Penerapan *Watermarking* pada Basis Data Relasional dan Dokumen XML

Andri Tanoto – NIM: 13503078

Program Studi Teknik Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
E-mail : if13078@students.if.itb.ac.id

Abstrak

Digital Watermarking atau yang lebih dikenal sebagai *watermarking* adalah suatu teknik untuk menyisipkan informasi tersembunyi ke dalam data *multimedia* (seperti citra, audio, video, dan text) dengan tujuan melindungi data tersebut dari duplikasi oleh pihak yang tidak berwenang. Penyisipan *watermark* pada data tidak akan merusak data digital yang akan dilindungi tersebut. *Watermarking* berguna untuk membuktikan kepemilikan, *copyright protection*, *authentication*, *fingerprinting*, *tamper profing*, dan *distribution tracing*.

Data digital merupakan aset berharga yang dimiliki oleh individu, perusahaan, ataupun organisasi. Beberapa data digital tersebut antara lain dapat berupa perangkat lunak, citra, audio, video, dan teks. Basis data relasional dan dokumen XML merupakan data yang akan menjadi kajian pada makalah ini. Penerapan prinsip *watermarking* pada basis data relasional dan dokumen XML merupakan suatu teknik baru dalam bidang *watermarking*. Kajian dan riset dalam bidang ini menjadi suatu topik yang menarik dan penting dalam dunia *watermarking* saat ini.

Penerapan prinsip w*atermarking* pada basis data relasional dan dokumen XML merupakan suatu metode baru untuk mendeteksi keaslian dan melakukan identifikasi karakterisitik suatu basis data relasional dan dokumen XML tersebut. Karena pada basis data tersimpan data penting milik perusahaan atau organisasi maka perusahaan atau organisasi tersebut tentu saja perlu melindungi basis data miliknya dari ancaman terhadap pencurian data. Prinsip w*atermarking* pada basis data relasional ini dapat diterapkan tanpa mempengaruhi nilai yang terdapat pada basis data tersebut. Teknik ini menggunakan suatu nilai spesifik pada posisi bit tertentu dari atribut suatu relasi. Pola pada posisi bit inilah yang menjadi *watermarking* pada basis data relasional. Sedangkan prinsip w*atermarking* pada dokumen XML dapat dilakukan dengan cara membangun suatu query dengan semantik yang baru untuk melakukan identifikasi kemungkinan penyisipan *watermarking* pada dokumen XML.

***Kata kunci**: digital watermarking,* basis data relasional, dokumen XML.

1. Pendahuluan

Perangkat lunak, citra, audio, video, dan teks merupakan aset yang sangat berharga bagi pemiliknya. Aset-aset tersebut tentu saja harus dilindungi dari pembajakan dan duplikasi tanpa ijin oleh pihak yang tidak bertanggung jawab. Oleh karena itu, diperlukan suatu metode atau teknik untuk menandai kepemilikan terhadap aset-aset tersebut. Salah satu teknik tersebut adalah dengan menggunakan *watermark* yang dapat menandai kepemilikan suatu aset tanpa merusak data yang terkandung di dalamnya.

Salah satu aset berharga yang dimiliki oleh perusahaan atau organisasi adalah basis data karena pada basis data tersimpan data penting milik perusahaan atau organisasi tersebut. Perusahaan tentu saja perlu melindungi basis data miliknya dari ancaman terhadap pencurian data. Penerapan prinsip *watermarking* pada basis data relasional merupakan suatu teknik baru dalam bidang keamanan basis data. Kajian dan riset dalam bidang ini menjadi suatu topik yang penting dalam dunia basis data saat ini.

Selain pada basis data relasional, prinsip *watermarking* ini dapat pula diterapkan pada dokumen XML. Dokumen XML merupakan format data yang telah menjadi standar baru dalam proses pertukaran data melalui internet. Penerapan prinsip watermarking pada dokumen XML bertujuan untuk mencegah adanya distribusi dan duplikasi dari dokumen XML secara ilegal.

Pada makalah ini akan dibahas mengenai *watermarking*, penerapan prinsip *watermarking* pada basis data relasional, dan penerapan prinsip *watermarking* pada dokumen XML.

2. *Watermarking*

Digital watermarking atau *watermarking* adalah teknik untuk menyisipkan informasi tertentu ke dalam data digital yang disebut *watermark* (tanda air). *Watermark*

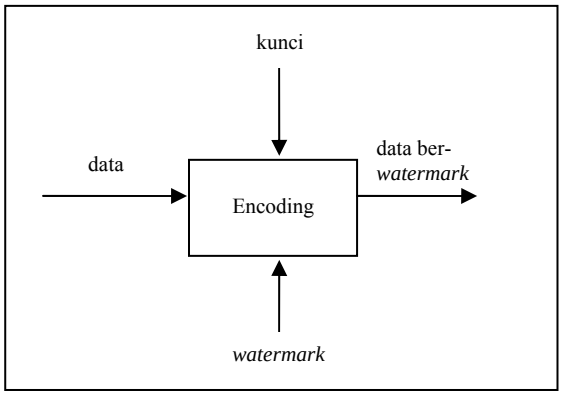
dapat berupa teks seperti informasi *copyright*, gambar berupa logo, data audio, atau rangkaian bit yang tidak bermakna. Penyisipan *watermark* dilakukan sedemikian sehingga *watermark* tidak merusak data digital yang dilindungi. Selain itu *watermark* yang telah disisipkan tidak dapat dipersepsi oleh indra manusia, tetapi dapat dideteksi oleh komputer dengan menggunakan kunci yang benar. W*atermark* yang telah disisipkan tidak dapat dihapus dari dalam data digital sehingga jika data digital tersebut disebar dan diduplikasi maka otomatis *watermark* di dalamnya akan ikut terbawa. *Watermark* di dalam data digital harus dapat diekstraksi kembali. *Watermarking* berguna untuk membuktikan kepemilikan, *copyright protection*, *authentication*, *fingerprinting*, *tamper profing*, dan *distribution tracing*.

Sejarah *watermarking* sudah dimulai sejak 700 tahun yang lalu. Pada akhir abad 13, pabrik kertas di Fabriano, Italia, membuat kertas yang diberi *watermark* atau tanda air dengan cara menekan bentuk cetakan gambar atau tulisan pada kertas yang baru setengah jadi. Ketika kertas dikeringkan, terbentuklah suatu kertas yang ber-*watermark.* Kertasini biasanya digunakan oleh seniman dan sastrawan untuk menulis karya mereka. Kertas yang sudah dibubuhi *watermark* tersebut sekaligus dijadikan identifikasi bahwa karya seni di atasnya adalah milik mereka.

Ide watermarking pada data digital dikembangkan di Jepang tahun 1990 dan di Swiss tahun 1993. *Digital watermarking* semakin berkembang seiring dengan semakin meluasnya penggunaan internet, objek digital seperti video, citra, dan suara yang dapat dengan mudah digandakan dan disebarluaskan.

Saat ini kebanyakan data dan informasi disajikan dalam bentuk format digital, baik berupa teks, citra, audio, maupun video. Produk digital lainnya, mempunyai beberapa karakteristik, antara lain penggandaan terhadap data digital juga mudah dilakukan dan hasilnya tepat sama dengan aslinya, mudah didistribusikan melalui *magnetic disk* maupun internet, dan perubahan sedikit pada citra tidak mudah dipersepsi oleh indera penglihatan.

Secara umum *watermarking* terdiri dari dua tahapan, yaitu tahap pertama, penyisipan *watermark* (*watermark embedding*) dan tahap kedua, ekstraksi atau pendeteksian *watermark* (*watermark detection*). Penyisipan *watermark* dapat dipandang sebagai superposisi data *watermark* pada data dengan cara tertentu sehingga superposisi tersebut tidak mempengaruhi persepsi virtual terhadap data. Berikut ini adalah gambar proses penyisipan *watermark* pada data.



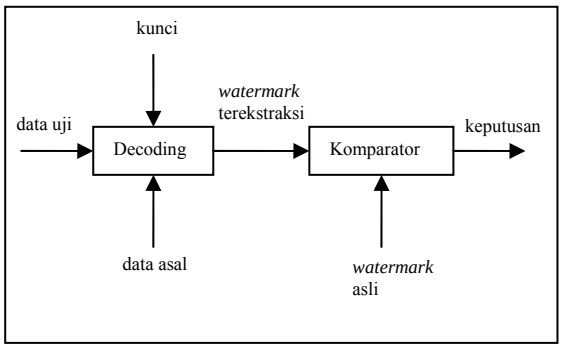
Gambar 1 Proses *Watermarking*

Seperti terlihat pada gambar di atas, sebuah *encoder* yang melakukan proses *watermarking* menerima masukan berupa data, *watermark*, dan kunci. Data sebelum dan sesudah proses *watermarking* hampir mirip secara statistik atau secara visual mempunyai persepsi yang sama.

Watermark harus dapat diekstraksi atau dideteksi kembali bergantung pada *nature* algoritma *watermarking*. Pada beberapa algoritma *watermarking, watermark* dapat diekstraksi dalam bentuk yang eksak, sedangkan pada sebagian algoritma yang lain, hanya dapat dideteksi apakah *watermark* terdapat di dalam citra sehingga prosedurnya dinamakan pendeteksian *watermark* (*watermark detection*).

Prosedur untuk melakukan ekstraksi dan verifikasi *watermark* dapat dilakukan dengan cara *decoder* menerima masukan berupa data uji (bisa berupa data yang ber- *watermark* atau tanpa *watermark,* bahkan data yang sudah mengalami distorsi), kunci k, dan menghasilkan *watermark* yang sudah terekstraksi. Decoder dapat mengikutsertakan data asal yang belum diberi *watermark* (*non-blind watermark*) atau tidak menggunakan data sama sekali (*blind watermark*) karena beberapa skema *watermarking* memang belum menggunakan data asal dalam proses ini untuk meningkatkan hasil ekstraksi yang lebih baik.

Selanjutnya, *watermark* yang sudah terekstraksi dibandingkan dengan *watermark* asli dengan suatu fungsi komparator (umumnya sebuah *correlator*) untuk menghasilkan keputusan berupa keluaran biner, 1 untuk menyatakan cocok dan 0 untuk menyatakan tidak cocok. Berikut ini adalah gambar yang memperlihatkan prosedur untuk melakukan ekstraksi dan verifikasi *watermark.*



Gambar 2 Proses Ekstraksi dan Verifikasi

Sebuah teknik *watermarking* yang handal harus memenuhi persyaratan berikut:

1. *Imperceptibility*: keberadaan *watermark* tidak dapat dipersepsi oleh indra visual. Hal ini bertujuan untuk menghindari gangguan pengamatan visual.
2. *Key uniqueness*: kunci yang berbeda seharusnya menghasilkan *watermark* yang berbeda. Ini berarti penggunaan kunci yang salah dapat menyebabkan hasil ekstraksi atau deteksi *watermark* yang salah pula.
3. *Noninvertibility*: secara komputasi sangat sukar menemukan *watermark* jika diketahui hanya data ber- *watermark* saja.
4. *Robustness*: *watermark* seharusnya tetap kokoh terhadap serangan yang dilakukan pada data ber-*watermark*. Ini berarti manipulasi yang dilakukan terhadap data ber- *watermark* tidak merusak *watermark* sehingga *watermark* masih dapat dideteksi.

Sebagian besar penelitian, publikasi, dan aplikasi di bidang *watermarking* ditujukan untuk citra digital, akan tetapi *watermarking* juga dapat diterapkan pada jenis data multimedia lain, seperti audio, video, dan teks. *Watermarking* pada video digital memerlukan teknik tertentu sehingga peralihan gambar dari satu *frame* ke *frame* lainnya harus tetap baik dan tidak terlihat dimodifikasi. *Watermarking* pada video digital memerlukan proses penyisipan yang lebih banyak, hal ini disebabkan ukuran file video digital yang relatif lebih besar daripada citra. Sedangkan untuk *watermarking* pada data audio, perlu ketelitian pada perancangan algoritma *watermarking* karena suara lebih sensitif daripada citra. Hal ini berarti suara digital lebih mudah rusak jika ditambahkan *watermark*. *Watermarking* pada dokumen teks menggunakan metode yang berbeda yaitu dengan cara menyisipkan spasi antara dua buah kata atau antara dua buah kalimat dalam suatu dokumen teks.

Selain pada citra, audio, video, dan teks, *watermarking* dapat juga diterapkan pada basis data relasional serta dokumen XML. Penerapan prinsip *watermarking* pada basis data relasional dan dokumen XML ini merupakan metode *watermarking* yang banyak dikaji akhir-akhir ini. Hal ini disebabkan adanya kebutuhan untuk melindungi hak kepemilikan terhadap basis data relasional yang merupakan aset penting suatu organisasi atau perusahaan dan juga hak kepemilikan terhadap dokumen XML yang telah menjadi format standar dalam pertukaran data melalui internet.

Beberapa jenis *digital watermarking* antara lain:

1. *Visible Watermark*
   Watermarking jenis ini merupakan watermarking yang memiliki tujuan untuk meningkatkan perlindungan akan hak cipta. Selain itu, watermarking jenis ini juga digunakan untuk mengidentifikasi kepemilikan dari sebuah karya (originalitas).
2. *Invisible Robust Watermark*
   Watermarking jenis ini untuk mendeteksi ketidaktepatan dari sebuah citra. Selain itu, jenis ini biasanya digunakan untuk menerangkan kepemilikan.
3. *Invisible Fragile Watermark*
   Watermarking jenis ini digunakan oleh perangkat kamera yang cukup handal. Dan proses *watermarking* dilakukan ketika pengambilan gambar.

3. *Watermarking* pada Basis Data Relasional

Penerapan prinsip *watermarking* pada basis data relasional adalah dengan cara menukarkan beberapa posisi bit dari atribut pada *record* yang mengandung nilai tertentu. *Record*, atribut, dan nilai bit yang spesifik tersebut ditentukan oleh suatu algoritma yang mengendalikan kunci privat yang hanya diketahui oleh pemilik basis data. Pola bit yang digunakan inilah yang menjadi *watermark* pada suatu basis data relasional. Hanya orang yang memiliki kunci privat sajalah yang dapat mendeteksi *watermark* pada basis data relasional tersebut dengan kemungkinan yang tinggi. Pendeteksian *watermark* tersebut tidak memerlukan akses terhadap data asli ataupun *watermark*-nya. Suatu *watermark* dapat dideteksi hanya dengan menggunakan bagian dari relasi yang mengandung *watermark*.

Teknik *watermarking* pada basis data relasional memiliki beberapa perbedaan dengan teknik *watermarking* pada data multimedia, perbedaan-perbedaan tersebut antara lain:

1. Data multimedia memiliki jumlah bit yang sangat banyak dan terkadang terdapat redundansi. Oleh karena itu, *watermark* harus disisipkan dengan meliputi banyak bit tersebut. Sementara pada basis data relasional yang terdiri dari *record* yang terpisah-pisah, *watermark* perlu disebar pada *record-record* tersebut.
2. Data multimedia relatif jarang berubah sedangkan pada basis data relasional dapat terjadi perubahan pada record-record karena adanya operasi *insert*, *update*, maupun *delete*.
3. Perubahan sedikit saja pada data multimedia akan menyebabkan perubahan pada persepsi

data multimedia tersebut, sedangkan pada basis data perubahan dapat dilakukan hanya dengan menghapus atau melakukan edit pada *record* tertentu tanpa diketahui oleh pemilik basis data tersebut.

Karena adanya perbedaan tersebut maka teknik *watermarking* yang digunakan pada data multimedia tidak dapat digunakan pada basis data relasional. Untuk menerapkan teknik ini maka suatu relasi pada basis data dianggap ekivalen dengan citra sedangkan setiap atributnya dianggap ekivalen dengan pixel. Tetapi analogi seperti ini tidak dapat langsung diterapkan secara sama persis karena pada kenyataannya relasi dan citra memiliki properti yang sangat berbeda jauh. Oleh karena itu, penerapan *watermarking* pada basis data relasional merupakan suatu hal yang menantang dan telah menarik banyak komunitas riset untuk mengkajinya secara serius. Prinsip *watermarking* pada basis data relasional pada dasarnya sama dengan teknik *watermarking* pada data biasa, hanya saja perlu dilakukan pengembangan dan inovasi lebih lanjut untuk penerapannya.

Ide dasar dari penerapan prinsip *watermarking* pada basis data relasional adalah dengan memastikan posisi bit pada atribut dari *record* tertentu pada basis data mengandung nilai yang spesifik.

3.1 Pemodelan *Watermarking*

Andaikan seorang bernama Alice mempunyai relasi R yang mengandung *n record*, dan Alice telah menandai sejumlah *w* record maka properti berikut harus dimiliki oleh pemodelan *watermarking*:

1. *Detectability*
   Alice harus bisa mendeteksi *watermark* miliknya dengan memeriksa *w* record dari suatu basis data. Jika pola bit (*watermark*) ternyata terdapat pada semua *w* record tersebut maka basis data tersebut adalah miliknya.
2. *Robustness*
   Watermark sebaiknya handal (*robust*) terhadap serangan untuk menghapus *watermark* tersebut.
3. *Incremental Updatability*
   Jika Alice memiliki relasi R yang sudah diberi *watermark* maka ia harus dapat melakukan *update* R tanpa menghilangkan *watermark*-nya. Selain itu, jika Alice melakukan penambahan atau penghapusan record maka *watermark* tersebut juga harus dapat di-*update*.
4. *Imperceptibility*
   Modifikasi pada basis data yang disebabkan oleh *watermarking* tidak boleh mempengaruhi kegunaan dari basis data tersebut. Selain itu, penggunaan perhitungan statistik seperti mean dan variansi dari atribut numerik tidak boleh terpengaruh secara signifikan.
5. *Blind System*
   Pendeteksian *watermark* sebaiknya tidak memerlukan informasi mengenai basis data asli maupun *watermark* itu sendiri. Properti ini sangat penting karena memungkinkan *watermark* dapat dideteksi pada salinan dari relasi suatu basis data.
6. *Key-Based System*
   Watermarking memiliki asumsi bahwa metode yang digunakan untuk menyisipkan *watermark* bersifat publik. Pertahanan terhadap *watermark* terletak pada pemilihan kunci privat.

3.2 *Benign Updates* dan *Malicious Attacks*

Karena suatu relasi pada basis data dapat di-update maka *watermark* pada relasi dapat dihilangkan dengan cara melakukan *benign updates* dan juga *malicious attacks*

.

Benign Updates

Andaikan seseorang mencuri data milik Alice tanpa menyadari adanya *watermark*. Selanjutnya orang tersebut dapat melakukan *update* terhadap data milik Alice dan menggunakannya. Pada kasus seperti ini, w*atermarking* harus tetap ada pada data yang dicuri tersebut meskipun data tersebut telah di-*update*.

Malicious Attacks

Andaikan seseorang yang mengetahui bahwa data yang dicurinya mengandung *watermark* dan ia mencoba untuk menghapus *watermark* tersebut atau mengakui kepemilikan terhadap data tersebut. Pada kasus seperti ini, w*atermarking* harus dapat melindungi data milik Alice tersebut dari berbagai malicious attack, antara lain:

1. *Bit Attack*
   Serangan ini merupakan cara paling sederhana untuk menghilangkan *watermark* dengan cara melakukan *update* terhadap sejumlah bit. Jika seseorang mampu untuk mengubah setiap bit data maka dengan mudah ia juga dapat meghilangkan *watermark.* Akan tetapi, orang tesebut juga membuat data menjadi tidak dapat digunakan lagi.
2. *Randomization Attack*
   Serangan ini dilakukan dengan cara mengisi nilai random pada sejumlah posisi bit tertentu. Serangan *zero out* dilakukan dengan mengisi sejumlah nilai bit dengan nilai nol. Serangan *bit flipping* dilakukan dengan mempertukarkan nilai bit 1 menjadi 0 dan sebaliknya.
3. *Rounding Attack*
   Serangan ini dilakukan dengan cara mencoba untuk menghilangkan atribut numerik dengan membulatkan semua nilai pada atribut tersebut. Serangan ini tidak lebih baik daripada serangan *bit attack*. Serangan ini dilakukan dengan menebak secara benar berapa jumlah bit yang terkandung pada *watermarking*. Jika tebakan jumlahnya terlalu rendah maka serangan tidak

akan berhasil. Jika tebakan jumlahnya terlalu besar maka serangan tersebut akan menurunkan kualitas data. Bahkan jika tebakannya benar pun data tersebut akan tetap berkurang kualitasnya.

4. *Subset Attack*
   Serangan ini dilakukan dengan cara mengambil sebagian dari *record* yang mengandung atribut ber-*watermark* dan berharap agar *record* yang terambil tersebut hilang.
5. *Mix and Match Attack*
   Serangan ini dilakukan dengan cara membuat relasi yang merupakan gabungan dari *record* yang saling *disjoint* dari beberapa relasi yang memiliki informasi yang sama.
6. *Additive Attack*
   Serangan ini dilakukan dengan cara menambahkan *watermark* secara manual sehingga data dianggap menjadi miliknya.
7. *Invertibility Attack*
   Serangan ini dilakukan dengan cara mencuri kepemilikan suatu data jika pada data tersebut ditemukan *watermark* yang palsu. Serangan ini dapat dilakukan secara *random*.

3.3 Algoritma

Pada bagian ini akan dijelaskan mengenai algoritma yang digunakan untuk penerapan *watermarking* pada basis data relasional. Teknik yang digunakan adalah dengan menandai atibut numerik dan mengasumsikan bahwa atribut yang ditandai tersebut tetap dapat diterima (*acceptable*) dan penandaan tersebut dianggap tidak berpengaruh (*non-obvious*). Semua atribut numerik dari relasi yang ada harus ditandai. Pemilik data bertanggung jawab untuk menentukan atribut mana saja yang cocok untuk ditandai.

Andaikan terdapat relasi basis data R dengan skema sebagai R (P,A_0,...,A_{v-1}), di mana P adalah atribut kunci (*primary key*). Dengan asumsi, semua v dari atribut A_0,...,A_{v-1} adalah kandidat untuk ditandai. Algoritma ini dibagi menjadi dua tahap yaitu penyisipan *watermark* dan pendeteksian *watermark*.

3.3.1 Penyisipan *Watermark*

Adanya penggunaan *Message Authenticated Code* (MAC) maka hanya pemilik basis data relasional yang mengetahui kunci privat K yang dapat dengan mudah menentukan *record* yang sudah ditandai. *Message Authenticated Code* (MAC) adalah fungsi hash satu arah yang bergantung pada sebuah kunci. Pada record yang terpilih, tentukan atribut mana saja yang akan ditandai di antara *v* kandidat atribut. Pada atribut yang terpilih, tentukan posisi bit di antara ξ *least significant bit* yang ditandai. Hasil dari pengujian bergantung pada kunci privat pemilik basis data relasional. Untuk menghapus sebuah watermark, seorang yang hendak melakukan serangan terhadap *watermark* harus menebak tidak hanya *record*-nya saja, tetapi sekaligus menebak atribut yang sudah ditandai dan juga posisi bit-nya. Kemudian, dilakukan pengisian nilai terhadap bit 0 atau 1 tergantung pada nilai hash yang didapat. Hasilnya dapat akan menyebabkan nilai atribut akan tidak berubah atau terjadi *increment* atau *decrement* dari bit tersebut. Oleh karena itu, terdapat nilai bit yang tidak berubah, mengalami *increment*, atau mengalami *decrement*.

Basis data relasional memungkinkan untuk mengisi nilai *null*. Jika nilai null ditemukan pada saat proses penandaan *record* maka nilai *null* tersebut akan dibiarkan tidak berubah.

Algoritma Penyisipan *Watermark*

```
// The private key K is known only to the owner of the database.
// The parameters γ, ν, and ξ are also private to the owner.

1) foreach tuple r ∈ R do
2)     if (F(r.P) mod γ equals 0) then          // mark this tuple
3)         attribute_index i = F(r.P) mod ν     // mark attribute A_i
4)         bit_index j = F(r.P) mod ξ           // mark j^th bit
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
```

Tabel Notasi Algoritma

η	Number of tuples in the relation
ν	Number of attributes in the relation available for marking
ξ	Number of least significant bits available for marking in an attribute
$1/\gamma$	Fraction of tuples marked
ω	Number of tuples marked
α	Significance level of the test for detecting a watermark
τ	Minimum number of correctly marked tuples needed for detection

3.3.2 Pendeteksian *Watermark*

Andaikan seorang bernama Alice mencurigai adanya relasi basis data S milik Bob yang merupakan hasil penjiplakan dari relasi basis data R milik Alice. Himpunan *record* dan atribut dari relasi S dapat merupakan himpunan bagian dari relasi R. Asumsi bahwa Bob tidak menghilangkan atribut kunci yang mengandung informasi berharga dan mengubah atribut kunci berarti akan menyebabkan basis data menjadi kurang berguna dari sudut pandang *user*.

Algoritma pendeteksian watermark pada dasarnya menggunakan teori kemungkinan (probabilitas). Tentukan apakah *record* s yang dipertimbangkan harus ditandai pada waktu penyisipan *watermark*. Kemudian, tentukan atribut dan posisi bit yang ditandai. Setelah itu, bandingkan antara nilai bit saat ini dengan nilai yang seharusnya diisi ketika penyisipan *watermark.*

Selanjutnya, dapat diketahui berapa jumlah *record* yang telah diuji dan berapa dari *record* tersebut yang mengandung nilai bit yang diharapkan. Dengan teori kemungkinan (probabilitas), hanya sejumlah minimum record tertentu yang harus mengandung bit bertanda yang cocok.

Andaikan terdapat semua kandidat atribut $A_0,...,A_{v-1}$ pada relasi S. Jika Alice menemukan sebuah record s yang harus ditandai pada atribut A_i dan Bob telah menghilangkan A_i maka Alice dapat menghilangkan atribut A_i tersebut. Begitu juga, jika sebuah atribut A_i harus ditandai tetapi nilai atribut tersebut adalah *null* maka atribut tersebut pun dapat dihilangkan.

Algoritma

```
// K, γ, ν, and ξ have the same values used for watermark insertion.
// α is the test significance level that the detector preselects.

1)  totalcount = matchcount = 0

2)  foreach tuple s ∈ S do
3)      if (F(s.P) mod γ equals 0) then        // this tuple was marked
4)          attribute_index i = F(s.P) mod ν   // attribute A_i was marked
5)          bit_index j = F(s.P) mod ξ         // j^th bit was marked
6)          totalcount = totalcount + 1
7)          matchcount = matchcount + match(s.P, s.A_i, j)

8)  τ = threshold(totalcount, α)               // see Section 4.2
9)  if (matchcount ≥ τ) 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
```

3.4 Relasi Tanpa *Primary Key*

Teknik *watermarking* yang dijelaskan pada bagian sebelumnya bekerja pada relasi yang memiliki *primary key*. *Primary key* akan muncul secara otomatis pada suatu basis data relasional.
Pada bagian ini akan dibahas mengenai penerapan *watermarking* pada relasi yang tidak memiliki *primary key.*

Asumsi terdapat relasi R yang terdiri dari satu atribut numerik A. Partisi bit pada atribut A menjadi dua kelompok, yaitu X bit untuk merepresentasikan pengganti *primary key* dan bit lainnya digunakan untuk ditandai. Teknik ini hanya akan berhasil jika pada X bit tidak terdapat duplikasi.

Jika relasi memiliki lebih dari satu atribut maka salah satu dari atribut tersebut akan dianggap sebagai *primary key* dan atribut lainnya yang akan ditandai.

4. *Watermarking* pada Dokumen XML

Saat ini banyak sekali data yang dipublikasikan dalam format XML (*eXtensible Markup Language*) maka dibutuhkan suatu perlindungan terhadap kepemilikan dari dokumen XML tersebut. Salah satu teknik yang dapat digunakan untuk menandai kepemilikan dokumen XML adalah teknik *watermarking.* Skema *watermarking* yang digunakan harus tetap memperhatikan kegunaan data dan beberapa semantik penting seperti atribut kunci dan *functional dependencies*. Salah satu skema yang akan dibahas untuk melakukan *watermarking* terhadap dokumen XML adalah dengan cara menghasilkan *query* dari semantik yang penting untuk melakukan identifikasi kemungkinan penyisipan *watermark* pada dokumen XML dan mengintegrasikan teknik penulisan ulang suatu *quey* untuk mengatasi ancaman perubahan dan reorganisasi data.

4.1 Pendahuluan

XML adalah suatu format data standar untuk merepresentasikan informasi dan saling bertukar data melalui internet. Dengan semakin banyaknya jumlah data komersial yang dipertukarkan dan dipublikasikan maka penyebaran dan duplikasi secara ilegal menjadi hal yang penting untuk diperhatikan.

Digital watermarking adalah salah satu cara yang banyak digunakan untuk melindungi informasi digital dari serangan terhadap kepemilikan. Dengan adanya penandaan pada data maka seseorang dapat membuktikan kepemilikannya terhadap data tersebut. Penerapan teknik *watermarking* pada dokumen XML merupakan suatu metode baru yang meliputi melakukan identifikasi elemen data dan struktur untuk *watermarking.* Data XML dibangun dari sejumlah elemen data dan struktur yang menghubungkan elemen data menjadi satu berdasarkan keterhubungannya. Untuk melakukan identifikasi setiap elemen data atau struktur maka harus diperhatikan juga hubungan antara setiap elemen data dan strukturnya.
Selain itu diperlukan juga fleksibilitas terhadap perubahan dan reorganisasi data. Ketika elemen data diidentifikasi pada keterhubungan dan strukturnya, data tersebut harus dapat dikonstruksi kembali untuk mencegah kesalahan identifikasi elemen. Format data XML yang fleksibel ini meyebabkan data XML mudah untuk dikonstruksi ulang tanpa kehilangan informasi yang terkandung di dalamnya.

Contoh dokumen XML:

```
<db>
  <publisher name="mkp">
    <author name="Stonebraker">
      <book>Readings in Database Systems</book>
      <book>XML Query Processing</book>
    </author>
    <author name="Hellerstein">
      <book>Readings in Database Systems</book>
      <book>Relational Data Integration</book>
    </author>
    ...
  </publisher>
  <publisher name="acm">
    ...
  </publisher>
  ...
</db>
```

Jika terdapat redundansi pada data XML maka hal ini dapat menurunkan kualitas *watermark*. Jika data yang redundan tersebut disisipkan *watermark* maka *watermark* tersebut akan mudah dihapus dengan cara membuat duplikasi yang identik sama.

Dua jenis teknik untuk melakukan *watermarking* pada dokumen XML yang akan dibahas pada makalah ini adalah teknik WmXML dan *Query-Preserving Watermarking.*

4.2 Teknik WmXML

Teknik WmXML dilakukan dengan cara mengukur kegunaan data (*data usability*) berdasarkan ketepatan hasil query, menggunakan query tersebut untuk melakukan identifikasi elemen, dan melakukan konstruksi query dengan semantik khusus.

4.2.1 *Data Usability*

Teknik *watermarking* yang efektif harus memenuhi dua kebutuhan utama, yaitu pertama, *watermark* harus dapat disisipkan tanpa mengurangi kegunaan dari data dan kedua, *watermark* yang disisipkan harus handal (*robust*) sehingga akan sulit untuk menghilangkan *watermark* tanpa mengurangi kegunaan data. Kegunaan data (*data usability*) adalah suatu ukuran yang sangat penting untuk menentukan kualitas teknik *watermarking.*

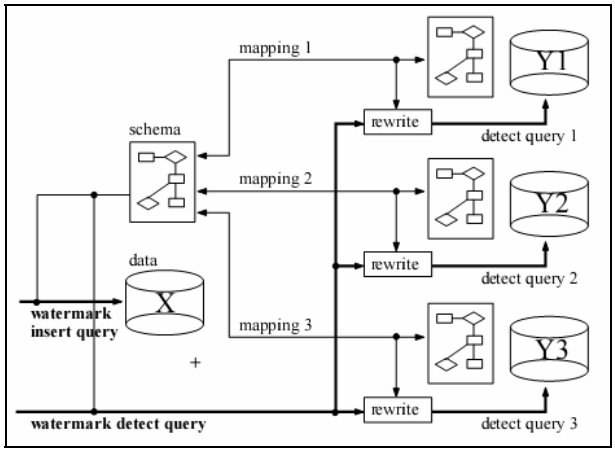
4.2.2 Identifikasi *Query*

Baik data elemen maupun unit struktur pada dokumen XML harus bisa digunakan untuk menyisipkan *watermark*. Karena seseorang dapat melakukan reorganisasi struktur dokumen XML untuk mencegah elemen data dan struktur dapat diidentifikasi dengan benar maka identitas yang dibuat oleh WmXML untuk elemen data dan struktur tersebut perlu dipisahkan dari organisasi fisik suatu data. *Query* merupakan salah satu penanda (*identifier*) data karena *query* dapat digunakan dengan mudah pada organisasi data yang berbeda hanya dengan menulis ulang *query* tersebut sesuai kebutuhan.

Skema watermarking yang digunakan pada teknik WmXML adalah sebagai berikut:

1. Inisialisasi
   Inisialisasi dilakukan dengan cara menentukan skema dan melakukan validasi data XML sesuai dengan skemanya. Kemudian perlu ditentukan sekumpulan *query template* untuk merepresentasikan kegunaan data. Kunci rahasia digunakan untuk memeilih sejumlah elemen data dan struktur untuk menyisipkan bit *watermark*. Selanjutnya, dibuatlah *query* sebagai penanda (*identifier*) dari elemen data atau struktur dan melindungi kumpulan *query* tersebut bersamaan dengan kunci rahasianya.
2. Penyisipan *watermark*
   Pada tahap ini dilakukan eksekusi *query* pada data asli untuk mendapatkan elemen data atau struktur unit. Selanjutnya bit *watermark* disisipkan pada elemen atau unit dengan algoritma penyisipan *watermark*.
3. Pendeteksian *watermark*
   Pada tahap ini dilakukan eksekusi *query* yang sama untuk mendapatkan elemen data atau struktur unit yang telah disisipkan dengan bit *watermark* untuk kemudian dilakukan rekonstruksi *watermark.* Karena skema dan data XML dapat dibuat ulang maka *query* juga harus ditulis ulang untuk mengorganisasikan data. Penulisan ulang suatu *query* dapat dilakukan sesuai dengan skema asli dan skema baru.

Berikut ini adalah gambar dari skema penyisipan *watermark.*



Gambar 3 Skema Penyisipan *Watermark*

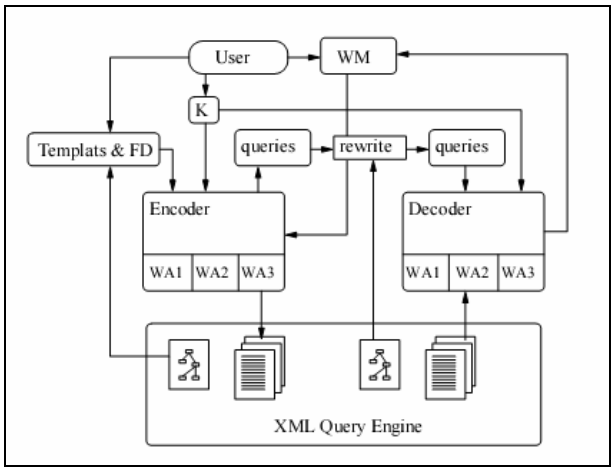
4.2.3 Arsitektur Sistem

Arsitektur sistem pada WmXML memiliki tiga komponen utama, yaitu XML *query engine*, *encoder*, dan *decoder*. XML *query engine* menyediakan akses antarmuka untuk data XML. Sedangkan *encoder* dan *decoder* bertanggung jawab untuk penyisipandan pendeteksian *watermark.* Ketika menyisipkan *watermark,* user memasukkan input berupa *watermark,* kunci rahasia, sekumpulan *query* bersamaan dengan kunci dan *functional dependency* yang diambil dari skema struktur data kepemilikan.

Selanjutnya, *encoder* akan menyisipkan *watermark* ke dalam data dan membangkitkan sekumpulan *query* yang kemudian disimpan oleh user. Pada pendeteksian *watermark,* user menyediakan kunci rahasia yang sama dan sekumpulan *query* untuk digunakan *decoder* menulis ulang *query* tersebut dan mendapatkan *watermark* dari data.

Karena XML dapat mengandung berbagai jenis tipe data, sistem harus menyediakan berbagai jenis algoritma *watermarking* untuk jenis-jenis data tersebut. Tipe data yang sudah didukung oleh sistem ini adalah data numerik dan data berupa gambar. Selain itu, sistem ini juga masih membutuhkan campur tangan user untuk melakukan penulisan ulang suatu *query*.

Berikut adalah gambar arsitektur sistem WmXML:



Gambar 4 Arsitektur Sistem

4.3 Dua Jenis Pendekatan pada *Watermarking* Dokumen XML

Pendekatan yang pertama merupakan teknik *watermarking* pada dokumen XML yang tidak terkompresi. Pendekatan ini berdasarkan pada teknik *watermarking* pada basis data relasional yang dilakukan oleh Agrawal. Pendekatan yang pertama ini sering disebut juga *selective approach* atau pendekatan selektif.

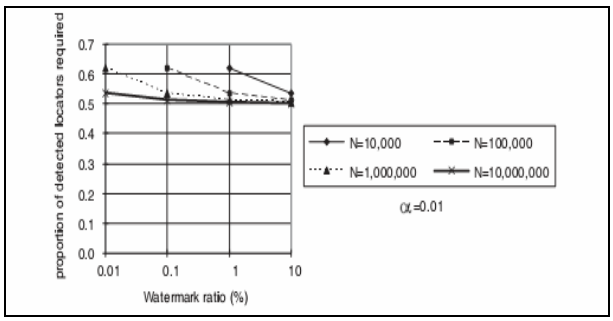
Pendekatan yang kedua merupakan pendekatan yang lebih menarik, yaitu teknik *watermarking* untuk dokumen XML yang sudah terkompresi. Dokumen XML merupakan dokumen yang seringkali di-*update* sehingga pada pendekatan ini akan diperkenalkan suatu skema *watermarking* yang baru untuk diterapkan pada suatu dokumen XML yang terkompresi. Pendekatan kedua ini disebut *compression approach* atau pendekatan kompresi.

4.3.1 Pendekatan Selektif

Pada pendekatan selektif untuk melakukan *watermarking* dokumen XML ini, *watermark* disebar secara acak (*random*) pada dokumen XML berdasarkan kunci rahasia yang dimiliki oleh pemilik data. Tujuannya adalah untuk membuat perubahan pada dokumen XML tanpa terjadi kesalahan selama proses berlangsung.

Sebelum menyisipkan *watermark*, perlu ditentukan terlebih dahulu sebuah *locator* yang dapat dianalogikan dengan *primary key* pada basis data relasional untuk menandai elemen mana yang sebaiknya diberi tanda. Tidak seperti teknik *watermarking* pada basis data relasional, *primary key* tidak perlu didefinisikan pada data XML. Asumsinya adalah pemilik data bertanggung jawab untuk memilih elemen mana yang sesuai untuk menjadi kandidat dari *locator*. Karakteristik terbaik untuk elemen yang dapat dijadikan *locator* adalah nilai elemen tersebut harus unik, tidak dapat dimodifikasi, dan memiliki ruang untuk *locator*. Contohnya adalah tag "ISBN" pada dokumen XML buku dapat dijadikan sebagai *locator*.

Sementara itu, untuk mendeteksi apakah suatu data XML itu asli maka pemilik data harus memberikan kunci rahasia miliknya dan juga file yang berisi *setting* untuk algoritma pendeteksian *watermark.* File tersebut berisi informasi mengenai tingkat signifikan, α, dan daftar kandidat *locator* dalam format XPath. Algoritma pendeteksian akan menemukan jumlah elemen yang ditandai dan *locator* pada data XML kemudian dilanjutkan dengan menghitung *hit rate*. Algoritma pendeteksian ini menggunakan fungsi treshold untuk menghitung bilangan integer terkecil, dinotasikan dengan *t*, sehingga jika *hit rate* lebih besar daripada *t* maka pemilik data dapat mengklaim kepemilikan terhadap dokumen tersebut dengan tingkat kepercayaan (1- α).



Gambar 5 Perbandingan *Locator*

Gambar tersebut menunjukkan perbandingan jumlah *locator* yang dibutuhkan untuk melakukan deteksi yang berhasil dengan tingkat kepercayaan 99% terhadap rasio *watermark*. Ini berarti pada dokumen XML yang berukuran kecil (N=10.000), jika 1% dari *record* diberi tanda maka hanya dibutuhkan 62% *locator* yang harus didetekesi pada tingkat kepercayaan sebesar 99%. Semakin tinggi rasio *watermark*, semakin sedikit jumlah locator yang dibutuhkan untuk dideteksi. Nilai konstanta yang diperlukan untuk perbandingan adalah sebesar 0,5 karena algoritma pendeteksian adalah bersifat probabilistik dan memerlukan lebih dari 50% *locator* yang terdeteksi untuk membedakan sebuah *watermark* dari kemungkinan adanya kejadian *random*. Secara umum, sebuah dokumen XML dengan banyak elemen memerlukan lebih sedikit *locator* yang harus dideteksi daripada dokumen XML dengan sedikit elemen.

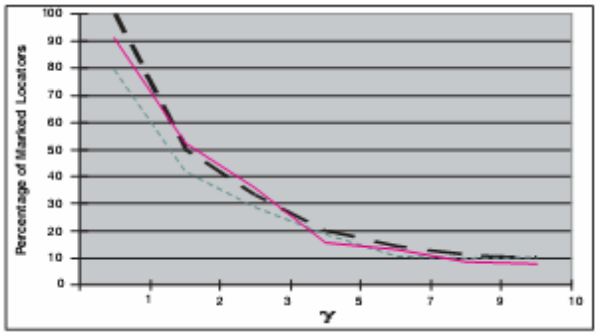
Percobaan pada pendekatan selektif

Berikut ini adalah tabel karakterisitik dari dataset yang akan diuji.

Documents	File size (KB)	No. of records	No. of elements available for marking (v)	No. of least significant bits for marking (ξ)
1998_statistics.xml	1227	1226	4	3
weblog.xml	89809	247024	2	3

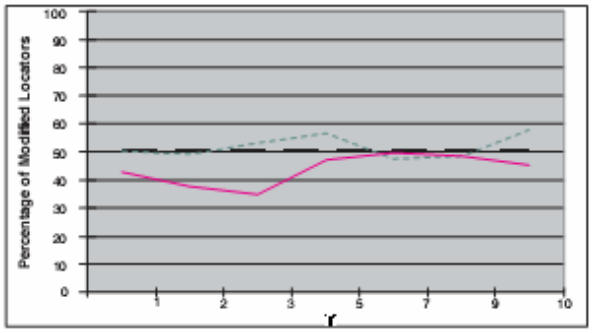
Dokumen pertama bernama "1998_statistics.xml" dan dokumen kedua adalah "weblog.xml". Berdasarkan percobaan yang dilakukan, terlihat bahwa persentase *locator* yang dimodifikasi pada dokumen pertama, yaitu "1998_statistics.xml" mengalami fluktuasi sekitar 50% sedangkan pada dokumen kedua, yaitu "weblog.xml" nilai persentasenya adalah kurang dari 50%.

Berikut adalah gambar grafik dari hasil percobaan pada dokumen "1998_statistics.xml"



Gambar 6 Dokumen "1998_statistics.xml"

Berikut adalah gambar grafik dari hasil percobaan pada dokumen "weblog.xml"



Gambar 7 Dokumen "weblog.xml"

Waktu yang dibutuhkan untuk melakukan algoritma penyisipan *watermark* pada dua dokumen XML tersebut dapat dilihat pada tabel berikut:

Watermark ratio (%)	100.00	50.00	33.33	20.00	14.29	11.11	10.00
1998_statistics.xml (sec)	0.891	0.891	0.861	0.891	0.871	0.871	0.851
weblog.xml (sec)	73.46	69.02	68.01	61.68	61.62	61.44	61.79

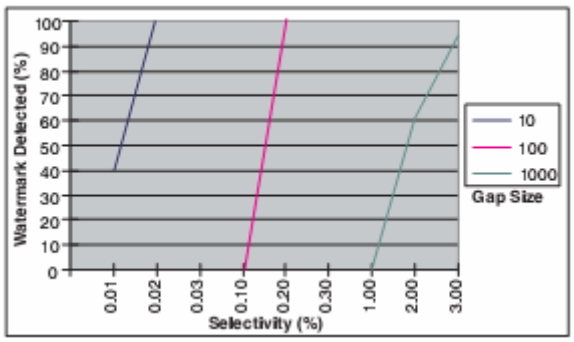
Hasil pada tabel tersebut memperlihatkan bahwa rasio *watermark* tidak memiliki pengaruh besar pada waktu *running* algoritma. Waktu input dan output pada saat pembacaan dokumen menjadi *overhead* utama yang menyebabkan peningkatan waktu *running* algoritma.

Serangan terhadap *watermark* yang disisipkan pada dokumen XML antara lain:

Subtractive Attack

Subtractive attack adalah serangan yang mencoba untuk menghapus keberadaan dari suatu *watermark*. *Subtractive attack* yang berhasil akan mengurangi jumlah *watermark* yang dibuat oleh pemilik dokumen sehingga *watermark* tidak akan bisa dijadikan bukti kepemilikan lagi. *Subset attack* adalah bentuk khusus dari *subtractive attack*, yaitu suatu serangan yang mencoba untuk melakukan penyalinan sebagian dokumen yang memiliki *watermark* dan mengurangi persentase *watermark* yang ditemukan pada dokumen.

Gambar berikut merupakan gambar yang menunjukkan kehandalan pendekatan selektif terhadap *subtractive attack*. Dokumen yang digunakan untuk percobaan ini adalah dokumen "weblog.xml"



Gambar 8 Tingkat Pertahanan "weblog.xml"

Pada gambar tersebut terlihat bahwa ketika ukuran *gap* sama dengan 10,9%, *watermark* dapat dideteksi dengan tingkat selektifitas hanya 0,02%. Ketika ukuran *gap* meningkat, tingkat selektifitas pun akan meningkat untuk mendeteksi lebih dari 90% *watermark.* Pada tingkat selektifitas 0,3%, pendeteksian *watermark* mencapai 100%. Untuk ukuran *gap* sebesar 1000, hanya sedikit tingkat selektifitas yang dapat mencapai 90% pendeteksian *watermark.* Hasil ini menandakan bahwa *watermark* yang disisipkan oleh skema ini memiliki penyebaran yang merata dan memiliki ketahanan yang baik terhadap *subset attack*.

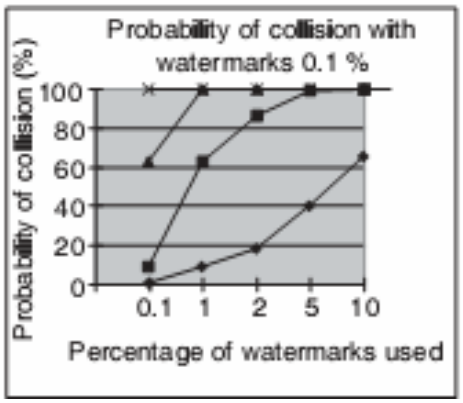
Additive Attack

Pada *additive attack*, serangan dilakukan dengan cara menambahkan *watermark* pada dokumen XML sehingga menjadi milik pihak yang mencuri data XML tersebut. Karena *watermark* yang ditambahkan belakangan dapat menimpa *watermark* asli, dokumen XML palsu tersebut akan memiliki elemen ber-*watermark* yang lebih banyak jumlahnya. Jika M adalah jumlah total elemen yang ber- *watermark,* L adalah jumlah elemen yang dapat diberi *watermark,* dam F adalah jumlah total *watermark* yang disisipkan belakangan, maka kemungkinan wilayah elemen yang saling *overlap* adalah sebagai berikut:

$$\begin{cases} 1 - \Pi_{i=0}^{F-1} \frac{(L-i)-M}{L-i}, & \text{if } M+F < L; \\ 0, & \text{if } M+F \geq L. \end{cases}$$

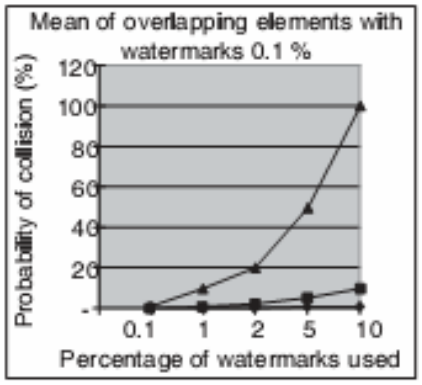
Mean dari overlap = L x (M/L) x (F/L).

Pada dokumen XML yang berukuran kecil (N=10.000), jika pemilik dokumen tersebut menggunakan 10% rasio *watermark* dan terdapat seseorang lain yang menyisipkan *watermark* dengan rasio *watermark* sebesar 0,1%, maka kemungkinan terjadinya overlapping adalah sebesar 65%. Gambar berikut menunjukkan rasio dokumen XML tersebut:



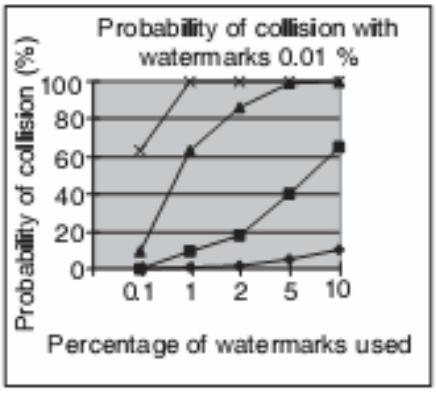
Gambar 9 Rasio Dokumen XML (N=10.000)

Sedangkan mean dari overlapping tersebut hanya satu seperti terlihat pada gambar berikut:



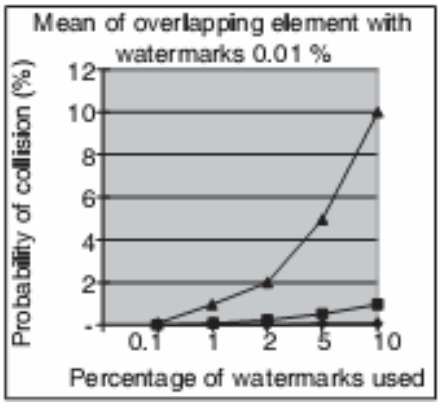
Gambar 10 Grafik Mean (N=10.000)

Pada dokumen XML yang berukuran besar (N=100.000), jika pemilik menggunakan rasio *watermark* sebesar 2% dan seseorang lain menyisipkan *watermark* dengan rasio 0,1% maka kemungkinan overlapping akan semakin tinggi, yaitu sebesar 85% dengan penggunaan rasio *watermark* sebesar 0,01%. Gambar berikut menunjukkan hal tersebut:



Gambar 11 Rasio Dokumen XML (N=100.000)

Sedangkan grafik mean dari overlapping tersebut dapat dilihat pada gambar berikut ini:



Gambar 12 Grafik Mean (N=100.000)

Percobaan yang dilakukan untuk menguji pendekatan selektif menunjukkan bahwa pendekatan selektif tahan terhadap *subtractive* dan *additive attack*. Parameter untuk menentukan *performance* dari pendekatan ini adalah sebagai berikut pertama, ukuran dokumen XML

(N), rasio *watermark* (γ), jumlah locator (v), dan tingkat signifikan dari pengujian (α). Pada dokumen XML yang berukuran besar dapat menggunakan rasio *watermark* yang lebih kecil untuk mendapatkan tingkat kepercayaan pendeteksian *watermark. Overhead* pada saat komputasi relatif lebih kecil jika dibandingkan waktu input output. *Overhead* ini juga berhubungan dengan ukuran dokumen.

Algoritma *watermark* ini juga dapat bertahan dari serangan yang berusaha untuk mengubah struktur data XML ber- *watermark* yang disebut dengan istilah *distortive attack.* Karena struktur XML berupa struktur pohon maka pengubahan menjadi struktur asli dapat dilakukan dengan menggunakan skema aslinya, pada kasus ini dapat dilakukan algoritma pendeteksian *watermark* untuk memeriksa apakah dokumen XML mempunyai tanda kepemilikan.

4.3.2 Pendekatan Kompresi

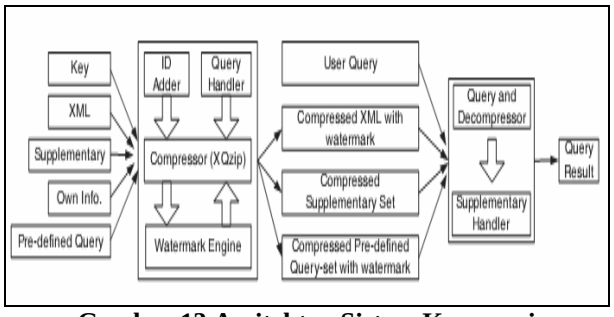
Kebanyakan teknik *watermarking* menggunakan data XML biasa. Pada pendekatan sebelumnya yaitu pendekatan selektif terbukti efisien dan efektif untuk membuktikan kepemilikan suatu dokumen XML, akan tetapi perlindungan data dari akses yang ilegal tidak diperhitungkan pada pendekatan selektif.

Pada bagian ini akan dibahas mengenai pendekatan *watermark* yang diterapkan pada data XML yang terkompresi dengan menyertakan bukti kepemilikan dan juga perlindungan pada keamanan data. Prinsipnya adalah data XML terkompresi tersebut tidak dapat dibuka secara langsung tanpa melakukan dekompresi terlebih dahulu. Penerapan teknik *watermarking* pada data terkompresi adalah suatu tindakan untuk mencegah penyalinan dan pembacaan data oleh pihak yang tidak berhak.

Sama dengan pendekatan selektif, pada pendekatan kompresi ini diperlukan kunci rahasia dari pemilik data untuk melakukan kompresi. Kunci rahasia ini digunakan untuk menentukan lokasi dari *watermark,* setiap salinan dari data akan diberi *watermark* dengan kunci rahasia yang unik dari pemilik data tersebut. Tanpa kunci rahasia maka data yang terkompresi tersebut tidak dapat dibaca.

Seseorang yang memiliki hak untuk mendapatkan data terkompresi tersebut dapat menggunakan *query* yang sudah didefinisikan sebelumnya oleh pemilik. Selain itu pemilik data juga dapat menentukan jumlah data yang dapat dilihat untuk masing-masing pihak yang berhak dengan memberikan kumpulan *query* yang berbeda-beda kepada masing-masing pihak tersebut.

Adapun arsitektur dari sistem kompresi ini dapat dilihat pada gambar berikut:



Gambar 13 Arsitektur Sistem Kompresi

Berikut adalah penjelasan dari arsitektur sistem kompresi tersebut:

ID Adder

ID Adder dibangun dari SAX *parser* dengan cara melakukan parsing pada dokumen XML secara sekuensial dan menyisipkan informasi kepemilikan pada dokumen XML. Informasi tersebut disimpan "*ownership node*" yang berada di tepat bawah akar pada struktur pohon dokumen XML. Label kepemilikan pada node tersebut merupakan nilai *hash* dari kunci rahasia pemiliknya. Untuk mendukung proses *update*, nilai unik ID ditambahkan pada masing-masing elemen untuk memudahkan proses kompresi.

Query Handler

Modul ini merupakan antarmuka yang memungkinkan pemilik data untuk mendefinisikan *query* bagi pihak yang diijinkan untuk mengakses data tersebut. Modul ini dapat memilih bagian mana saja dari dokumen XML yang dapat dilihat kemudian mengubah bagian tersebut menjadi suatu dokumen XML dan melanjutkan proses dokumen XML tersebut ke modul *compressor*.

Compressor* dan *Watermark Engine

Kunci rahasia, fungsi *hash*, dan nilai *gap* digunakan untuk menentukan posisi byte yang akan ditandai dengan *watermark*. Nilai *gap* yang kecil akan menghasilkan lebih banyak *watermark* yang disisipkan pada blok terkompresi. Setelah semua blok dikompresi dan diberi *watermark* maka blok-blok tersebut akan digabung menjadi satu file.

Query* dan *Decompressor

Kumpulan *query* terkompresi yang telah didefinisikan sebelumnya akan didekompresi terlebih dahulu. Kemudian pihak yang memiliki wewenang akan menggunakan *query* tersebut. Solusi dari *query* akan didekompresi dan diproses oleh *supplementary handler* jika *update* diperlukan. Pada proses dekompresi, *query* dan *decompressor* akan melakukan fungsi *hash* pada kunci rahasia dan nilai *gap* untuk menentukan elemen yang memiliki *watermark* dan melakukan *recovery* dokumen tersebut.

Supplementary Handler

Terdapat dua tahapan untuk menangani file tambahan ketika melakukan proses *update* data XML yang terdekompresi. Pertama, modul ini akan menghilangkan

content yang sudah *out-dated* (tidak berlaku) yang telah didefinisikan sebelumnya. Kemudian hasilnya diserahkan pada *ID Adder* yang akan melakukan update *content* tersebut. Selanjutnya, parser akan memeriksa setiap atribut dari masing-masing elemen dan jika atribut bertanda "*added*" maka parser akan menyisipkan nilai yang berkorespondensi dengan elemen. Hasilnya berupa dokumen XML.

Pada sistem *watermark* terkompresi ini, jumlah *watermark* pada masing-masing blok terkompresi dibatasi. Sistem akan melakukan proses pertukaran byte pada *locator* yang ditunjuk oleh fungsi *hash*. Jika lokasi satu byte yang dipilih berjumlah dua kali atau genap maka pertukaran byte tidak akan dilakukan pada lokasi byte tersebut. Untuk memastikan setiap blok mengandung paling sedikit satu buah *watermark* maka jumlah *watermark* pada setiap blok kompresi harus ganjil. Jika m adalah jumlah blok maka jumlah *watermark* dapat ditentukan dengan menggunakan rumus berikut:

$$Number\ of\ mark = \begin{cases} m, & \text{when } m \text{ is odd;} \\ m+1, & \text{otherwise.} \end{cases}$$

Percobaan pada pendekatan kompresi

Beberapa percobaan yang dilakukan untuk mengukur ketahanan sistem *watermarking* terkompresi ini anatara lain:

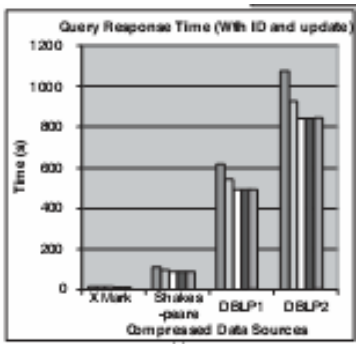
Efektifitas Sistem *Watermarking* Terkompresi

Data dari dokumen yang terkompresi didapatkan dengan menggunakan pasangan {kunci,*gap*} yang sama ketika melakukan kompresi. Untuk menguji efektifitas dari sistem maka dibuatlah skenario pengujian sebagai berikut: pertama, kunci rahasia dan nilai *gap* yang digunakan berbeda, kedua, kunci rahasia sama tetapi nilai *gap* yang digunakan berbeda, ketiga, kunci rahasia yang digunakan berbeda tetap nilai *gap* sama, dan keempat, kunci rahasia dan nilai *gap* yang digunakan sama. Jika pasangan {kunci,*gap*} yang digunakan salah maka *query* dan *decompressor* tidak dapat menentukan lokasi dari pola elemen yang ditandai dan gagal untuk melakukan dekompresi data yang dibutuhkan.

Waktu Respon *Query*

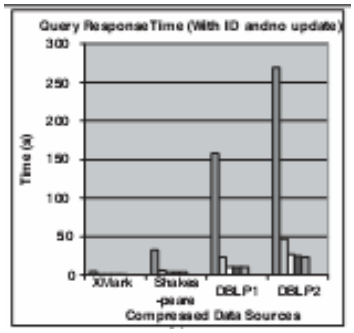
Pada percobaan untuk waktu respon *query* ini digunakan tiga skenario yang berbeda, yaitu pertama, adanya ID dan *update* diperbolehkan, kedua, ID diperbolehkan tetapi *update* tidak diperbolehkan, dan ketiga adanya ID dan *update* tidak diperbolehkan. Pengujian *query* ini dilakukan dengan cara mendapatkan seluruh dokumen dan melakukan proses yang melibatkan dekompresi secara penuh.

Gambar berikut menunjukkan hasil percobaan dari skenario pertama:



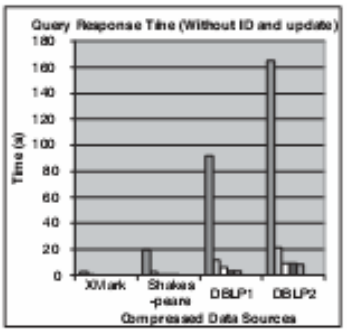
Gambar 14 Hasil Percobaan Pertama

Gambar berikut menunjukkan hasil percobaan dari skenario kedua:



Gambar 15 Hasil Percobaan Kedua

Gambar berikut menunjukkan hasil percobaan dari skenario ketiga:



Gambar 16 Hasil Percobaan Ketiga

Dari hasil percobaan pertama dan kedua terdapat perbedaan besar antara waktu respon query pada kasus *update* dan pada kasus yang tidak dilakukan *update*. Hal ini terlihat pada nilai *gap* yang kecil dan ukuran file yang besar, pada kasus seperti ini penanganan terhadap dokumen XML tambahan akan sangat "mahal" dan membutuhkan banyak waktu. Akan tetapi jika nilai *gap*-nya sangat kecil maka waktu yang diperlukan untuk melakukan proses *recovery* dari elemen yang bertanda dari dokumen XML yang terkompresi menjadi sangat penting dan waktu untuk menangani dokumen tambahan tersebut menjadi kurang penting.

Pada skenario percobaan ketiga yang tidak memperbolehkan adanya ID dan *update*, hasilnya sama dengan yang didapat dari percobaan kedua. Pada kasus

ini sebagian besar waktu digunakan untuk melakukan proses *recovery* dari elemen yang bertanda dari dokumen XML yang terkompresi. Hal ini menunjukkan bahwa waktu respon yang dibutuhkan untuk menangani ID merupakan fungsi yang linier terhadap ukuran dokumen dan tidak membutuhkan banyak biaya *overhead*.

Kehandalan Teknik *Watermarking*

Salah satu serangan untuk menguji kehandalan teknik *watermarking* adalah dengan cara mendapatkan data dari dokumen terkompresi tanpa menggunakan query. Serangan ini diklasifikasikan sebagai *flipping attack* atau *averaging attack*.

Flipping Attack

Serangan *flipping attack* dilakukan dengan menghancurkan *watermark* dengan cara menukarkan posisi byte tertentu. Karena suatu dokumen terkompresi terdiri dari banyak blok yang terkompresi, serangan dapat dilakukan pada setiap blok dan menggabungkan semua blok tersebut untuk membentuk sebuah dokumen terkompresi yang baru. Jika satu blok saja salah termodifikasi maka dokumen terkompresi tersebut tidak dapat didekompresi oleh karena itu serangan harus dilakukan dengan menebak pola dari setiap bok dengan benar.

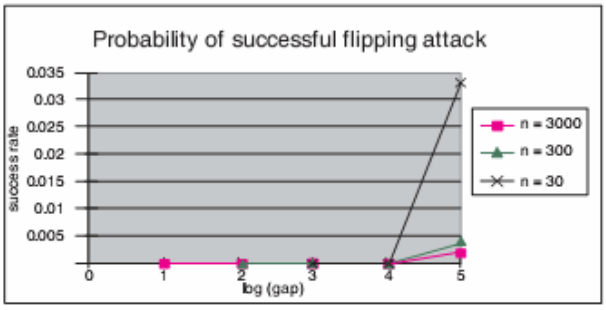
Asumsikan seseorang mengetahui jumlah *locator* yang ditandai, *m*,dan ukuran data, *n*. Jika dua elemen bertanda terletak pada lokasi yang sama, dua elemen tersebut akan saling meniadakan tanda di masing-masing elemen tersebut. Kemungkinan menebak pola dengan benar dapat dihitung dengan rumus berikut:

$$\frac{1}{\Sigma_{r=1,3,\ldots,m} nCr}$$

,di mana 0<m≤n dan m adalah bilangan ganjil.

Tingkat kesuksesan akan menurun sebanding dengan meningkatnya ukuran blok pada nilai *gap* tertentu. Jika ukuran blok sudah sesuai, penurunan pada nilai *gap* akan menurunkan juga tingkat kesuksesan. Sebagai conoh, ukuran blok yang kecil (n=30) dan nilai *gap* yang besar (*gap*=100.000) akan memberikan tingkat kesuksesan yang semakin tinggi, kebanyakan tingkat kesuksesan bernilai sekitar 3%, yang relatif rendah.

Gambar berikut menunjukkan hasil percobaan pada *flipping attack*:



Gambar 17 Hasil Percobaan *Flipping Attack*

Averaging Attack

Averaging attack adalah serangan yang mencoba untuk membangkitkan suatu dokumen yang tidak mengandung *watermark* dari sejumlah dokumen sampel yang ber-*watermark*. Serupa dengan *flipping attack*, pada *averaging attack* serangan dilakukan pada blok per blok data yang terkompresi dengan melakukan analisis semua blok terkompresi dari dokumen sampel yang tersedia dan menggunakan informasi yang didapatkan untuk membuat sebuah blok baru.

Percobaan dilakukan dengan melakukan simulasi *averaging attack* dengan menggnakan Xmark dan Shakespeare dataset. Untuk setiap dataset, terdapat 99 sampel yang tersedia. Kemudian dilakukan pembangkitan ukuran blok secara acak (*random*) sebagai target serangan. Untuk mendapatkan *averaging attack* yang berhasil maka semua blok buatan tersebut dibutuhkan untuk melakukan serangan tersebut.

Berikut ini adalah tabel dari hasil percobaan *averaging attack:*

Gap Size	1	5	10	100	1000	10000	100000
XMark	Fails	Fails	Fails	5	3	3	3
Shakespeare	Fails	Fails	Fails	5	3	3	3

Berdasarkan tabel tersebut, ditemukan bahwa serangan yang dilakukan pada data terkompresi dapat berhasil dengan mudah jika nilai *gap* besar. Jika nilai gap lebih kecil dari 10 maka semua serangan akan gagal karena kesalahan byte yang sedikit sudah cukup untuk mencegah dari serangan dekompresi data. Dari hasil tersebut dapat dilihat ketika nilai *gap* kecil, sistem akan semakin handal terhadap *averaging attack.* Selain itu, terdapat pula nilai *tradeoff* antara nilai *gap* dengan waktu kompresi.

Beberapa serangan lain yang sering dilakukan pada dokumen XML yang memiliki *watermark* antara lain:

1. *Data Alteration*
   Serangan ini dilakukan dengan dengan cara mengubah atau memodifikasi elemen atau struktur dari data untuk menghancurkan *watermark.*
2. *Data Reduction*
   Serangan ini dilakukan dengan cara memilih subset dari semi-struktur data dan menghilangkan sisanya.

3. *Data Reorganization*
   Serangan ini dilakukan dengan cara melakukan reorganisasi data sesuai dengan skema baru dan menyusun ulang elemen datanya
4. *Redundancy Removal*
   Serangan ini dilakukan dengan cara melakukan identifikasi dan menghapus redundansi pada data.

5. Kesimpulan

Penerapan teknik *watermarking* pada basis data relasional merupakan kajian yang penting dan menarik yang sangat berguna untuk menandai suatu kepemilikan terhadap suatu basis data. Teknik *watermarking* tersebut haruslah handal (*robust*) terhadap serangan yang berusaha menghilangkan *watermark* suatu basis data (*malicious attack*).

Pada penerapan teknik *watermarking* pada dokumen XML, ketika penyisipan *watermark* dilakukan maka user harus memberi masukan berupa sebuah *watermark,* kunci rahasia, sekumpulan *query* yang sudah didefinisikan sebelumnya bersama dengan kunci dan *functional dependency* yang ditemukan dari skema sturktur data. Selanjutnya encoder akan menyisipkan *watermark* ke dalam data dan membangkitkan sekumpulan *query* yang akan disimpan oleh user.

Pada pendeteksian *watermark*, user harus menyediakan kunci rahasia yang sama dan kumpulan query sebagai input pada decoder untuk menulis ulang *query* dan mendapatkan *watermark* dari data. Karena XML dapat mengandung berbagai jenis tipe data, sistem harus menyediakan berbagai jenis algoritma *watermarking* untuk jenis-jenis data tersebut. Tipe data yang sudah didukung oleh sistem ini adalah data numerik dan data berupa gambar. Selain itu, sistem ini juga masih membutuhkan campur tangan user untuk melakukan penulisan ulang suatu *query*.

Sebuah *watermark* akan dapat dibuat ulang jika serangan yang dilakukan terhadap *watermark* tersebut tidak menghancurkan kegunaan datanya atau dengan kata lain jika serangan yang dilakukan berhasil maka data akan menjadi tidak berguna lagi.

DAFTAR PUSTAKA

[1] Attalah, M., S. Wagstaff. (1999). *Watermarking with Quadraric Residue*. Proc.of IS&T/SPIE Conference on Security and Watermarking of Multimedia Contents.

[2] Agrawal, Rakesh, Jerry Kiernan. (2002). *Watermarking Relational Databases*. IBM Almaden Research Center.

[3] Bender, W., D. Gruhl, N.Morimoto. (1995). *Tecniques for Data Hiding*. Proc. of the SPIE 2420 Storage and Retrieval for Image and Video Databases III.

[4] Benjamin, S., B. Schwartz, R. Cole. (1999). *Accuracy of ACARS Wind and Temperature Observations Determined by Collocation*. Weather and Forecasting.

[5] Boney, Laurence, Ahmed H. Tewik, Khaled N. Hamdy. (1996). *Digital Watermarks for Audio Signals.* Proceedings of International Conference on Multimedia Computing and Systems.

[6] Cheng, J., W. Ng. (2004). *Xqzip: Querying Compressed XML Using Structural Indexing.* Proc. Of the EDBT.

[7] Collberg, Christian, Clark Thomborson. (1999). *Software Watermarking: Models and Dynamic Embeddings*. Proceedings of the Principles of Programming Language. POPL.

[8] Cox, I.J., M.L. Miller, J.A. Bloom. (2001). *Digital Watermarking*. Morgan Kaufmann Publishers, Inc. San Fransisco.

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

[10] Dugelay, J-L., Roche S. (2000). *A Survey of Current Watermarking Techniques*. Information Hiding Techniques for Steganography and Digital Watermarking. Artech House.

[11] Flum, J., M. Frick, M. Grohe. (2001). *Query Evaluation via Tree Decompositions.* International Conference on Databases Theory (ICDT). Lecture Notes in Computer Science. Springer.

[12] Gross, David, Amblard. (2003). *Query-preserving Watermarking of Relational Databases and XML Documents.* ACM.

[13] Hartung, Frank, Bernd Girod. (1998). *Watermarking of Uncompressed and Compressed Video*. Signal Processing.

[14] Katzenbeisser, S., F. A. P. Petitcolas. (2000). *Information Hiding: Techniques for Steganography and Digital Watermarking.* Computer Security Series: Artech House.

[15] Khanna, S., F. Zane. (2000). *Watermarking Maps: Hiding Information in Structured Data*. Symposium on Discrete Algorithms (SODA).

[16] Libkin, L., L. Wong. (1997). *Query Languages for Bags and Aggregate Functions*. Journal of Computer and System Sciences.

[17] Milo, T., D. Suciu, V. Vianu. (2000) *Typechecking for XML Transformers.* Symposium on Principles of Databases Systems (PODS).

[18] Maes, M. (1998). *The Histogram Attack on Fixed Depth Image Watermarks.* Proc. of the 2nd International Workshop on Information Hiding. Springer-Verlag Lecture Notes in Computer Science.

[19] Munir, Rinaldi. (2006). *Diktat Kuliah IF5054 Kriptografi*. Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung.

[20] Ng, Wilfred, Ho-Lam Lau. (2005). *Effective Approaches for Watermarking XML Data*. Department of Computer Science, The Hong Kong University of Science and Technology, Hong Kong.

[21] Papadimitriou, C.H., M. Yannakakis. (1991). *Optimization, Approximation, and Complexity Classes.* Journal of Computer and System Sciences.

[22] Schneier, B. (1996$). *Applied Cryptography*. John Wiley: Second Edition.

[23] Sion, Radu, Mikhail Atallah, Sunil Prabhakar. (2002). *Watermarking Relational Databases.* Computer Sciences, Purdue University.

[24] Sion, Radu, Mikhail Atallah, Sunil Prabhakar. (2003). *Right Protection for Relational Data.* Proceedings of ACM SIGMOD International Conference on Management of Data. ACM Press.

[25] Sion, Radu, Mikhail Atallah, Sunil Prabhakar. (2004). *Resilient Information Hiding for Abstract Semi-Structures*. Proceedings of Workshop on Digital Watermarking. IWDW 2003.

[26] Swanson, Mitchell, Bin Zhu, Ahmed H. Tewik. (1996). *Transparent Robust Image Watermarking*. Proceedings of the 1996 SPIE Conf. on Visual Communications and Image Proc.

[27] Wolfe, G., J.L. Wong, M. Potkonjak. (2001). *Watermarking Graph Partitioning Solutions*. DAC.

[28] Yu, Cong, Lucian Popa. *Constraint-based XML Query Rewriting for Data Integration*. (2004). Proceedings of the 2004 ACM SIGMOD international conference on Management of Data. ACM Press.

[29] Zhou, Xuan, HweeHwa Pang, Kian-Lee Tan, Dhruv Mangla. (2005). *WmXML: A System for Watermarking XML Data*. Proceedings of the 31st VLDB Conference, Trondheim, Norwegia.