Jurnal Masyarakat Informatika (Jumanji)
Volume 01 Nomor 01 Oktober 2017

Model Enkripsi XML Pada *Output* DFXML untuk Pengamanan *Metadata* Bukti Digital

Danar Cahyo Prakoso
Magister Informatika, Forensika Digital
Fakultas Teknologi Industri
Universitas Islam Indonesia
Jl. Kaliurang KM 14,5 Yogyakarta
danarcahyop@gmail.com

Yudi Prayudi
Pusat Studi Forensika Digital
Fakultas Teknologi Industri
Universitas Islam Indonesia
Jl. Kaliurang KM 14,5 Yogyakarta
prayudi@uii.ac.id

***Abstrak*— DFXML (Digital *Forensics* XML) adalah sebuah *tool* forensik yang dikembangkan untuk menghasilkan *output* dalam bentuk dokumen XML. *Tools* ini dirancang untuk menampilkan *metadata* dari *file* hasil *disk imaging* dari perangkat elektronik. Umumnya *output* DFXML berupa dokumen XML dalam bentuk *plaintext.* Hal ini memunculkan permasalahan dalam aspek keamanan data, yaitu bentuk *plaintext* dari dokumen XML memungkinkan dibaca dengan mudah oleh setiap orang. Untuk itu diusulkan pendekatan XML *security* sebagai solusi untuk keamanan dokumen XML hasil dari DFXML. Solusi yang diusulkan adalah dalam bentuk *automatic encryption tool* yang mampu melakukan enkripsi dokumen XML secara fleksibel dan otomatis. Usulan ini masih bersifat model yang dapat menunjukkan bahwa konsep enkripsi XML yang nantinya dikembangkan mampu meningkatkan keamanan informasi data pada *output plaintext* dokumen DFXML dan memberikan kemudahan bagi siapa saja yang ingin melakukan enkripsi dari dokumen XML.**

Kata kunci— XML, Enkripsi, Tools, Digital Forensic, Metadata, DFXML

I. I. PENDAHULUAN

A. Latar Belakang

Kesulitan dalam merekap *metadata* bukti digital adalah salah satu tantangan dalam dunia forensika digital. Selain itu terdapat problem lain yaitu setiap *tools* untuk kepentingan pembacaan hasil akuisisi *disk imaging* memiliki keterbatasan serta menghasilkan *metadata* yang berbeda-beda. Untuk itu, seorang analis forensika digital memerlukan sebuah *tools* untuk dapat membantu memudahkan kesulitan tersebut. Salah solusi yang tersedia adalah memanfaatkan bantuan DFXML *tools* yang dikembangkan oleh Garfinkel [1].

DFXML (*Digital Forensics XML*) adalah bahasa XML yang dirancang untuk menampilkan *metadata* berbagai *file* hasil *disk imaging. Tools* ini akan menghasilkan abstraksi data yang dapat digunakan untuk kebutuhan pembacaan hasil *disk imaging* oleh *tools* forensika lainnya. DFXML memungkinkan untuk berbagi informasi yang terstruktur diantara *tools-tools* lainnya yang saling terorganisasi. DFXML *tools* merupakan modul berbasis Python (dfxml.py) dan dapat dengan mudah dikembangkan untuk membuat program analisis forensik lainnya melalui modifikasi bahasa *script* [1]. Dengan demikian DFXML akan memfasilitasi para pengembang aplikasi forensika lainnya untuk memanfaatkan data-data hasil proses *disk imaging* sesuai dengan kebutuhan aplikasi yang dikembangkannya.

Keluaran dari DFXML akan menghasilkan dokumen XML yang berisi informasi dengan baris kode dalam jumlah yang sangat banyak. Untuk memudahkan pembacaan dokumen XML tersebut maka dalam penelitian sebelumnya telah dikembangkan solusi *tools web-based* yang mampu menampilkan dan menambahkan informasi *metadata* tambahan serta menyimpannya ke dalam *file* XML yang lain. *Tool* tersebut dikembangkan antara lain untuk memenuhi kebutuhan *investigator* dalam membaca data hasil proses *disk imaging* menggunakan DFXML. Selain itu, informasi yang dihasilkan dari DFXML sangatlah terbatas, sementara proses investigasi memerlukan

penambahan informasi forensik lainnya yang umumnya tidak tercover saat proses *disk imaging,* seperti penambahan *metadata case number, evidence number, unique description, investigator* dan lainnya. Untuk itu *tools* yang dikembangkan memberikan solusi untuk pembacaan dan penambahan informasi *metadata* yang diperlukan [2].

Namun di sisi lain, keluaran *tools web-based* tersebut masih berupa *plaintext* sehingga siapa saja bisa dengan mudah melihat atau bahkan memodifikasi isi dari dokumen XML tersebut dengan menggunakan *tools text editor*. Risiko keamanan dokumen atau informasi menjadi sesuatu yang sangat penting untuk diperhatikan untuk diamankan [3]. Lalu lintas data dalam jaringan, akan menimbulkan risiko keamanan, baik risiko dari pencurian, modifikasi ataupun akses tidak sah bagi orang yang tidak mempunyai hak akses [4].

Untuk mengatasi hal tersebut itu maka dalam penelitian ini dikembangan model *tools* yang mampu melakukan pengamanan dokumen XML melalui pendekatan enkripsi. Selanjutnya paper ini akan membahas tentang prinsip dasar DFXML, keamanan XML serta usulan model konsep pengamanan *output* DFXML.

II. DFXML (DIGITAL FORENSIC XML)

Metadata sangatlah penting untuk proses investigasi forensik. Kualitas *metadata* akan meningkatkan kualitas pada hasil analisis [5]. DFXML memberikan salah satu solusi untuk melakukan pembacaan *metadata* yang didapat dari *imaging* sebuah perangkat elektronik. DFXML adalah kumpulan kode XML yang mampu merangkum *metadata* informasi *file*. DFXML dapat menggambarkan dan menampilkan *metadata* perangkat penyimpanan, *metadata file*, kumpulan *hash*, dan asal usul dokumen yang digambarkan dengan *tag-tag* XML [1]. Pertukaran informasi antara aplikasi satu dan lain dalam bentuk dokumen XML dapat diproses secara baik dan lebih optimal ketika dokumen XML tersebut disimpan dengan tepat [6].

Digital *Forensics* XML (DFXML) adalah sebuah bahasa XML yang memungkinkan terjadinya perubahan struktur informasi forensik. DFMXL dapat menampilkan informasi data untuk kepentingan investigasi forensik, menampilkan lokasi dari *file system*, *file*, entri Microsoft Windows *Registry*, JPEG EXIF, dan informasi teknis lainnya yang dapat dimanfaatkan untuk kepentingan analis forensika digital. [1].

Penggunaan DFXML adalah untuk mengembangkan *tools* yang memiliki kemampuan untuk mendeskripsikan informasi forensik seperti, *hashing* kriptografi, informasi forensik lokasi *file* pada *hard drive*, dan *metadata* nama *file* dan *timestamp* [1]

III. XML DAN XML ENCRYPTION

A. XML

XML adalah salah satu bahasa *Markup Language* disederhanakan dari SGML (*Standard Generalized Markup Language*). XML dikembangkan oleh W3C yang mempunyai tujuan untuk lebih menyempurnakan pada teknologi HTML yang telah menjadi dasar layanan berbasis *web* sekarang ini, dalam fungsinya XML memiliki dua fungsi yaitu sebagai format dokumen dan format pertukaran data pada sebuah sistem [7].

B. XML Encryption

Enkripsi adalah salah satu cara untuk menyimpan data kedalam bentuk yang sifatnya tidak terbaca dan tersandikan dalam *file* XML[7]. Dalam algoritma simetris Enkripsi dan Dekripsi menggunakan kunci yang sama untuk menyandikan atau membongkarnya [8]. Terdapat problem terkait pengamanan dokumen XML, khususnya dalam XML *Encryption* adalah, bagaimana cara menerapkan teknologi *Cryptography* ke dalam sebuah dokumen XML yang tidak mempengaruhi atau mengkontaminasi isi dari dokumen tersebut. Masalah ini timbul disebabkan karena XML merupakan sebuah dokumen data yang tersusun, sehingga setiap sistem yang mengandalkan pertukaran datanya melalui XML sangat membutuhkan data tersebut [7].

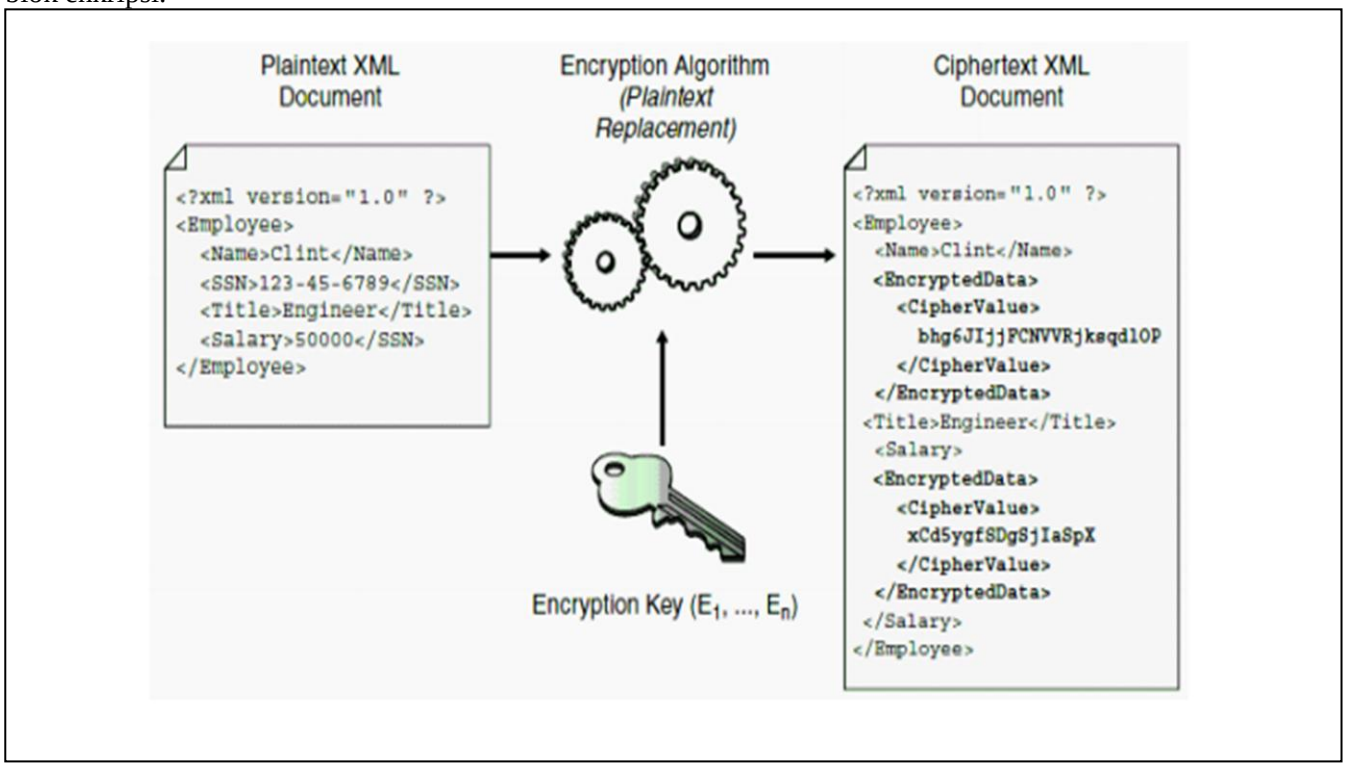
Teknologi *Cryptography* bertujuan untuk membuat segala bentuk informasi ke dalam bentuk data yang tersandi dan sifatnya rahasia agar tidak mudah dibaca oleh manusia. Oleh karena perbedaan kedua sifat inilah yang menyebabkan sebuah dokumen XML yang dienkripsi akan mengalami perubahan terhadap struktur yang sudah ada sebelumnya [7].

Menurut [7], ada 3 jenis aturan pemrosesan XML *encryption* yaitu aplikasi, penyandi *(Encryptor*) dan pembongkar sandi (*Decryptor*):

1) Aplikasi, istilah aplikasi berkaitan dengan suatu atau beberapa struktur yang digunakan untuk proses penerapan XML *Encryption*.

2) *Encryptor* (Penyandi), *Encryptor* berguna untuk membuat semua elemen XML *Encryption* tergabung dan aktif termasuk kunci, data yang tersandi, dan semua atribut. Selain itu juga berguna dalam hal penyimpanan kunci yang berguna untuk penyandian.

3) *Decryptor* (Pembongkar Sandi), berguna untuk melakukan proses *decoding* dan dekripsi pada data yang sudah tersandi sebelumnya kepada aplikasi untuk diproses lebih lanjut. Pada Gambar 1 adalah contoh potongan blok enkripsi.



Gambar 1. Contoh potongan blok enkripsi dari [7].

Enkripsi dapat didefinisikan untuk memproses data yang kemudian dikenal sebagai *cleartext* atau *plaintext* dan mengubahnya dengan menggunakan kunci kriptografi untuk menghasilkan *chipertext*, yang tidak dapat dikenali oleh pihak yang tidak mempunyai hak atau wewenang. Sebaliknya Dekripsi merupakan proses untuk mengubah *chipertext* menjadi *cleartext* kembali agar dapat dibaca [9].

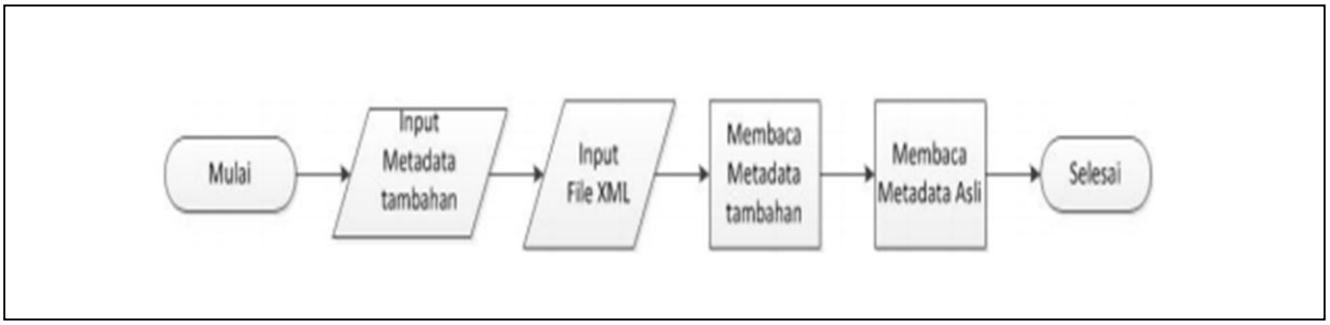
IV. RISET SEBELUMNYA

Pada kesempatan sebelumnya sudah pernah dilakukan penelitian tentang *output* DFXML tersebut, dalam penelitian tersebut melakukan rekayasa terhadap *output* XMLnya yaitu membuat sebuah sistem yang mampu melakukan penambahan *metadata* untuk kepentingan aktivitas forensik dan menampilkannya kembali ke dalam bentuk *web based*.

Penambahan *metadata* ini menjadi penting karena dalam proses *disk imaging* umumnya informasi *metadata*nya terbatas, demikian pula pada *output* DFXMLnya sifatnya juga terbatas. Informasi seperti *investigator, case* dan lain sebagainya yang belum tercover dalam output DFXML menjadi hal yang perlu ditambahkan untuk kepentingan aktivitas forensik.

Dalam penelitian tersebut diusulkan sebuah solusi untuk dapat melakukan penambahan *metadata* pada *output* DFXML dan menampilkannya kedalam bentuk *web-based.* Usulan tersebut diimplementasikan ke dalam sebuah sistem berbasis *web*, sistem tersebut dapat menambah informasi *metadata* secara fleksibel sesuai dengan kebutuhan pengguna.

Gambar 2 menunjukan alur penggunaan sistem yang sudah pernah dikembangkan pada riset sebelumnnya. Alur pertama yaitu pengguna dapat menentukan *metadata* apa saja yang akan ditambahkan dan memasukkannya kedalam *field* yang ada pada sistem secara fleksibel sesuai kebutuhan pengguna.

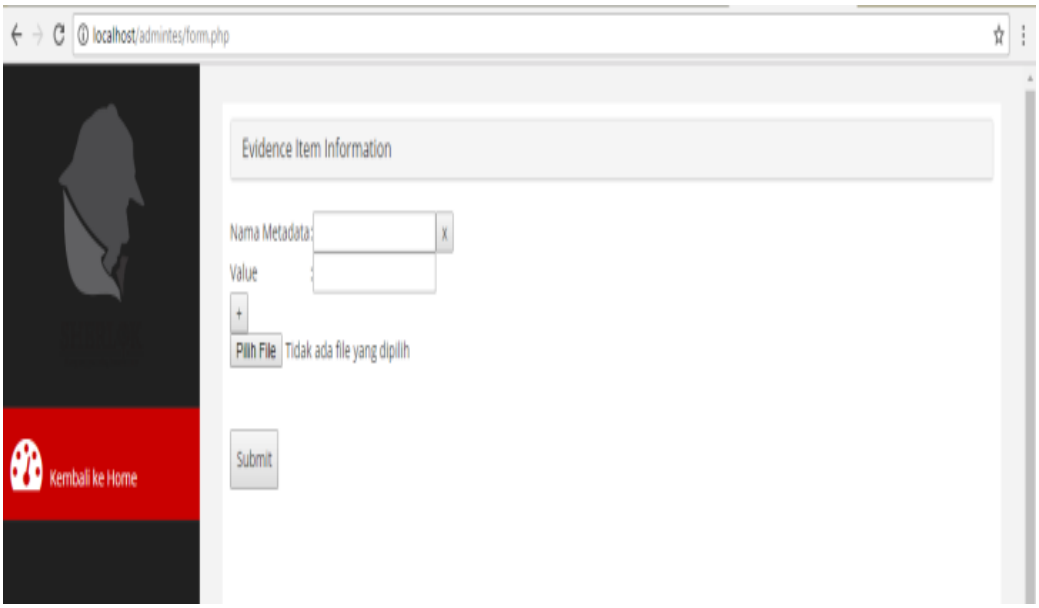


Gambar 2. Alur sistem pada riset sebelumnya [2]

Setelah menentukan dan memasukkan *metadata* yang diinginkan, selanjutnya adalah memasukkan File XML hasil DFXML. Sistem akan melakukan penambahan *metadata* yang dimasukkan tersebut secara otomatis kedalam dokumen XML yang baru.

Sistem tersebut akan menambahkan *metadata* ke dalam dokumen XML yang baru, namun tidak mengurangi *metadata* lain pada dokumen XML aslinya. Tambahan *metadata* tersebut berupa penambahan tag XML sesuai dengan apa yang sudah pengguna tentukan sebelumnya yang berguna untuk kepentingan aktivitas digital forensik.

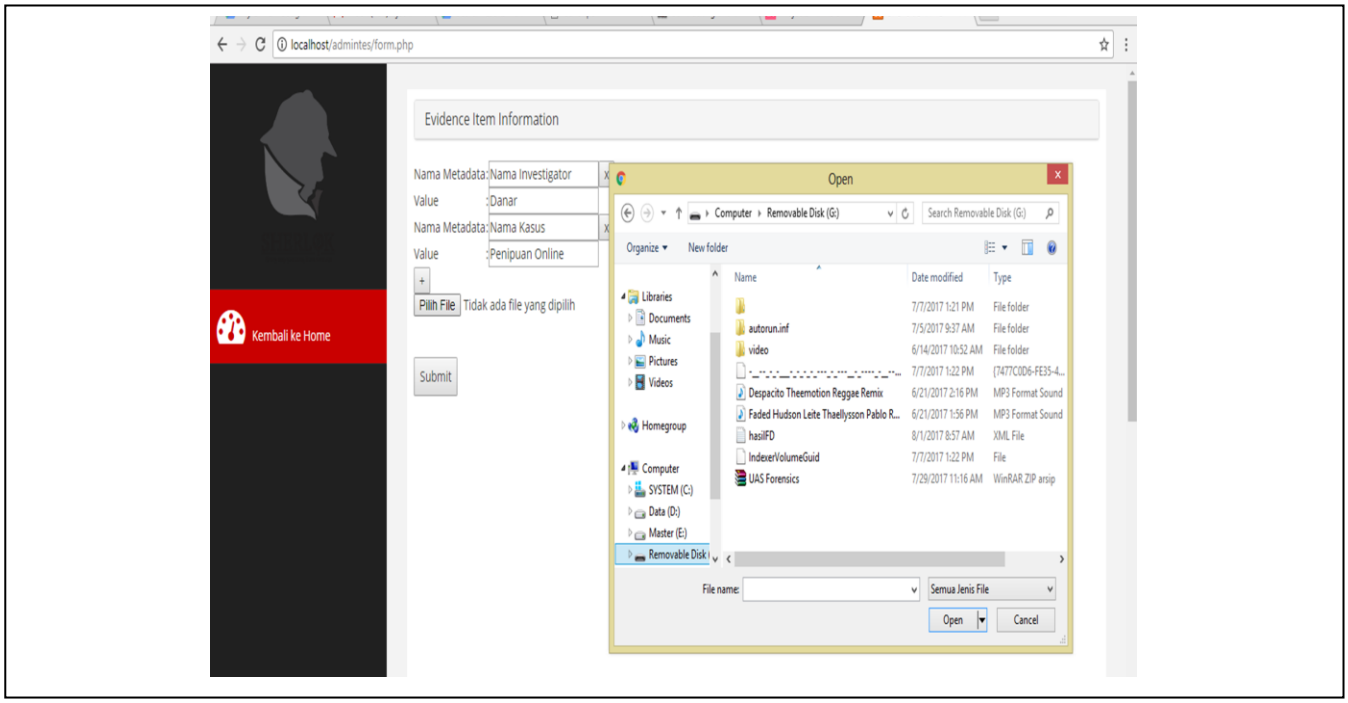
Berikut akan dijelaskan mengenai sistem yang sudah pernah dikembangkan sebelumnya. Gambar 3 menunjukkan tampilan pada sistem yang dikembangkan pada halaman ini terdapat dua *field* yaitu Nama Metadata dan Value dimana dua *field* ini bisa ditambah secara fleksibel tergantung kebutuhan *user* dengan cara menekan tombol + (plus) yang sudah tersedia. Nama metadata dan value yang ditambahkan ini akan disisipkan ke file XML hasil dari DFXML [2].



Gambar 3. Halaman untuk memasukan metadata tambahan [2]

Selain terdapat fungsi untuk menambah *field* tersebut, juga terdapat tombol × (delete) yang berfungsi untuk menghapus *field* jika pengguna ingin mengurangi *field* yang sudah terlanjur ditambahkan [2].

Setelah mengisi form sesuai kebutuhan, maka langkah selanjutnya adalah memilih file XML hasil DFXML dengan cara klik tombol Pilih file. Pada Gambar 4 berikut menunjukkan proses untuk memilih dokumen XML yang akan ditambahkan *metadata*nya

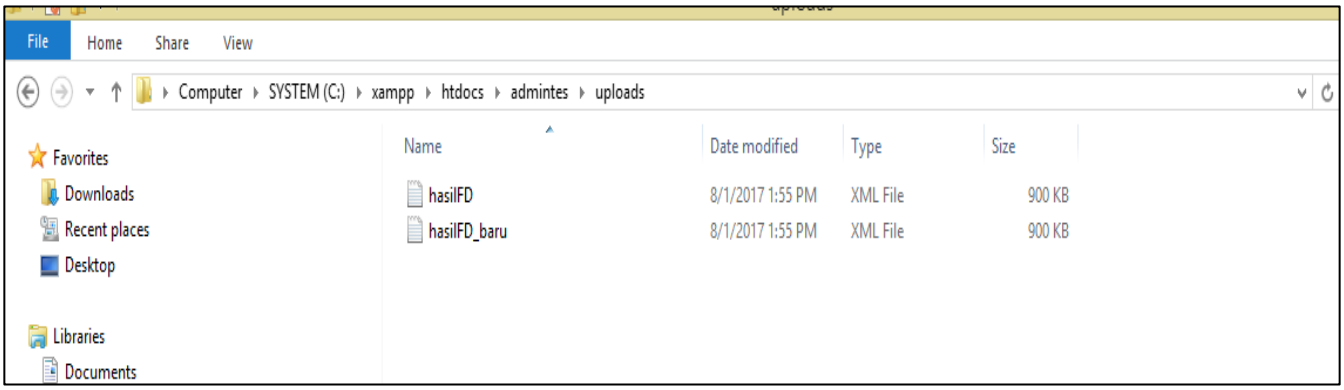


Gambar 4. Proses memilih dokumen XML yang akan ditambah *metadata* [2]

Selanjutnya sistem akan menyisipkan *metadata* tambahan tersebut kedalam dokumen XML baru tanpa mengurangi *metadata* asli. Setelah dokumen XML baru terbuat lengkap dengan *metadata* tambahan tadi maka sistem akan menampikan seluruh *tag* XML pada dokumen tersebut kedalam bentuk tabel untuk mempermudah pembacaanya, seperti terlihat pada Gambar 5.

Gambar 5. Halaman tabel [2]

Dokumen XML asli maupun dokumen XML yang sudah direkayasa akan tersimpan pada direktori server *localhost* pada komputer. Seperti terlihat pada gambar 6 berikut

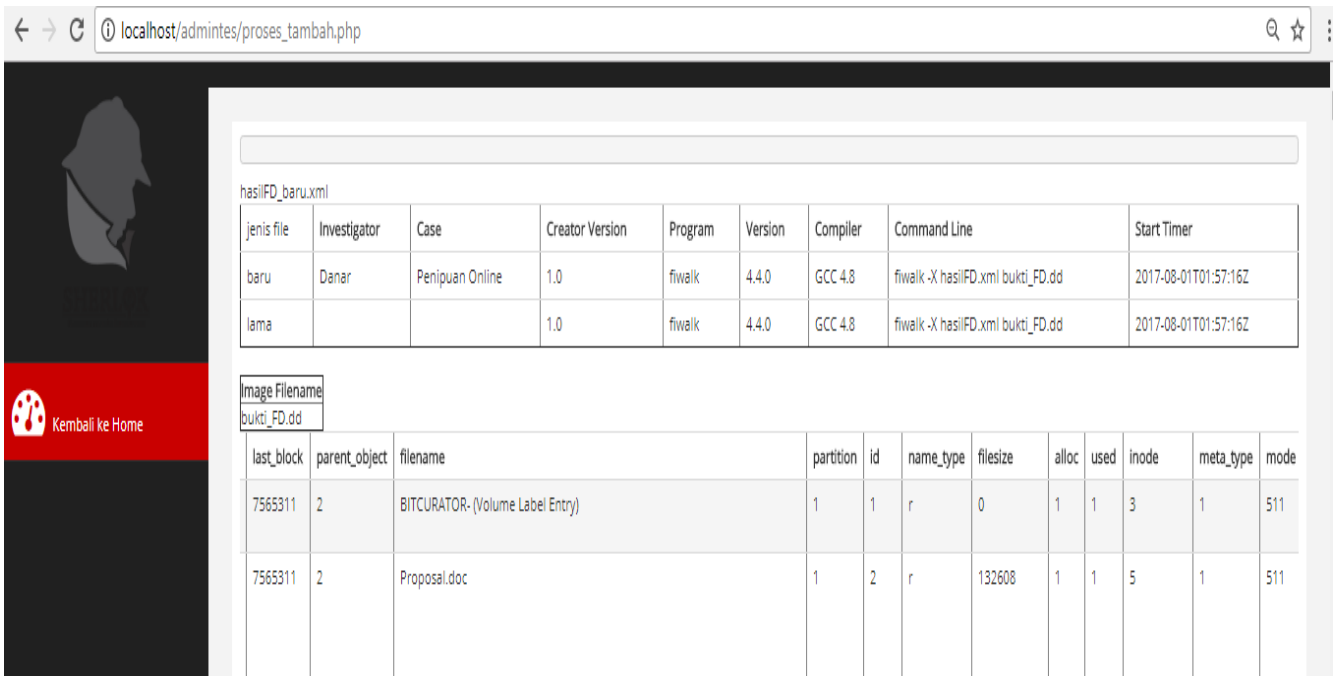


Gambar 6. Direktori penyimpanan dokumen XML [2]

Untuk membedakan antara dokumen XML asli dan mana dokumen XML yang sudah direkayasa sistem yang dikembangkan ini akan menambahkan keterangan di akhir nama dokumen XML dengan penambahan keterangan '_baru' secara otomatis.

Terlihat pada Gambar 6 yang menunjukan direktori pada server *localhost* dalam direktori tersebut terdapat dua dokumen XML yaitu hasilFD dan hasilFD_baru. Dokumen hasilFD adalah dokumen asli yang belum dilakukan penambahan *metadata* sebelumnya, pada dokumen yang kedua terdapat dokumen hasiFD_baru yang merupakan dokumen baru yang sudah ditambakan *metadata* sesuai dengan apa yang pengguna masukkan sebelumnya.

Kemudian ketika membuka kedua dokumen XML tersebut dengan menggunakan *text editor* maka hasilnya adalah terlihat pada Gambar 7 sebagai berikut, adalah potongan XML dari dokumen yang asli dan belum ditambahkan *metadata* tambahan.



Gambar 7. Potongan XML dari dokumen yang asli [2]

Pada gambar 7 yang merupakan dokumen XML asli keluaran DFXML. Dalam elemen *tag* <metadata> hanya terdapat satu *metadata* yaitu *Disk Image*, belum terdapat informasi tambahan lainnya.

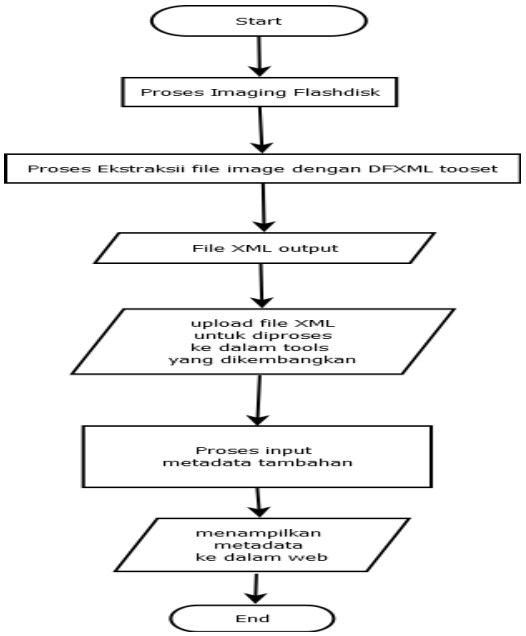
Setelah membuka dokumen asli keluaran DFXML, selanjutnya akan dibandingkan dengan dokumen XML baru yang sudah diproses oleh sistem seperti terlihat pada Gambar 8 berikut.

Gambar 8. File XML baru yang sudah ditambahkan metadata [2]

Sistem akan menghasilkan dokumen XML baru seperti terlihat pada Gambar 8, selanjutnya sistem akan menyisipkan pada elemen *parent tag* <*metadata*> dengan *metadata* baru yaitu *tag* <*Investigator*>dan <*Case*> lengkap dengan *value*nya yaitu 'Danar' yang berada diantara *tag* <*Investigator*> dan 'Penipuan Online' yang berada diantara *tag* <*Case*>. *Tag* ini adalah hasil dari *input* pengguna pada *field* yang ada sebelumnya. Pengguna bisa dengan fleksibel untuk menambah *tag* dan memberi nama *tag* tersebut sesuai kebutuhan yang diperlukan untuk kepentingan forensik.

V. PENGEMBANGAN MODEL

Cara kerja penggunaan DFXML dapat dilihat pada Gambar 9 sebagai berikut.

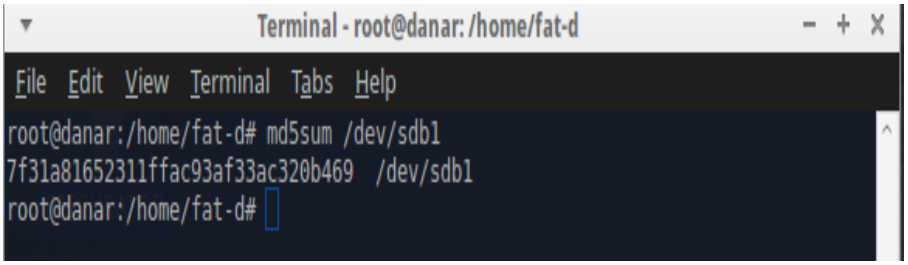


Gambar 9. Diagram Alur Proses Bisnis Sistem yang Pernah dikembangkan

Dalam forensika digital proses akuisisi dan *disk imaging* adalah hal yang penting dilakukan. Hal ini bertujuan untuk menjaga integritas barang bukti digital. Ketika melakukan proses analisis terhadap barang bukti digital, seorang *examiner* dilarang mengakses langsung barang bukti digital karena dikhawatirkan akan mengkontaminasi barang bukti yang ada.

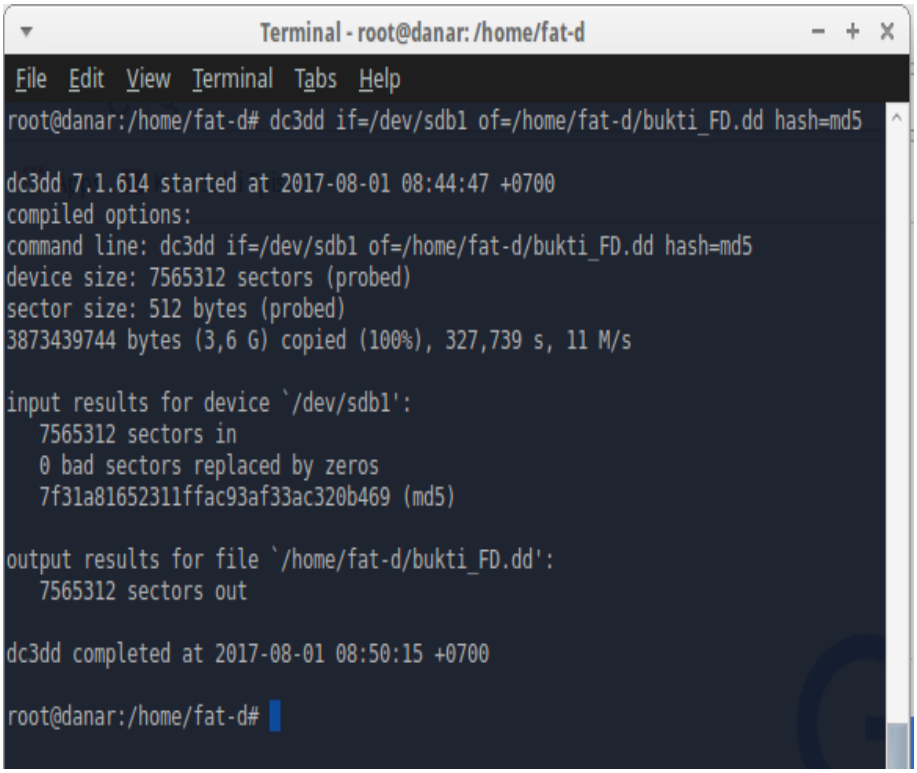
A. Proses Akuisisi

Pada Gambar 10 berikut ini adalah sejumlah langkah proses akuisisi barang bukti berupa *flashdisk* 4GB. Pada saat proses disk *imaging* perlu dilakukan terlebih dahulu pemeriksaan md5 *hash* dari *flashdisk* yang terdeteksi dengan cara mengetikkan perintah md5sum /dev/sdb1. Kode *hashing* yang muncul berfungsi untuk memastikan bahwa *flashdisk* tidak terkontaminasi.



Gambar 10. Proses melihat nilai *hash* pada flashdisk 4GB

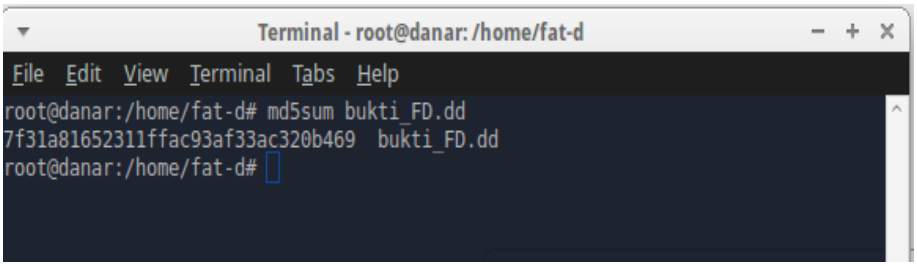
Pada Gambar 10 tersebut, terdapat keterangan md5 *hash* dari *flashdisk* yang diakuisisi ke dalam *file* dan hasilnya sama dengan pemeriksaan awal flashdisk yaitu sama-sama memiliki kode *hash* 7f31a81652311ffac93af33ac320b469.



Gambar 11. Proses melakukan akuisisi menggunakan tools dc3dd

Selanjutnya Gambar 11 menunjukkan cara untuk menjalankan aktivitas disk *imaging* melalui bantuan perintah dc3dd tools. Perintah ini dijalankan dengan cara: dc3dd diketikan pada *terminal*, diikuti dengan if=/dev/sdb1 yang merupakan informasi *source flashdisk* yang akan diakuisisi, kemudian of=/home/fat-d adalah *output* tempat *direktori* yang akan menampung hasil akuisisi tersebut. Selanjutnya /bukti_FD.dd adalah nama

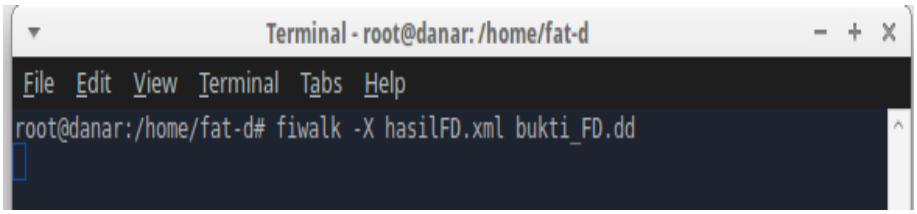
file hasil akuisisi, kemudian hash=md5 adalah jenis *hash* yang digunakan. Uraian langkah tersebut terlihat pada Gambar 12.



Gambar 12. Proses melihat nilai *hash* pada *file* akuisisi flashdisk 4GB

B. *Menggunakan DFXML tools*

Setelah mendapatkan *file image* maka tahap selanjutnya adalah mencoba ekstraksi *file image* tadi dengan menggunakan DFXML *tool*.



Gambar 13. Perintah menjalankan DFXML tools untuk ekstraksi *file* akuisisi *Flashdisk* 4GB

Petunjuk untuk menggunakan DFXML *tool* terlihat pada Gambar 13. Perintah yang pertama adalah mengetikan perintah Fiwalk–x, untuk menghasilkan XML *output*. Sedangkan hasilFD.xml adalah nama *file* hasil *output* dari proses tersebut, dan bukti_FD.dd adalah *file image* dari dc3dd *tool imager* yang akan diekstraksi dengan DFXML tool.

Setelah perintah itu dijalankan dengan baik, maka akan dihasilkan keluaran file XML yang akan menggambarkan *metadata* atau ekstraksi dari *flashdisk* tersebut.

```
</metadata>
<creator version='1.0'>
  <program>fiwalk</program>
  <version>4.4.0</version>
  <build_environment>
    <compiler>GCC 4.8</compiler>
  </build_environment>
  <execution_environment>
    <command_line>fiwalk -X hasilFD.xml bukti_FD.dd</command_line>
    <start_time>2017-08-01T01:57:16Z</start_time>
  </execution_environment>
</creator>
<source>
  <image_filename>bukti_FD.dd</image_filename>
</source>
```

Gambar 14. Contoh potongan xml hasil ekstraksi *file* akuisisi flashdisk

Potongan XML pada Gambar 14 menjelaskan informasi *command line* digunakan untuk menghasilkan *file* XML tersebut, kemudian juga nama *file imaging* hasil akuisisi yang diekstraksi lengkap disebutkan informasi *timestamp* kapan ekstraksi tersebut dilakukan.

Selain itu output XML tersebut juga akan memberikan informasi *file* apa saja yang ada didalam *flashdisk* tersebut. Sebagai contoh adalah pada Gambar 15.

```
<fileobject>
  <parent_object>
    <inode>2</inode>
  </parent_object>
  <filename>Proposal.doc</filename>
  <partition>1</partition>
  <id>2</id>
  <name_type>r</name_type>
  <filesize>132608</filesize>
  <alloc>1</alloc>
  <used>1</used>
  <inode>5</inode>
  <meta_type>1</meta_type>
  <mode>511</mode>
  <nlink>1</nlink>
  <uid>0</uid>
  <gid>0</gid>
  <mtime prec="2">2016-12-30T14:45:40</mtime>
  <atime prec="86400">2017-06-20T17:00:00</atime>
  <crtime prec="2">2017-03-04T20:43:52</crtime>
  <byte_runs>
   <byte_run file_offset='0' fs_offset='3474067456' img_offset='3474067456' len='132608'/>
  </byte_runs>
  <hashdigest type='md5'>ac9a38abcbd45a4cedd0f7d7c9646bbd</hashdigest>
  <hashdigest type='sha1'>3f081dcbd3b2a5781877f28126dba7656cc4645e</hashdigest>
</fileobject>
```

Gambar 15. Contoh potongan xml *output* DFXML

Dengan adanya DFXML para investigator bisa saling bertukar informasi *metadata* barang bukti digital tanpa harus bertukar *file* akuisisi.

Pada konsep yang sudah dijelaskan sebelumnya, terlihat bahwa *output file* XML yang bersifat *plaintext* bisa dengan mudah terlihat atau bahkan dimodifikasi. Maka pengembangan sistem yang dilakukan dalam penelitian ini adalah membangun model *automatic encryption tools* untuk meningkatkan keamanan metadata bukti digital.

VI. HASIL DAN ANALISIS

Penelitian kali ini mengusulkan sebuah model untuk melakukan enkripsi terhadap *file* XML. Model ini kemudian dikenal dengan nama *Automatic Encryption Tools*. Gambar 16 menujukkan Model *Automatic Encryption Tools* yang akan dikembangkan.

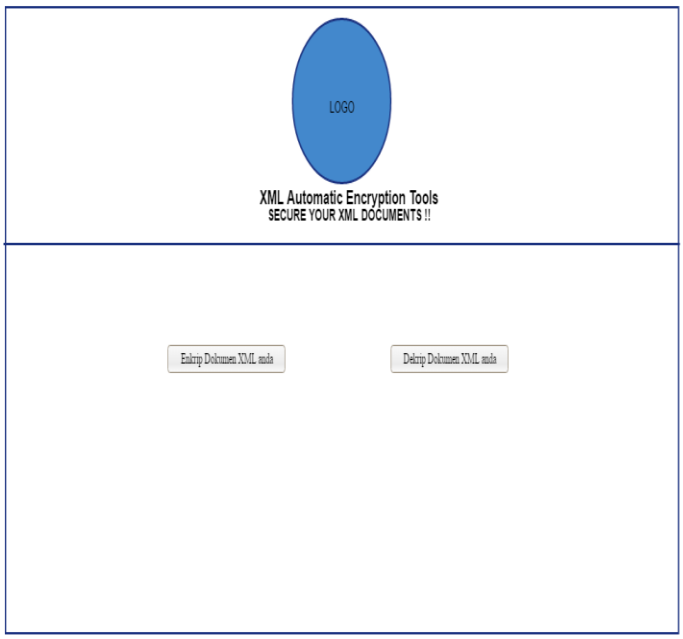
Gambar 16. *Flowchart Automatic encryption tools*

Merujuk pada penggunaan sistem yang telah dikembangkan sebelumnya, *output* DFXML adalah berupa *file* XML yang masih bersifat *plaintext*. Konsep usulan *automatic encryption tools* ini akan mempermudah dalam melakukan enkripsi XML *plaintext* tersebut.

Pengguna bisa melakukan penyandian berdasarkan *tag* elemen yang ada pada *file* XML tersebut, pada sistem *automatic encryption tools* tersebut akan dilakukan pembacaan seluruh *tag* elemen, kemudian pengguna dapat memilih elemen *tag* mana yang akan dilakukan penyandian sesuai dengan kebutuhan pengguna.

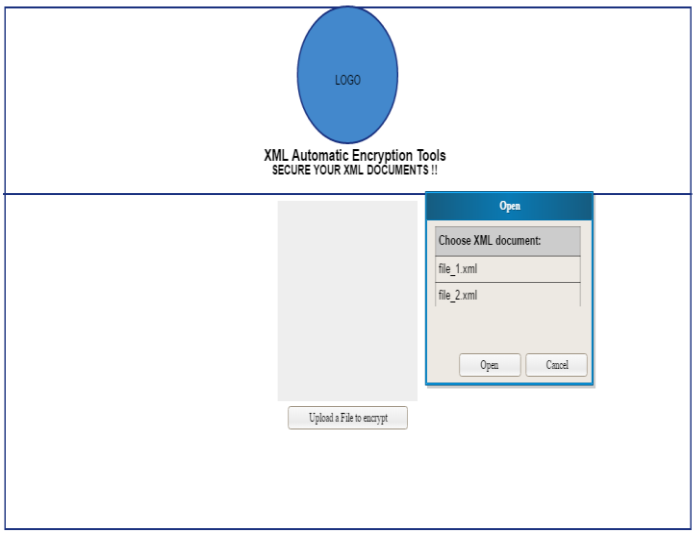
Selain fasilitas untuk memilih elemen *tag* mana yang akan dipilih, pengguna bisa melakukan penyandian terhadap keseluruhan isi dokumen XML berdasarkan *header tag* elemen yang ada pada dokumen XML tersebut.

Pada fungsi selanjutnya konsep *automatic encryption tools* ini juga terdapat pilihan dekripsinya, yaitu untuk melakukan pembongkaran sandi yang berguna untuk membaca *plaintext* XML yang sebelumnya sudah dilakukan penyandian.



Gambar 17. *Mockup Home Page Automatic encryption tools*

Gambar 17 adalah rancangan halaman utama, Pada halaman ini akan ditampilkan pilihan untuk melakukan Enkripsi atau Dekripsi dokumen XML

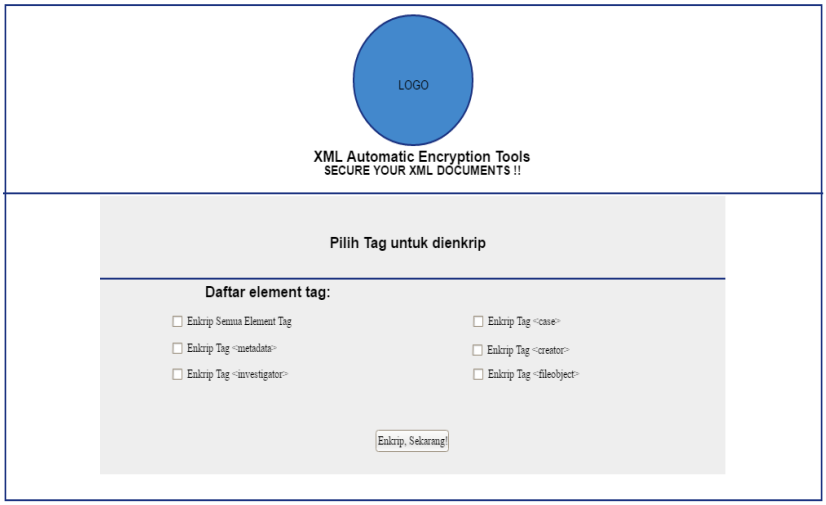


Gambar 18. *Mockup Upload Page Untuk Enkrip Dokumen XML*

Selanjutnya, Gambar 18 menunjukan proses upload dokumen XML yang hendak dienkripsi, setelah dokumen diupload *automatic encryption tools* akan melakukan pembacaan dan melakukan *listing tag* apa saja yang terdapat pada dokumen tersebut.

Pada model tersebut, Informasi *Tag* akan muncul secara otomatis pada *automatic encryption tools*, adalah hasil pembacaan dari dokumen XML yang sudah diupload sebelumnya.

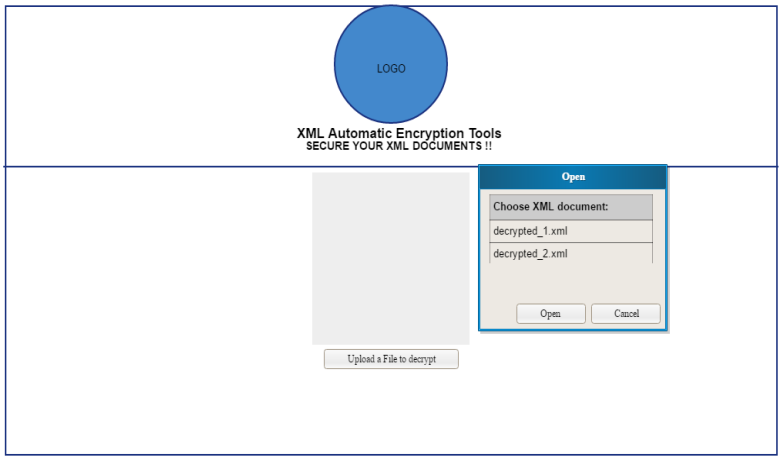
Pengguna dapat memilih elemen XML mana yang akan dienkrip dengan cara men*check-list* terhadap *tag-tag* yang terbaca oleh sistem. Setelah melakukan pemilihan elemen mana yang akan dienkrip maka langkah selanjutnya adalah menekan tombol enkrip pada sistem.



Gambar 19. *Mockup listing* elemen *tag* XML

Setelah dilakukan upload *file* XML, maka *automatic encryption tools* akan melakukan *listing* tag yang ada pada dokumen XML, pada Gambar 19 adalah halaman daftar elemen *tag* hasil pembacaan *automatic encryption tools*.

Pada halaman tersebut pengguna bisa melakukan pemilihan *tag* mana saja yang saja akan dilakukan enkripsi. Setelah dilakukan *check-list* pilihan *tag* elemen yang akan dienkripsi, maka proses penyandian akan berlangsung dan dokumen XML yang sudah dienkripsi akan otomatis terunduh. *Automatic encryption tools* juga menyediakan fungsi untuk melakuan dekripsi dokumen XML yang sudah dienkripsi tersebut.



Gambar 20. *Mockup Upload Page Untuk Dekrip Dokumen XML*

Gambar 20 adalah halaman untuk *upload* dokumen XML yang sudah dienkrip sebelumnya menggunakan *automatic encryption tools*, setelah upload selesai maka proses dekripsi (pembongkaran sandi) dilakukan dan *automatic encryption tools* akan membongkar sandi pada dokumen XML yang sudah terenkrip tersebut sehingga dokumen XML bisa dibaca dan terbuka.

Konsep ini masih berbentuk model, belum melangkah lebih jauh dalam bentuk implementasi pemrograman. Penelitian ini juga dapat dijalankan dengan menggunakan satu algoritma enkripsi, *user* tidak mempunyai opsi untuk memilih algoritma enkripsi lainnya. Pemilihan algoritma yang tepat untuk kepentingan enkripsi masih perlu dilakukan kajian lebih lanjut. Secara prinsip model yang diusulkan telah memenuhi beberapa kebutuhan dasar dari permasalahan keamanan dokumen XML. Implementasi lebih lanjut dari model yang dikembangkan ini dapat dilakukan tidak hanya untuk kepentingan keamanan dokumen output dari DFXML namun juga dapat diterapkan pada dokumen XML apapun.

VII. KESIMPULAN

DFXML tool adalah sebuah alat yang bisa melakukan pembacaan metadata terhadap sebuah *file* akuisisi (*image file*) dari sebuah perangkat elektronik. *Output* dari pembacaan tersebut umumnya disajikan kedalam bentuk dokumen XML.

Pada penelitian sebelumnya telah dikembangkan sebuah sistem yang berguna untuk melakukan pembacaan dan rekayasa *metadata* terhadap dokumen XML. Keluaran dari DFXML *tool* yang berupa *file plaintext* yang sifatnya kurang aman. Untuk itu diusulkan dalam paper ini sebuah model *automatic encryption tools* yang akan memudahkan untuk melakukan enkripsi dokumen XML. Adanya enkripsi tersebut diharapkan akan meningkatkan keamanan dari dokumen XML yang berisi informasi penting dari sebuah barang bukti digital tersebut.

Penelitian ini menghasilkan sebuah model untuk kepentingan *automatic encryption tools* dan belum mencapai pada tahap implementasi, untuk itu saran untuk penelitian selanjutnya adalah dapat mengimplementasikan konsep tersebut dalam bentuk implementasi aplikasi sesungguhnya.

VIII. DAFTAR PUSTAKA

[1] S. Garfinkel, "Digital forensics XML and the DFXML toolset," *Digit. Investig.*, vol. 8, no. 3–4, pp. 161–174, 2012.

[2] Danar Cahyo Prakoso, *Pengembangan dfxml*. Yogyakarta: Fakultas Teknologi Industri, Teknik Informatika , Univesitas Islam Indonesia, 2015.

[3] F. Maruf, "Merging of Vigenére Cipher with XTEA Block Cipher to Encryption Digital Documents," *Ijca*, vol. 132, no. 1, pp. 27–33, 2015.

[4] Y. Prayudi and T. K. Priyambodo, "Study on Cryptography as a Service (CAAS)," *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, vol. 4, no. 10, pp. 150–156, 2014.

[5] A. J. Nelson, E. Q. Steggall, and D. D. E. Long, "Cooperative mode: Comparative storage metadata verification applied to the Xbox 360," *Digit. Investig.*, vol. 11, no. SUPPL. 2, pp. S46–S56, 2014.

[6] N. Palsetia, G. Deepa, F. Ahmed Khan, P. S. Thilagam, and A. R. Pais, "Securing native XML database-driven web applications from XQuery injection vulnerabilities," *J. Syst. Softw.*, vol. 122, pp. 93–109, 2016.

[7] B. Susanto, "Pemrograman XML Security," no. September, pp. 1–40, 1998.

[8] V. Sankar and G. Zayaraz, "Securing confidential data in XML using custom level encryption," *2016 Int. Conf. Comput. Power, Energy, Inf. Commun. ICCPEIC 2016*, pp. 227–230, 2016.

[9] T. Wahyuningrum, P. Studi, D. T. Telekomunikasi, A. Teknik, T. Sandhy, and P. Purwokerto, "IMPLEMENTASI XML ENCRYPTION (XML Enc) MENGGUNAKAN JAVA," vol. 4, pp. 17–28, 2012.