoritma Kriptografi Stream Cipher*, p. 34.

[6] Arpaci-Dusseau dan R. H, “Hard Disk Drive,” dalam *"Operating Systems: Three Easy Pieces, Chapter: Hard Disk Drives"*, Arpaci-Dusseau Books, 2014.

[7] R. Nugraha dan T. Sumarno , “Analisis Keamanan Protokol GSM,” *Otentikasi*, p. 5.

[8] T. T. Nielsen dan J. Wigard, “Performance Enhancements in a Frequency Hopping GSM Network,” New York, Kluwer Academic Publisher, 2002, p. 22.

[9] L. Torvalds, “git,” Kernel, [Online]. Available: https://git.kernel.org/pub/scm/git/git.git/tree/. [Diakses 23 07 2017].

[10] C. Robert, “Codes and Ciphers: Julius Caesar,” dalam *the Enigma and the Internet*, Cambridge University Press, 2002, p. 11.

[11] A. J. Menezes dan P. C. Van Oorschot, “Handbook of Applied Cryptography,” CRC Press, 1996, pp. 169-190.

[12] D. Nugraha, “Implementasi Gr-GSM untuk decoding sinyal GSM,” 2013.