Kriptografi Suara dengan Teknik Inferensi *Data Subchunk* pada Berkas WAV 16 Bit Tanpa Kompresi (16-bit PCM WAV)

R. Tomi Akhmad Fadlan (NIM 13504094)
Program Studi Teknik Informatika STEI ITB, Bandung 40132, email: if14094@students.if.itb.ac.id

Abstract *– Makalah ini membahas sebuah teknik dalam kriptografi suara. Teknik ini mengadaptasi teknik pemecahan piksel pada kriptografi visual. Informasi suara dapat disimpan dalam berbagai macam format berkas. Salah satunya adalah berkas WAV. Di dalam berkas WAV terdiri dari dua buah chunk. Chunk yang digunakan untuk penyembunyian informasi suara ini adalah WAV chunk, lebih tepatnya pada data subchunk. Nantinya setiap nilai 2 byte sample pada data subchunk diubah untuk semua shares (dalam berkas WAV juga) sehingga apabila dijumlahkan menghasilkan nilai yang sama seperti aslinya. Berkas WAV asli hanya dapat didengarkan dengan memperdengarkan semua berkas WAV share tersebut secara bersamaan.*

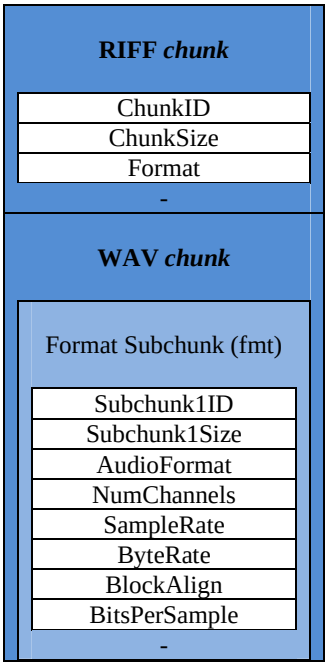
Kata Kunci*: audio cryptography, kriptografi suara, WAV, data subchunk.*

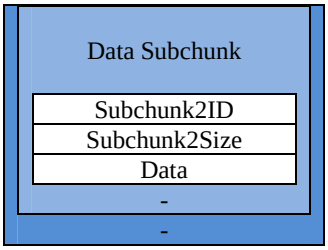
1. PENDAHULUAN

Saat ini salah satu teknik penyembunyian informasi yang sedang berkembang adalah kriptografi visual. Sebuah gambar dapat dipecah dalam tingkat piksel menjadi beberapa gambar (*share*) dengan ukuran yang sama yang tidak mengandung arti. Gambar-gambar tersebut baru akan memiliki arti yang sama dengan gambar awalnya apabila digabung dengan teknik penumpukan *(overlay)*. Berangkat dari sini, muncul gagasan untuk menerapkan teknik tersebut pada suara.
Suara dapat dikemas/direpresentasikan sebagai sebuah berkas dengan format tertentu, seperti WAV, MP3, WMA, OGG, AIFF, dan lain-lain. Di antara semuanya, berkas WAV merupakan format yang paling umum dan dapat diputar pada hampir semua pemutar multimedia. Oleh karena itu, penulis memilih berkas suara dengan format WAV sebagai plainteks.
Bagian potongan (*chunk*) dalam berkas WAV dalam merepresentasikan suara yang dikandungnya adalah RIFF *chunk* dan WAV *chunk* (yang terdiri subchunkdari *format subchunk* (fmt) dan *data subchunk*). RIFF *chunk* memberikan informasi bahwa berkas tersebut adalah sebuah berkas RIFF – berkas WAV adalah keluarga dari format berkas RIFF – dengan *data chunk* berupa "WAV *chunk*" yang terdiri dari fmt dan *data subchunk*. Fmt mengandung informasi mengenai format kompresi suara dan jumlah *channel* sedangkan *data subchunk* mengandung isi dari suara yang dijelaskan strukturnya pada fmt. Apabila berkas suara hendak dipecahkan untuk keperluan penyembunyian informasi, pemecahan dan pendistribusian *data subchunk* ke sejumlah n *share*. merupakan cara yang tepat. Setiap *share* merupakan sebuah berkas WAV yang berisi frekuensi-frekuensi sembarang dan tidak sama dengan berkas WAV asli tetapi fmt-nya sama. Namun apabila setiap nilai *byte* pada data *subchunk* semua *share* dijumlahkan maka akan menghasilkan nilai yang sama dengan byte pada berkas aslinya.
Perlu diketahui, nilai 2 byte untuk jenis berkas PCM WAV 16 bit ini berkisar antara -32768 to 32767 (*signed integer*). Setiap nilai 2 byte mencerminkan tinggi titik gelombang (amplitudo), baik positif (ke atas) maupun negatif (ke bawah) dari garis sunyi gelombang. Keadaan seperti ini dapat dimanfaatkan untuk mengimplementasikan teknik inferensi gelombang, yaitu apabila dua amplitudo positif maka akan saling menguatkan dan apabila salah satu atau kedua amplitudo negative maka akan saling melemahkan.

2. FORMAT BERKAS WAV

Format berkas WAV, seperti yang telah dijelaskan, secara global meliputi RIFF *chunk* dan WAV *chunk*. WAV *chunk* terdiri dari dua *subchunk* yaitu *format subchunk* (fmt) dan *data subchunk*. Di masing-masing *chunk* terbagi lagi menjadi beberapa bagian dengan ukuran byte tertentu. Berikut gambaran lengkapnya.





Gambar 1 Gambaran Lengkap Format Berkas WAV

2.1. RIFF *Chunk*

RIFF *chunk* (*header*) menjelaskan format berkas yang akan diputar. RIFF *chunk* terdiri dari chunkId, chunkSize, dan format.

1. ChunkId
   ChunkId berisi tulisan “RIFF” dalam ASCII
   (ukuran bagian ini 4 *byte*).
2. ChunkSize
   ChunkSize merupakan jumlah byte dari keseluruhan chunk tetapi tidak termasuk chunkId dan chunkSize
   (ukuran bagian ini 4 *byte*).
3. Format
   Format berisi tulisan “WAVE” dalam ASCII
   (ukuran bagian ini 4 *byte*).

2.2. *Fmt*

Fmt menjelaskan format data suara pada bagian data dalam *data subchunk*. Fmt terdiri dari subchunk1Id, subchunk1Size, audioFormat, numChannels, sampleRate, byteRate, blockAlign, bitsPerSample, extraParamSize, dan extraParams.

1. Subchunk1Id
   Subchunk1Id berisi tulisan “fmt” dalam ASCII
   (ukuran bagian ini 4 *byte*).
2. Subchunk1Size
   Subchunk1Size merupakan jumlah byte dari keseluruhan subchunk pada fmt tetapi tidak termasuk subchunk1Id dan subchunk1Size
   (ukuran bagian ini 4 *byte*).
3. AudioFormat
   AudioFormat berisi nilai yang menyatakan jenis kompresi pada data.Oleh karena berkas WAV tidak dikompresi (PCM – *Pulse Code Modulation*) maka bernilai 1
   (ukuran bagian ini 2 *byte*).
4. NumChannels
   Jumlah channel suara, contoh: untuk Mono bernilai 1, untuk Stereo bernilai 2
   (ukuran bagian ini 2 *byte*).
5. SampleRate
   SampleRate berisi nilai sample rate dari suara, contohnya: 44100 untuk 44,1 kHz atau 44100 *sample* per detik
   (ukuran bagian ini 4 *byte*).
6. ByteRate
   Hasil perhitungan `sampleRate * numChannels * bitsPerSample/8`
   (ukuran bagian ini 4 *byte*).
7. BlockAlign
   Hasil perhitungan `numChannels * bitsPerSample/8` yang menyatakan jumlah *byte* untuk setiap *sample*
   (ukuran bagian ini 2 *byte*).
8. BitsPerSample
   Jumlah bit untuk setiap sample, yaitu 16 bit
   (ukuran bagian ini 2 *byte*).

2.3. *Data Subchunk*

Data subchunk berisi ukuran dan data suara yang sesungguhnya pada bagian data. *Data subchunk* terdiri dari subchunk2Id, subchunk2Size, dan data.

1. Subchunk2Id
   Subchunk1Id berisi tulisan “data” dalam ASCII
   (ukuran bagian ini 4 *byte*).
2. Subchunk2Size
   Subchunk2Size merupakan ukuran (n byte) bagian data
   (ukuran bagian ini 4 *byte*).
3. Data
   Data berisi representasi suara sesungguhnya yang informasinya telah disebutkan pada fmt
   (ukuran bagian ini n *byte*).

3. DISTRIBUSI *DATA SUBCHUNK*

Data subchunk berisi *sample-sample* suara yang runtut dengan *channel* masing-masing. Untuk mendistribusikan nilai 2 byte sebuah data subchunk ke masing-masing share digunakan prosedur berikut.

```
Int[]  void  get2ByteShares(byte[]  realByte,
int numShares)
{
       Int[] Temp = new Integer[numShares];
       Int SumCount = 0
       Int x = toInteger(realByte);
       For (int I = 0; I < numShares; i++)
       {
              Int y = randomInteger();
              while ((SumCount + y > 32767)
&& (SumCount + y < -32768) && ((SumCount + y
== x) && (I == numShares - 1)))
              {
                     Y = randomInteger();
              }
              Else
              {
                     SumCount += y;
                     Temp[i] = y;
              }
       }
       Return temp;
}
```

Prosedur di atas menghasilkan nilai sembarang untuk nilai 2 byte pada berkas WAV share untuk posisi *sample* di *channel* yang sama.
Berikut algoritma utama untuk membentuk keseluruhan *share* dengan *channel-channel*-nya pada

sebuah *sample* memanfaatkan hasil prosedur di atas.

```
Share[]  void  getShares(int[][]  2ByteShares,
int numChannel, int numShares)
{
        Share[] Temp = new Shares[numShares];
        For (int I = 0; I < numShares; i++)
        {
                For   (int   j   =   0;   j   <
numChannel; j++)
                {
                        Temp[i].channel[i]    =
2ByteShares[i][j];
                }
        }
        Return temp;
}
```

Keseluruhan prosedur di atas diulang kembali untuk *sample* berikutnya.

4. PENGGABUNGAN *SHARE* MELALUI INFERENSI

Penggabungan share untuk mendapatkan informasi suara asli dapat dilakukan dengan mendengarkan keselurahan berkas WAV share secara bersamaan (tidak telat sedikit pun) karena efek inferensi baru dapat dirasakan ketika setiap titik gelombang yang terdengar dari setiap share bertepatan dengan titik gelombang yang sama untuk share yang lain.
Setiap channel dari share harus tepat diwakili oleh sebuah speaker dengan masing-masing speaker volumenya sama sehingga rentang amplitudonya sama.
Berikut representasi tabel dari berkas-berkas share yang dihasilkan dan kemudian diperdengarkan untuk satu buah *sample*.

Sample I	Share 1	Share 2	Share 3	Suara Asli
Channel A	0x1a	0x04	0x00	0x1e
Channel B	0x2e	0xa3	0x02	0xd3
Channel C	0x59	0x7e	0x1d	0xf4

Tabel 1 Representasi Tiga Share yang Diperdengarkan pada Satu Sample

5. ANALISIS HASIL

Analisis untuk teknik ini dapat dilakukan pada hasil pembentukan *share*, yaitu berkas WAV *share*. Untuk lebih detilnya dapat dilihat dari tingkat *sample*. Kasus yang dianalisis antara lain: (1) kumpulan *share* yang lengkap, (2) kumpulan *share* yang tidak lengkap, (3) volume yang berbeda pada sebuah *speaker*, dan (4) pemutaran yang tidak berbarengan.

5.1. Kumpulan Share yang Lengkap

	S1	S2	S3	S123	Asli
Ch. L	0x1a	0x04	0x00	0x1e	0x1e
Ch. R	0x2e	0xa3	0x02	0xd3	0xd3

Tabel 2 Analisis Kumpulan Share yang Lengkap

Pemutaran kumpulan share yang lengkap menghasilkan suara *sample* sama dengan aslinya.

5.2. Kumpulan Share yang Tidak Lengkap

	S1	S2	S3	S123	Asli
Ch. L	0x1a	0x04	-	0x1e	0x1e
Ch. R	0x2e	0xa3	-	0xd1	0xd3

Tabel 3 Analisis Kumpulan *Share* yang Tidak Lengkap

Pemutaran kumpulan share yang lengkap menghasilkan suara *sample* tidak sama dengan aslinya.

5.3. Volume Berbeda pada *Speaker* (Misalnya *Share* 2 dengan Faktor Perbesaran k <> 1)

	S1	k.S2	S3	S123	Asli
Ch. L	0x1a	k(0x04)	0x00	0x1e + (k - 1) (0x04)	0x1e
Ch. R	0x2e	K(0xa3)	0x02	0xd3 + (k − 1) (0xa3)	0xd3

Tabel 4 Analisis Perbedaan Volume

Pemutaran kumpulan share yang lengkap menghasilkan suara *sample* tidak sama dengan aslinya.

5.4. Pemutaran *Share* Tidak Bersamaan (Misalnya S1 Telat)

	S1	S2	S3	S123	Asli
Ch. L	-	0x04	0x00	0x04	0x1e
Ch. R	-	0xa3	0x02	0xa5	0xd3

Tabel 5 Analisis Ketidakbersamaan Pemutaran *Share*

Oleh karena *Share* 1 telat dan tidak pada waktu seharusnya maka *Share* 1 tidak dapat mengikuti inferensi yang sedang dialami *Share* 2 dan *Share* 3. Hasilnya suara *sample* tidak sama dengan aslinya.

4. Akurasi pemutaran berkas dan sinkronisasi *clock* pada sejumlah pemutar berkas *share* sangat diperlukan untuk proses pemerolehan informasi melalui penggabungan *share*.

6. KESIMPULAN

Dari hasil analisis didapat beberapa kesimpulan, antara lain:

1. Teknik inferensi pada kriptografi suara cocok digunakan untuk berkas WAV tanpa kompresi.
2. Teknik inferensi pada kriptografi suara ini tidak dapat diterapkan Skema (k, n), yaitu dengan sebagian berkas share, informasi suara asli dapat didengar. Hal ini dikarenakan proses penggabungan *share* merupakan operasi penjumlahan yang nilainya harus tepat.
3. Pemerolehan informasi suara asli untuk *channel* dan *share* yang banyak memakan banyak sumber daya pemutar berkas WAV ditambah banyak speaker.

DAFTAR REFERENSI

[1] Sapp, Craig S. Modifikasi terakhir 20 Januari 2003. *Microsoft WAVE Soundfile Format*. http://ccrma.stanford.edu/courses/422/projects/WaveFormat/ (diakses pada tanggal 30 Januari 2007).

[2] Naor Moni dan Adi Shamir. -. *Visual Cryptography*. -.

[3] Electrical Engineering Training Series. -. *Pulse-Code Modulation*. Integrated Publishing. http://www.tpub.com/neets/book12/49l.htm (diakses pada tanggal 3 Januari 2008).

[4] Cisco Systems. Modifikasi terakhir 2 Februari 2006. *Waveform Coding Techniques*. http://www.cisco.com/warp/public/788/signalling/waveform_coding.pdf (diakses pada tanggal 3 Januari 2008).