

**TUGAS MAKALAH  
KEAMANAN JARINGAN INFORMASI  
Dosen : DR. BUDI RAHARDJO**

**Digital Timestamping: Suatu Tinjauan Komprehensif  
dan Usulan Model Skema Implementasi**

Oleh :  
**Lala Septem Riza**  
**23205301**



**MAGISTER TEKNOLOGI INFORMASI  
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA  
INSTITUT TEKNOLOGI BANDUNG  
2006**

# Daftar Isi

Daftar Isi.....	i
<i>Digital Timestamping</i> : Suatu Tinjauan Komprehensif dan Usulan Model Skema Implementasi.....	1
Abstrak .....	1
1. Pendahuluan .....	1
2. Tinjauan <i>Digital Time Stamping</i> .....	2
2.1 Metode atau Algoritma <i>Timestamp</i> .....	4
2.2 Skema <i>Digital Time Stamping</i> .....	8
2.2.1 Skema <i>Trusted Third Party (TTP)</i> .....	9
2.2.2 Skema Terdistribusi ( <i>Trust Distributed</i> ).....	12
2.3 Metode Penentuan Waktu.....	12
3. Usulan Skema <i>TSA Distributed</i> Untuk Implementasi <i>Digital TimeStamping</i> .....	14
4. Kesimpulan.....	18
5. Diskusi dan Penelitian Lebih Lanjut.....	18
Daftar Pustaka .....	19

# ***Digital Timestamping: Suatu Tinjauan Komprehensif dan Usulan Model Skema Implementasi***

Lala Septem Riza  
23205301

## **Abstrak**

Saat ini, penentuan waktu pembuatan dan modifikasi suatu dokumen digital menjadi sesuatu yang diperhatikan misalkan pada aplikasi e-trading dan penentuan waktu pada hak cipta dan paten. Digital timestamping merupakan salah satu service yang menjaga integritas dengan cara menambahkan catatan waktu pada suatu dokumen. Tiga perhatian utama dalam *digital timestamping* yang akan dibahas dalam makalah ini adalah *cryptography hash function*, skema implementasi, dan metode penetapan waktu.

Pada Akhir makalah ini, diusulkan suatu model TSA *Distributed*. Model ini diharapkan memiliki tingkat kepercayaan yang lebih tinggi dibanding skema TTP dan lebih sederhana dibanding skema trusted distributed. Implementasi skema ini juga mangakomodir verifikasi dan claim terdapat timestamp dokumen pada TSA yang berbeda.

**Kata Kunci:** *Digital Timestamping, Cryptographic hash function, trust third party, trusted distributed, skema TSA Distributed*

## **1. Pendahuluan**

*Digital Time-stamping* merupakan suatu proses untuk menjaga integritas suatu dokumen dengan cara menambahkan catatan waktu ketika pembuatan atau modifikasi dari dokumen tersebut dilakukan. Untuk menggambarkan dengan lebih jelas mengenai *digital time-stamping* ini, diberikan dua pertanyaan umum terhadap suatu dokumen yaitu:

1. Siapakah penulis atau pemilik dokumen ini ?
2. Kapan dokumen ini dibuat dan kapan terakhir kali dokumen dimodifikasi ?

Jawaban dari pertanyaan pertama merupakan nama seorang, perusahaan, atau ID orang/perusahaan yang merepresentasikan suatu identitas diri. Sedangkan jawaban

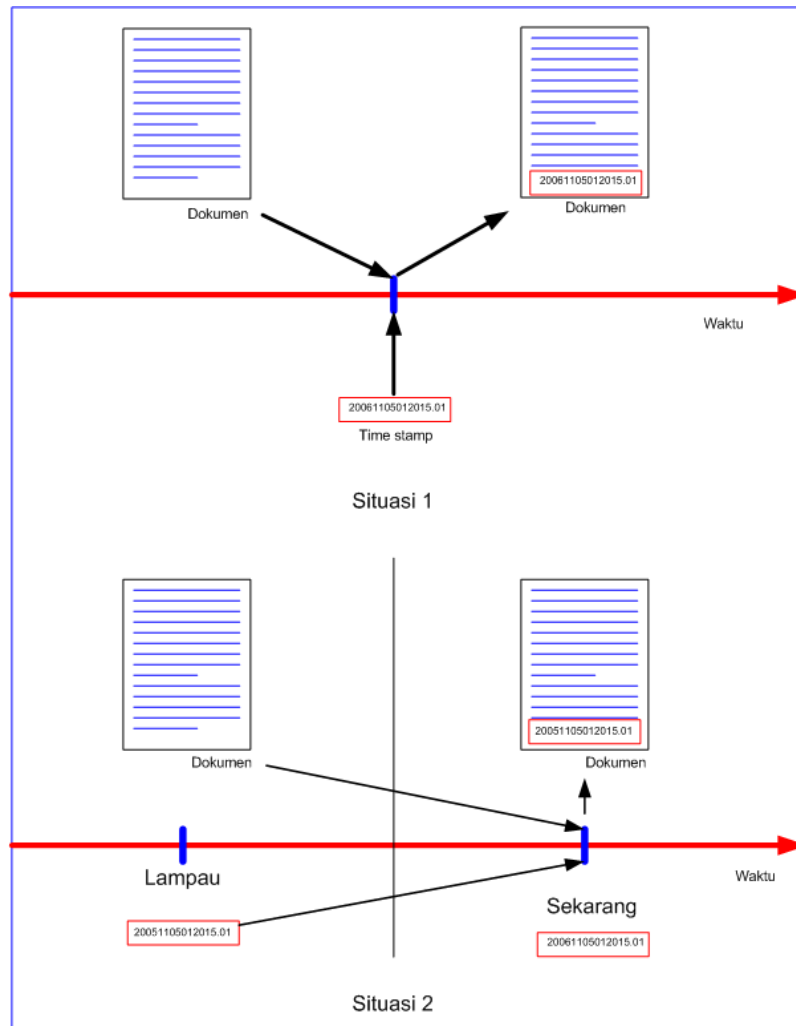
dari pertanyaan kedua adalah waktu dari pembuatan atau modifikasi dokumen, misalkan keterangan waktu dengan format *zulu time*: YYYYMMDDhhmmss[.s...]. *Digital signature* merupakan suatu teknik untuk mengimplementasikan jawaban dari pertanyaan pertama pada dokumen digital. Sedangkan *digital time-stamping* merupakan implementasi untuk jawaban pertanyaan kedua.

Penentuan waktu keberadaan dokumen digital sangat penting dalam beberapa aplikasi antara lain untuk penandatanganan kontrak secara elektronik dan *e-trading*. Aplikasi *digital time stamping* juga digunakan untuk memproteksi hak cipta dengan menentukan siapa yang menemukan pertama kali atas suatu temuan, aplikasi ini juga dapat digunakan sebagai claim atas temuan tersebut. Karena bersifat non repudiation, *digital timestamp* dapat pula digunakan sebagai barang/alat bukti.

Pada makalah ini, akan dibahas mengenai *digital time-stamping* yang meliputi tinjauan secara menyeluruh yang meliputi cryptography hash function, skema implementasi dan penetapan waktu untuk *timestamp* serta usulan suatu skema baru dalam implementasi *digital timestamping* ini.

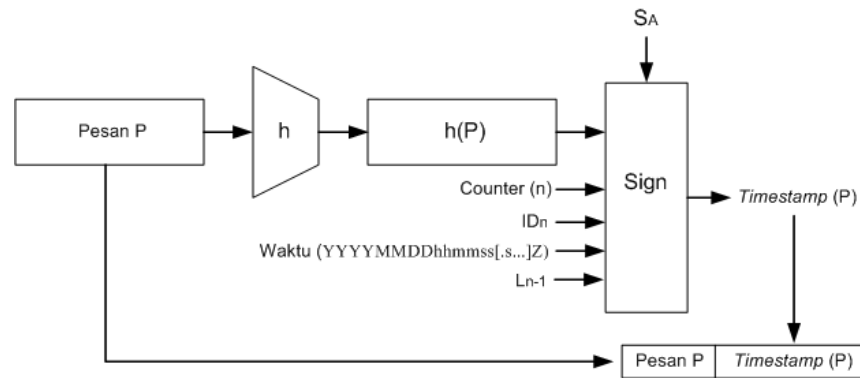
## **2. Tinjauan Digital Time Stamping**

Terdapat dua situasi dalam melakukan *digital time stamping* pada suatu dokumen digital seperti diilustrasikan dalam Gambar 1. Situasi pertama, setelah dilakukan *time stamp*, dokumen diasumsikan dibuat sesuai waktu dilakukan *time stamp*, jadi tanpa memperdulikan kapan sebenarnya dokumen tersebut dibuat. Sedangkan situasi kedua, dilakukan *time stamp* dengan waktu yang disesuaikan dengan waktu sebenarnya dari pembuatan dokumen. Penelitian ini hanya mengkaji untuk situasi pertama sehingga tidak mempertimbangkan kapan sebenarnya dokumen dibuat, tapi hanya mempertimbangkan kapan dokumen dilakukan *time stamp*.



**Gambar 1. Situasi Dalam Melakukan Time Stamp**

Penyederhanaan proses *digital time stamping* diilustrasikan pada Gambar 2. Proses tersebut dimulai dari client yang ingin melakukan timestamp dokumen P. Dokumen tersebut selanjutnya dikonversi menjadi  $h(P)$  dengan fungsi hash. Dengan ditambahkan counter(n), ID client ( $ID_n$ ), waktu (YYYYMMDDhhmmss) dan *link information*( $Ln$ ),  $h(P)$  dimasukkan proses *digital signature*(Sign) dengan kunci  $S_A$  menjadi  $Timestamp(P)$ .  $Timestamp(P)$  merupakan timestamp dari pesan P yang kemudian dikirimkan kembali ke client.



**Gambar 2. Proses Digital Time Stamping**

Dari proses pada gambar diatas, fokus perhatian *time stamping* meliputi:

1. Metode atau algoritma ( $h$ ) yang digunakan untuk mendapatkan *message digest* ( $h(P)$ ). Metode ini biasa disebut sebagai *cryptographic hash function*. *Hash function* ini akan dijelaskan lebih detil pada subbab lain. Sedangkan metode "sign" merupakan proses *digital signature*. Pembahasan mengenai digital signature tidak akan dibahas dalam tulisan ini secara detil.
2. Skema implementasi *time stamping* untuk mendapatkan  $Timestamp(P)$ . Secara singkat dapat dijelaskan bahwa skema implementasi time stamping terbagi dalam dua konsep yaitu skema dengan *trust third party* (TTP) dan skema dengan *distributed trust*. Penjelasan dari tiap – tiap konsep ini akan diuraikan pada subbab lain.
3. Metode penentuan waktu *timestamp*. Metode penentuan waktu ini akan dibahas juga pada subbab dibawah ini.

Stuart Haber dan W. Scott Stornetta ([1]) merupakan orang yang memperkenalkan pertama kali konsep mengenai *digital time stamping* ini.

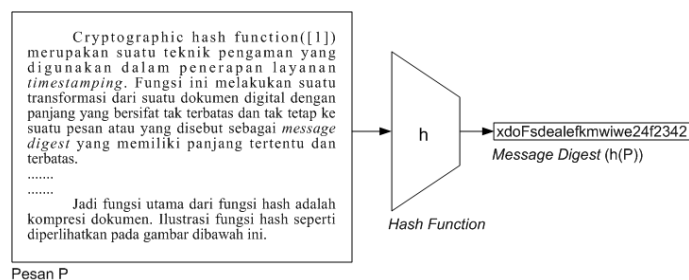
### 2.1 Metode atau Algoritma *Timestamp*

Metode atau algoritma yang digunakan didalam melakukan *digital time-stamping* adalah *cryptographic hash function* dan *digital signature*.

*Digital signature*([2]) merupakan suatu teknik pengamanan sederhana didalam memberikan hubungan antara dokumen dengan pengarang/penciptanya. Seperti halnya tanda tangan pada sebuah dokumen, *digital signature* juga memiliki tujuan untuk authorisasi, kepemilikan, dan mencegah penyangkalan. Akan tetapi *digital signature* memiliki kelebihan dibanding tanda tangan biasa yaitu dengan *digital signature* dimungkin tanda tangan/*signature* seseorang dapat berubah sesuai dengan

waktu, kebutuhan dan keinginannya. Algoritma yang digunakan dalam *signature* antara lain algoritma RSA (Rivest, Shamir, dan Adleman, [3]), skema GQ signature, DSA. Pada tulisan ini tidak akan dibahas mengenai *digital signature* secara lebih detail lagi.

*Cryptographic hash function*([1]) merupakan suatu teknik pengamanan yang digunakan dalam penerapan layanan *timestamping*. Fungsi ini melakukan suatu transformasi dari suatu dokumen digital dengan panjang yang bersifat tak terbatas dan tak tetap ke suatu pesan atau yang disebut sebagai *message digest* yang memiliki panjang tertentu dan terbatas. Jadi fungsi utama dari fungsi hash adalah kompresi dokumen. Ilustrasi fungsi hash seperti diperlihatkan pada gambar dibawah ini.



**Gambar 3. Cryptographic Hash Function**

Fungsi hash harus memiliki sifat sebagai berikut:

1. *One way function*.
2. *Collision free*.
3. Mudah (dalam arti: cepat, komputasi yang ringan) untuk mendapatkan suatu nilai hash dari suatu pesan/dokumen.

Definisi dari *colliosion free* seperti diperlihatkan pada kotak dibawah ini.

**Definisi 1.**

Misalkan  $x$  merupakan suatu pesan. Sebuah fungsi hash  $h$  dikatakan bersifat *weakly collision free* untuk  $x$  jika secara komputasi tidak mungkin menemukan pesan  $x' \neq x$  sedemikian sehingga  $h(x') = h(x)$ .([4])

**Definisi 2.**

Sebuah fungsi hash  $h$  merupakan *strongly collision free* jika secara komputasi tidak mungkin menemukan pesan  $x$  dan  $x'$  sedemikian sehingga  $x' \neq x$  dan  $h(x') = h(x)$ .([4])

Dari definisi diatas, sebuah fungsi hash  $h$  disebut sebagai *strongly collision free* jika dan hanya jika secara komputasi tidak mungkin menemukan pesan  $x$  sedemikian sehingga  $h$  bukan *weakly collision free* untuk  $x$ .

Sifat lain yang dimiliki oleh fungsi hash adalah *one-way function*. Definisi dari *one-way function* sebagai berikut [4].

**Definisi 3.**

Sebuah hash function  $h$  merupakan *one-way function* jika diberikan sebuah *message digest*  $z$  maka secara komputasi tidak mungkin untuk menemukan pesan  $x$  sedemikian sehingga  $h(x) = z$ .

Dari definisi 2 dan 3 dapat dibuktikan bahwa sifat *strongly collision free* berimplikasi memiliki sifat *one-way function*.

Seperti telah dijelaskan diawal bahwa pesan (dokumen) memiliki panjang digit yang berubah – ubah sedangkan panjang digit dari *message digest* bersifat tetap dan terbatas. Dalam implementasinya biasanya pesan memiliki panjang yang lebih besar dibandingkan panjang dari *message digest*. Hal ini menyebabkan peluang terjadinya *collision* tidaklah sama dengan nol, seperti diperlihatkan pada teorema 1 dibawah ini.

**Teorema 1.**

Misalkan  $h: X \rightarrow Z$  merupakan sebuah fungsi hash dimana  $|X|$  dan  $|Z|$  adalah terbatas dan  $|X| \geq 2|Z|$ . Misalkan  $A$  adalah algoritma invers untuk  $h$ . maka terdapat peluang algoritma Las Vegas yang menemukan sebuah *collision* untuk  $h$  dengan peluang paling sedikit  $\frac{1}{2}$ . [4]

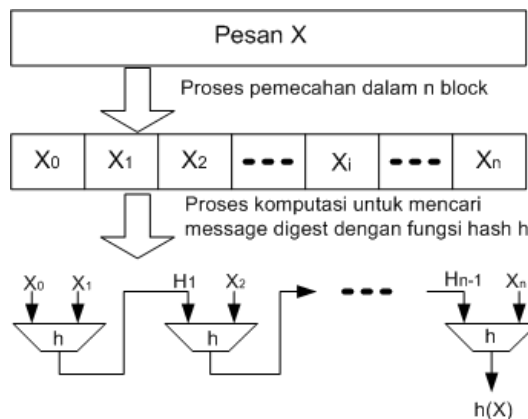
**Pembuktian :**

Pembuktian dari teorema diatas dapat dilihat pada ([4]).

Untuk mempertahankan sifat *strongly collision free* pada hash function dengan pesan/dokumen yang memiliki panjang tak terbatas, digunakanlah metode *extending hash function/iterated hash function*[4][5] .

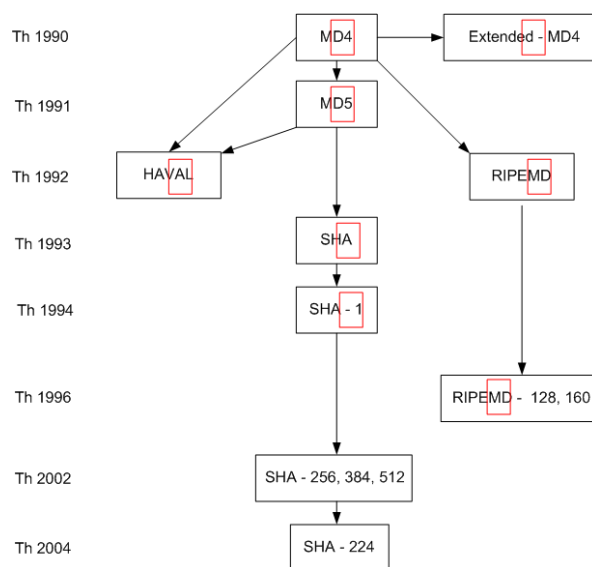
Prinsip dasar dari *extending hash function* adalah melakukan pemotongan dari pesan yang berukuran panjang menjadi potongan pesan yang memiliki ukuran lebih kecil dan dipastikan dengan ukuran tersebut nilai hashnya bersifat *strongly collision free*. Kemudian dari tiap pesan kecil tersebut dapat diperoleh nilai hash dari masing –

masing pesan. Semua nilai hash digabung menjadi satu nilai hash gabungan yang bersifat *strongly collision free* (lihat Gambar 4).



**Gambar 4. Proses *Extended Hash Function***

Beberapa contoh algoritma fungsi hash yang umum digunakan antara lain MD4[6][4], SHA1[7] dan MD5[8][9] yang merupakan perbaruan dari MD4. Keterkaitan dan perkembangan dari algoritma hash tersebut, seperti diperlihatkan pada Gambar 5. Kotak merah pada label pada gambar, menunjukkan bahwa algoritma tersebut terbukti telah terjadi *collision*. Saat ini, National Institute of Standard and Technology (NIST) telah menjadikan SHA-224, SHA-256, SHA-384, dan SHA-512 sebagai standard fungsi hash yang baru. Standard fungsi hash yang baru saat ini masih terus dalam kajian apakah fungsi hash ini bersifat *collision free* karena fungsi tersebut masih turunan dari MD5 yang telah terbukti bersifat *collision*.

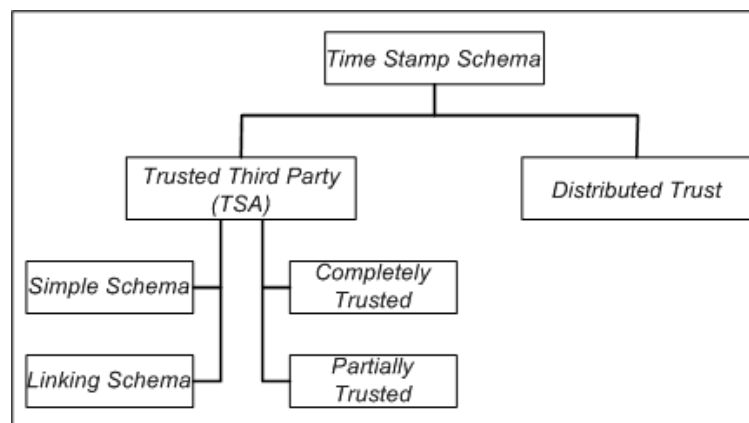


**Gambar 5. Keterkaitan dan Histori Algoritma Hash (Diambil dari Slide Perkuliahan Prof. Sean Murphy, <http://www.isg.rhul.ac.uk/node/106>)**

## 2.2 Skema *Digital Time Stamping*

Didalam beberapa literatur[10][11], skema *digital time stamping* dikelompokkan kedalam 3 tipe dasar yaitu: skema sederhana (*simple scheme*), skema terangkai (*linking scheme*), dan skema terdistribusi (*distributed scheme*). Disamping pembagian tersebut, peneliti lain juga mengklasifikasikan *time stamping* berdasarkan tekniknya dengan membagi kedalam 2 konsep[12] yaitu konsep skema dengan menggunakan pihak ketiga terpercaya (*trusted third party/TTP*) dan berdasarkan konsep terdistribusi (*distributed trust*). Konsep yang berdasarkan *trusted third party* terbagi lagi dalam 2 jenis yaitu pihak ketiga yang dipercaya penuh (*completely trusted*) dan dipercaya sebagian (*partially trusted*).

Pada dasarnya pengelompokan tersebut diatas memiliki kesamaan karena pada skema simple dan linking digunakan TTP sebagai dasar teknik implementasinya. Sehingga diagram pengelompokan skema *time stamp* dapat diilustrasikan seperti pada Gambar 6 dibawah ini.



**Gambar 6. Klasifikasi Skema Time Stamp**

Saat ini, konsep dengan *trusted third party* dengan skema *linking* dan konsep *distributed trust* banyak dikaji oleh para peneliti. Beberapa peneliti cenderung memilih skema dengan TTP dan beberapa peneliti yang lain lebih memilih skema terdistribusi. Satu sama lain memiliki alasan dan argumen masing – masing. Untuk implementasi skema dengan pihak ketiga, berdasarkan X.509 Public Key Infrastructure Time-Stamp Protocol (TSP) [13], persyaratan Time Stamping Authority / TSA yang berperan sebagai TTP adalah sebagai berikut

1. Untuk menggunakan sumber penetapan waktu yang terpercaya.
2. Untuk menyertakan nilai waktu tersebut untuk setiap *time-stamp token*.

3. Untuk menyertakan urutan bilangan yang unik untuk setiap *time-stamp token* yang baru.

Permasalahan atau pertanyaan yang muncul ketika mengimplementasikan skema dengan konsep TSA adalah seberapa percayakah terhadap pihak ketiga (TSA) atas keamanan data/dokumen kita, bagaimana tingkat keakuratan penetapan waktu, bagaimana melakukan deteksi, verifikasi, audit atau perbaikan jika terjadi kesalahan/serangan pada *timestamp*, disamping itu Jon Busey dalam ([14]) memberikan pertanyaan yang perlu diperhatikan, yaitu:

1. Bagaimana jika pemilik file/dokumen tidak ingin orang lain mengetahui keberadaan file tersebut?
2. Bagaimana kalau sertifikasi *timestamp* pada suatu dokumen ingin dicabut ?
3. Jika karena sesuatu hal TSA tidak dapat dipercaya atau TSA mengalami kebangkrutan sehingga ditutup, apa yang terjadi dengan database dokumen kita? pihak mana yang kemudian akan mengambil alih database dokumen?

Disamping pertanyaan tersebut diatas, diperlukan juga suatu standarisasi antar TSA sehingga dapat mengantisipasi jika terjadi gugatan atas suatu dokumen pada TSA yang berbeda.

Dilain pihak, penerapan konsep secara terdistribusi menurut H. Massias dalam[15] tidak memungkinkan dilakukan dalam lingkungan profesional karena memerlukan waktu yang lama untuk melakukan timestamp pada satu dokumen.

Penjelasan mengenai skema dengan *trusted third party* (TTP) dan *distributed trust*, akan dijelaskan secara lebih detil pada subbab dibawah ini.

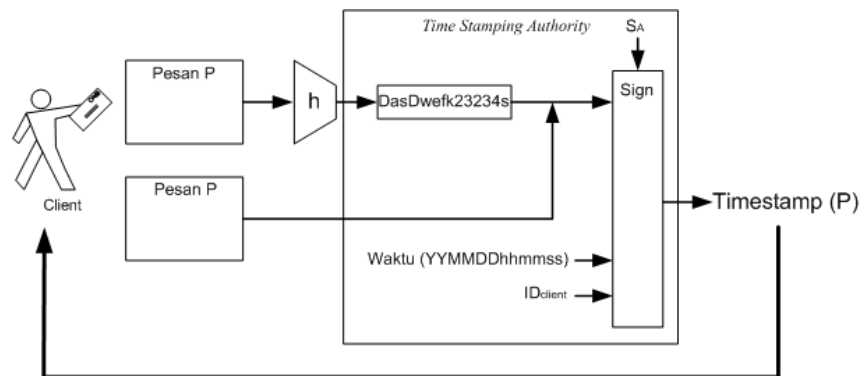
### **2.2.1 Skema *Trusted Third Party* (TTP)**

#### **2.2.1.1 Skema Sederhana (*Simple Scheme*)**

Dalam skema ini, *timestamp* akan dihasilkan oleh *Time Stamping Authority* (TSA) sebagai pihak ketiga yang telah dipercaya (*trusted third party*). Skema simple adalah sebagai berikut (lihat Gambar 7) [10]:

1. *Client* mengirimkan dokumen P (atau sebuah nilai hash dari dokumen tersebut) ke TSA.
2. TSA membubuhkan waktu t dan  $ID_{client}$  dan digabungkan dengan dokumen seperti sebagai berikut ( $ID_{client}, t, P$ ).

3. TSA mengembalikan dua nilai yaitu  $t$  dan *signature*  $s = \text{sig}_{\text{TSA}}(\text{ID}, t, P)$  ke *client*.



Gambar 7. Skema Simple

Metode ini memiliki kelemahan terletak pada tingkat kepercayaan TSA, yaitu antara lain rentannya manipulasi waktu ( $t$ ) oleh TSA, kemudian jika terjadi kesalahan akan sulit untuk dilakukan pelacakan/audit.

### 2.2.1.2 Skema Terangkai (*Linking Scheme*)

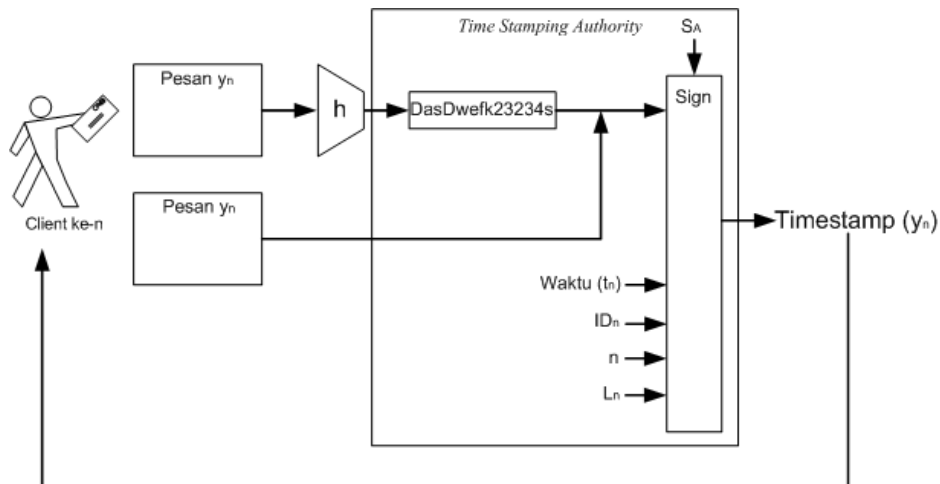
Ide yang mendasari skema terangkai ([10], [1], [15]) ini adalah menghasilkan sebuah *timestamp* dari suatu dokumen dengan melibatkan *timestamp* dari dokumen yang lain (misalkan dengan *timestamp* dari dokumen sebelumnya), sehingga membentuk suatu rantai *time stamp*. Alasan pembentukan rantai ini adalah mempersulit kemungkinan manipulasi waktu dari *time stamp* khususnya jika dibandingkan dengan skema sederhana.

Langkah – langkah implementasi dari skema terangkai adalah sebagai berikut (lihat Gambar 8) [15]:

Misalkan  $S_k$  adalah algoritma signature dari TSA,  $H$  adalah fungsi hash,  $y_n$  adalah dokumen yang akan di timestamp, dengan  $n$  adalah permintaan yang ke- $n$  ke TSA.

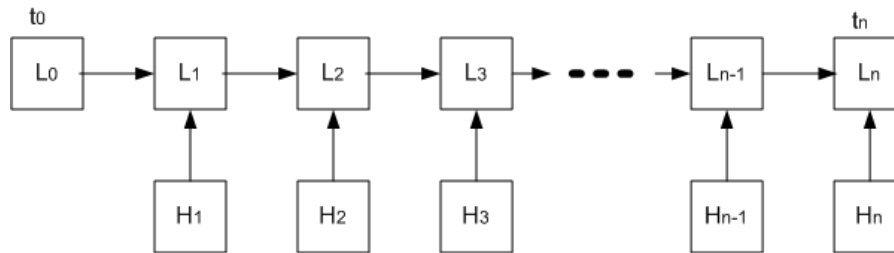
1. *Client* mengirimkan dokumen  $y_n$  (atau nilai hash dokumen) dan ID client ( $\text{ID}_n$ ) ke TSA.
2. TSA mengirimkan kembali ke *client* :
 
$$s = S_k(n, t_n, \text{ID}_n, y_n, L_n)$$
 dimana,
 
$$L_n = (t_{n-1}, \text{ID}_{n-1}, y_{n-1}, H(L_{n-1}))$$
3. Ketika terdapat permintaan ke  $n+1$ , TSA akan mengirimkan  $\text{ID}_{n+1}$  ke client. ( $s, \text{ID}_{n+1}$ ) adalah timestamp dari dokumen  $y_n$ .

$L_n$  merupakan informasi yang terangkai.



Gambar 8. Skema Terangkai (*linking scheme*)

Model linking dari informasi  $L_n$  lebih jelasnya dapat dilihat pada Gambar 9.



Gambar 9. Model Linking Timestamp

Verifikasi dari *timestamp* dokumen  $y_n$  yaitu dengan cara melakukan pengecekan pada timestamp dengan  $ID_{n+1}$ ,  $(n+1, t_{n+1}, ID_{n+1}, y_{n+1}; L_{n+1})$ , sehingga diperoleh  $L_{n+1} = (n, t_n, ID_n, y_n; L_n)$ .

Skema terangkai ini memiliki tingkat keakuratan, keamanan dan kepercayaan yang baik karena setiap dokumen yang dilakukan *timestamp* diberi nomor terurut, waktu  $t$  yang diberikan saling terkait dengan pemberian waktu time stamp dokumen lain. Disamping itu, jika ada pihak ketiga yang melakukan pergantian terhadap dokumen dan ID, misal  $(n, t_n, ID_n, y_n; L_n)$  diganti dengan  $(n, t_n, ID'_n, y'_n; L_n)$  maka ketika dilakukan verifikasi akan terdapat perbedaan pada nilai hash dari  $L_{n+1}$ .

Walaupun skema ini lebih baik dibandingkan skema sederhana, skema ini memiliki kelemahan yaitu jika terdapat suatu dokumen yang ingin dicabut *timestamp* atau jika terdapat client yang ingin merahasiakan ID dan dokumennya, bagaimana keterkaitan (*link*) yang baru

Variasi lain dari skema skema terangkai ini adalah skema atau metode dengan menggunakan struktur pohon (*tree structure*) [15] yang memiliki prinsip adanya suatu rangkaian antar dokumen yang secara bersamaan di lakukan *timestamp* disamping rangkaian *timestamp* dokumen sebelum dan sesudahnya.

Skema *trust third party* (TTP) dengan metode *linking* ini merupakan metode yang saat ini banyak digunakan. Unizeto CERTUM (<http://time.certum.pl>) merupakan salah satu contoh TSA.

### 2.2.2 Skema Terdistribusi (*Trust Distributed*)

Skema terdistribusi[10][1][15] merupakan skema tidak bersifat terpusat pada pihak ketiga seperti halnya konsep skema dengan menggunakan *Trust Third Party* (TTP). Skema ini didasari konsep bahwa setiap *user* memiliki skema *signature* (ID) sehingga setiap user dapat melakukan *sign* pada dokumen.

Langkah – langkah pada skema terdistribusi adalah sebagai berikut.

Misalkan *client* ingin melakukan *timestamp* pada dokumen *y*.

1. Pertama, dengan menggunakan pembangkit *pseudo-random* dihitung suatu nilai sebanyak  $k : V_1, V_2, \dots, V_k$ . Nilai ini merepresentasikan ID seseorang, sehingga terdapat sebanyak  $k$  ID.
2. Kemudian, dokumen *y* dikirimkan ke semua ID tersebut.
3. Untuk setiap orang menambahkan waktu ke dokumen sebelum dilakukan *sign* dan dikirim kembali ke *client*.
4. *client* menyimpan  $k$  *signature* sebagai *timestamp* dari dokumen *y*.

Untuk menjamin bahwa *client* melakukan manipulasi pada pemberian waktu *timestamp* maka harus dipastikan bahwa  $k$  cukup besar. Secara konseptual, proses *timestamp* dengan menggunakan skema ini membutuhkan *execution time* yang lama untuk melakukan *timestamp* satu dokumen dan sangat rumit karena membutuhkan banyak sekali pihak ketiga.

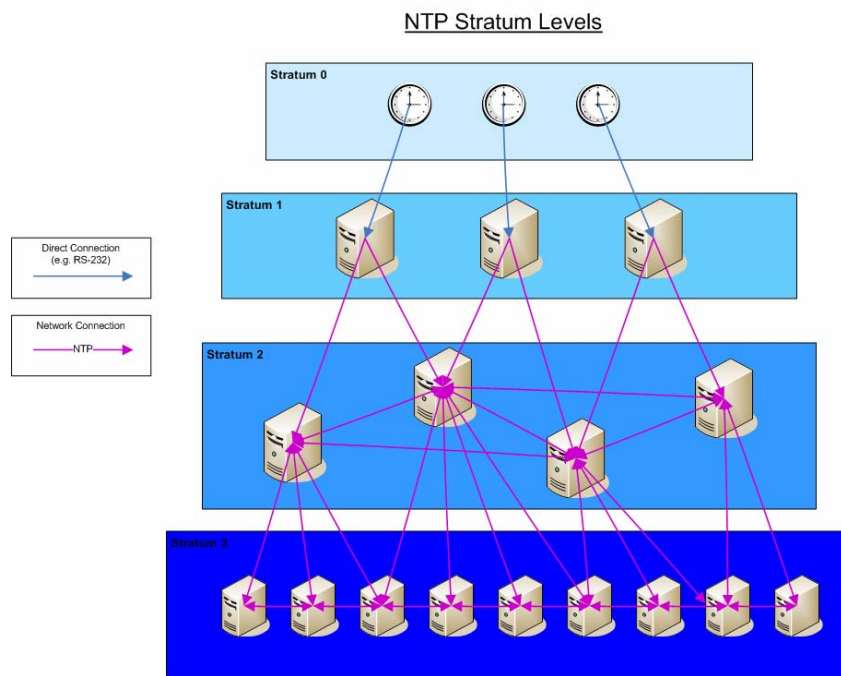
### 2.3 Metode Penentuan Waktu

Waktu merupakan elemen penting didalam implementasi *digital timestamp* sehingga dibutuhkan akurasi dan presisi yang tinggi. Format waktu yang digunakan dalam *timestamp* adalah format Universal Time (UT) atau dikenal juga sebagai Greenwich Mean Time (GMT) yang merupakan basis dari sistem waktu sedunia.

Sebuah contoh format GMT adalah YYYYMMDDhhmmss[.s...]z, dimana YYYY menunjukkan tahun, MM menunjukkan bulan, DD menunjukkan tanggal, hhmmss[.s...] berturut – turut menunjukkan jam, menit dan detik dengan presisi bisa mencapai nanosecond, simbol z menunjukkan zulu time atau universal time.

Penentuan waktu secara networking dapat menggunakan protocol Network Time Protocol (NTP). NTP merupakan sebuah protocol untuk sinkronisasi waktu dari sistem komputer packet-switched. NTP menggunakan port UDP 123 pada layer transport. NTP ini pertama kali didesain oleh Dave Mills dari University of Delaware.

NTP menggunakan sistem hirarki clock strata seperti yang diilustrasikan pada gambar dibawah ini.



**Gambar 10.** Level Stratum/lapisan NTP (diambil dari [http://en.wikipedia.org/wiki/Network\\_time\\_protocol](http://en.wikipedia.org/wiki/Network_time_protocol)) Stratum 0 merupakan GPS clock atau radio clock. Stratum 0 tidak terhubung dengan jaringan akan tetapi terhubung secara lokal dengan komputer (misal via RS-232). Stratum 1 disebut sebagai timeserver yaitu komputer yang melayani permintaan dari level dibawahnya. Stratum 2 dan seterusnya yaitu komputer yang mengirimkan permintaan ke stratum 1.

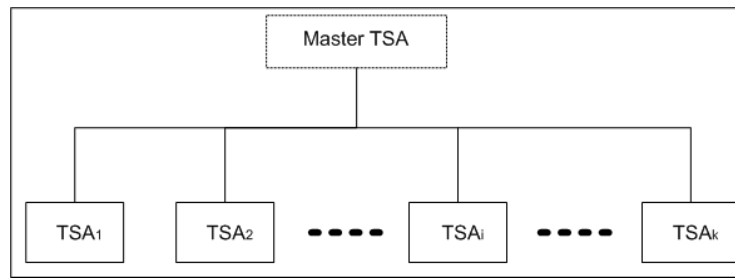
Salah satu contoh implementasi NTP adalah BELNET (<http://www.belnet.be/services/ntp/ntp.html>). BELNET berlokasi di Brussels dan Louvain memberikan layanan NTP yang menawarkan akurasi sekitar 100ms.

### 3. Usulan Skema TSA *Distributed* Untuk Implementasi *Digital TimeStamping*

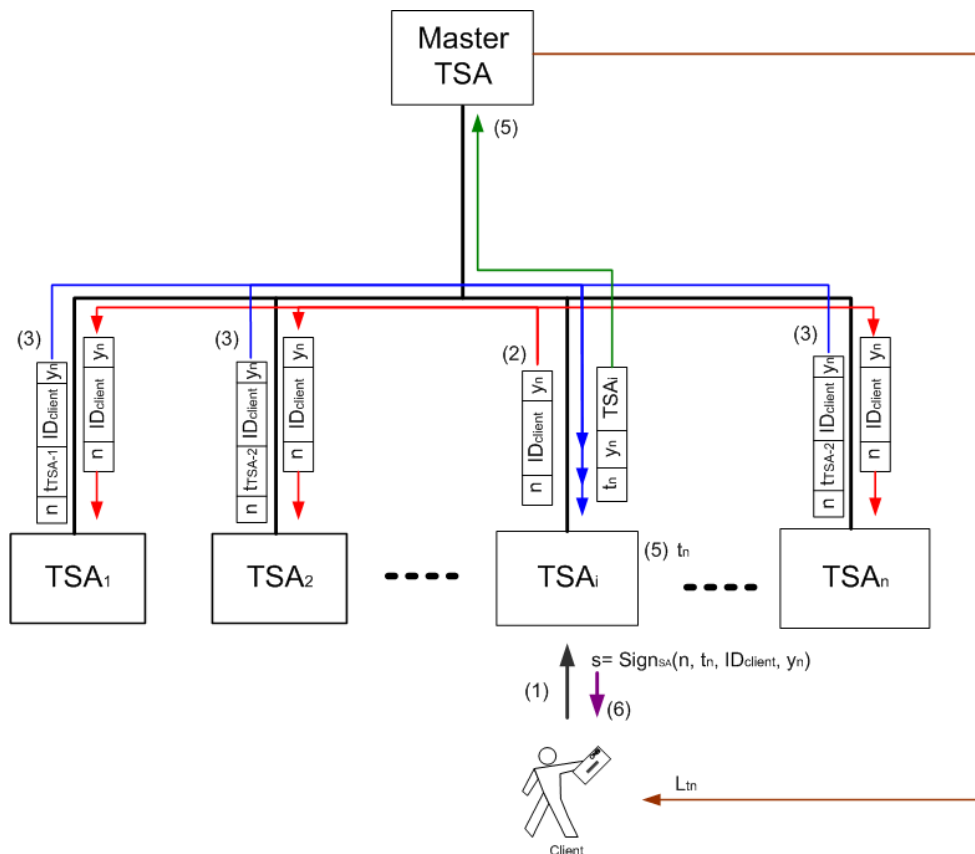
Setelah membahas *digital timestamping* pada bab tinjauan, penulis menawarkan suatu konsep skema implemetasi digital timestamping. Konsep yang menjadi usulan penulis adalah konsep skema TSA *Distributed* yaitu skema yang menggabungkan antara skema terdistribusi dengan skema trust third party (TTP) dengan TSA sebagai TTP. Ide dasar dari konsep ini adalah meningkatkan tingkat kepercayaan TSA sebagai TTP dengan menerapkannya skema terdistribusi, tetapi juga memiliki sifat yang sederhana sebagaimana implementasi *timestamp* dengan menggunakan skema TTP.

Langkah – langkah pada skema TSA *Distributed* adalah sebagai berikut: Terdapat sejumlah  $k$  TSA yang terkoneksi dengan 1 master TSA (lihat Gambar 11 dan 12) dan masing – masing TSA memiliki ID (misalkan  $TSA_1, TSA_2, \dots, TSA_k$ ).

1. Client ke-  $n$  mengirimkan nilai hash dari dokumennya ( $y_n$ ) dan  $ID_{client}$  ke suatu TSA yang memiliki ID  $TSA_i$ .
2. Oleh  $TSA_i$ , dokumen yang masuk diberi counter ( $n$ ) dan kemudian  $TS_{TSA-i} = (n, ID_{client}, y_n)$  dikirimkan keseluruh TSA yang lain untuk meminta waktu ( $t_{TSA}$ ).
3. TSA yang menerima permintaan  $TS_{TSA-i}$  akan mengirimkan kembali  $TS_{TSA-i} = (n, t_{TSA}, ID_{client}, y_n)$  dimana  $t_{TSA}$  adalah waktu saat itu di masing – masing TSA.
4. Dengan menggunakan pembangkit pseudo random dari sejumlah  $n$  TSA yang mengirimkan  $t_{TSA}$  ( $n=k$  jika semuanya mengirimkan  $t_{TSA}$ ) ditentukan TSA pilihan (misalkan  $TSA_f$ ) yang waktunya dijadikan timestamp ( $t_n$ ).
5. Kemudian  $TSA_i$  mengirimkan  $TS_{TSA-i} = (t_n, y_n, TSA_i)$  ke master TSA untuk dilakukan linking. Dari proses linking ini  $TS_{TSA-i}$  akan dirangkai dengan timestamp dokumen lain dari semua TSA dan hasilnya adalah  $L_{tn}$ .  $L_{tn}$  dikirimkan ke client.
6. Client akan menerima  $s = Sign_{SA}(n, t_n, ID_{client}, y_n)$  dari  $TSA_i$ .



Gambar 11. Struktur TSA dan Master TSA pada Skema TSA Distributed



Gambar 12. Alur Kerja Skema TSA Distributed

Analogi peran Master TSA adalah seperti seorang petugas administrasi atau tata usaha bagian surat menyurat. Petugas ini bekerja berdasarkan tanggal surat yang diterimanya. Dari informasi tanggal yang tertera pada kop surat, maka beberapa surat akan disusun sesuai urutan tanggalnya. Kadangkala untuk memudahkan memasukkan surat baru atau mengambil surat dalam rak, petugas akan memberi pembatas kertas yang mengklasifikasikan waktu berdasarkan hari, bulan atau tahunnya. Sehingga surat yang berada dalam rentang pembatas yang sama akan memiliki klasifikasi yang sama. Sama halnya dengan petugas administrasi ini, master TSA melakukan proses ini secara digital. Dokumen atau data yang diterima dari seluruh TSA dilakukan pengecekan berdasarkan waktu *timestamp*-nya. Dokumen yang memiliki waktu

*timestamp* yang sama diletakkan pada tempatnya sama juga. Kumpulan dokumen dengan waktu yang sama diurutkan dengan dokumen lain berdasarkan waktu *timestamp*-nya. Pembatas yang mengklasifikasi waktu dapat dibuat secara acak dengan rentang waktu yang telah ditentukan dan kemudian dienkripsi, hal ini untuk menghindari pelacakan dari pihak ketiga yang bertujuan memasukkan atau mengambil data tertentu secara ilegal.

Dokumen yang telah dilakukan *timestamp* oleh TSA dan yang dikirimkan ke master TSA (misalkan:  $TS_{TSA-i} = (t_n, y_n, TSA_i)$ ) akan dirangkaikan dengan dokumen *timestamp* dari berbagai TSA yang memiliki waktu yang sama dan *chiphertext* dari pembatas klasifikasi waktu untuk menambah tingkat kepercayaan. Contoh rangkaian dokumen yang diperoleh dari penyusunan rangkaian ini adalah

$$L_{tn} = h(t_n, y_n, TSA_i, x_n, TSA_j, \dots, z_n, TSA_m, E_p)$$

dimana,

- $h$  adalah fungsi hash
- $t_n$  adalah waktu *timestamp*
- $y_n, x_n, z_n$  adalah dokumen yang memiliki waktu *timestamp* yang sama ( $t_n$ )
- $TSA_i, TSA_j, TSA_m$  adalah secara berturut – turut ID TSA untuk dokumen  $y_n, x_n, z_n$ .
- $E_p$  adalah *chiphertext* dari pembatas klasifikasi waktu tepat sebelum waktu  $t_n$

#### Verifikasi:

Client dapat melakukan verifikasi *timestamp* dari dokumennya dengan cara:

1. Client melakukan pengecekan  $s = \text{Sign}_{SA}(n, t_n, ID_n, y_n)$ .
2. Client mengirimkan  $t_n$  dan  $y_n$  ke master TSA.
3. Dari master TSA, client akan memperoleh  $L_{tn}$ .
4. Cocokkan  $L_{tn}$  yang diterima dari master TSA dengan yang dipunyai oleh client.

#### Claim:

Dua client atau lebih yang mempertanyakan (*claim*) *timestamp* dokumen satu sama lainnya dapat melakukan verifikasi dengan mengirimkan waktu dan dokumen masing masing *client* ke master TSA, kemudian mencocokkan  $L_{tn}$  masing – masing client.

#### State of the Art:

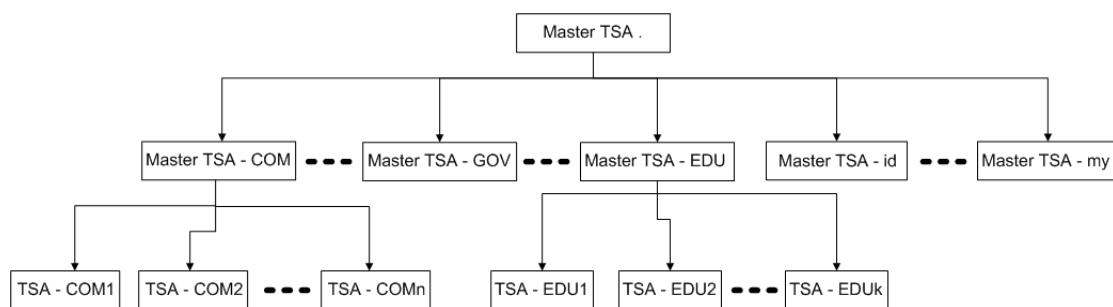
Pada tabel dibawah ini dijelaskan persamaan dan perbedaan antara konsep TTP, terdistribusi dan TSA Distributed.

**Tabel 1. Persamaan dan perbedaan antar skema implementasi timestamp**

	Trusted Third Party	Trusted Distributed	TSA Distributed
<b>Penentuan Waktu</b>	Oleh TSA sbg TTP	Oleh Random Client	Oleh Random TSA
<b>Jumlah Pihak yang terlibat</b>	Satu TSA	Sejumlah Client/ID User	Sejumlah TSA dan master TSA
<b>Verifikasi Dokumen</b>	TSA	tidak dapat dilakukan	Master TSA
<b>Claim dokumen</b>	Tidak dapat dilakukan untuk TSA yang berbeda	tidak dapat dilakukan	dapat dilakukan baik dengan TSA yang sama atau berbeda
<b>Tingkat Kepercayaan</b>	Completely Trusted	minimal trusted	Terdapat sejumlah TSA yang dipercaya (partially trusted)
<b>Jaringan Komputer</b>	Jaringan antar TSA - Client	Jaringan terkoneksi untuk semua Client/ID User	Jaringan antar TSA - TSA, TSA - Client, TSA - Master TSA
<b>Proses Time stamp</b>	Sederhana	Rumit	Sederhana
<b>Tempat Penyimpanan Data (hash dokumen)</b>	TSA	Client	TSA

Skema TSA Distributed seperti halnya skema secara terdistribusi memerlukan jaringan dengan bandwidth yang cukup besar khususnya untuk antar TSA. Pembentukan TSA dapat diatur oleh suatu lembaga yang terpercaya dan memiliki otoritas. Dengan skema ini, setiap TSA diharuskan memiliki standard *timestamping* yang sama agar dapat melakukan komunikasi antar TSA. Standard *timestamping* disamping sebagai protocol komunikasi antar TSA juga berguna sebagai pengatur, audit, legalitas dan keterjaminan atas keamanan dokumen pada pelaksanaan *timestamping*.

Skema dengan TSA *Distributed* memberikan struktur yang lebih bagus dibandingkan skema yang lain. Struktur TSA memberikan legalitas, kepercayaan dan otorisasi yang lebih jelas terhadap TSA. Gambar dibawah ini adalah salah satu contoh struktur TSA yang mungkin dapat diimplementasikan.



**Gambar 13. Struktur TSA Global**

Struktur TSA yang diilustrasikan pada Gambar 13 terlihat bahwa struktur ini mirip dengan struktur yang ada pada domain name system (DNS) pada penamaan server di internet.

#### **4. Kesimpulan**

Dari pembahasan makalah ini, dapat disimpulkan betapa pentingnya implementasi digital timestamping untuk dokumen digital. Untuk implementasi digital time stamping diperlukan suatu kajian yang mendalam khususnya pada cryptography hash function, skema implementasi, dan penetapan waktu timestamp. Ketiga hal tersebut telah dibahas dalam makalah ini. Tinjauan lain yang perlu dikaji namun tidak dibahas dalam makalah ini adalah tinjauan hukum, bisnis, dan implementasi teknis.

Skema implementasi secara TSA Distributed, yang diusulkan oleh penulis, secara konseptual memiliki tingkat kepercayaan yang lebih tinggi dibandingkan TTP tapi juga lebih sederhana dibandingkan skema secara *trust distributed*. Disamping dapat mengakomodasi claim terhadap timestamp dokumen pada TSA yang berbeda, skema ini memiliki struktur yang jelas dan menciptakan standard bersama antar TSA didunia.

#### **5. Diskusi dan Penelitian Lebih Lanjut**

Dalam makalah ini, telah diusulkan suatu konsep baru mengenai skema TSA distributed untuk implementasi digital timestamping. Konsep ini masih perlu dilakukan kajian lebih jauh untuk mengetahui kelayakan dalam implementasinya. Masalah lain yang perlu diteliti dan didiskusikan antara lain:

1. Standard teknis mengenai proses timestamp. Standard teknis ini meliputi hash function dan penetapan waktu yang digunakan, spesifikasi minimum untuk jaringan informasi, jenis dan nilai QoS yang harus dipenuhi dan tingkat keamanannya.
2. Analisa techno-economic dan benefit yang diperoleh oleh masyarakat global dengan menerapkan skema TSA distributed ini.
3. Uji coba implementasi dan problem solving dari implementasi digital timestamping.

## Daftar Pustaka

- [1] S. Haber, and W.S. Stornetta. How to timestamp a digital document. *Journal of Cryptology*, 3(2):99 – 112, 1991.
- [2] C. Mitchel, F. Piper, and P. Wild. Digital Signature. G. Simmons, editor, *Contemporary Cryptology: The Science of Information Integrity*, Hal. 325-378. IEEE pre, 1991.
- [3] R. Rivest, A. Shamir, and L. Adleman, A Method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, Vol. 21, Hal 120 – 126. 1978.
- [4] D. R. Stinson. *Criptography The Theory and Practice*, Hal. 232 – 257. CRC Press, 1995.
- [5] K. Gibson. *Extending Hash Function*. 1998.
- [6] R. Rivest. The md4 message digest algorithm. Dalam S. Vanstone, editor, *Advances in Cryptology – Proceeding of Crypto’90*, No. 537 LNCS, Hal. 303-311. Springer-Verlag, 1991.
- [7] wikipedia website – <http://en.wikipedia.org/wiki/SHA1>
- [8] R. Rivest. The md5 message-digest algorithm. Request for Comments (RFC) 1321, Internet Activities Board, Internet Privacy Task Force, April 1992.
- [9] wikipedia website – <http://en.wikipedia.org/wiki/MD5>
- [10] A. Cilaro, A. Mazzeo, L. Romano, G. P. Saggese, G. Cattaneo. A web services based architecture for digital time stamping. *Journal of Web Engineering*, Vol. 0, No. 0. Rinton Press. 2003.
- [11] International Organization for Standardization and Intenational Electrotechnical Commissioin (2002), ISO/IEC Standard 18014: Information Technology – Security Techniques – Time Stamping Services.
- [12] H. Massias, X. Serret Avila, J. –J. Quisquarter. Design of a secure timestamping service with minimal trust requirement. TIMESEC Project (Feredal Government Project, Belgium).
- [13] rfc 3161, <http://www.ietf.org/rfc/rfc3161.txt>
- [14] J. Busey. *A Proposal for Distributed Digital Time-Stamping*. 2004.
- [15] H. Massias and J. –J Quisquater. Time and cryptography. Technical report, TIMESEC Project (Federal Government Project, Belgium), 1997.