Pemanfaatan Algoritma BFS dan Knuth-Morris-Pratt dalam Pendeteksian *Backdoor* Pada Server

Aryya Dwisatya Widigdha 13512043
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13512043@std.stei.itb.ac.id

***Abstract*—Backdoor merupakan sebutan bagi sebuah script yang dapat digunakan oleh seorang cracker untuk menjalankan *command* pada server yang padanya telah tertanam *backdoor* sehingga server tersebut berada dalam kendali *cracker*.**

Makalah ini membahas pendeteksian script *backdoor* pada sebuah server menggunakan algoritma Breadth First Search (BFS) untuk pembangkitan pohon link menuju file yang akan di cek dan algoritma Knuth-Morris-Pratt (KMP) untuk *pattern matching* antara pattern pengindikasi *backdoor* dengan isi dari sebuah file.

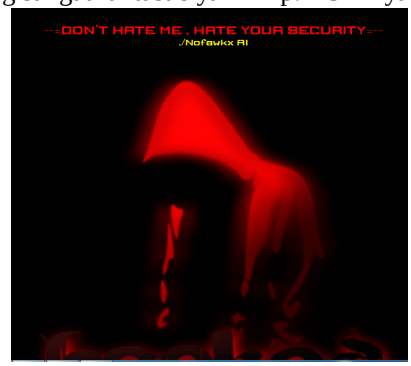
I. PENDAHULUAN

Perkembangan teknologi yang sangat pesat mengakibatkan penggunaan website sebagai sarana komunikasi dan publikasi menjadi semakin lumrah. Sayangnya, dalam penggunaan teknologi tersebut seringkali faktor keamanan menjadi topik yang tidak diprioritaskan. Padahal, faktor keamanan merupakan faktor yang harus menjadi perhatian utama demi keberlangsungan suatu website dan server yang menjadi tempat penyimpanannya.

Salah satu kasus keamanan yang sering ditemui pada sebuah website maupun server adalah adanya *backdoor* yang memungkinkan *attacker* untuk menguasai server selayaknya pengguna yang sah. Akibatnya, *attacker* dapat bertindak sesuka hati kepada stus yang ada di sana. Ia bisa merusak situs yang ada, menyadap informasi yang diberikan oleh user, bahkan men-*take-over* server untuk tujuan yang tidak dapat diperkirakan oleh pemilik server.

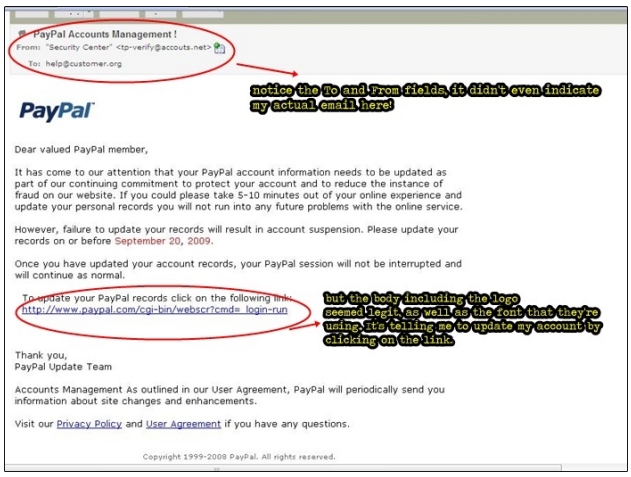
Salah satu akibat dari tertanamnya *backdoor* pada sebuah server adalah *defacing* yakni pengubahan halaman baik halaman utama maupun halaman lain dari suatu situs secara tidak legal dengan halaman lain yang dikehendaki oleh *attacker*.

Kasus *defacing* yang paling fenomenal adalah kasus situs KPU pada tahun 2004 yang mana nickname Xnuxer melakukan pembobolan terhadap website KPU dan selanjutnya mengganti nama-nama partai dengan nama-nama yang tak lazim dan lucu. Padahal menurut kabar yang beredar website KPU tersebut dibangun dengan budget yang sangat fantastis yakni Rp.125 milyar.



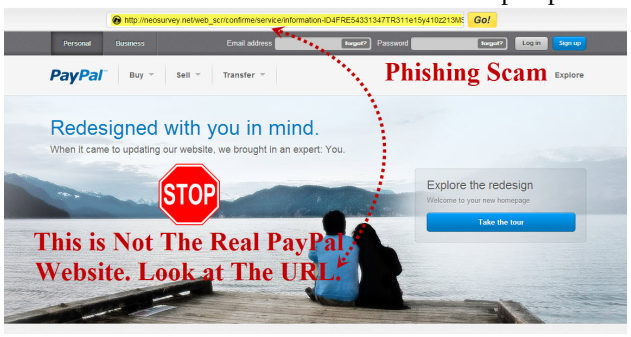
Gambar 1 - Contoh tampilan website yang di deface

Selain *defacing*, server yang terpasang *backdoor* juga dapat digunakan oleh pemilik *backdoor* untuk membuat server *proxy*, downloader, *bot* server, spam, dan phising. Bayangkan betapa berbahayanya ketika ada *cracker* yang melakukan aksi nya dengan menggunakan *proxy* server dari server yang kita miliki. Tentu, IP address yang tercatat adalah IP address kita dan hal tersebut akan menjadi sebuah masalah ketika ada laporan dari pihak yang dirugikan. Selain itu, ketika server digunakan sebagai downloader padahal ada keterbatasan *bandwidth* maka akan ditimbulkan kerugian yang tidak sedikit akbat bocornya *bandwidth* atas tindakan yang tidak diharapkan. Terlebih ketika dikombinasikan dengan bot server yang memungkinkan semua orang mengirim request ke bot agar server menjalankan aksi tertentu.



Gambar 2- Contoh email spam

Tidak hanya itu, hal mengerikan lain yang dapat terjadi ketika sebuah server telah terpasang *backdoor* adalah server tersebut dapat digunakan sebagai mesin untuk mengirim email spam. Akiabtnya, akan banyak keluhan yang datang akibat email spam yang terkirim dan jumlahnya pasti tidak sedikit. Keadaan ini dapat diperparah ketika server yang di take-over digunakan sebagai media penipuan atau phising. Akibatnya, bisa jadi pemilik server yang terjerat oleh hukum *cybercrime* lantaran memiliki halaman untuk melakukan penipuan.



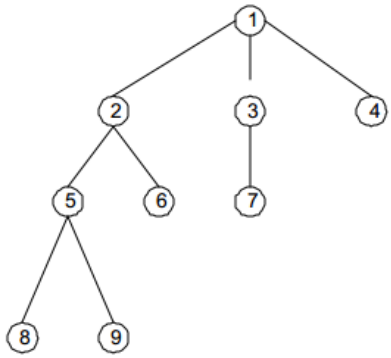
Gambar 3- Contoh Scam Page untuk Phising

Oleh karenanya, setiap server harus dapat dipastika terbebas dari *backdoor* yang dipasang oleh *attacker* demi terjaminnya keamanan server dan pengguna dari tindak ilegal oleh *attacker* yang tidak hanya dapat merugikan pemilik server, tetapi juga pengguna server.

II. DASAR TEORI

A. BFS (Breadth First Search)

BFS (Breadth First Search) merupakan algoritma pencarian dengan membangkitkan semua kemungkinan simpul pada suatu level untuk selanjutnya membangkitkan simpul pada level selanjutnya pada simpul yang paling awal dibangkitkan pada level yang sama.



Gambar 4 - Pembangkitan Simpul pada BFS

Algoritma BFS dapat diterapkan dengan mengikuti langkah-langkah berikut ini:

1. Kunjungi simpul 1
2. Kunjungi simpul anak dari simpul 1 terurut dari pembangkitan yang paling awal.
3. Kunjungi simpul anak dari simpul yang dikunjungi untuk selanjutnya mengunjungi simpul anak pada simpul tetangga.
4. Ulangi langkah tersebut dari simpul anak hingga tidak ada simpul lain yang dapat ditelusuri atau dibangkitkan.

Atau dalam pseudocode:

```
procedure BFS(inputv:integer)
{ Traversal graf dengan algoritma pencarian BFS.
Masukan: v adalah simpul awal kunjungan
Keluaran: semua simpul yang dikunjungi dicetak ke layar
}
Deklarasi
w: integer
q : antrian;
procedure BuatAntrian(input/outputq : antrian)
{ membuat antrian kosong, kepala(q) diisi 0 }
procedure MasukAntrian(input/outputq:antrian, inputv:integer)
{ memasukkan v ke dalam antrian q pada posisi belakang }
procedure HapusAntrian(input/outputq:antrian,output v:integer)
{ menghapus v dari kepala antrian q }
function AntrianKosong(inputq:antrian) ® boolean
{ truejika antrian q kosong, falsejika sebaliknya }

Algoritma:
BuatAntrian(q)    { buat antrian kosong }
write(v)  { cetak simpul awal yang dikunjungi }
dikunjungi[v]<--true  { simpul v telah dikunjungi, tandai dengantrue}
MasukAntrian(q,v) { masukkan simpul awal kunjungan ke dalam antrian}
{ kunjungi semua simpul graf selama antrian belum kosong }
while notAntrianKosong(q) do
        HapusAntrian(q,v)  { simpul v telah dikunjungi,
```

```
hapus dari antrian }
        For tiap simpul w yang bertetangga dengan simpul v do
                if not dikunjungi[w] then
                        write(w)  {cetak simpul yang dikunjungi}
                        MasukAntrian(q,w)
                        dikunjungi[w]¬true
                endif
        endfor
endwhile
{AntrianKosong(q) }
```

Dalam penerapan pada *programming*, BFS dapat diterapkan dengan memanfaatkan struktur data queue. Setiap simpul baru yang dibangkitkan dimasukkan ke dalam sebuah queue untuk selanjutnya ketika semua simpul pada level yang sama sudah dibangkitkan dan dimasukkan pada queue, dilakukan penelusuran pada simpul di head dari queue. Proses ini dilakukan terus menerus hingga queue kosong. Pada saat inilah proses pencarian menggunakan algoritma BFS berakhir.

B. KMP (Knuth-Morris-Pratt)

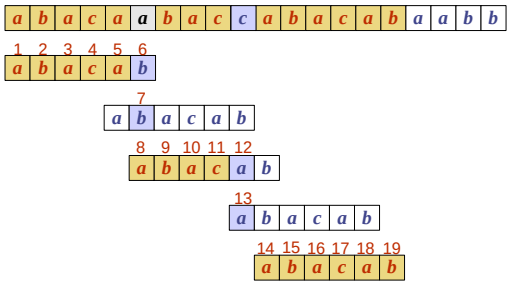
KMP (Knuth-Morris-Pratt) merupakan algoritma pencocokan *pattern* dalam sebuah teks dengan mencocokkan pattern dengan text dengan urutan dari kiri ke kanan. Urutan ini sama dengan urutan pencocokan pattern dengan algoritma brute force, tapi pada KMP terdapat penambahan fungsi sehingga pergeseran dan pencocokan terkesan lebih pintar.

Pada KMP terdapat fungsi pinggiran sebagai kecerdasan tambahan algoritma tersebut agar pencocokan kata menjadi semakin singkat dengan hasil yang maksimal. Fungsi pinggiran merepresentasikan panjang karakter dari prefix dan suffix yang sama dari suatu pattern. Sebagai contoh pattern P: “abaaba” didefinisikan fungsi pinggiran sebagai berikut:

J	1	2	3	4	5	6
P[j]	A	b	a	c	a	b
B(j)	0	0	1	0	1	0

J merupakan panjang karakter yang diberikan dari pattern. P[j] merupakan karakter ke-j pada pattern. B(j) merupakan nilai terbesar dari prefix dan suffix yang sama ketika panjang pattern adalah J.

Contoh penerapan KMP pada pencocokan string dengan text (T) : abacaabaccabacabaabb dengan P: abacab



Gambar 5- Proses pencocokan text menggunakan KMP

Kompleksitas Waktu KMP

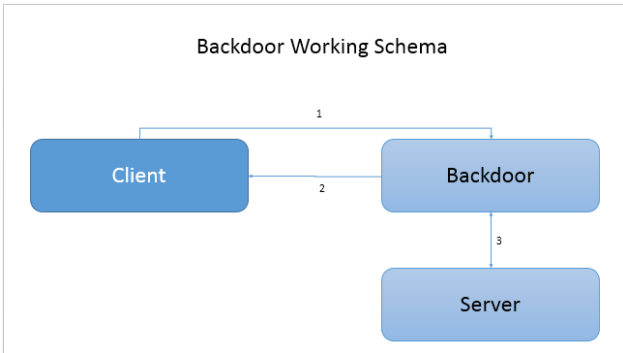
Untuk menghitung fungsi pinggiran KMP memiliki kompleksitas waktu O(m), untuk pencarian string kompleksitas waktunya O(n), sedangkan kompleksitas waktu keseluruhan adalah O(m+n). Sangat cepat dibandingkan algoritma lain yakni *bruteforce* yang tidak menerapkan prinsip “otak buatan”.

Kelebihan & kekurangan Algoritma KMP

Salah satu kelebihan algoritma KMP ketimbang algoritma brute force adalah adanya otak buatan yang memungkinkan tidak terjadinya pencocokan yang tidak berguna sehingga pencocokan menjadi semakin sedikit dan kompleksitas semakin kecil. Meskipun demikian, KMP memiliki kekurangan yakni kurang signifikan dalam memberikan hasil yang optimal ketika jumlah apphabet semakin banyak karena ada kemungkinan kesalahan pencocokan.

C. Backdoor

Backdoor merupakan sebuah alat yang digunakan untuk menerobos *autentifikasi* normal dengan jalan ilegal sehingga seseorang dimungkinkan untuk mengakses komputer secara remote bahkan men-take over komputer tanpa akses yang sah.Ada banyak macam *backdoor* yang dapat digunakan untuk mengakses komputer secara ilegal, tapi pada kasus ini hanya akan dibahas *backdoor* yang dibuat menggunakan PHP karena basis serangan ditujukan kepada web server.



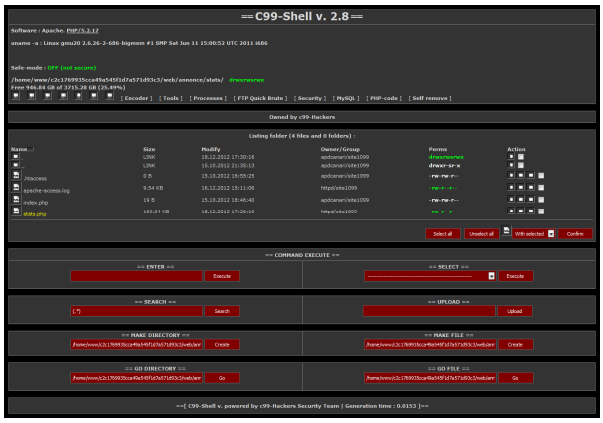
Gambar 6- Skema Kerja Backdoor

Skema kerja *backdoor* dapat dijelaskan dengan sederhana aykni sebagai berikut. Perbedaaan warna antara Client, *Backdoor*, dan Server menunjukkan perbedaan lokasi. Client dan server beada di mesin yang

berbeda sedangkan *backdoor* berada di mesin server.

Kerja *backdoor* dimulai ketika client mengirimkan request *command* yang harus dieksekusi oleh *backdoor*. Selanjutnya, *backdoor* akan mengeksekusi *command* yang diminta di server karena memang berada dalam satu mesin. Untuk setiap *command* yang dieksekusi terdapat respon yang berbeda-beda. Respon inilah yang akan dikirimkan oleh *backdoor* ke client sehingga *client* dan server seolah-oalh berinteraksi secara langsung.

Backdoor memberikan keleluasaan bagi seseorang untuk mengeksekusi perintah selayaknya pada command line sehingga secara tida langsung komputer berada pad akemndali pemilik *backdoor*. Pada PHP, ada berbagai fungsi yang dapat digunakan untuk menjalankan perintah seperti pada command line yakni fungsi system(), passthru(), exec(), dan shell_exec(). Dengan bermodal fungsi tersebut dan *backdoor* yang tertanam pada sebuah web server, seseorang dapat dengan mudah mengobrak-abrik sebuah server tanpa harus melewati autentifikasi.



Gambar 7- Contoh backdoor c99 yang tertanam di server

III. PEMBAHASAN

Untuk melakukan pendeteksian *backdoor* di server sebenarnya dapat dilogikakan dengan sederhana yakni dengan mengecek keseluruhan file .php yang terdapat pada suatu server apakah terdapat fungsi yang ditengarai digunakan untuk *backdoor*ing. Namun, logika yang terlalu luas tersebut menghasilkan adanya ketidak samaan hasil dalam berbagai percobaan, oleh karenanya pada kasus ini dilakukan standarisasi mengena algoritma yang akan digunakan dalam melakukan pendeteksian.

Untuk melakukan pendeteksian *backdoor* pada suatu web server dapat digunakan preudocode algoritma sebagai berikut:

```
procedure deteksiBackdoor(input
rootFolder : string)
{ Mencetak nama file yang ditengarai
sebagai backdoor yang terpasang pada
sebuah web server }

Deklarasi
    queueOfLink  : queue
    tmpLink :   string
Algoritma
    Bangkitkan ling yang ada di
rootFolder dengan BFS
    Masukkan semua link hasil
pembangkitan ke dalam queueOfLink
    while queueOfLink != Empty
        tmpLink =   dequeue
queueOfLink
        if(tmpLink mengandung .php)
            if(ketika dicek dengan KMP
terdapat fungsi untuk backdooring)
                link
            else
                do nothing
        else
            if(tmpLink adalah
direktori)
                bangkitkan link dan
masukkan ke dalam queue
            else
                do nothing
```

Pseudocode tersebut akan mencetak link file setiap kali ditemukan fugnsi untuk *backdoor*ing pada file. Sebagai pengaplikasian pseudocode tersebut diberikan sebuah direktory web dengan struktur direktory dan file seperti berikut:

- Backup
  - 2012
    - Desember
      - Index.php
  - 2013
    - Januari
      - Index.php
      - Trial.php
      - Connect.php
      - Readme.txt
    - Juni
      - Config.php
    - Desember
      - Index.old.php
  - 2014
- Config
  - Connect.php
- Css
  - Boostrap.css
- Files
- Images
  - Header.jpg
  - Footer.jpg
  - Background.jpg

Asumsikan file Config/Connect.php dan & Backup/2012/Desemer/Index.php mengandung fungsi PHP yang dapat digunakan sebagai *backdoor*.

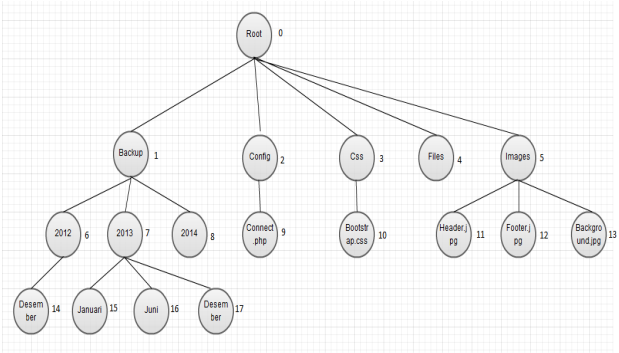
Pada kasus dengan struktur direktori diatas, pendeteksian akan dimulai dengan algoritma BFS yakni dengan memabgnkitkan link awal dari folder root yakni: Backup, Config, Css, Files, dan Images. Karena tidak ada link yang berekstensi .php maka pendeteksian akan dilakukan dengan membangkitkan link dari link yang pertama kali dibangkitkan yakni Backup yang akan menghasilkan 2012, 2013, dan 2014. Karena tidak ada file yang berekstensi .php penelusuran akan dilanjutkan dnegan membangkitkan link dari direktori kedua yakni Config yang akan menghasilkan file Connect.php. Karena ada file yang mengandung ekstensi .php maka dilakukan pengecekan isi file Connect.php menggunakan algoritma KMP untuk mencari fungsi php yang diindikasi digunakan sebagai *backdoor* yakni system(), exec(), passthru(), shell_exec(). Bila ditemukan pattern tersebut dalam file maka nama file akan ditampilkan dan penelusuran dilanjutkan.

Penelusuran dilanjutkan dengan membangkitkan Bootstrap.css dari folder CSS, karena file tersebut tidak berekstensi .php maka tidak dilakukan pengecekan. Pengecekan dilanjutkan dengan membangkitkan simpul anakan dari folder Images. Karena semua file berekstensi .jpg maka tidak dilakukan pengecekan pada file tersebut.

Sampai pada level tiga, folder desember dibangkitkan dari folder 2012, folder Januari, Juni, dan Desember dibangkitkan dari folder 2013, sedangkan pada folder induk 2014 tidak ada link yang dapat dibangkitkan.

Pada level keempat dilakukan penelusuran lafi dari anakan folder Desember yakni file index.php, karena terdapat ekstensi .php maka dilakukan pengecekan *backdoor*. Begitu pula pada anakan dari folder Januari yakni Index.php, Trial.php, dan Connect.php. Readme.txt tidak dicek karena tidak menganduk ekstensi .php. Pun, anakan dari folder Juni yakni Config.php serta anakan dari folder Desemver yakni index.old.php dilakukan pengecekan karena memiliki ekstensi file .php.

Berikut merupakan ruang pohon status pembangkitan:



Gambar 8 - Pohon Ruang Status BFS

Bila dilakukan eksekusi program maka akan dihasilkan keluaran seperti berikut.

```
Directory: C:\Users\adwisatya\Documents\NetBeansProjects\BackdoorScanner\src\root\Backup\2012\Desember
Index.php is a backdoor
Directory: C:\Users\adwisatya\Documents\NetBeansProjects\BackdoorScanner\src\root\Backup\2013\Desember
Index.old.php is not a backdoor
Directory: C:\Users\adwisatya\Documents\NetBeansProjects\BackdoorScanner\src\root\Backup\2013\Januari
Connect.php is a backdoor
Directory: C:\Users\adwisatya\Documents\NetBeansProjects\BackdoorScanner\src\root\Backup\2013\Januari
Index.php is not a backdoor
Directory: C:\Users\adwisatya\Documents\NetBeansProjects\BackdoorScanner\src\root\Backup\2013\Januari
Readme.txt is not php file
Directory: C:\Users\adwisatya\Documents\NetBeansProjects\BackdoorScanner\src\root\Backup\2013\Januari
Trial.php is not a backdoor
Directory: C:\Users\adwisatya\Documents\NetBeansProjects\BackdoorScanner\src\root\Backup\2013\Juni
Config.php is not a backdoor
Directory: C:\Users\adwisatya\Documents\NetBeansProjects\BackdoorScanner\src\root\Backup
2014 is not php file
Directory: C:\Users\adwisatya\Documents\NetBeansProjects\BackdoorScanner\src\root
Config is not php file
Directory: C:\Users\adwisatya\Documents\NetBeansProjects\BackdoorScanner\src\root
Css,Bootstrap.css is not php file
Directory: C:\Users\adwisatya\Documents\NetBeansProjects\BackdoorScanner\src\root
Files is not php file
Directory: C:\Users\adwisatya\Documents\NetBeansProjects\BackdoorScanner\src\root
Images is not php file
Directory: C:\Users\adwisatya\Documents\NetBeansProjects\BackdoorScanner\src\root\Images
Header.jpg is not php file
Directory: C:\Users\adwisatya\Documents\NetBeansProjects\BackdoorScanner\src\root\Images
Footer.jpg is not php file
Directory: C:\Users\adwisatya\Documents\NetBeansProjects\BackdoorScanner\src\root\Images
Background.jpg is not php file
BUILD SUCCESSFUL (total time: 0 seconds)
```

Gambar 9- Hasil Eksekusi Program

Pada gambar hasil eksekusi nampak bahwa dilakukan pengecekan pada setiap link yang ada. Pengecekan pertama adalah mengecek apakah link tersebut berekstensi .php selanjutnya bila memenuhi barulah dilakukan pengecekan isi dari file tersebut untuk mencari fungsi-fungsi yang dapat digunakan untuk *backdoor*ing menggunakan algoritma KMP.

IV. KESALAHAN UMUM

Kesalahan umum yang mungkin terjadi dalam percobaan pendeteksian *backdoor* pada sever menggunakan algoritma BFS dan KMP adalah pendefinisian urutan simpul yang dibangkitkan pada algoritma BFS dan tabel pembatas pada KMP.

V. KESIMPULAN

Berdasarkan pengelitian, percobaan, dan pembahasan tersebut didapatkan bahwa algoritma BFS dan KMP dapat digunakan untuk melakukan pendeteksian *backdoor* pada suatu web server. Hanya saja, cara ini tidak selalu berhasil ketika digunakan fungsi lain yang belum didefinisikan atau dimasukkan ke dalam daftar fungsi yang di larang.

Dalam penelitian selanjutnya, diharapkan dilakukan pula analisa untuk fungsi lain yang ditengarai dapat digunakan untuk *backdoor*ing sehingga akurasi pendeteksian bisa meningkat.

VI. UCAPAN TERIMA KASIH

Puji syukur atas semua yang telah Allah berikan sehingga makalah ini dapat selesai dengan cepat dan tepat. Tidak pula lupa saya memberikan ucapan terima kasih kepada Pak Rinaldi Munir yang menjadikan tugas menulis makalah sebagai salah satu tugas wajib kuliah sehingga semangat menulis saya semakin tinggi. Tidak lupa pula terima kasih kepada Bu Masayu yang sudah membimbing saya selama satu semester ini sehingga dengan amteri yang beliau berikan makalah ini dapat tersusun dengan baik.

REFERENCES

[1] Munir, Rinaldi. "Diktat Kuliah IF2211 Strategi Algoritma", Program Studi Teknik Informatika Sekolah Teknik Elektro dan InformatikaBandung, 2009.

[2] Dr. Andrew Davison, "Pattern Matching, WiG Lab, CoE,2006-2007.

[3] "How to Find *Backdoor* PHP shell script on a server".Anrej. http://msfbt.wordpress.com/tag/*backdoor*/.Diakses pada 14 Mei 2014.

[4] "Java *Backdoor* Scanner".Aryya Dwisatya Widigdha. https://github.com/adwisatya/JBS. Diakses pada 18 Mei 2014.

[5] http://www.zone-h.org/mirror/id/22376117. Diakses pada 16 Mei 2014.

SUMBER GAMBAR

[1] Contoh tampilan website yang di deface: http://www.zone-h.org/mirror/id/22376117

[2] Contoh email spam : http://2.bp.blogspot.com/-Jqrn26-knUY/UNCdK-_osII/AAAAAAAAAC0/SrQdsaTYeG8/s1600/c99-hackers.png

[3] Contoh Scam Page untuk Phising : http://1.bp.blogspot.com/-zgxhzx9Y3N0/UR_w3k0p2pI/AAAAAAAACn0/CNIB2G-yvtA/s1600/2-16-2013+2-39-23+PM.jpg

[4] Contoh *backdoor* c99 yang tertanam di server : http://2.bp.blogspot.com/-Jqrn26-knUY/UNCdK-_osII/AAAAAAAAAC0/SrQdsaTYeG8/s1600/c99-hackers.png

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 19 Mei 2014

Aryya Dwisatya Widigdha 13512043