Penggunaan *XML Database Xindice* pada Aplikasi Kriptografi menggunakan Data *XML* untuk Keamanan Distribusi Data

Edy Winarno
Fakultas Teknologi Informasi, Universitas Stikubank Semarang
edy_winarno@yahoo.co.id

Abtract

In order to send data or information processed by using cryptography people can use XML (eXtensible Markup Language). XML data are general texts file which consist of several different tags which are defined by the XML document maker. In this research, the XML data were processed by using Password Based Encryption cryptography. This method consists of encryption and decryption meant to alter plaintext into chipertext which in turn creates chipertext from plaintext by using password key.

[2]The implementation of the program in this research is a prototype of Hospital's web facilitated with encyption menus and decryption data which are meant to protect the saved data which are in turn accessed and processed by authorized people. The implementation of the program is built as a web application with Java Server Page (JSP) in NetBeans 6.0 with Password Based Encryption cryptography provided by Java Cryptography Extension (JCE).

Keywords : *XML, XML database Xindice, Password Based Encryption, encryption, decryption.*

1. Pendahuluan

Untuk mengirimkan informasi atau data yang diolah menggunakan teknik kriptografi salah satunya menggunakan data XML yang merupakan singkatan dari *eXtensible Markup Language* (Noprianto,2004). Sebuah data XML merupakan sebuah file teks biasa yang berisikan berbagai tag yang didefinisikan sendiri oleh pembuat dokumen XML tersebut.

Data XML sering disimpan sebagai file data dalam sebuah *filesystem* atau sebagai teks atau karakter dalam *relational databases*. Namun, data XML dalam ukuran yang besar akan sulit dalam pengolahan datanya. *Relational Databases* tidak cocok untuk pengolahan data-data XML yang mempunyai struktur tertentu. Penggunaan teks dalam sebuah *relational databases* dapat dengan mudah disimpan sebagai sebuah data, tapi akan kesulitan ketika memberikan perintah atau *query* terhadap struktur data XML yang telah disimpan. *Database* yang mampu mengolah dan bisa mengakomodasi data-data XML adalah XML *Database* Xindice. Database ini merupakan sebuah sistem basis data yang dapat mengakomodasi data-data XML yang digunakan untuk memudahkan dalam penyimpanan dan pengolahan data XML

Data XML ini kemudian diolah menggunakan proses enkripsi dan dekripsi menggunakan teknik kriptografi. Data XML digunakan untuk proses enkripsi dan dekripsi yang dilakukan pada tag tertentu sesuai yang diperlukan. Enkripsi dan dekripsi tidak dilakukan pada sebuah kesatuan dokumen XML, tapi dilakukan dengan memilih pada tag-tag tertentu sesuai yang diinginkan.

Teknik kriptografi adalah teknik pengiriman data yang dikirimkan ke tempat tujuan lain dan disamarkan sedemikian rupa sehingga data itu tidak bisa dimengerti oleh pihak yang tidak berhak. Data asli yang akan dikirimkan dan belum mengalami penyandian disebut dengan istilah *plaintext*, dan setelah disamarkan dengan suatu cara penyandian maka *plaintext* ini akan berubah menjadi *ciphertext*. Teknik untuk mengubah *plaintext* menjadi *ciphertext* disebut sebagai enkripsi *(encryption),* sedangkan teknik untuk mengubah kembali *chipertext* menjadi *plaintext* disebut dekripsi *(decryption).*

Salah satu contoh aplikasi dari teknik kriptografi dalam penelitian ini digunakan pada proses penyandian data rekam medis pada sebuah web rumah sakit yang memiliki beberapa bagian unit kerja dimana masing-masing bagian tersebut memiliki kewenangan masing-masing untuk mengetahui, menyimpan dan memproses data pasien sehingga tingkat kerahasiaan dan keamanan data pasien dapat dijaga. Setiap bagian dari rumah sakit tersebut memiliki kunci untuk dapat mengenkripsi dan mendekripsikan data yang diterima, sehingga tiap bagian mempunyai kewenangan masing-masing terhadap kerahasiaan dan keamanan data pasien.

Dalam proses enkripsi dan dekripsi data terdapat beberapa metode, salah satunya menggunakan metode *Password Based Encryption*. Metode *Password Based Encryption* merupakan metode enkripsi dengan algoritma kunci simetrik yang menggunakan kesepakatan kunci yang sama untuk proses enkripsi dan dekripsinya.

2. Metode Penelitian

2.1 Perangkat Keras

Perangkat keras yang digunakan dalam membuat aplikasi pada penelitian ini adalah sebuah Laptop dengan spesifikasi Intel(R) Celeron(R) M CPU 430 @1,73 GHz, 1,49 GB of RAM, dan Harddisk 80GB.

2.2 Perangkat Lunak

Perangkat lunak yang digunakan dalam membangun aplikasi Penggunaan XML *Database* Xindice pada Aplikasi Kriptografi menggunakan Data XML untuk Keamanan Distribusi Data adalah sebagai berikut.

1. *XML Database* Xindice sebagai basis data yang mengakomodasi data-data XML.
2. NetBeans IDE 6.0 dan Macromedia Dream Weaver 8 untuk membuat *prototype* program aplikasi kriptografi pada Web
3. JSP (*Java Server Page*) sebagai perangkat lunak pembantu untuk editor program
4. *Browser* Mozilla Firefox untuk menampilkan aplikasi program.
5. Microsoft Windows XP Professional sebagai sistem operasi yang digunakan untuk membangun sistem.

2.3 Subyek Penelitian

Yang menjadi subyek dalam penelitian ini adalah sistem distribusi data rekam medis pasien di sebuah contoh rumah sakit yang terdapat beberapa bagian penting yang memiliki kewenangan dan hak masing-masing untuk mengakses data dan untuk mengetahui mengenai informasi dari data yang dibutuhkan.

2.4 Spesifikasi Sistem

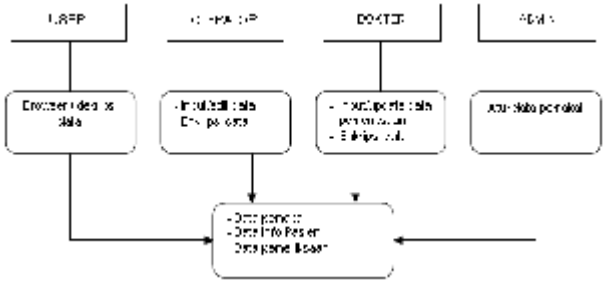
Aplikasi yang akan dibangun meliputi spesifikasi sistem sebagai berikut :

1. Aplikasi berupa pembuatan sebuah *prototype* web dari sebuah contoh rumah sakit yang dapat ditampilkan dalam sebuah *browser* yang berisi tentang informasi mengenai rumah sakit dan data – data yang bisa diakses oleh siapapun.
2. Aplikasi dilengkapi dengan informasi data rekam medis pasien rumah sakit yang diatur dengan sebuah model sistem sederhana. Sistem yang dirancang memiliki beberapa bagian yang terdiri dari administrator, dokter, operator dan *user* (pasien) dimana masing-masing bagian memiliki hak sendiri-sendiri sesuai dengan distribusi password yang telah disepakati untuk memasukkan data, menambah data, mengenkripsi dan mendekripsi data, dengan tujuan agar data dapat disampaikan kepada pihak yang berhak dan berwenang untuk mengolah dan mengakses informasi data.
3. Aplikasi dibuat dengan menambahkan sebuah basis data menggunakan XML *database* Xindice. Xindice merupakan sebuah *database* yang dapat mengakomodasi data-data XML yang dikembangkan dalam penelitian ini. Dibandingkan dengan RDBMS (*Relational Database Management System*) penggunaan *XML Database* dapat lebih memudahkan dalam proses aplikasi dan pengembangan data XML.
4. Aplikasi yang dibuat menggunakan proses enkripsi dan dekripsi dengan program java

dengan metode *Password Based Encryption* (PBE) yang telah disediakan dalam JCE (*Java Cryptography Extension*). JCE merupakan sebuah ekstensi kriptografi yang dikembangkan oleh Java yang memiliki beberapa pilihan metode kriptografi yang dapat digunakan secara bebas untuk pengembangan kriptografi.

2.5 Perancangan Desain Sistem

Tahap perancangan desain sistem merupakan tahap penggambaran atau identifikasi komponen-komponen fungsional yang digunakan dalam perencanaan pengembangan sistem. Tahap perancangan sistem ini bertujuan untuk mendesain sistem yang lengkap dan jelas yang akan digunakan dalam implementasi yang ditujukan untuk memenuhi kebutuhan pemakai sistem.



Gambar 1. Rancangan Desain Sistem

Gambar 1 diatas merupakan rancangan desain dari sistem kriptografi PBE yang digunakan untuk keamanan distribusi data pada aplikasi web sebuah rumah sakit. Informasi pada sistem aplikasi dapat dijelaskan secara umum yang terdiri dari beberapa bagian entitas yaitu :

1. *Administrator*

   Administrator terlebih dahulu harus melakukan *login*, kemudian sistem akan memberikan konfirmasi login. Setelah login diterima, *administrator* dapat melakukan perubahan pada data pemakai dimana administrator mengatur administrasi perubahan dan penambahan data pemakai. Data pemakai dalam sistem ini adalah dokter dan *operator* yang masing-masing mempunyai kunci *password* masing-masing dalam membuka akses dalam sistem.

2. *Operator*

   Operator melakukan *login* terlebih dahulu, kemudian setelah mendapat hak akses maka operator mempunyai hak untuk mengisi data-data pasien, menambah dan menghapus data pasien, serta melakukan proses enkripsi dan dekripsi menggunakan kunci *password* yang sama dengan yang diberikan kepada tiap-tiap user (pasien), kemudian menyimpan data pasien untuk dipergunakan oleh dokter dalam penambahan data rekam medis, dan untuk diakses oleh *user* (pasien).

3. Dokter

   Dokter juga melakukan *login* terlebih dahulu, kemudian setelah mendapat hak akses maka dokter dapat membuka data pasien sesuai dengan nomor identitas yang telah disepakati dengan operator dan *user*. Kemudian dokter juga dapat menambahkan data tambahan pemeriksaan yang bertujuan untuk memberikan *update* data rekam medis bagi seorang pasien. Data rekam medis ini merupakan data riwayat penyakit dan kesehatan pasien yang hanya dapat dibaca oleh si pasien melalui proses dekripsi. Oleh dokter data rekam medis ini kemudian dilakukan proses enkripsi dan disimpan untuk nantinya dapat diakses oleh pihak pasien. Proses enkripsi menggunakan *password* nomor identitas masing-masing pasien untuk memudahkan pasien pada saat membuka dan mengakses data riwayat rekam medis dirinya.

4. *User* (pasien)

   Pasien merupakan entitas terakhir yang dapat membuka akses untuk mengetahui data-data rekam medis dirinya dengan cara memasukkan *login password* berupa nomor identitas pasien yang telah disepakati dengan operator. Pasien dapat mengetahui data informasi tentang dirinya dan dapat melakukan proses dekripsi menggunakan *password* yang telah diberikan untuk membuka data rekam medis yang terenkripsi.

2.6 Perancangan menggunakan XML database Xindice

XML database Xindice merupakan sebuah *server* yang didesain untuk menyimpan data-data XML yang disebut dengan *collections.*

Collections hampir mirip dengan tabel – tabel pada *Relational Database Management Systems* (RDBMS). Dalam xindice data disimpan dalam sebuah basis data dan data tersebut digunakan sebagai sebuah *document collection.* Dalam sebuah *database* terdiri dari beberapa *child collection*.

XML *Database* Xindice digunakan sebagai *server* basis data yang didesain dari awal untuk menyimpan data XML, atau yang saat ini lebih dikenal dengan *native* XML.

Keuntungan utama penggunaan basis data *native* XML adalah memudahkan pengolahan data aplikasi yang berupa data XML. Karena dengan dipergunakannya *server* basis data *native* XML, aplikasi tidak perlu melakukan pemetaan dari data XML ke struktur data tertentu. Aplikasi dapat langsung menyimpan data XML, dan mendapatkan kembali data XML dalam format yang tidak memerlukan perubahan.

Pada saat ini, Xindice menggunakan perintah XPath untuk bahasa *query*-nya dan XML:DB *Update* untuk bahasa *update*-nya. Selain itu, fungsionalitas Xindice juga bisa diakses melalui aplikasi lain, tidak hanya dengan program Java, tapi bisa juga dengan menggunakan API XML-RPC, yang merupakan salah satu mekanisme komunikasi antar aplikasi menggunakan protokol HTTP.

2.7 Enkripsi dan Dekripsi Data XML

Pada sistem aplikasi ini menggunakan proses enkripsi dan dekripsi dengan metode PBE (*Password Based Encryption*) with MD5 and DES. Enkripsi dan dekripsi dilakukan pada struktur data base XML yang telah dibuat, yaitu dengan cara mengenkripsi dan mendekripsi tag-tag XML yang diperlukan. Metode PBE *with* MD5 *and* DES adalah metode kriptografi simetrik yang telah disediakan pada JCE (*Java Cryptography Extension*) sebagai sebuah metode enkripsi dan dekripsi yang aplikatif terhadap pemrograman java.

Tag XML yang telah dipilih kemudian akan dienkripsi menggunakan metode PBE with MD5 and DES untuk menghasilkan data *chipertex*. Data *chipertex* merupakan data hasil pengolahan enkripsi dari sebuah data *plaintex* yang merupakan tampilan sebuah data dari sebuah tag XML. Pada proses ini sebuah data pada tag XML akan diubah dan dikombinasikan dengan kunci *password* yang dimasukkan sehingga akan menghasilkan tampilan data *chipertex* yang merupakan tampilan karakter acak yang tidak dapat dibaca.

Data *chipertex* yang merupakan data hasil pengolahan enkripsi dari sebuah data *plaintex* kemudian akan didekripsi kembali menjadi tampilan data *plaintex*. Pada proses dekripsi ini data *chipertex* akan dikombinasikan dengan kunci *password* sehingga akan menghasilkan data *plaintex* yang sama seperti aslinya. Kunci *password* yang digunakan juga harus sama dengan kunci *password* pada saat melakukan proses enkripsi.

III. Hasil dan Pembahasan

3.1. Pembahasan Pengolahan data pada XML Database Xindice

Sebagai *database native* XML, Xindice memiliki kelebihan dalam bekerja menggunakan data XML. Aplikasi dapat bekerja dengan data XML secara fleksibel. Beberapa kelebihan tersebut di antaranya adalah:

1. Dokumen-dokumen XML disimpan di dalam koleksi (*collection*) yang dapat menerima perintah *query* secara keseluruhan. Aplikasi dapat membuat satu koleksi untuk satu jenis dokumen, atau satu koleksi untuk banyak jenis dokumen. Xindice mendukung penuh pilihan manapun yang dipilih.
2. Untuk melakukan *query* atau pencarian pada dokumen, dipergunakan sintaksis XPath sebagaimana yang didefinisikan oleh W3C (*World Wide Web Consortium*). Fitur ini menyediakan mekanisme yang fleksibel untuk melakukan pencarian pada dokumen dengan melakukan navigasi dan membatasi output dari *tree* yang dihasilkan.
3. Untuk mempercepat kinerja pencarian, aplikasi dapat mendefinisikan indeks terhadap elemen atau atribut. Hal ini akan meningkatkan performa pencarian secara cepat dan signifikan.
4. Memiliki fungsi XUpdate yang merupakan mekanisme penyegaran dokumen berbasis XML, yang dilakukan disisi *server*. Modifikasi dari hasil perintah XUpdate

dapat diaplikasikan pada satu dokumen, atau juga pada keseluruhan dokumen.

5. Xindice mengimplementasikan dukungan penuh kepada standar pengolahan data XML yang disebut dengan XML:DB *Application Programming Interface*. Seperti JDBC yang merupakan standar pengembangan aplikasi basis data, maka XML:DB merupakan standar pengembangan aplikasi XML.
6. Untuk mendukung administrasi basis data, Xindice menyediakan seperangkat alat bantu berbasis konsol, yang merupakan implementasi dari semua fungsionalitas yang dapat diprogram melalui XML:DB API. Sehingga, admin dari basis data dapat melakukan perawatan terhadap basis data tanpa perlu membuat program Java tersendiri.
7. *Server* Xindice dibangun secara modular, sehingga sangat mudah untuk menambah atau menghapus komponen agar *server* sesuai dengan kebutuhan.

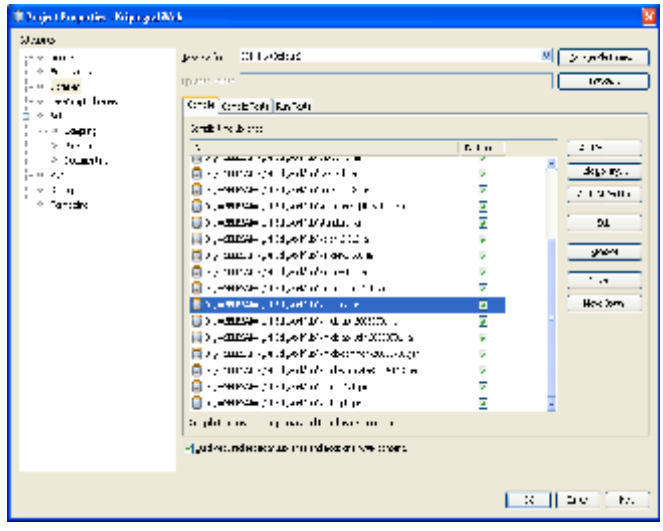
3.2. Pemrograman Menggunakan XML *Database* Xindice

Pada penelitian ini dipergunakan Netbeans 6.0 untuk proses pengembangan sistem, sehingga dalam penjelasan persiapan sistem akan dipergunakan Netbeans 6.0 sebagai acuan.

Setelah melakukan download dari situs Xindice, maka ada beberapa file Jar yang merupakan kumpulan pustaka Java yang harus disertakan pada sembarang aplikasi yang akan mengakses basis data Xindice, yaitu:

- **xindice.jar** – berisi kelas utama Xindice yang dipergunakan oleh aplikasi klien pengakses database Xindice.
- **xmldb-common.jar**, **xmldb-api.jar**, **xmldb-api-sdk.jar**, **xmldb-xupdate.jar** – berisi implementasi XML:DB API dan XUpdate API.
- **xml-apis.jar** – berisi Java XML APIs.
- **xerces.jar** – berisi parser XML Xerces.
- **xalan.jar** – berisi mesin XSLT Xalan.
- **commons-logging.jar** – berisi paket logging Jakarta Commons Logging.

Untuk menambahkan jar tersebut pada aplikasi Netbeans, menggunakan perintah Project Properties, klik node Libraries, klik tombol Add Jar/Folder dan mengarahkan ke file-file jar tersebut. Berikut adalah tampilan Netbeans Project Properties seperti terlihat pada gambar 2. di bawah ini :



Gambar 2. Project Properties dari Netbeans

Setelah pustaka-pustaka yang diperlukan telah didefinisikan, maka project tersebut telah dapat menggunakan kelas-kelas dalam pustaka Xindice.

Untuk memulai perintah *query* dengan XPath maupun *update* dengan XUpdate, maka terlebih dahulu koneksi ke *server* basis data Xindice bisa dilakukan.

Xindice menggunakan arsitektur modular seperti JDBC, sehingga urutan proses koneksi memiliki kesamaan yang erat dengan JDBC. Pada kode di atas, terdapat instansiasi dari kelas org.apache.xindice.client. xmldb.DatabaseImpl, yang merupakan *driver* untuk koneksi ke basis data tertentu, yang umumnya ditemui pada aplikasi yang menggunakan JDBC.

Setelah koneksi berhasil dilakukan, maka kelas Database Manager, yang telah mendaftarkan instan database, dapat dipergunakan sebagai *entry point* untuk mengakses basis data Xindice.

3.3. Pemrosesan Dokumen XML pada *Database* Xindice

Dokumen yang akan diproses pada *database* xindice adalah dokumen pasien.xml dan pemakai.xml seperti yang terlihat pada listing program 1 dan 2 di bawah ini.

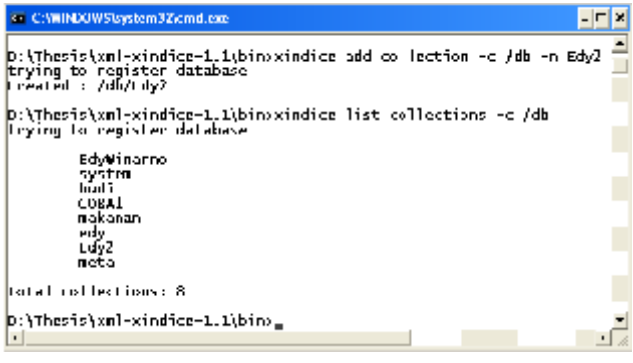
```
<daftar_pasien>
  <pasien>
    <nomorinduk>1518989</nomorinduk>
    <nama>Edy Winarno</nama>
    <nomoridentitas>003</nomoridentitas>
    <tempatlahir>Salatiga / 15 Nopember
     1975</tempatlahir>
    <jeniskelamin>Laki-Laki</jeniskelamin>
    <golongandarah>A</golongandarah>
    <tinggi>165</tinggi>
    <berat>68</berat>
    <telepon>09999999</telepon>
    <pekerjaan>swasta</pekerjaan>
    <alergiobat>Asetaminofen</alergiobat>
    <alamat>Semarang</alamat>
    <riwayatpemeriksaan />
  </pasien>
</daftar_pasien>
```

Listing 1. Program pasien.xml

```
<daftar_pemakai>
 <pemakai>
    <idpemakai>1</idpemakai>
    <nama>edy</nama>
    <sandi>admin</sandi>
    <peran>Administrator</peran>
 </pemakai>
  </daftar_pemakai>
```

Listing 2. Program pemakai.xml

Langkah pertama adalah membuat sebuah koleksi (*collection*) untuk dimasukkan dalam basis data Xindice yang akan digunakan sebagai tempat untuk menyimpan data XML yang telah disiapkan. Pembuatan koleksi seperti ditunjukkan pada tampilan program pada gambar 6 di bawah ini :



Gambar 3. Penambahan sebuah koleksi dalam xindice

3.4. Modifikasi Dokumen XML pada Basis Data dengan XUpdate

Seperti perintah pada basis data RDBMS (*Relational Data Base Management System*), maka modifikasi dokumen pada Xindice juga terdiri dari *INSERT*, *UPDATE* dan *DELETE*. Masing-masing perintah digunakan untuk memproses dan memodifikasi setiap adanya perubahan data yang diinginkan pada basis data. Untuk setiap tahap perintah akan dipergunakan perintah menggunakan konsolnya.

Perintah XUpdate ini bisa sangat panjang dan memerlukan modifikasi yang sering, maka pada utilitas konsol, perintah tersebut disimpan di dalam file teks tersendiri, untuk kemudian disertakan nama filenya pada setiap perintah modifikasi dokumen.

Adapun nilai dari variabel *xupdate* yang akan menentukan efek dari eksekusi servis tersebut, yaitu bisa bertipe INSERT, UPDATE ataupun DELETE.

3.5. Pembahasan Kriptografi

Data-data XML yang telah diolah menggunakan proses penyimpanan dan modifikasi pada *database* Xindice selanjutnya akan diproses menggunakan teknik kriptografi. Teknik kriptografi dilakukan dengan cara mengenkripsi dan mendekripsi tag-tag XML yang diperlukan. Enkripsi dan dekripsi dilakukan menggunakan kriptografi *Password Based Encryption* yang telah disediakan oleh JCE (*Java Crytography Extension*) dengan algoritma yang digunakan adalah metode PBEwithMD5andDES. Sebelum dilakukan proses enkripsi dan dekripsi, urutan pengolahan data yang digunakan mulai dari proses pengambilan data sampai ke menampilkan data yang telah dienkripsi dan didekripsi adalah sebagai berikut:

1. Registrasi *Server Database*

   Registrasi ini dilakukan pada saat awal proses dijalankan, yaitu dengan pengaktifan *server database* yang digunakan sebagai tempat penyimpanan dan pemrosesan data secara keseluruhan.

2. Pengambilan data *Collection* pada *Database*

   Pada proses ini dilakukan pengambilan data yang telah disimpan pada basis data yaitu

berupa collection yang didalamnya berisi dokumen-dokumen XML yang akan diproses menggunakan kriptografi.

3. Instantiasi Xpath dan Xupdate

   Pada proses ini dilakukan pengaturan dan modifikasi dokumen XML menggunakan XpathQueryService dan XupdateQueryService.

4. Proses Enkripsi pada Tag yang diperlukan

   Enkripsi dilakukan pada tag yang ditentukan sesuai dengan yang dipilih. Enkripsi bisa dilakukan lebih dari satu tag.

5. Proses Dekripsi pada Tag yang diperlukan

   Pada proses dekripsi dilakukan hal yang sama seperti pada proses enkripsi. Dekripsi juga bisa dilakukan lebih dari satu tag.

6. Menampilkan informasi data hasil Enkripsi dan Dekripsi

   Hasil proses enkripsi dan dekripsi kemudian ditampilkan sebagai hasil tampilan akhir pada aplikasi web berupa hasil enkripsi (*chipertext*) dan hasil dekripsi (*plaintext*)

3.6. Pembahasan Proses Enkripsi dan Dekripsi

Proses enkripsi dan dekripsi dilakukan dengan menggunakan *Password Based Encryption* yang diperoleh dari JCE *(Java Cryptography Extension*). Ada 3 class yang digunakan pada penggunaan teknik kriptografi ini yaitu :

1. PBEParameterSpec Class

   Javax.crypto.spec.PBEParameterSpec class merupakan class yang disediakan sebagai pembawa *salt* dan *itetarion count* untuk membantu proses enkripsi dan dekripsinya.

2. The PBEKeySpec Class

   Javax.crypto.spec.PBEKeySpec merupakan class yang digunakan untuk memproses *password* yang digunakan dalam proses enkripsi dan dekripsinya.

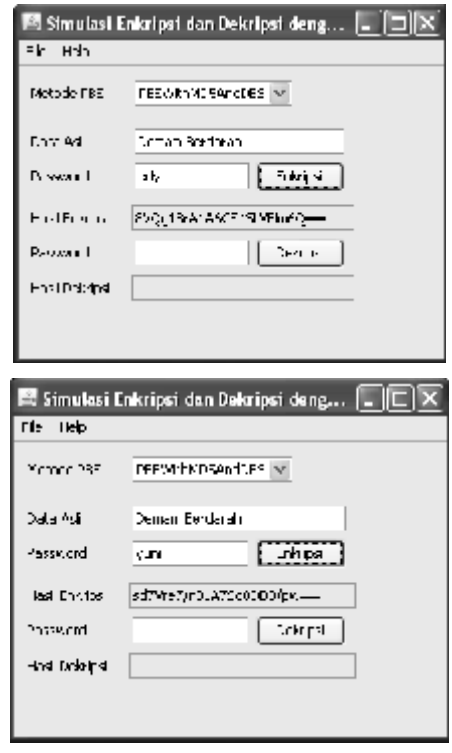
3. The SecretKeyFactory Class

   Javax.crypto.SecretKeyFactory class merupakan class yang digunakan untuk mengkonversikan kunci yang digunakan pada proses enkripsi dan dekripsinya.

Seperti pada class JCE yang lainnya, SecretKeyFactory dibuat menggunakan metode getInstance ( ) method. Penulisan pada kode program pada java dituliskan :

private static String *METHOD* = "PBEWithMD5AndDES";

Penggunaan kunci *password* yang berbeda pada proses enkripsi dengan menggunakan data plainteks yang sama akan dihasilkan data chiperteks yang berbeda pula. Gambar 7 menunjukkan simulasi hasil data chiperteks yang berbeda dari 2 buah proses enkripsi menggunakan data plainteks yang sama dan dengan kunci *password* yang berbeda.

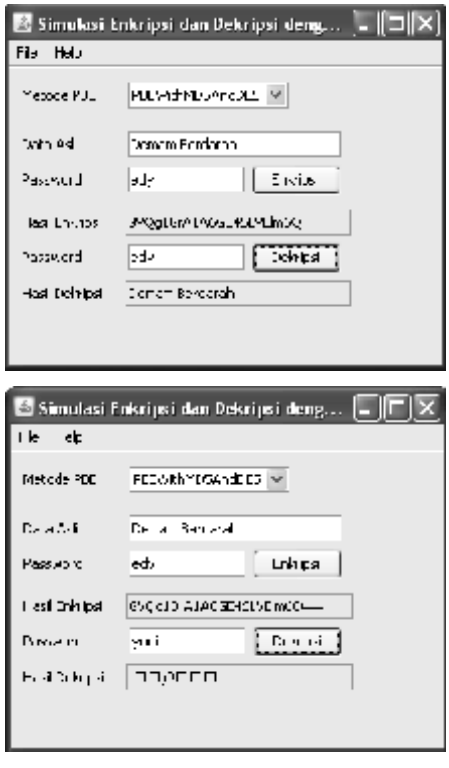


Gambar 4. Hasil enkripsi dengan kunci *password* berbeda

Data plainteks akan diproses menjadi data chiperteks melalui proses pengolahan pada proses enkripsi menggunakan metode *Password Based Encryption.* Data plainteks akan diolah dengan kunci *password*, *salt* dan *iteration count* sehingga akan didapatkan data terenkripsi yang berupa chiperteks. *Salt* digunakan untuk menambah sebuah *string* dari *byte-byte* yang random pada *password. Iteration count* digunakan untuk menambah perhitungan waktu yang dibutuhkan untuk mengkonversi sebuah

password menjadi sebuah kunci. Data plainteks dan kunci tersebut akan diproses pada proses *chiper* untuk menghasilkan *encrypted data* yang disebut dengan chiperteks.

Pada proses dekripsi, data chiperteks yang merupakan hasil pengolahan proses enkripsi dapat dikembalikan lagi menjadi data plainteks atau data asli sebelumnya. Proses ini merupakan kebalikan dari proses enkripsi. *Encrypted data* yang berupa data chiperteks bersama-sama dengan *password* akan diproses untuk menghasilkan data plainteks kembali. *Password, salt, iteration count*, dan chiperteks akan diubah menjadi sebuah kunci dan akan diproses menjadi data plainteks seperti semula. Pada proses dekripsi, *password* akan sangat menentukan pada hasil proses pengembalian data chiperteks menjadi data plainteks. *Password* yang sama dengan *password* saat proses enkripsi akan dapat mengembalikan data chiperteks menjadi data plainteks aslinya. Gambar 5 memperlihatkan simulasi perbedaan data hasil dekripsi antara *password* yang benar dan yang salah pada proses aplikasi.

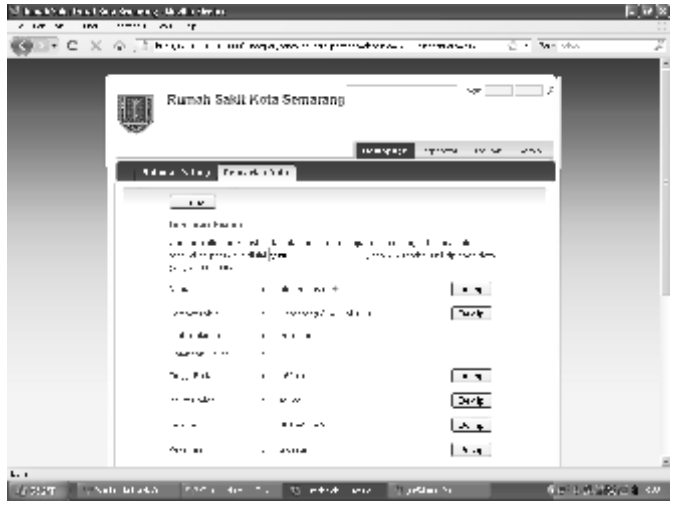


Gambar 5. Hasil dekripsi menggunakan *password* berbeda

3.7. Pembahasan Implementasi Enkripsi dan Dekripsi pada Web

Aplikasi penerapan kriptografi *Password Based Encryption* untuk keamanan distribusi data menggunakan data base XML yang dibuat dalam penelitian ini adalah sebuah implementasi program pada pertukaran data rekam medis pasien di sebuah web rumah sakit dengan dilakukan proses enkripsi dan dekripsi pada bagian *user*, *operator* dan dokter. Operator dan dokter dapat melakukan proses enkripsi dan dekripsi serta menyimpannya untuk digunakan pada bagian yang lain. Sedangkan user atau pasien hanya dapat melakukan proses dekripsi dari data yang telah dienkripsi oleh *operator* dan dokter.

Pada bagian operator, enkripsi dan dekripsi dilakukan untuk menyembunyikan data informasi pasien yang dapat dipilih sesuai dengan tag XML yang dipilih dan juga untuk mengembalikan data informasi pasien yang telah dienkripsi. Sedangkan pada bagian dokter, enkripsi dilakukan untuk menyembunyikan data pemeriksaan pasien dan mengembalikan data pemeriksaan pasien yang telah dienkrip. Hasil implementasi program pada web dapat dilihat pada gambar 6 di bawah ini:



Gambar 6. Hasil Implementasi Program pada web

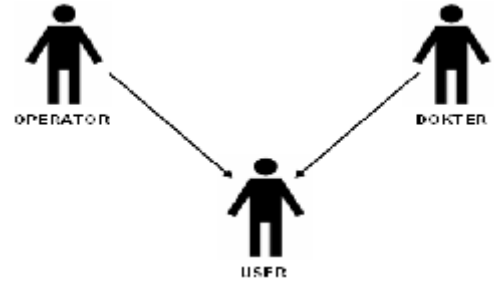
3.8. Analisa Sistem Keamanan Pada Implementasi Web

Implementasi yang dibuat merupakan sebuah web rumah sakit yang merupakan sebuah sistem informasi pelayanan rumah sakit terhadap

kerahasiaan data rekam medis pasien. Implementasi ini dilengkapi dengan menu enkripsi dan dekripsi untuk menyembunyikan atau merahasiakan data. Data rekam medis seorang pasien menjadi sangat penting untuk dijaga kerahasiaannya yaitu untuk melindungi data rekam medis pasien dari pihak yang tidak berwenang untuk mengaksesnya. Kerahasiaan data dapat dibaca dengan memasukkan *password* yang telah disepakati sebelumnya. Data chiperteks hasil enkripsi dapat didekripsi menjadi sebuah data plainteks kembali dengan menggunakan kunci *password* tersebut.

Seorang *operator* dapat memasukkan data rekam medis pasien dan dapat melakukan enkripsi data sesuai yang diperlukan. Oleh dokter, data tersebut kemudian diproses untuk dilakukan penambahan data riwayat rekam medis pasien sesuai hasil pemeriksaan. Dokter tidak dapat membaca seluruh data yang telah dienkripsi oleh *operator*, karena dokter tidak memiliki kunci *password* untuk mendekripsi data dari *operator*. Dokter memiliki hak untuk melakukan proses enkripsi pada data riwayat rekam medis pasien yang diperiksanya. Kunci *password* yang digunakan dokter untuk melakukan proses enkripsi ini berbeda dengan kunci *password* yang diberikan oleh *operator* saat melakukan proses enkripsi saat memasukkan data pasien. Demikian juga sebaliknya, *operator* tidak memilik hak untuk mengakses data pasien pada bagian yang diisikan oleh dokter. *Operator* tidak mempunyai kunci *password* yang dibuat oleh dokter. Satu-satunya yang dapat mengakses untuk mendekripsi data yang diisikan oleh pihak *operator* dan dokter adalah pasien atau *user*.

Pasien atau *user* mendapatkan 2 buah kunci *password* yang masing – masing diberikan oleh pihak *operator* dan pihak dokter. Kunci *password* pertama diberikan oleh *operator* untuk mendekripsi data yang telah dienkripsi oleh pihak *operator*, dan kunci *password* yang kedua diberikan oleh dokter untuk mendekripsi data yang telah dienkripsi oleh pihak dokter. Hubungan distribusi pembagian kunci dapat dilihat pada gambar 7 di bawah ini.



Gambar 7. Hubungan distribusi pembagian kunci *password*

Dari sistem distribusi kunci *password* diatas dapat dilihat bahwa hanya *user* yang memiliki 2 buah kunci *password* untuk digunakan pada proses dekripsi data yang telah dienkripsi oleh pihak *operator* ataupun pihak dokter. Pihak *operator* tidak mempunyai kunci *password* dari pihak dokter, sehingga *operator* tidak bisa melakukan proses dekripsi pada data yang telah dienkripsi oleh dokter. Demikian pula pihak dokter tidak bisa melakukan proses dekripsi pada data yang telah dienkripsi oleh pihak *operator*, sehingga pihak dokter tidak dapat melakukan proses dekripsi pada data yang telah dienkripsi oleh pihak *operator*.

Dari segi keamanan distribusi data, sistem ini bisa diaplikasikan untuk menyimpan dan menyembunyikan data-data rahasia dari seorang pasien pada sebuah rumah sakit agar tidak dapat diakses oleh pihak yang tidak berkepentingan.

4. Kesimpulan dan Saran

4.1 Kesimpulan

Dari hasil penelitian dan pembahasan yang telah dilakukan dapat diambil kesimpulan sebagai berikut.

1. XML *database* Xindice dapat digunakan sebagai sebuah sistem basis data khusus untuk menyimpan, mengolah dan mengakomodasi data-data XML yang memiliki struktur data tertentu.
2. Dengan digunakannya *server* basis data XML maka aplikasi tidak perlu melakukan pemetaan dari data XML ke struktur data tertentu. Aplikasi dapat langsung menyimpan data XML, dan mendapatkan kembali data XML dalam format yang tidak memerlukan perubahan.

3. Dengan menggunakan XML database Xindice akan dapat mempermudah penyimpanan dan modifikasi data XML, karena s*erver* Xindice dibangun secara modular, sehingga sangat mudah untuk menambah atau menghapus komponen agar *server* dapat digunakan sesuai dengan kebutuhan.
4. Aplikasi penerapan kriptografi dapat dilakukan pada dokumen yang memiliki format data XML yaitu dengan cara melakukan proses enkripsi dan dekripsi pada tag / bagian tertentu sesuai yang diinginkan.

4.2 Saran

Dari hasil penelitian yang telah dilakukan maka penulis dapat memberikan beberapa saran untuk penelitian berikutnya, yaitu:

1. Membuat perbandingan antara XML *database* Xindice dengan sistem basis data yang lain dan mengimplementasikan pada kasus yang lebih luas.
2. Membuat implementasi program yang lebih kompleks dan variatif untuk mencakup sistem distribusi data yang lebih besar, dan tidak hanya digunakan pada distribusi data rekam medis di sebuah rumah sakit tapi juga bisa dikembangkan pada instansi-instansi dan lembaga yang lain yang membutuhkan.
3. Sistem aplikasi dapat dikembangkan menjadi sebuah sistem yang memiliki kehandalan dalam sistem keamanan jaringan, sehingga distribusi data akan lebih aman dari serangan *hacker*.

Daftar Pustaka

[1] Abdul Kadir, 2004, *Dasar Pemrograman Web Dinamis Dengan Java Server Page (JSP)*. Penerbit Andi, Yogyakarta.

[2] Bruce S., 1996, *Applied Cryptography, Protocols, Algorithms,and Source Code in C.* John Wilwey & Sons, Inc

[3] David Hook, 2005, *Beginning Cryptography With Java.* Wiley Publishing,Inc, Indianapolis, USA

[4] Hartono, 2003, *Pemakaian Kriptografi Kunci Publik Denagn Algoritma RSA Untuk Keamanan Data XML*. Universitas Gadjah Mada Yogyakarta.

[5] John E.S., alih bahasa: Dwi Prabantini, 2002, *Just XML*. Penerbit Andi,Yogyakarta.

[6] Noprianto, 2004, *Mengenal XML.* Info Linux 12/2004

[7] Stallings W., 1999, *Cryptography and Network Security Principles and Practice second edition*. Prentice Hall, New Jersey, USA

[8] Theodore W. Leung, 2004, *Professional XML Development with Apache Tools: Xerces, Xalan, FOP, Cocoon, Axis, Xindice*, Wrox Press, Wiley Publishing,Inc, Indianapolis, USA

[9] Umniati, Mukodim, 2002, *Perbandingan Algoritma dalam Enkripsi antara Conventional Cryptosystems dan Public Key Cryptosystems.* Proceeding, Komputer dan Sistem Intelijen (KOMMIT 2002)

[10] Wrox Books, 2002, "*Profesional XML Databases*",http"//tutorials.freeskills.com/read/category/81/id/137