Investigasi Forensika Pada Log Web Server untuk Menemukan Bukti Digital Terkait dengan Serangan Menggunakan Metode Hidden Markov Models

Triawan Adi Cahyanto
Magister Teknik Informatika
Fakultas Teknologi Industri UII
Sleman, Yogyakarta
mastriawan@gmail.com

Yudi Prayudi
Magister Teknik Informatika
Fakultas Teknologi Industri UII
Sleman, Yogyakarta
prayudi@uii.ac.id

***Abstract*—Log merupakan suatu file yang berisi data atau informasi mengenai daftar tindakan, kejadian, dan aktivitas yang telah terjadi di dalam suatu sistem. Informasi tersebut akan berguna apabila sistem mengalami kegagalan sehingga akan dapat dicari penyebabnya berdasarkan log file yang terdapat pada sistem tersebut. Namun, penggalian informasi yang relevan dari data yang terkait dengan serangan pada log file itu menjadi tugas yang sulit bagi seorang administrator sistem. Dari beberapa data log file yang ada, belum tentu data log tersebut sesuai dengan data log yang diinginkan dan dicari. Suatu sistem yang kompleks memiliki beberapa data log file, namun yang saat ini sering dipakai, salah satunya adalah log file yang terdapat pada web server, karena pada era saat ini, dukungan aplikasi berbasis web kian maju tetapi berbanding terbalik dengan faktor keamanan yang dimiliki aplikasi web tersebut. Saat ini, terdapat beberapa perangkat lunak yang digunakan untuk melakukan analisa log seperti IDS (intrusion detection system), dan program berbasis signature lain, yang dapat digunakan untuk menemukan serangan terhadap aplikasi berbasis web pada log web server. Dari beberapa perangkat lunak yang ada, ada beberapa fitur yang mestinya harus diperbaharui terkait dengan kemampuan perangkat lunak agar dapat melacak keberadaan pelaku berdasarkan ip address log yang tersimpan, statistik berdasarkan alamat IP yang tersimpan pada log, dan visualisasi tampilan log, agar log file yang dianggap sebagai log si penyerang dapat ditampilkan sehingga memudahkan administrator sistem dalam mencari log yang terkait dengan serangan dari sekian banyaknya log yang tersimpan pada sistem. Berdasarkan ide tersebut, perangkat lunak yang akan dibuat, diharapkan dapat meningkatkan kualitas dan informasi isi dari data log yang diberikan, meringkas, dan menghasilkan output yang dapat dibaca dengan mudah oleh administrator sistem. Penelitian ini menerapkan metode hidden markov models untuk mendeteksi serangan terhadap aplikasi berbasis web kemudian hasilnya mampu menganalisa dan melakukan perhitungan statistik serta teknik pembelajaran yang diperoleh dari data pada log web server. Selanjutnya menggunakan data domain name system (DNS) blacklist dan informasi GeoIP untuk mengidentifikasi identitas penyerang yang potensial.**

Keywords—Investigasi Forensika, Log Web Server, Hidden Markov Models

I. PENDAHULUAN

Pada era saat ini, *web server* dan aplikasi berbasis web merupakan target serangan yang popular bagi kalangan peretas. Selain kemudahan dalam mengakses, ketersediaan *resource* konten aplikasi melalui jaringan (internet) membuat peretas memiliki banyak waktu untuk melakukan analisis dan serangan terhadap *resource* target serangan. *Web server* merupakan sebuah aplikasi yang terdapat pada *server* yang melayani permintaan HTTP atau HTTPS dari *browser* dan mengirimkannya kembali dalam bentuk halaman-halaman web [1].

Data setiap pengunjung yang mengirimkan permintaan atau ketika mengakses aplikasi berbasis web akan disimpan pada suatu *file* yang dinamakan *log* dalam hal ini yang terdapat pada *web server*. Data pengunjung yang terdapat pada *log web server* akan sangat bermanfaat apabila nantinya terdapat suatu permasalahan yang terjadi terhadap *web server* tersebut, misalnya kasus peretasan aplikasi web (*deface*). Dengan memeriksa satu per satu setiap catatan yang tersimpan pada *log*, maka data-data seorang peretas akan diketahui. Data peretas tersebut, dapat diketahui salah satu caranya adalah dengan melihat dari alamat *IP* yang dipakai untuk mengakses *web server*. Namun, catatan yang tersimpan dalam *log*, tentunya berisi catatan seluruh pengunjung yang mengakses *web server* tersebut, akan menjadi tidak efisien apabila metode pencarian data itu diperiksa satu per satu dari banyaknya data yang tersimpan.

Berdasarkan permasalahan tersebut, ada beberapa perangkat lunak yang dapat digunakan untuk melakukan identifikasi pelaku terkait dengan serangan ke *log web server* diantaranya IDS (*Intrusion Detection System*). Aplikasi IDS yang populer saat ini adalah Snort. Snort dapat melakukan deteksi adanya penyusupan terhadap *log web server* dengan cara menganalisis data *log* dan menyesuaikan dengan *signature* yang dimiliki oleh Snort tersebut. Dalam melakukan identifikasi, snort hanya dapat melakukan analisis *log*, tetapi tidak dapat melakukan proses-proses pemilihan *log* yang akan dipakai dan ditampilkan kepada penggunanya, sehingga apabila data-data *log* yang tersimpan tidak sesuai format *log*

standard atau bahkan rusak maka data tersebut sudah tidak layak untuk dilakukan identifikasi.

Berkaitan dengan hal tersebut, melalui penelitian ini diharapkan mampu untuk membuat dan mengembangkan tools untuk melakukan analisa terhadap data *log* terkait serangan yang tertuju ke aplikasi berbasis web. Analisa yang dilakukan yakni melalui *log file* yang tersimpan pada *web server* kemudian akan dicari data *log* setiap pengunjung yang mengakses *resource* yang terdapat pada *web server*, sehingga dapat diidentifikasi pengunjung yang melakukan aktivitas *browsing* secara umum dan pengunjung yang melakukan aktivitas dengan maksud untuk melakukan serangan (*anomaly detection*). Banyaknya data *log* tersebut, akan menyebabkan *record* yang terdapat pada *log file* menjadi semakin banyak. Dengan semakin banyaknya data *log*, terdapat *record* pada *log* yang tidak terkait dengan proses serangan, *record* tersebut disebut dengan *false alarm*.

Untuk mengurangi *false alarm* yang disebabkan banyaknya data *log*, maka akan dilakukan *filtering* dan statistik terhadap *log* menggunakan metode *hidden markov models*. *Hidden markov models* merupakan perluasan dari rantai *markov* dimana *state* atau kondisinya tidak dapat diamati secara langsung (tersembunyi) tetapi hanya dapat diobservasi melalui suatu himpunan pengamatan lain [5]. Untuk melakukan investigasi forensika terhadap pelaku kejahatan (*attacker*) yang menyerang *resource* maka *tools* yang nantinya dibuat, akan diintegrasikan dengan *dataset geolocation* berdasarkan lokasi geografis penyerang yang diperoleh dari alamat IP (walaupun mungkin tidak akurat) yang tersimpan pada *record log* [11].

Data *log* asli yang tersimpan pada web server akan dilakukan proses pelatihan data, pengujian data, pembentukan *dataset* kemudian dilakukan proses identifikasi terkait aktivitas pengguna yang mengakses *resource* serta identifikasi halaman dan struktur direktori *path* yang diakses dan proses akhir dari kerangka konsep tersebut adalah investigasi forensika dengan menampilkan tampilan *mapping* berdasarkan IP *geolocation* dan berdasarkan data DNS *blacklist* [2].

II. METODE PENELITIAN

Tahapan yang dilakukan dalam penelitian ini adalah

a) Mempelajari tentang *Hidden Markov Models*

b) Mempelajari penggunaan *Hidden Markov Models* dalam mendeteksi serangan terhadap *log web server*.

c) Mengerjakan contoh masalah yang terdapat pada *log web server* menggunakan *Hidden Markov Models* dimulai dari identifikasi karakter *url parameter* yang terdapat pada *log web server*, pemilihan data (*training* data), pengujian data (data *testing*), analisa data (*analysis* data) dan kesimpulan (*summary*).

III. HIDDEN MARKOV MODELS

Penggunaan *Hidden Markov Models* (HMM) untuk pembuatan jenis aplikasi meningkat pesat di era saat ini. Beberapa kasus yang seringkali dijadikan permasalahan menggunakan HMM adalah prediksi cuaca dan prediksi pasar saham [9]. Dari permasalahan tersebut, analisa menggunakan HMM telah terbukti menjadi alat yang ampuh [5]. Penelitian ini akan membahas mengenai identifikasi terkait serangan dengan melakukan analisa terhadap data *log web server* menggunakan HMM, sehingga nantinya dapat melindungi aplikasi berbasis web dari serangan atau eksploitasi terkait dengan *input validation flaws*.

Beberapa langkah-langkah proses *hidden markov models* pada penelitian ini diantaranya : identifikasi *log web server,* fase *training,* fase *testing,* fase analisis, dan dokumentasi.

A. Identifikasi Log Web Server

Aktivitas apa saja yang dilakukan oleh pengguna sistem akan selalu dicatat oleh sistem ke dalam bentuk *file log*. Banyaknya aktivitas yang dilakukan pengguna sistem akan menyulitkan proses pencarian data tertentu yang terdapat pada *log* terutama data yang terkait dengan serangan ke *log web server*. Identifikasi *log web server* yakni melakukan pengecekan terhadap integritas *log web server*. Seorang peretas kemungkinan akan mencoba untuk mengubah data *log* agar susah dilacak saat proses investigasi forensik. Pada web server *apache* (*httpd*), sering kali tidak memanfaatkan mekanisme perlindungan terhadap *log file* dalam mode konfigurasi *default*. Oleh karena itu, proses identifikasi *log* dibuat untuk memeriksa *log* agar dapat mendeteksi *tamper data*. Deteksi *tamper data* terhadap *log* dibuat dengan menerapkan algoritma *Grubbs outlier test* [3].

$$G = \frac{X_{\max} - \overline{X}}{\sigma_n} \quad \text{.... (1)}$$

G merupakan maksimum *delay* yang ditemukan pada *log*.
$X_{\max}$ merupakan nilai maksimum
$\overline{X}$ merupakan rata-rata aritmatika
σ_n merupakan standar deviasi *delay* waktu *request* yang tersimpan pada *log*.

Tabel I merupakan *dataset* hasil perhitungan *timestamp*, antara waktu ketika *record log* terakhir diakses (*last date*) dengan tanggal saat ini (*current date*).

TABEL I. DATASET PERHITUNGAN TIMESTAMP

N	0.1	0.075	0.05	0.025	0.01
3	1.15	1.15	1.15	1.15	1.15
4	1.42	1.44	1.48	1.48	1.49
5	1.6	1.64	1.67	1.71	1.75
6	1.73	1.77	1.82	1.89	1.94
7	1.83	1.88	1.94	2.02	2.1
8	1.91	1.96	2.03	2.13	2.22
9	1.98	2.04	2.11	2.21	2.32
10	2.03	2.1	2.18	2.29	2.41

Nilai 0,1, 0,075, 0,05, 0,025, 0,01 merupakan nilai kritikal dari *grubss test*.

Kalkulasi = 0,6400-0,6062 / 0,0166 = 2.04

Dari hasil tersebut maka bandingkan dengan yang terdapat pada tabel dimana nilai N = 7, nilainya 1,94. Sedangkan berdasarkan nilai *Gcalc* > *Gcritical* maka dapat dikatakan bahwa data tersebut mempunyai *timestamp* yang berkaitan sehingga dapat dipastikan bahwa tidak ada *tamper data* yang terjadi pada *record log* yang diidentifikasi.

B. Fase Training

Fase *training* merupakan fase yang dilakukan untuk membangun dan melatih *ensembles* HMM pada setiap *query* URL *parameter* dari suatu aplikasi web. Hal ini dilakukan untuk memproses masukan yang nantinya akan dipelajari dari data *training* dan kemudian akan dibandingkan dengan data *testing*. Pada fase *training* ini, proses pembelajaran dan beberapa jumlah masukan data akan diubah. Setiap huruf akan diubah dengan karakter A dan setiap digit akan diubah dengan karakter N, sedangkan karakter lainnya akan tetap disimpan. Setiap *client* hanya diperbolehkan berkontribusi sekali per aplikasi web untuk *generate* data *training* untuk menghindari perubahan pada *dataset*. Nilai-nilai yang ditambahkan ke fase *training* ini akan diatur jika kode respon HTTP 2xx/3xx teramati, sehingga itu akan menunjukkan operasi normal. *Training* HMM merupakan fase komputasi yang intensif, pada penelitian ini menggunakan maksimal 150 observasi, sedangkan observasi minimum yang dibutuhkan pada deteksi berbasis HMM adalah 40. Proses *training* HMM *ensembles* menggunakan algoritma *baum-welch*, sedangkan rumus untuk hmm λ : (A,B,π) adalah sebagai berikut [3]:

$$\overline{a_{ij}} = \frac{\sum_k W_k \sum_{t=1}^{Tk} \alpha_t^k a_{ij} b_j(O_{t+1}^{(k)}) \beta_{t+1}^{(k)}(j)}{\sum_k W_k \sum_{t=1}^{Tk} a_t^k(i) \beta_t^k(i)} \quad ...(2)$$

$$\overline{b_{ij}} = \frac{\sum_k W_k \sum_{O_t(k)=v_j} \alpha_t^k(i) \beta_t^k(i)}{\sum_k W_k \sum_{t=1}^{Tk} \alpha_t^k(i) \beta_t^k(i)}$$

Dimana A merupakan matrik transisi, $a \in A$
B adalah matrik observasi, $b \in B$
π adalah 1 yang merupakan inisial distribusi awal HMM (dipilih secara acak)
$W_k = \frac{1}{Pk}$, $k \in [1\mathrm{K}]$ merupakan *inverse* probabilitas dari perkiraan model urutan training.
Variabel α, β merupakan hasil dari prosedur *forward-backward*.

C. Fase Testing

Penelitian ini menggunakan algoritma *viterbi* untuk menguji urutan masukan. Algoritma *viterbi* akan menghitung urutan paling mungkin dari kondisi yang tersembunyi terhadap urutan probabilitas. Dalam kasus ini, kemungkinan terjadinya nilai permintaan URL yang diberikan akan dihitung. Jika URL berisi beberapa pasang *parameter* atau nilai, maka aturan minimum akan diterapkan. Prosedur yang digunakan untuk melakukan deteksi ditunjukkan sebagai berikut [6] :

Algoritma HMM-based Anomaly Detection

```
1.  for all requests i do
2.     if |samples per web application j| ≥
3.        threshold then
4.         for all URL query parameters k do
5.           for all ensembles l do
6.             p_jkt = Viterbi decode(k, hmm_jkt)
7.           end for
8.         end for
9.       if min(p_jk0 ... jkn) ≤ threshold then
10.         return true
11.      end if
12.     end if
13. end for
```

Jika urutan masukkan berisi simbol yang tidak terdapat pada fase training maka probabilitas akan menjadi *valid* dan menurun secara signifikan, sehingga akan menandai *requests* sebagai suatu serangan [3].

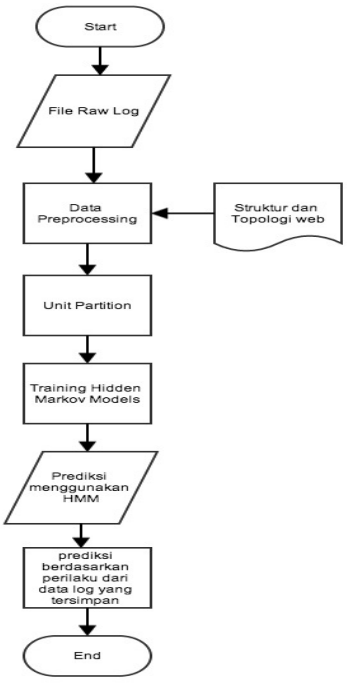
D. Fase Analisis

Untuk melakukan analisa serangan terkait dengan data-data *log web server*, penelitian ini akan mengklasifikan *session* web yang mencurigakan yang dilakukan oleh manusia dan yang berupa *bot* otomatis (*tools*) [4]. Dengan menggunakan teknik dari fase *training* dan *testing* yang nantinya digunakan untuk mengukur dan mengevaluasi serangan yang terdeteksi apakah berhasil atau gagal, kemudian kedua klasifikasi sesi dan kuantifikasi serangan akan membantu untuk menyusun dokumentasi. Dari *file log*, data-data mempunyai dua nilai yang dikirim dari *web server* ke *client* yaitu nilai kode respon HTTP dan jumlah byte *response*. Respon atau 'status' kode tersebut dijelaskan pada Tabel II.

TABEL II. RESPON KODE LOG WEB SERVER

Status	Keterangan
401,403	Unsuccessful .htacess login
404	Unsuccessful static cgi-bin scanner
408,503	Slowloris denial of service attempt
414	Unsuccessful buffer overflow attempt
500	Server -side error

Fase identifikasi, *training* dan *testing* merupakan urutan yang dibangun agar dapat menghasilkan pelatihan yang *representative* sehingga *dataset* akhir untuk *log web server* yang terkait dengan serangan dapat diprediksi dan disajikan dengan benar. Data tersebut bukan hanya berisi mengenai *log* urutan proses serangan melainkan juga terdapat data *log* urutan akses halaman tetapi yang terkait dengan serangan. Data *log web server* yang dipakai menggunakan *sanitizer log* yang diperoleh dari *honeynet project* [7]. Gambar 1 merupakan *flowchart* untuk memprediksi pengakses konten web menggunakan HMM.



Gambar 1 Flowchart HMM

Penelitian ini menetapkan variabel acak untuk keadaan transisi probabilitas distribusi A dan pengamatan probabilitas emisi simbol distribusi B menggunakan algoritma *Baum – Welch*. Algoritma *Baum – Welch* dipakai untuk melatih dan mendapatkan model yang optimum dari parameter A dan B. Kemudian untuk memulihkan kondisi tersembunyi dari *dataset* yang sudah ditentukan variabel acak nya, digunakan algoritma *Viterbi*.

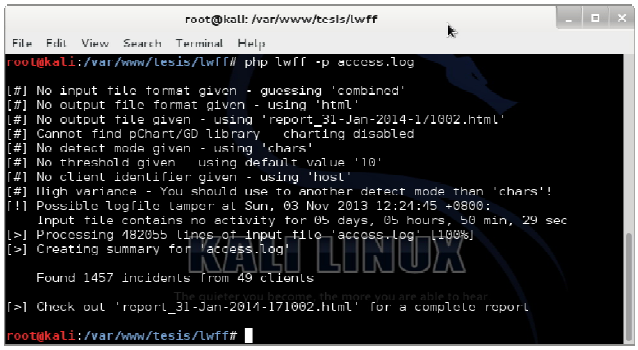
Hasil percobaan yang terdapat pada penelitian ini terdapat pada table 3. Tabel 3 menjelaskan hasil dari model statistik berdasarkan fase *training* dan *testing* sehingga hasilnya dapat memprediksi dari model yang akan dipakai menggunakan HMM. Dengan model HMM, dapat dicari proses yang paling mungkin untuk mendekati perilaku pengguna. Hasil data pengujian dari *log web server* yaitu jumlah data serangan yaitu 53% atau disebut dengan *true positive* sedangkan jumlah data yang diperkirakan terkait dengan proses serangan yaitu 82% atau disebut *false positive*.

Tabel 3 Statistik Model dengan HMM

Data Set 1		**Data Set 2**	
Training data	6420	Testing data	2034
Initial A = [0.5 0.5 ; 0.5 0.5]		Convergence A = 0.075 0.05 ; 0.025 0.01	
		Convergence B = ~	
Initial B = [1/643 , …, 1/643]		TP = 75 FP = 50 TN = 25	
		Precision = 53%	
		Recall = 82%	

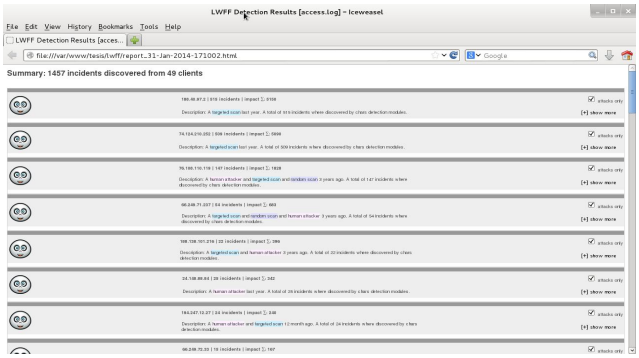
E. Dokumentasi

Dokumentasi merupakan laporan terkait dengan pencarian data-data yang terdapat pada *log web server*. Laporan-laporan yang terdapat pada penelitian ini dapat di-*generate* dalam bentuk html, json, xml, dan csv. Berikut ini merupakan contoh analisa *log web server*:



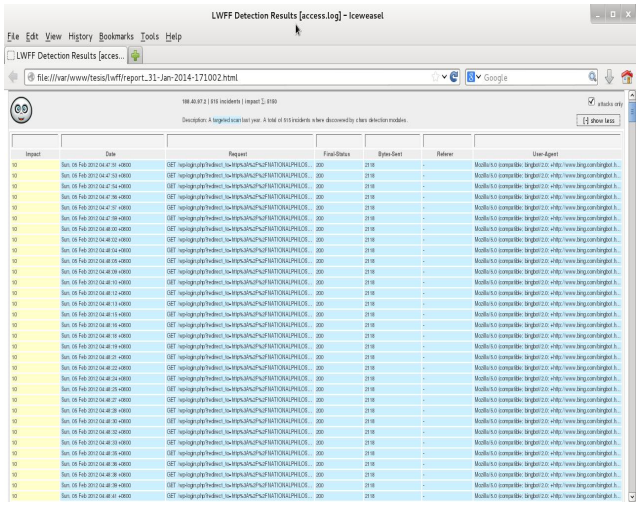
Gambar 2 Analisa log web server

Sedangkan hasil *output report*-nya adalah sebagai berikut:



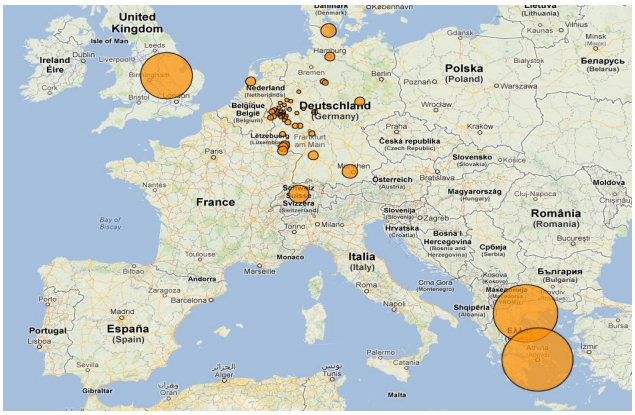
Gambar 3 Report file access.log

Detail laporan hasil analisa dari *file* access.log adalah sebagai berikut :



Gambar 4 Detail Laporan Serangan file access.log

Prosedur pelacakan dan pemetaan untuk mengidentifikasi pengguna internet berdasarkan lokasi geografis diperoleh berdasarkan alamat IP. Alamat IP (alamat yang dialokasikan setiap hari oleh penyedia akses internet) merupakan suatu proses yang dapat dihandalkan untuk melacak posisi geografis. Berikut ini merupakan salah satu data *ip geolocation* berdasarkan data yang terdapat pada *log web server*.



Gambar 5 IP Geolocation berdasarkan record log

IV. KESIMPULAN

Berdasarkan uraian yang telah dituliskan, dapat diambil beberapa kesimpulan, yaitu:

1. Metode *Hidden Markov Models* merupakan solusi yang efektif dalam mengolah data *log* terkait dengan serangan, dimana proses pemecahan masalahnya dilakukan dengan tiga tahap dasar yaitu evaluasi permasalahan dengan algoritma *forward-backward*, membaca permasalahan dengan algoritma *viterbi*, dan mempelajari permasalahan dengan algoritma *baum-welch*.
2. Model HMM dapat mengenali proses atau perilaku pengguna yang mengakses *resource* web tertentu melalui observasi yang dilakukan dari tahap pelatihan, pengujian dan perhitungan statistik yang menghasilkan nilai *precision* = 53% dan *recall* = 80% yang berarti bahwa jumlah data serangan yang terdapat pada *log* lebih sedikit dibandingkan jumlah data yang dianggap sebagai serangan.
3. *Domain Name System* (DNS) *blacklist* dan IP *Geolocation* merupakan fitur yang digunakan untuk melakukan *monitoring* secara *visual* terhadap alamat IP yang diperoleh ketika proses identifikasi menggunakan *Hidden Markov Models.*
4. Investigasi forensika dapat dilakukan dengan menguji keaslian data *log web server* menggunakan algoritma *grubbs outlier test* untuk memastikan bahwa data *log* tersebut tidak mengalami modifikasi dengan melihat *timestamp log* saat tersimpan di web server dengan melihat *timestamp log* terakhir diakses.

DAFTAR PUSTAKA

[1] Almgren, Debar, Dacier.2001.*A Lightweight Tool For Detecting Web Server Attacks.* Ruschlikon. Switzerland.

[2] C.Wolf, J. Müller, M. Wolf, and D.P.D.J. Schwenk.2012.*Webforensik forensische Analyse Von Apache httpd log files*.

[3] Corona, 2009.*HMM-Web: A Framework for The Detection of Attacks Against Web Applications*. http://pralab.diee.unica.it/sites/default/files/Corona_ICC2009.pdf

[4] D. Doran and S.S. Gokhale.2011.*Web robot detection techniques: overview and limitations*. Data Mining and Knowledge Discovery, 22(1):183–210

[5] Firdaniza, 2006.*Hidden Markov Models.* http://eprints.uny.ac.id/7203/1/M-1 - Firdaniza, Nurul G, Akmal.pdf

[6] G.D. Forney Jr.1973.*The Viterbi Algorithm*.Proceedings of the IEEE, 61(3):268–278

[7] Honeynet Project. http://www.honeynet.org/challenges/2010_5_log_mysteries

[8] Kruegel, Vigna, Robertson.2005. *A Multi-model Approach To The Detection of Web-based Attack*. Elsevier.

[9] Munir, Rinaldi.2010. *Teori Dasar Hidden Markov Models*. http://informatika.stei.itb.ac.id/~rinaldi.munir/Probstat/2010-2011/Makalah2010/MakalahProbstat2010-025.pdf

[10] Salama and Marie.2011.*Web Server Logs Preprocessing for Web Intrusions Detection*. CIS vol 4 no 4: Canada.

[11] Sans Whitepapers. https://www.sans.org/reading-room/whitepapers/logging/detecting-attacks-web-applications-log-files-2074