WATERMARKING
BASIS DATA RELASIONAL

Hermanto Ong – NIM : 13503069

Program Studi Teknik Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
E-mail : if13069@students.if.itb.ac.id

Abstrak

Makalah ini membahas mengenai kebutuhan *watermarking* terhadap basis data relasional, yang bertujuan untuk mencegah terjadinya pembajakan terhadap data yang tersimpan dalam basis data relasional, mengidentifikasi karakteristik yang unik dari data relasional, dan menyediakan properti-properti yang diharapkan dari sebuah sistem *watermarking* untuk data relasional. Sebuah *watermark* dapat diterapkan terhadap relasi dalam basis data yang memiliki atribut-atribut sedemikian sehingga sedikit perubahan pada nilai data tidak akan mempengaruhi aplikasinya.

Teknik *watermarking* terhadap basis data relasional menjamin bahwa beberapa posisi bit dari atribut dalam sebuah *tuple* mengandung nilai spesifik. *Tuple*, atribut dalam sebuah *tuple*, posisi bit dalam sebuah atribut, dan nilai bit spesifik secara algoritmik ditentukan berdasarkan kontrol dari sebuah kunci privat, yang hanya diketahui oleh sang pemilik data. Pola-pola bit inilah yang membentuk *watermark*. Hanya orang yang mengetahui kunci privat sajalah yang dapat mendeteksi *watermark* dengan probabilitas tinggi. Proses pendeteksian *watermark* tidak membutuhkan akses baik terhadap data aslinya maupun terhadap *watermark* itu sendiri. *Watermark* dapat dideteksi bahkan dalam sebuah himpunan bagian yang kecil dari relasi yang diberi *watermark*, dengan syarat himpunan bagian tersebut masih mengandung beberapa *mark* (tanda).

Hasil analisis terhadap teknik *watermarking* yang digunakan pada basis data relasional menunjukkan bahwa teknik tersebut cukup tangguh dan kuat terhadap berbagai bentuk serangan dan manipulasi terhadap data. Di samping itu, algoritma *watermarking* yang digunakan juga memiliki performansi yang cukup tinggi sehingga dapat diaplikasikan dalam dunia nyata.

***Kata kunci**: watermarking*, *watermark*, *mark*, basis data relasional, data relasional, kunci privat.

1. Pendahuluan

Pembajakan terhadap aset-aset digital seperti perangkat lunak, citra, video, audio, dan teks telah lama menjadi perhatian penting dari sang pemiliknya. Proteksi terhadap aset-aset ini biasanya dilakukan berdasarkan penyisipan *digital watermark* ke dalam data [4] [9] [11]. Proses penyisipan *watermark* tersebut menghasilkan sedikit penurunan kualitas dari objek yang diberi *watermark*. Penurunan kualitas inilah yang disebut *mark* dan kumpulan dari *mark* tersebut membentuk *watermark*. *Mark* tidak boleh memiliki dampak yang signifikan terhadap kegunaan dari data (data tidak boleh menjadi rusak) dan *mark* juga harus ditempatkan sedemikian sehingga seorang *attacker* tidak dapat menghancurkan *mark* tersebut tanpa membuat data menjadi rusak. Oleh sebab itu, *watermarking* tidak mencegah proses penggandaan tetapi mencegah penggandaan ilegal dengan menyediakan suatu mekanisme pembuktian kepemilikan dari objek yang digandakan dan didistribusikan secara ilegal tersebut.

Peningkatan penggunaan basis data dalam aplikasi di luar jangkauan proteksi pemrosesan data yang telah disediakan, menumbuhkan kebutuhan yang sama untuk *watermarking* terhadap basis data. Sebagai contohnya, dalam industri semikonduktor, data parametrik pada bagian-bagian semikonduktor disediakan oleh tiga buah perusahaan: *Aspect*, *IHS*, dan *IC Master* [1].

Masing-masing mempekerjakan sejumlah besar pegawai untuk secara manual mengekstrak spesifikasi bagian semikonduktor dari *datasheet*. Selanjutnya mereka melakukan lisensi terhadap basis data ini dengan harga yang sangat tinggi. Di samping itu, terdapat pula beberapa perusahaan yang telah melakukan kompilasi terhadap koleksi data pelanggan dan bisnis yang sangat besar. Dalam kehidupan industri keilmuan, aset utama dari perusahaan adalah basis data dari berbagai informasi. Beberapa contoh di atas menunjukkan bahwa sekarang ini data telah menjadi salah satu aset paling berharga yang dimiliki oleh perusahaan.

Keberadaan internet telah memberikan tekanan yang besar terhadap penyedia data agar memberikan layanan yang memudahkan pengguna untuk mencari dan mengakses data dalam basis data secara *remote*. Di satu sisi, kecenderungan ini merupakan suatu keuntungan bagi para pengguna. Namun di sisi lain, kecenderungan ini juga menghasilkan suatu ancaman baru berupa pencurian data dari penyedianya. Oleh sebab itu, dibutuhkan suatu cara atau kemampuan untuk mengidentifikasi pembajakan terhadap data yang tersimpan di dalam suatu basis data.

Pengelolaan hak dari data relasional melalui *watermarking* telah menjadi suatu topik penting untuk penelitian dalam bidang basis data. Relasi basis data yang dapat diberi *watermark*, memiliki atribut-atribut sedemikian sehingga sedikit perubahan dari nilai data tidak akan mempengaruhi aplikasinya. Permasalahannya adalah adakah himpunan data di dalam dunia nyata yang dapat menerima sejumlah kecil kesalahan (*error*) tanpa menurunkan tingkat kegunaan dari data tersebut.

Data meteorologi yang digunakan dalam membangun model prediksi cuaca, memenuhi persyaratan tersebut [2]. Vektor angin dan ketepatan suhu dalam data ini diperkirakan sekitar 1.8 m/s dan 0.5º C [2]. Kesalahan (*error*) yang dihasilkan oleh *watermarking* dapat diakomodasi melalui batas toleransi dalam data ini. Contoh lainnya, hasil segmentasi pelanggan dari sebuah perusahaan barang tidak akan terpengaruh jika penyedia data pelanggan menambah atau mengurangi beberapa data dari sejumlah kecil transaksi.

1.1 Teknik *Watermarking* Baru Untuk Data Relasional

Terdapat banyak literatur yang membahas mengenai *watermarking* terhadap data multimedia [4] [9] [11]. Kebanyakan teknik yang digunakan berawal dari *watermarking* terhadap citra diam [10] dan kemudian berkembang terhadap video [8] dan audio [3] [6]. Meskipun banyak sekali teknik *watermarking* yang dapat dipelajari dari literatur-literatur tersebut, terdapat suatu tantangan baru dalam bidang *watermarking* sehubungan dengan perbedaan karakteristik dari data relasional dan data multimedia [1]. Perbedaan tersebut mencakup:

a. Sebuah objek multimedia terdiri atas sejumlah besar bit dengan tingkat perulangan bit (redundansi) yang cukup tinggi. Oleh sebab itu, *watermark* memiliki tempat yang cukup luas untuk proses penyembunyiannya. Sementara itu, suatu relasi basis data terdiri atas kumpulan *tuple*, di mana setiap *tuple* merepresentasikan objek yang berbeda (terpisah). Untuk itu, penyembunyian *watermark* harus disebar terhadap seluruh objek yang terpisah.
b. Posisi spasial/temporal relatif bagian-bagian dari suatu objek multimedia umumnya tidak berubah. Di sisi lain, kumpulan *tuple* membentuk sebuah relasi, dan di dalam relasi tidak terdapat urutan yang pasti dari *tuple-tuple* penyusunnya.
c. Bagian-bagian dari sebuah objek multimedia tidak dapat dibuang atau diganti tanpa menimbulkan perubahan pada objek yang dapat diamati secara jelas. Akan tetapi, pembajakan terhadap sebuah relasi dapat dilakukan dengan membuang beberapa *tuple* dalam relasi atau menggantinya dengan *tuple* dari relasi lain.

Perbedaan-perbedaan inilah yang menyebabkan teknik *watermarking* pada data multimedia tidak dapat diterapkan secara langsung pada data relasional. Untuk

membahas hal ini lebih jauh, sebuah relasi dapat dipetakan menjadi sebuah citra dengan memperlakukan setiap nilai atribut sebagai sebuah piksel [1]. Namun, citra yang telah didefinisikan tersebut tidak memiliki sebagian besar properti yang dimiliki oleh citra asli. Sebagai contohnya, piksel yang bersebelahan dalam sebuah citra asli umumnya memiliki keterhubungan yang tinggi dan asumsi ini telah menjadi basis bagi banyak teknik *watermarking* seperti *predictive coding* untuk menentukan lokasi *watermark* [7]. Beberapa teknik lainnya menerapkan sebuah transformasi (misalnya *discrete Fourier*, *discrete cosine*, *Mellin-Fourier*, *Wavelet*) terhadap citra, menyisipkan *watermark* dalam ruang transformasi, dan kemudian melakukan inversi dari transformasi tersebut [7]. *Noise*, yang disebabkan oleh sinyal *watermarking*, disebar kepada seluruh citra. Aplikasi langsung dari teknik-teknik ini terhadap sebuah relasi akan menimbulkan kesalahan (*error*) dalam semua nilai atribut, yang mungkin tidak dapat ditoleransi. Di samping itu, *watermark* seperti ini mungkin tidak dapat bertahan bahkan terhadap perubahan kecil dari relasi.

Teknik-teknik *watermarking* pada teks memanfaatkan properti khusus dari teks berformat. *Watermark* sering disisipkan dengan cara mengubah spasi antara kata-kata dan baris-baris dari teks [13]. Beberapa teknik bergantung pada penyusunan ulang kata-kata pada beberapa kalimat dalam teks [1]. Meskipun teknik-teknik ini mungkin berguna untuk *watermarking* pada relasi yang mengandung *CLOBs* (*character large binary objects*), penerapannya pada relasi yang hanya mengandung tipe data sederhana masih diragukan.

1.2 Ide Dasar Teknik *Watermarking* Untuk Data Relasional

Watermarking data relasional memiliki tantangan teknik dan penerapan praktis yang signifikan sehingga membutuhkan perhatian serius dari komunitas riset basis data. Kebutuhan terhadap sebuah sistem *watermarking* perlu ditetapkan dan diikuti dengan pembangunan teknik-teknik tertentu. Teknik-teknik tersebut kemungkinan besar tetap menggunakan prinsip-prinsip *watermarking* yang telah ada. Meskipun demikian, teknik-teknik tersebut juga memerlukan peningkatan dan inovasi baru dari teknik-teknik yang sudah ada sekarang ini.

Di dalam makalah ini akan dibahas mengenai kebutuhan *watermarking* tersebut. Untuk menjamin kemungkinan penerapan (*feasibility*) dari *watermarking* data relasional, akan dibahas juga sebuah teknik yang efektif untuk memenuhi kebutuhan tersebut. Teknik ini memberi tanda (*mark*) hanya pada atribut-atribut numerik dan mengasumsikan bahwa atribut-atribut yang diberi tanda tersebut dapat menerima (mentoleransi) sedikit perubahan pada nilainya. Ide dasar dari teknik ini adalah untuk menjamin bahwa beberapa posisi bit untuk beberapa atribut dari sejumlah *tuple* mengandung nilai yang spesifik. *Tuple*, atribut dalam sebuah *tuple*, posisi bit dalam sebuah atribut, dan nilai bit spesifik secara algoritmik ditentukan berdasarkan kontrol dari sebuah kunci privat, yang hanya diketahui oleh sang pemilik data. Pola-pola bit inilah yang membentuk *watermark*. Hanya orang yang mengetahui kunci privat sajalah yang dapat mendeteksi *watermark* dengan probabilitas tinggi. Hasil analisis terhadap teknik *watermarking* yang digunakan pada basis data relasional menunjukkan bahwa teknik tersebut cukup tangguh dan kuat terhadap berbagai bentuk serangan dan manipulasi terhadap data.

2. Model

Misalkan Alice adalah pemilik dari relasi R yang mengandung η *tuple*, dengan ω *tuple* yang telah ditandai [1]. Properti-properti berikut ini harus terpenuhi:

Detectability **–** Alice seharusnya dapat mendeteksi *watermark*-nya dengan memeriksa ω *tuple* dari sebuah basis data yang dicurigai. Jika pola bit (*watermark*) yang dimilikinya berhasil ditemukan dalam seluruh ω *tuple*, Alice memiliki alasan yang kuat untuk mencurigai telah terjadinya pembajakan. Meskipun demikian, sangat beralasan bagi Alice untuk curiga jika polanya ditemukan dalam setidaknya r *tuple* ($r \leq \omega$), di mana r bergantung pada ω dan sebuah nilai awal α**,** yang disebut level signifikan dari tes. Nilai dari r sangat

menentukan bahwa kemungkinan Alice akan menemukan pola bitnya dalam setidaknya *r tuple* dari ω *tuple* adalah lebih kecil dari α.

Robustness – *Watermark* seharusnya tahan terhadap berbagai serangan yang bertujuan untuk menghapusnya. Misalkan Mallory adalah seorang *attacker*. Dia mengubah ζ *tuple* dari relasi *R* milik Alice. *Watermark* dikatakan aman dari penghapusan jika serangan tidak dapat menghancurkan *mark* dari setidaknya *r tuple*, di mana *r* bergantung pada ω dan α.

Incremental Updatability – Setelah memberi *watermark* pada *R*, Alice seharusnya dapat melakukan perubahan (*update*) pada *R* di masa yang akan datang tanpa menghancurkan *watermark* tersebut. Ketika Alice menambah/menghapus *tuple* atau memodifikasi nilai dari atribut *R*, *watermark* seharusnya dapat di-*update* secara inkremental. Untuk itu, nilai *watermark* seharusnya dapat dihitung ulang untuk *tuple* yang ditambahkan atau dimodifikasi.

Imperceptibility – Modifikasi yang disebabkan oleh *mark* seharusnya tidak mengurangi kegunaan dari basis data. Selain itu, ukuran statistika yang umum digunakan seperti rata-rata dan variansi dari atribut-atribut numerik seharusnya tidak terpengaruh secara signifikan.

Blind System – Deteksi *watermark* seharusnya tidak memerlukan baik pengetahuan mengenai basis data aslinya maupun *watermark* itu sendiri. Properti ini sangat kritis sebab hal ini memungkinkan *watermark* untuk dideteksi dalam sebuah relasi basis data hasil penggandaan, terlepas dari perubahan yang dilakukan kemudian terhadap relasi aslinya.

Key-Based System – Sesuai dengan Kerckhoffs [12], sistem *watermarking* seharusnya mengasumsikan bahwa metode yang digunakan untuk menyisipkan sebuah *watermark* bersifat publik. Oleh sebab itu, perlindungan hanya bergantung kepada pemilihan kunci privat.

2.1 *Benign Update* dan *Malicious Attack*

Karena relasi dalam basis data dapat di-*update*, *mark* yang terkandung dalam sebuah relasi dapat dibuang melalui *benign update* (perubahan yang lunak) maupun *malicious attack* (serangan yang berbahaya) [1].

Benign Update – Misalkan Mallory berhasil mencuri data milik Alice tanpa menyadari bahwa data tersebut sudah diberi *watermark*. Mallory mungkin mengubah data yang dicurinya ketika ia sedang menggunakannya. *Watermarking* seharusnya bersifat sedemikian sehingga Alice tidak kehilangan *watermark* dalam data miliknya yang dicuri walaupun data tersebut telah diubah.

Malicious Attack – Mallory mungkin mengetahui bahwa data yang dicurinya mengandung sebuah *watermark*. Oleh sebab itu, ia mungkin mencoba untuk menghapus *watermark* tersebut atau mencoba cara lainnya sehingga kepemilikan data tidak lagi dapat dibuktikan. Sistem *watermarking* seharusnya dapat melindungi Alice dari berbagai jenis serangan berbahaya yang dilakukan oleh Mallory, seperti:

Bit Attack – Merupakan percobaan serangan berbahaya untuk menghancurkan *watermark* yang paling sederhana dengan mengubah beberapa bit. Jika Mallory dapat mengubah seluruh bit, ia dapat dengan mudah menghancurkan *watermark*. Meskipun demikian, ia juga akan membuat data tersebut menjadi tidak berguna. Untuk itu, keefektifan sebuah serangan seharusnya mempertimbangkan hubungan di antara jumlah bit-bit yang diubah oleh Mallory dan Alice sebab setiap perubahan dapat menyebabkan terjadinya kesalahan (*error*). Kesalahan yang terlampau banyak tentu saja akan membuat data menjadi tidak berguna.

Randomization Attack – Merupakan serangan berbahaya yang berusaha untuk memberikan nilai acak pada beberapa posisi bit. Serangan *zero out* mengubah nilai dari beberapa posisi bit menjadi nol. Sementara serangan *bit flipping* membalikkan nilai dari beberapa posisi bit. *Benign update* dapat juga dimodelkan sebagai sebuah *randomization attack*.

Rounding Attack – Mallory mungkin mencoba untuk membuang *mark* yang terkandung dalam sebuah atribut numerik dengan cara melakukan pembulatan terhadap semua nilai dari atribut. Serangan ini tidak lebih baik dari *bit attack* yang sudah dibahas di atas. Mallory harus menebak dengan tepat jumlah posisi bit yang digunakan dalam *watermarking*. Jika ia menebak terlalu rendah, serangan yang dilakukannya tidak akan berhasil. Jika ia menebak terlalu tinggi, ia akan menurunkan kualitas dari data tersebut. Bahkan jika tebakannya tepat, data tersebut tidak akan dapat bersaing dengan data aslinya sebab nilai dari data tersebut sudah tidak tepat lagi.

Subset Attack – Mallory mungkin mengambil sebuah himpunan bagian dari *tuple* atau atribut sebuah relasi yang sudah diberi *watermark* dan berharap bahwa *watermark* tersebut hilang.

Mix and Match Attack – Mallory mungkin membuat relasinya sendiri dengan mengambil *tuple-tuple* yang tidak berhubungan (*disjoint*) dari relasi-relasi yang mengandung informasi serupa.

Additive Attack – Mallory mungkin menambahkan *watermark* miliknya kepada relasi milik Alice yang sudah diberi *watermark* dan mengklaim kepemilikan dari relasi tersebut.

Invertibility Attack – Mallory mungkin melakukan *invertibility attack* [5] untuk mengklaim kepemilikan jika ia berhasil menemukan sebuah *watermark* yang disisipkan.

3. Algoritma

Pada bagian ini akan dibahas sebuah teknik untuk *watermarking* relasi dalam basis data dan akan dibuktikan bahwa teknik ini dapat memenuhi kebutuhan yang sudah dijelaskan sebelumnya. Teknik ini memberi tanda hanya pada atribut numerik dan mengasumsikan bahwa atribut yang diberi tanda tersebut sedemikian sehingga sedikit perubahan dalam beberapa nilainya dapat diterima. Tidak seluruh atribut numerik dari sebuah relasi perlu ditandai. Pemilik data bertanggung jawab untuk menentukan atribut mana saja yang akan ditandai.

Watermarking akan diterapkan pada sebuah relasi basis data R yang memiliki skema R (P, A_0, ..., A_{v-1}), di mana P adalah atribut *primary key* [1]. Untuk penyederhanaan, diasumsikan bahwa semua v atribut A_0, ..., A_{v-1} adalah kandidat untuk ditandai. Semua atribut tersebut adalah atribut numerik dan nilainya sedemikian sehingga perubahan dalam ξ *least significant bit* untuk setiap atribut tersebut bersifat *imperceptible* [1].

Celah γ adalah sebuah parameter kontrol yang menentukan jumlah *tuple* yang ditandai, $\omega \approx \eta/\gamma$ [1]. Sering kali terjadi *trade-off* antara γ dan ξ yang menentukan perluasan kesalahan (*error*) yang disebabkan oleh sebuah nilai atribut. Jika *tuple* yang ditandai lebih sedikit, sangat memungkinkan untuk melakukan perubahan yang lebih besar pada nilai dari atribut yang diberi tanda.

Notasi	Keterangan
η	Jumlah *tuple* dalam relasi
v	Jumlah atribut dalam relasi yang tersedia untuk ditandai
ξ	Jumlah *least significant bit* yang tersedia untuk menandai sebuah atribut
$1/\gamma$	Fraksi dari *tuple* yang ditandai
ω	Jumlah dari *tuple* yang ditandai
α	Level signifikan dari tes untuk pendeteksian sebuah *watermark*
r	Jumlah minimum *tuple*, yang telah ditandai dengan benar, yang dibutuhkan untuk pendeteksian

Gambar 1: Notasi

Nilai dari atribut A_i dalam tuple $r \in R$ dinyatakan dengan $r.A_i$. Gambar 1 menunjukkan parameter-parameter penting yang digunakan dalam algoritma ini [1]. Algoritma ini memanfaatkan *message authenticated code* yang akan dibahas kemudian.

3.1 *Message Authenticated Code*

Sebuah fungsi hash satu arah H beroperasi pada sebuah pesan masukan M dengan panjang yang berubah-ubah dan mengembalikan sebuah nilai hash h dengan panjang yang tetap, misalkan, $h = H(M)$ [1].

Fungsi tersebut memiliki karakteristik tambahan sebagai berikut:

a. Diberikan M, mudah untuk menghitung h.
b. Diberikan h, sulit untuk menghitung M sedemikian sehingga $H(M) = h$.
c. Diberikan M, sulit menemukan pesan lain M' sedemikian sehingga $H(M) = H(M')$.

Beberapa fungsi satu arah dideskripsikan dalam [15]. MD5 dan SHA adalah dua buah pilihan yang baik untuk H.

Message authenticated code adalah sebuah fungsi hash satu arah yang bergantung pada sebuah kunci. Misalkan F adalah sebuah *message authenticated code* yang mengacak nilai dari atribut *primary key* $r.P$ dari *tuple* r dan mengembalikan sebuah nilai bilangan bulat. F diberi umpan sebuah kunci privat K, yang hanya diketahui oleh pemiliknya. *Message authenticated code* yang digunakan adalah [15]:

$$F(r.P) = H(K \circ H(K \circ r.P))$$

di mana $\circ$ merepresentasikan konkatenasi.

3.2 Penyisipan *Watermark*

Gambar 2 di bawah merupakan algoritma penyisipan *watermark* [1]. Baris kedua menentukan apakah *tuple* yang sedang dipertimbangkan akan diberi tanda atau tidak. Karena penggunaan *message authenticated code*, hanya pemilik yang mengetahui kunci privat K sajalah, dapat dengan mudah menentukan *tuple* yang sudah diberi tanda. Untuk sebuah *tuple* yang dipilih, baris ketiga menentukan atribut yang akan diberi tanda di antara v kandidat atribut. Untuk sebuah atribut yang dipilih, baris keempat menentukan posisi bit di antara ξ *least significant bit* yang akan diberi tanda. Sekali lagi, hasil dari tes pada baris ketiga dan keempat bergantung kepada kunci privat dari pemiliknya. Untuk menghapus sebuah *watermark*, *attacker* harus menebak tidak hanya *tuple*, tetapi juga atribut yang diberi tanda dalam sebuah *tuple* dan juga posisi dari bitnya.

Subrutine *mark* mengubah bit yang dipilih menjadi nol atau satu bergantung pada nilai hash yang diperoleh dalam baris ketujuh. Oleh sebab itu, hasil dari baris kesembilan (baris kesebelas) adalah nilai dari atribut tetap atau mengalami pengurangan (penambahan). Sebagai akibatnya, *marking* mengurangi beberapa nilai dari sebuah atribut, menambah nilai atribut yang lain, dan tidak mengubah beberapa nilai atribut lainnya.

Basis data umumnya memperbolehkan atribut-atribut untuk memiliki nilai *null*. Jika sebuah nilai atribut yang *null* ditemui pada saat akan memberi tanda sebuah *tuple*, *mark* tidak diterapkan terhadap nilai *null* tersebut. Sebagai gantinya, nilai atribut tersebut dibiarkan tetap seperti semula.

3.3 Pendeteksian *Watermark*

Misalkan Alice mencurigai bahwa relasi S yang diumumkan oleh Mallory merupakan hasil pembajakan dari relasi R miliknya. Himpunan *tuple* dan atribut dalam S bisa jadi merupakan sebuah himpunan bagian dari R. Diasumsikan bahwa Mallory tidak membuang atribut *primary key* atau mengubah nilai dari *primary key* sebab *primary key* mengandung informasi berharga dan mengubahnya akan membuat basis data menjadi kehilangan kegunaannya dari sudut pandang pengguna.

Algoritma pendeteksian *watermark*, yang ditunjukkan dalam Gambar 3, memiliki sifat probabilistik [1]. Baris ketiga menentukan apakah *tuple* s yang sedang dipertimbangkan telah diberi tanda (*mark*) pada saat penyisipan *watermark* dilakukan. Baris keempat dan kelima menentukan atribut dan posisi bit yang telah diberi tanda (*mark*). Subrutine *match* membandingkan nilai bit sekarang dengan nilai yang seharusnya diberikan untuk bit tersebut oleh algoritma penyisipan *watermark*.

Jumlah *tuple* yang sudah dites (*totalcount*) dan jumlah *tuple* yang mengandung nilai bit yang diharapakan (*matchcount*) dapat diketahui dari baris kedelapan. Dalam kerangka kerja probabilistik, hanya dibutuhkan sejumlah tertentu *tuple* yang harus mengandung bit yang sesuai dengan bit yang telah diberi tanda. *Matchcount* dibandingkan dengan *minimum count* yang dihasilkan oleh fungsi *threshold* untuk keberhasilan tes pada level signifikan α yang dipilih.

```
// Kunci privat K diketahui hanya oleh pemilik basis data.
// Parameter γ, v, dan ξ juga bersifat privat terhadap pemilik basis data.

1)  foreach tuple r ∈ R do
2)      if (F(r.P) mod γ equals 0) then          // tuple ini diberi tanda (mark)
3)          attribute_index i = F(r.P) mod v     // atribut A_i diberi tanda (mark)
4)          bit_index j = F(r.P) mod ξ           // bit ke-j diberi tanda (mark)
5)          r.A_i = mark(r.P, r.A_i, j)

6)  mark(primary_key pk, number v, bit_index j) return number

7)      first_hash = H(K ∘ pk)

8)      if (first_hash is even) then
9)          set the j^th least significant bit of v to 0
10)     else
11)         set the j^th least significant bit of v to 1

12)     return v
```

Gambar 2: Algoritma Penyisipan *Watermark*

```
// K, γ, v, dan ξ memiliki nilai yang sama dengan nilai yang digunakan pada saat penyisipan
   watermark.
// α adalah level signifikan tes yang dipilih terlebih dahulu oleh pendeteksi.

1)  totalcount = matchcount = 0

2)  foreach tuple s ∈ S do
3)    if (F(s.P) mod γ equals 0) then          // tuple ini mengandung tanda (mark)
4)          attribute_index i = F(s.P) mod v   // atribut A_i mengandung tanda (mark)
5)          bit_index j = F(s.P) mod ξ         // bit ke-j mengandung tanda (mark)
6)          totalcount = totalcount + 1
7)          matchcount = matchcount + match(s.P, s.A_i, j)

8)  r = threshold(totalcount, α)
9)  if (matchcount ≥ r) then suspect piracy

10) match(primary_key pk, number v, bit_index j) return int

11)   first_hash = H(K ∘ pk)

12)   if (first_hash is even) then
13)      return 1 if the j^th least significant bit of v is 0 else return 0
14)   else
15)      return 1 if the j^th least significant bit of v is 1 else return 0
```

Gambar 3: Algoritma Pendeteksian *Watermark*

Gambar 3 mengasumsikan untuk penyederhanaan bahwa semua kandidat atribut A_0, ..., A_{v-1} berada dalam relasi *S*. Jika Alice menemukan sebuah *tuple s* di mana di dalamnya ia seharusnya telah memberi *mark* pada atribut A_i (baris keempat), tetapi Mallory telah menghilangkan A_i maka ia dapat mengabaikan *tuple* tersebut. Begitu juga jika sebuah *tuple* ditemukan dengan atribut A_i seharusnya sudah diberi *mark*, tetapi A_i memiliki sebuah nilai *null* maka *tuple* tersebut diabaikan. Dengan demikian, nilai dari *matchcount* dan *totalcount* juga tidak boleh terpengaruh (tetap).

3.4 Relasi Tanpa *Primary Key*

Teknik *watermarking* yang dijelaskan di atas, didasarkan atas keberadaan sebuah *primary key* dalam relasi yang akan diberi *watermark*. *Primary key* umumnya muncul dalam situasi dunia nyata dan teknik *watermarking* di atas mengasumsikan bahwa relasi yang akan diberi *watermark* juga memiliki sebuah *primary key*. Jika asumsi ini tidak dipenuhi, maka teknik *watermarking* di atas harus dimodifikasi.

Misalkan relasi *R* mengandung sebuah atribut numerik *A*. Bit-bit dari atribut *A* dipartisi menjadi dua kelompok. χ bit dari nilai *r.A* digunakan sebagai pengganti *primary key* dari *tuple r* dan sisa ξ bit digunakan untuk pemberian tanda (*mark*) [1]. Dengan demikian, skema yang telah dijelaskan sebelumnya dapat tetap dipakai. Cara ini dapat bekerja hanya jika *r.A* tidak memiliki banyak duplikasi. Terlalu banyaknya duplikasi dalam χ bit dari nilai *r.A* akan menghasilkan banyak *mark* yang identik, yang dapat dieksploitasi oleh seorang *attacker*.

Jika relasi memiliki lebih dari satu atribut, salah satu dari atribut tersebut dapat digunakan sebagai pengganti *primary key* dan atribut lainnya untuk pemberian tanda (*mark*). Atribut yang memiliki duplikasi paling sedikit sebaiknya digunakan sebagai pengganti *primary key*. Pengganti *primary key* tersebut juga dapat disebar ke lebih dari satu atribut untuk mereduksi duplikasi. Kekurangan dari cara ini adalah jika salah satu dari atribut ini dihilangkan oleh Mallory, Alice tidak akan dapat mendeteksi *watermark* tersebut.

4. Analisis

Pada bagian ini akan dianalisis properti-properti dari teknik *watermarking* untuk basis data relasional [1].

4.1 *Cumulative Binomial Probability*

Percobaan berulang yang tidak saling berhubungan disebut percobaan *Bernoulli* jika hanya terdapat dua buah kemungkinan hasil untuk setiap percobaan dan probabilitasnya tetap sama selama percobaan dilakukan. Misalkan $b(k; n, p)$ adalah kemungkinan bahwa n percobaan *Bernoulli* dengan probabilitas p untuk kesuksesan dan $q = 1 - p$ untuk kegagalan, menghasilkan k buah kesuksesan dan $n - k$ buah kegagalan. Maka diperoleh:

$$b(k; n, p) = \binom{n}{k} p^k q^{n-k} \qquad (1)$$

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} 0 \le k \le n \qquad (2)$$

Jumlah kesuksesan dalam n percobaan dinyatakan dengan S_n. Kemungkinan dicapai setidaknya k buah kesuksesan dalam n percobaan, kemungkinan binomial kumulatif, dapat dituliskan sebagai berikut:

$$P\{S_n \ge k\} = \sum_{i=k}^{n} b(i; n, p) \qquad (3)$$

Untuk lebih ringkasnya, kemungkinan binomial kumulatif di atas dapat didefinisikan sebagai berikut:

$$B(k; n, p) := \sum_{i=k}^{n} b(i; n, p) \qquad (4)$$

4.2 Kerangka Kerja Probabilistik

Pada bagian ini akan dibahas subrutine *threshold* yang digunakan pada baris kedelapan dari Gambar 3. Misalkan totalcount = ω pada saat Alice menjalankan algoritma pendeteksian *watermark*. Untuk itu, Alice melihat pada ω buah bit dan

mengamati jumlah bit yang nilainya sesuai (cocok) dengan nilai yang diberikan oleh algoritma penyisipan *watermark*. Kemungkinan bahwa setidaknya r dari ω buah bit acak – masing-masing bit sama dengan nol atau satu dengan kemungkinan yang sama, tidak bergantung kepada bit lainnya – sesuai (cocok) dengan nilai yang diberikan adalah $B(r; \omega, 1/2)$. Dengan kata lain, kemungkinan setidaknya r buah bit sesuai (cocok) adalah $B(r; \omega, 1/2)$. Oleh sebab itu, subrutine:

subrutine *threshold*(ω, α) **return** *count*

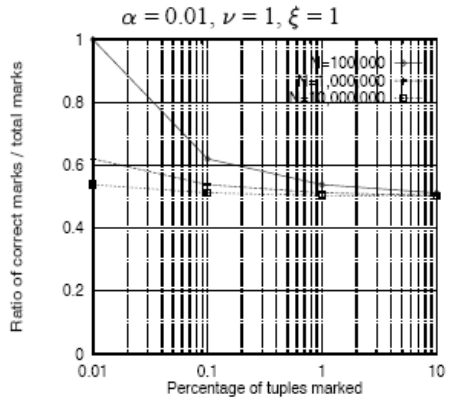
mengembalikan r minimum sedemikian sehingga $B(r; \omega, 1/2) < \alpha$.

Level signifikan α menentukan tingkat kepercayaan terhadap sistem dalam melakukan kesalahan. Oleh sebab itu, α adalah kemungkinan bahwa Alice akan menemukan *watermark* miliknya dalam sebuah relasi basis data yang tidak diberi *mark* olehnya. Dengan memilih nilai α yang rendah, Alice dapat meningkatkan kepercayaannya bahwa jika algoritma pendeteksian *watermark* menemukan *watermark* miliknya dalam sebuah relasi yang dicurigai, relasi tersebut kemungkinan besar memang merupakan sebuah hasil penggandaan ilegal (pembajakan).

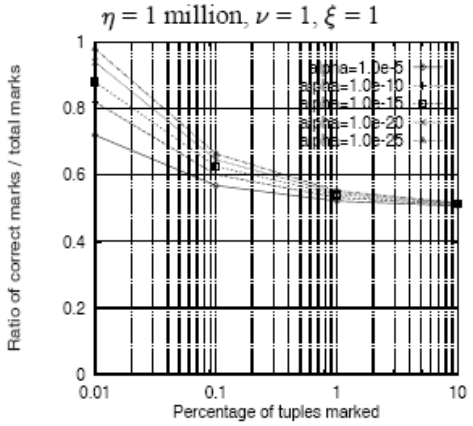
4.3 Kemungkinan Pendeteksian

Dari subbab 4.2 di atas dapat dilihat bahwa kemungkinan pendeteksian dari sebuah *watermark* bergantung pada level signifikan α dan jumlah *tuple* yang diberi *mark* ω. ω sendiri bergantung pada jumlah *tuple* dalam relasi η dan parameter celah γ.

Pendeteksian *Watermark* – Gambar 4 menggambarkan proporsi dari *tuple* yang diberi *mark*, yang harus memiliki nilai *watermark* yang benar untuk kesuksesan pendeteksian (dinyatakan dengan r/ω). Hasil untuk relasi dengan ukuran yang berbeda-beda juga telah digambarkan, dengan mengambil asumsi nilai $\alpha = 0.01$. Sumbu X mengubah-ubah persentase dari *tuple* yang diberi *mark* (fraksi $1/\gamma$ diekspresikan sebagai persentase). Persentase dari *tuple* yang diberi *mark* 0.01%, 0.1%, 1.0%, dan 10% berkorespondensi dengan nilai γ 10000, 1000, 100, dan 10.



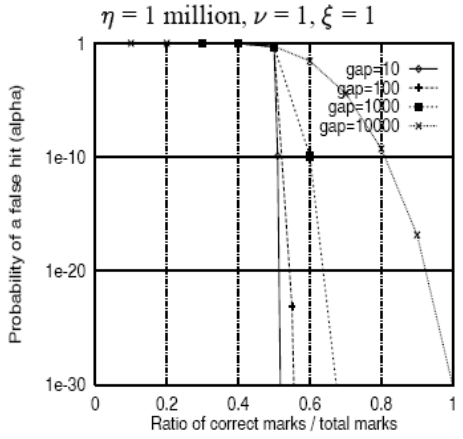
Gambar 4: Proporsi Dari *Tuple* Yang Diberi *Watermark* Dengan Benar Untuk Kebutuhan Pendeteksian



Gambar 5: Proporsi Dari *Tuple* Yang Diberi *Watermark* Dengan Benar, Yang Dibutuhkan Untuk Nilai α Yang Menurun (Alpha ≡ α)

Gambar di atas menunjukkan bahwa proporsi yang dibutuhkan dari *tuple* yang diberi *mark* dengan benar menurun ketika persentase dari *tuple* yang diberi *mark* meningkat. Proporsi ini juga menurun ketika jumlah *tuple* dalam relasi meningkat. Kita membutuhkan lebih dari 50% *tuple* yang diberi *mark* dengan benar untuk membedakan sebuah *watermark* dari sebuah kemunculan yang kebetulan, tetapi dengan suatu pilihan γ yang tepat, persentase ini dapat dibuat menjadi lebih kecil dari 51%. Gambar ini juga menunjukkan bahwa untuk relasi yang lebih besar, kita dapat menandai suatu persentase yang lebih kecil dari jumlah total *tuple* dan masih memelihara/menjaga kemampuan pendeteksian dari *watermark*.

Dalam Gambar 5, ditunjukkan proporsi yang dibutuhkan dari *tuple* yang diberi *mark* dengan benar untuk berbagai jenis nilai α. Hasilnya ditunjukkan untuk satu juta *tuple* relasi. Oleh sebab itu, kita harus menemukan secara proporsional sebuah jumlah yang lebih besar dari *tuple* yang diberi *mark* dengan benar ketika nilai α menurun. Dan yang tidak kalah pentingnya adalah untuk nilai α yang sangat rendah, proses pendeteksian *watermark* harus tetap dimungkinkan. Bahkan untuk $\gamma = 10000$ dan tersedia hanya 100 *tuple* yang diberi *mark* dari sejuta *tuple*, tetap dimungkinkan untuk mendeteksi *watermark* apabila 82% dari *mark*-nya memiliki nilai yang benar untuk α serendah $1.0e-10$. Untuk $\gamma = 10$, kurang dari 52% *tuple* yang diberi *mark* dibutuhkan untuk memiliki *mark* yang benar untuk setiap nilai α.



Gambar 6: Peluang Pemulihan Dengan Kesempatan *r Tuple* Memiliki Nilai *Watermark* Yang Benar Dari ω *Tuple* (Gap $\equiv \gamma$)

False Hits – Gambar 6 mengilustrasikan ketangguhan sistem terhadap *false hits*. Grafik tersebut telah digambari untuk satu juta *tuple* relasi dan untuk nilai-nilai parameter *gap* γ yang berbeda. Sumbu-X merepresentasikan berbagai variasi dari rasio *r tuple* yang diberi *mark*, yang memiliki nilai *watermark* yang benar terhadap jumlah total dari *tuple* yang diberi *mark* ω. Sumbu-Y menunjukkan $B(r; \omega; 1/2)$, yang merupakan kemungkinan kesalahan menemukan setidaknya *r tuple* yang memiliki nilai *watermark* yang benar dari ω *tuple*.

Grafik tersebut menunjukkan terdapatnya sebuah penurunan kemungkinan yang tajam bahwa lebih dari 50% *tuple* akan memiliki sebuah *mark* yang benar. Untuk $\gamma = 10$, kemungkinan bahwa 51% atau lebih dari *tuple* yang diberi *mark* memiliki sebuah *mark* yang benar adalah sebesar 1.29e-10. Bahkan untuk nilai γ yang sangat besar = 10000, kemungkinan bahwa 80% atau lebih dari *tuple* yang diberi *mark* memiliki sebuah *mark* yang benar dengan kesempatan hanya sebesar 5.58e-10. Oleh sebab itu, dengan memilih nilai untuk γ dan α yang tepat, *false hits* dapat dibuat sangat tidak mungkin (mustahil) terjadi.

4.4 *Robustness*

Pada bagian ini akan dianalisis ketangguhan (*robustness*) dari teknik *watermarking* basis data relasional terhadap berbagai bentuk serangan berbahaya yang mungkin dilakukan. Misalkan, Alice telah memberi *mark* pada ω ($\approx \eta/\gamma$) *tuple*. Untuk mendeteksi *watermark* miliknya, Alice menggunakan level signifikan α yang menentukan jumlah minimum *tuple r* dari ω yang harus memiliki *mark*-nya secara utuh.

4.4.1 *Bit-Flipping Attack*

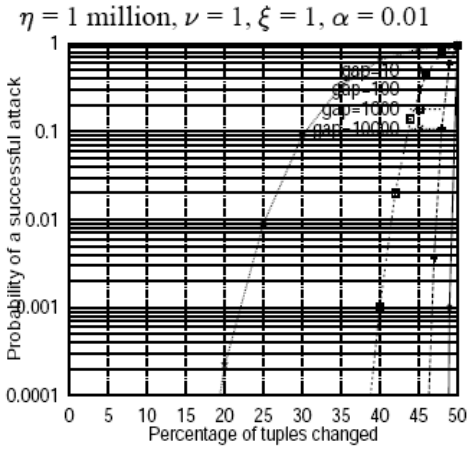
Dalam serangan bentuk ini, Mallory mencoba untuk menghancurkan *watermark* milik Alice dengan cara membalikkan nilai (*flip*) pada posisi bit yang diduganya telah diberi *mark*. Analisis dan hasil adalah sama untuk *zero-out attack* dan *randomization attack*.

Asumsikan bahwa Mallory secara tidak terduga mengetahui nilai dari parameter v dan ξ yang digunakan oleh Alice. Nilai dari ξ diasumsikan sama untuk semua atribut v. Karena Mallory tidak mengetahui posisi bit mana yang telah diberi *mark*, ia secara acak memilih ζ *tuple* dari η *tuple*. Untuk setiap *tuple* yang dipilih, ia membalikkan (*flip*) seluruh bit dalam seluruh posisi bit ξ dalam seluruh atribut v. Untuk mencapai keberhasilan, ia harus dapat membalikkan setidaknya $\bar{r} = \omega - r + 1$ *mark*.

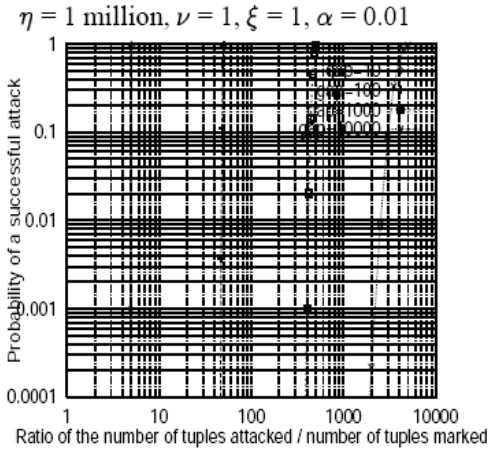
Peluang bahwa serangan ini akan berhasil dapat diperkirakan (diestimasi) sebagai berikut:

$$\sum_{i=\bar{r}}^{\omega} \frac{\binom{\omega}{i}\binom{\eta-\omega}{\zeta-i}}{\binom{\eta}{\zeta}} \qquad (5)$$

Mallory melakukan *sampling* tanpa penukaran ζ *tuple* dari η *tuple* potensial yang sudah diberi *mark*, dan berharap bahwa setidaknya $\bar{r}$ *tuple* dari ω *tuple* yang diberi *mark* akan muncul dalam *sample*-nya.



Gambar 7: Peluang Serangan Yang Berhasil (*Gap* ≡ γ)



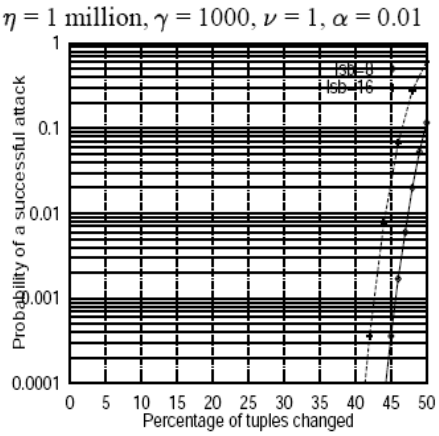
Gambar 8: Kesalahan Yang Berlebihan Dalam Menghancurkan Sebuah *Watermark* (*Gap* ≡ γ)

Peluang Keberhasilan – Gambar 7 menunjukkan peluang keberhasilan dari serangan di atas pada satu juta *tuple* relasi dengan nilai α = 0.01. Kita mengasumsikan bahwa Alice telah memberi *mark* hanya pada *least significant bit* dari satu atribut dan informasi ini secara tidak terduga dapat diketahui oleh Mallory. Kita telah melakukan pada sumbu-X variasi persentase *tuple* yang diubah. Untuk γ = 10000, jika Mallory mengubah 40% dari *tuple*, ia memiliki 64% peluang untuk menghancurkan *watermark*. Untuk γ = 1000, ia harus mengubah 46% dari *tuple* untuk mendapatkan 44% peluang keberhasilan. Peluangnya untuk berhasil hanya 11% jika ia mengubah 48% dari *tuple* ketika γ = 100. Berapapun yang nilainya kurang dari 50% tidak akan menghancurkan *watermark* untuk γ = 10. Perhatikan bahwa bukan ketertarikan Mallory untuk membalikkan (*flip*) lebih dari 50% bit yang diberi *mark*. Dalam kasus tersebut, Alice dapat mendeteksi *watermark* dengan membalikkan ulang bit-bit yang bersesuaian dan menerapkan algoritma pendeteksian *watermark* pada data yang telah ditransformasikan.

Gambar 8 menunjukkan kesalahan (*error*) berlebihan yang disebabkan oleh Mallory dalam menghancurkan sebuah *watermark*, yang diekspresikan sebagai rasio dari jumlah total *tuple* yang diserang terhadap jumlah *tuple* yang diberi *mark*. *Ketika* γ = 10000, Mallory harus mengubah hampir 5000 kali jumlah *tuple* yang diberi *mark* untuk menghancurkan *watermark*. Oleh sebab itu, data Mallory akan mengandung lebih dari tiga kali urutan besar kesalahan dibandingkan dengan versi Alice. Rasio ini menjadi 500, 50, dan 5 untuk γ = 1000, 100, dan 10. Untuk itu, Alice dapat memilih sebuah nilai γ bergantung kepada nilai toleransi data terhadap kesalahan dan memaksa Mallory untuk menghasilkan kesalahan yang lebih besar (dan oleh karena itu membuat data Mallory tidak seperti yang dikehendaki).

Gambar 8 juga mengatakan bahwa jika Mallory dibatasi untuk membuat perubahan terhadap data hanya sebanyak Alice, ia tidak dapat menghancurkan *watermark* kepunyaan Alice. Jadi, jika data sedemikian sehingga terdapat sebuah batas jumlah perubahan yang dapat ditoleransinya tanpa membuatnya menjadi tidak berguna dan Alice menyediakan jumlah maksimum

perubahan yang mungkin, maka Mallory tidak akan bisa membuang *watermark* tersebut.



Gambar 9: Peluang Dari Sebuah Serangan Yang Berhasil Ketika ($lsb \equiv \xi$) Ditaksir Terlalu Rendah Oleh 1

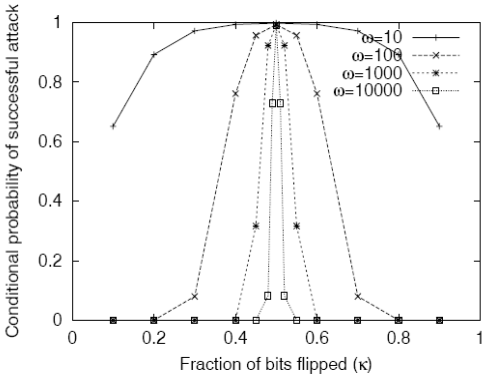
Bermacam-macam ξ – Sekarang pertimbangkan kasus di mana Alice memberi *mark* pada satu dari $\xi > 1$ *least significant bit*, tetapi hanya dalam sebuah atribut. Pertama, asumsikan Mallory bagaimanapun caranya dapat mengetahui nilai eksak dari ξ. Meskipun demikian, Mallory tidak akan mengetahui posisi bit mana yang telah diberi *mark* dalam sebuah *tuple* yang spesifik. Setelah memutuskan pada sebuah *tuple*, ia harus membalikkan semua ξ *least significant bit* dari atribut untuk dapat menghancurkan sebuah *mark*. Jika kita membandingkan peluang dari sebuah serangan yang berhasil terhadap persentase *tuple* yang diubah selama skenario ini, akan diperoleh grafik yang identik dengan Gambar 7. Perbedaannya adalah bahwa Mallory memiliki ξ kali kesalahan yang berlebihan. Mallory harus mengubah ξ bit dalam sebuah nilai atribut sementara Alice hanya mengubah satu bit. Sebuah nilai ξ yang lebih besar juga menuju ke kesalahan yang lebih besar dalam data Mallory.

Misalkan, Mallory tidak mengetahui nilai eksak dari ξ yang digunakan oleh Alice. Gambar 9 menunjukkan apa yang terjadi ketika Mallory menaksir terlalu rendah nilai dari ξ hanya oleh satu bit. Kita mengambil η = satu juta, γ = 1000, ν = 1, dan α = 0.01, dan menghitung peluang dari sebuah serangan yang berhasil untuk nilai ξ yang berbeda-beda dan persentase *tuple* yang diubah. Bandingkan Gambar 9 dengan Gambar 7 untuk kasus di mana γ = 1000. Dibandingkan terhadap Gambar 7, bidang sebelah kanan menunjukkan sebuah reduksi dalam peluang dari serangan yang berhasil. Kenyataannya, Gambar 9 tidak mengandung nilai untuk ξ = 4 dan 2. Hal ini disebabkan oleh peluang keberhasilan menjadi sangat kecil (kurang dari 0.0001).

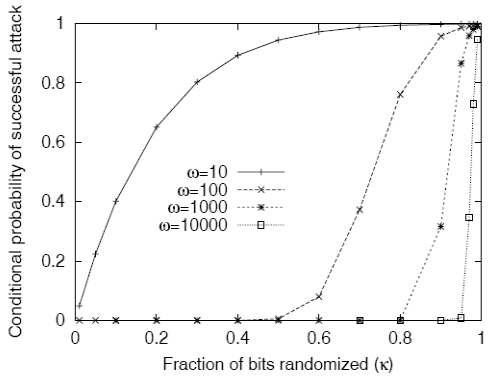
Dalam kenyataannya, sangat sulit bagi Mallory untuk menebak nilai eksak dari ξ yang digunakan oleh Alice. Jika ia menaksir terlalu tinggi, ia akan menyebabkan kesalahan yang lebih besar; jika ia menaksir terlalu rendah, peluang keberhasilannya menjadi berkurang. Oleh sebab itu, Alice dapat secara efektif menggunakan parameter ξ untuk menggagalkan usaha Mallory.

Bermacam-macam ν – Alice memiliki tambahan fleksibilitas untuk memberi *mark* pada salah satu dari ν atribut. Sayangnya Mallory harus mengubah nilai dari seluruh ν atribut dalam sebuah *tuple*. Analisis untuk memberi *mark* pada ν atribut identik dengan analisis untuk memberi *mark* pada ν *least significant bit*. Dengan demikian, Alice harus menggunakan seluruh atribut yang dapat diberinya *mark* tanpa mempengaruhi kualitas dari data secara signifikan.

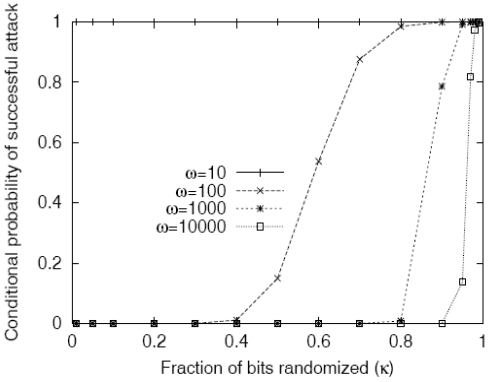
Berikut ini adalah beberapa gambar peluang bersyarat dari keberhasilan berbagai jenis *bit-flipping attack*:



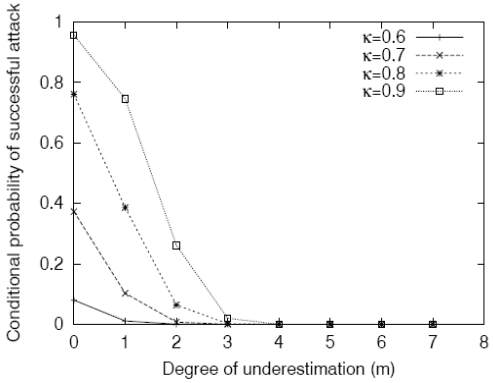
Gambar 10: Peluang Bersyarat Dari Sebuah Keberhasilan *Deterministic Bit-Flipping Attack* (α = 0.01)



Gambar 11: Peluang Bersyarat Dari Sebuah Keberhasilan *Randomized Bit-Flipping Attack* ($\alpha = 0.01$)



Gambar 12: Peluang Bersyarat Dari Sebuah Keberhasilan *Randomized Bit-Flipping Attack* ($\alpha = 0.0001$)



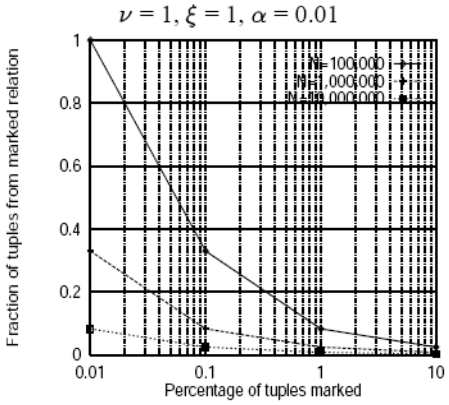
Gambar 13: Peluang Bersyarat Dari Sebuah Keberhasilan *Randomized Bit-Flipping Attack* Ketika ξ Ditaksir Terlalu Rendah ($\alpha = 0.01$, $\omega = 100$, $\xi = 8$)

4.4.2 *Mix-and-Match Attack*

Dalam serangan *Mix-and-Match*, Mallory mengambil κ bagian *tuple* dari relasi R Alice dan mencampurkan mereka dengan *tuple* dari sumber-sumber lainnya untuk menciptakan relasi S miliknya yang memiliki ukuran yang sama dengan R. Kita memberikan sebuah kasus analisis yang sederhana untuk serangan ini. Untuk Alice dapat mendeteksi *watermark* miliknya dalam S, dibutuhkan:

$$\kappa \frac{\eta}{\gamma} + \frac{1}{2}(1-\kappa)\frac{\eta}{\gamma} \geq r \qquad (6)$$

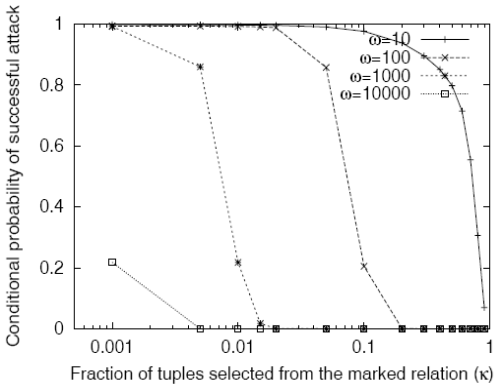
Terminologi kedua pada bagian sebelah kiri dari pertidaksamaan di atas muncul karena algoritma pendeteksian *watermark*, ketika diterapkan kepada sebuah relasi yang tidak diberi *mark*, dapat diharapkan untuk menemukan bit yang bersesuaian dalam setengah dari *tuple*. Nilai dari r bergantung kepada η, γ, dan α.



Gambar 14: Fraksi *Tuple* Minimum Dari Relasi Yang Diberi *Watermark*, Yang Dibutuhkan Untuk Pendeteksian

Gambar 14 memberikan nilai minimum dari κ di mana Alice tetap dapat mendeteksi *watermark* untuk berbagai nilai dari η dan γ (diekspresikan sebagai persentase dari *tuple* yang diberi *mark*). Kita telah mengambil $\alpha = 0.01$ dan $\nu = \xi = 1$ dan mengekstrak nilai r dari Gambar 4. Kita melihat bahwa jika Alice telah memberi *mark* 10% dari *tuple*, ia dapat mendeteksi *watermark* miliknya bahkan ketika Mallory membajak kurang dari 2.5% *tuple* dari 100000 *tuple* relasi miliknya. Fraksi dari *tuple* yang dapat

dibajak oleh Mallory dan menghindari deteksi, menurun secara cepat ketika ukuran relasi milik Alice meningkat.



Gambar 15: Peluang Bersyarat Dari Sebuah Keberhasilan *Mix-and-Match Attack* (α = 0.01)

4.4.3 *Additive Attack*

Dalam *additive attack*, Mallory menyisipkan *watermark* miliknya ke dalam data milik Alice. Klaim kepemilikan orisinil dapat diselesaikan dengan cara mencari bagian-bagian yang saling bertindihan (*overlapping*) dari kedua *watermark* di mana di dalamnya nilai bit dari *mark* mengalami konflik dan menentukan pemilik *mark* mana yang menang. Pemenang pasti sudah menimpa bit dari yang kalah dan oleh sebab itu pasti sudah menyisipkan *watermark*-nya kemudian. Bergantung kepada level signifikan yang dipilih untuk tes, sangat memungkinkan untuk tidak mencapai sebuah keputusan jika hanya terdapat sedikit *mark* yang bertabrakan. Untuk itu, memiliki lebih banyak *tuple* yang diberi *mark* (nilai γ yang lebih kecil) meningkatkan tabrakan dan peluang untuk mencapai sebuah keputusan.

4.4.4 *Invertibility Attack*

Watermark tiruan (palsu) digunakan untuk mengklaim kepemilikan dalam sebuah *invertibility attack* [5]. Serangan jenis ini mentranslasikan ke dalam kemungkinan Mallory dapat menemukan sebuah kunci yang menghasilkan sebuah *watermark* yang memuaskan untuk beberapa nilai α. Kunci yang ditemukan oleh Mallory tidak perlu sesuai dengan kunci yang digunakan oleh Alice dalam menyisipkan *watermark* miliknya. Untuk nilai α yang tinggi, Mallory dapat menemukan kunci tersebut secara kebetulan melalui percobaan yang berulang-ulang dari nilai-nilai kunci yang berbeda. Serangan ini dapat dihalangi dengan menggunakan nilai α yang rendah (misalnya 1.0e-10).

4.5 *Trade-Off* Perancangan

Teknik *watermarking* basis data relasional memiliki empat parameter penting yang dapat diselaraskan, yaitu:

i) α, level signifikan tes,
ii) γ, parameter *gap* yang menentukan fraksi dari *tuple* yang diberi *mark*,
iii) ν, jumlah atribut dalam relasi yang tersedia untuk pemberian *mark*, dan
iv) ξ, jumlah *least significant bit* yang tersedia untuk pemberian *mark*.

Berdasarkan analisis yang dijelaskan pada bagian ini, telah diringkas dalam Gambar 16 *trade-off* penting ketika memilih nilai untuk parameter-parameter ini.

$\downarrow \alpha$	$\downarrow$ false hits	$\uparrow$ missed watermarks
$\downarrow \gamma$	$\uparrow$ robustness	$\uparrow$ data errors
$\uparrow \nu$	$\uparrow$ robustness	
$\uparrow \xi$	$\uparrow$ robustness	$\uparrow$ data errors

Gambar 16: *Trade-Off* Perancangan

5. Kesimpulan

Berdasarkan pembahasan di atas mengenai *watermarking* terhadap basis data relasional, dapat ditarik kesimpulan sebagai berikut:

1. Identifikasi dari manajemen hak terhadap data relasional melalui *watermarking* merupakan suatu masalah yang sangat penting dan menantang untuk penelitian (riset) dalam bidang basis data.
2. Sebuah *watermark* dapat diterapkan terhadap relasi dalam basis data yang memiliki atribut-atribut sedemikian sehingga sedikit perubahan pada nilai data tidak akan mempengaruhi aplikasinya
3. Teknik *watermarking* yang digunakan pada basis data relasional merupakan suatu teknik yang cukup tangguh dan kuat terhadap berbagai bentuk serangan dan manipulasi data.

4. Algoritma *watermarking* basis data relasional yang dibahas dalam makalah ini juga memiliki performansi yang cukup tinggi sehingga dapat diaplikasikan dalam dunia nyata

DAFTAR PUSTAKA

[1] R. Agrawal dan J. Kiernan. Watermarking Relational Databases. In *Proc. of the 28th VLDB Conference*, Hong Kong, China, 2002.

[2] S. Benjamin, B. Schwartz, dan R. Cole. Accuracy of ACARS Wind and Temperature Observations Determined by Collocation. *Weather and Forecasting*, 14:1032–1038, 1999.

[3] L. Boney, A. H. Tewfik, dan K. N. Hamdy. Digital Watermarks for Audio Signals. In *International Conference on Multimedia Computing and Systems*, Hiroshima, Japan, June 1996.

[4] I. J. Cox dan M. L. Miller. A Review of Watermarking and The Importance of Perceptual Modeling. In *Proc. of Electronic Imaging*, February 1997.

[5] S. Craver, N. Memon, B.-L. Yeo, dan M. M. Yeung. Resolving Rightful Ownerships with Invisible Watermarking Techniques: Limitations, Attacks, and Implications. *IEEE Journal of Selected Areas in Communications*, 16(4):573–586, 1998.

[6] S. Czerwinski. Digital Music Distribution and Audio Watermarking. `http://citeseer.nj.nec.com`. Tanggal akses: 5 Oktober 2006 pukul 20:00.

[7] J.-L. Dugelay dan S. Roche. A Survey of Current Watermarking Techniques. In S. Katzenbeisser dan F. A. Petitcolas, editors, *Information Hiding Techniques for Steganography and Digital Watermarking*, chapter 6, pages 121–148. Artech House, 2000.

[8] F. Hartung dan B. Girod. Watermarking of Uncompressed and Compressed Video. *Signal Processing*, 66(3):283–301, 1998.

[9] N. F. Johnson, Z. Duric, dan S. Jajodia. *Information Hiding: Steganography and Watermarking – Attacks and Countermeasures*. Kluwer Academic Publishers, 2000.

[10] Joseph J. K. ´O Ruanaidh, W. J. Dowling, dan F. M. Boland. Watermarking Digital Images for Copyright Protection. *IEEE Proceedings on Vision, Signal and Image Processing*, 143(4):250–256, 1996.

[11] S. Katzenbeisser dan F. A. Petitcolas, editors. *Information Hiding Techniques for Steganography and Digital Watermarking*. Artech House, 2000.

[12] A. Kerckhoffs. La Cryptographie Militaire. *Journal des Sciences Militaires*, 9:5–38, January 1883.

[13] N. Maxemchuk. Electronic Document Distribution. Technical Journal, AT&T Labs, September 1994.

[14] R. Munir. Bahan Kuliah IF5054 Kriptografi. Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung, 2006.

[15] B. Schneier. *Applied Cryptography*. John Wiley, second edition, 1996.