Analisis dan Implementasi *Honeypot* Menggunakan *Dionaea* Sebagai Penunjang Keamanan Jaringan

Triawan Adi Cahyanto[1)], Hardian Oktavianto[2)], Agil Wahyu Royan[3)]
[1,2,3)]*Jurusan Teknik Informatika, Fakultas Teknik, Universitas Muhammadiyah Jember*
Email : [1)]triawanac@unmuhjember.ac.id, [2)]hardian@unmuhjember.ac.id,
[3)]agilwahyu.r@gmail.com

ABSTRAK

Honeypot merupakan salah satu paradigma baru dalam keamanan jaringan yang bertujuan untuk mendeteksi kegiatan yang mencurigakan, membuat jebakan untuk penyerang (*attacker*) serta mencatat aktivitas yang dilakukan penyerang. *Dionaea* merupakan salah satu kategori *honeypot low interaction* sebagai penerus *Nephentes. Dionaea* membuat emulasi layanan palsu yang akan dijadikan sebagai target utama serangan. Penelitian yang dilakukan dengan membuat simulasi terhadap kinerja sistem. *Honeypot* dibangun menggunakan sistem operasi pada lingkungan *virtual*. Pengujian sistem menggunakan teknik penyerangan *port scanning* dan *exploit* layanan sistem. Hasil penyerangan akan tersimpan pada *log* yang terdapat pada *honeypot*. *Dioanea* berhasil diterapkan untuk menjebak penyerang dimana data penyerangan yang tercatat pada *log* berupa *exploitasi* ke MySQL, Layanan SMB dan Layanan MSRPC.

Kata kunci: *Honeypot, Dionaea, Exploit,* Keamanan Jaringan

1. PENDAHULUAN

Perkembangan teknologi telah menjadikan salah satu media seperti internet menjadi media yang utama dalam pertukaran informasi. Tidak semua informasi dapat diakses untuk umum. Internet merupakan jaringan luas dan bersifat publik, oleh karena itu diperlukan suatu usaha untuk menjamin keamanan informasi terhadap data atau layanan yang menggunakan internet (Cahyanto, 2015). *Honeypot* merupakan sistem yang didesain menyerupai sistem yang asli dan dibuat dengan tujuan untuk diserang atau disusupi sehingga sistem yang asli tetap aman dan terhindar dari serangan (Umayah, 2012). Trafik jaringan yang menuju sistem asli akan dialihkan menuju *Honeypot*, sehingga semua trafik yang menuju ke *Honeypot* layak dicurigai sebagai trafik yang berupaya melakukan serangan atau trafik normal. Sistem *honeypot* memungkinkan untuk melakukan pendeteksian terhadap trafik tersebut, dengan cara melakukan pengawasan intensif. *Honeypot Dionaea* berlisensi kode terbuka (*open source*). Penelitian ini bertujuan untuk melakukan implementasi *honeypot* menggunakan *Dionaea* ke sistem *virtual*.

2. TINJAUAN PUSTAKA

2.1 *Honeypot*

Honeypot adalah suatu cara membuat sistem palsu atau layanan palsu yang berfungsi untuk menjebak pengguna yang mempunyai tujuan buruk atau menangkal usaha-usaha yang dapat merugikan sistem atau layanan (Nugroho, 2013). *Honeypot* merupakan pengalih perhatian penyerang, agar penyerang seolah-olah berhasil membobol dan mengambil data dari sebuah jaringan, padahal sesungguhnya data tersebut tidak penting dan lokasi tersebut sudah terisolir (Purbo, 2008). Saat ini, *honeypot* tidak hanya berfungsi atau bertujuan untuk menjebak penyerang, namun juga

bermanfaat untuk para *administrator* maupun *security analyst* dalam rangka menganalisa aktivitas apa saja yang dilakukan oleh penyerang ketika mengakses sistem *honeypot*. Secara umum terdapat dua tipe *honeypot*, yaitu:

1. *Low Interaction Honeypot*, tipe *honeypot* yang dibuat untuk mensimulasikan *service* (layanan) seperti pada *server* yang asli. Misalnya *Service FTP, Telnet, HTTP*, dan *service* lainnya.
2. *High Interaction Honeypot*, tipe *honeypot* yang menggunakan keseluruhan *resource* sistem, dimana *honeypot* yang dibangun nanti benar-benar persis seperti sistem yang asli. *Honeypot* jenis ini bisa berupa satu keseluruhan sistem operasi beserta aplikasi yang berjalan didalamnya.

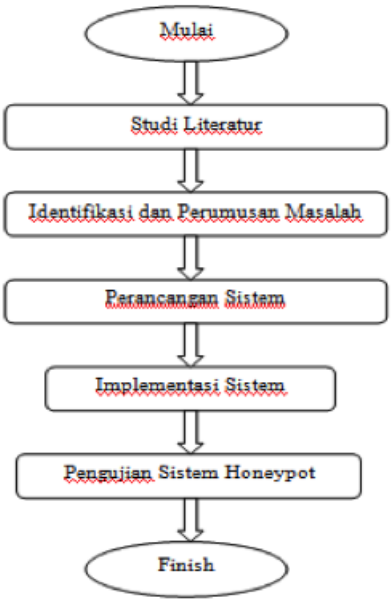
2.2 *Dionaea*

Dionaea adalah *honeypot* yang bersifat *Low Interaction Honeypot* yang diciptakan sebagai pengganti *Nepenthes* (Sentanoe, 2015). *Dionaea* menggunakan bahasa pemrograman *python* sebagai bahasa *scripting*, *libemu* untuk mendeteksi *shellcode*, mendukung *Ipv6* dan *TLS*. *Dionaea* bertujuan untuk mendapatkan duplikasi data dari *malware* (Ion, 2015). Perangkat lunak (*software*) cenderung memiliki *bug*, yang seringkali dapat dieksploitasi oleh pihak lain untuk memperoleh informasi atau keuntungan.

Dionaea memiliki kemampuan untuk mendeteksi dan mengevaluasi *payload* agar dapat memperoleh salinan *malware*. Dalam mendeteksi *payload*, *dionaea* menggunakan *libemu*. Setelah *dionaea* memperoleh lokasi berkas yang diinginkan penyerang agar diunduh dari *shellcode*, *dionaea* akan mencoba untuk mengunduh berkas tersebut. Protokol untuk mengunduh berkas tersebut menggunakan *tftp* dan *ftp* yang diimplementasikan menggunakan bahasa pemrograman *python* (*tftp.py* dan *ftp.py*) sebagai bagian dari *dionaea*. Berkas diunduh melalui *http* yang dilakukan dalam modul *curl* yang memanfaatkan *libcurl http*.

3. METODE PENELITIAN

3.1 Kerangka Konsep Penelitian



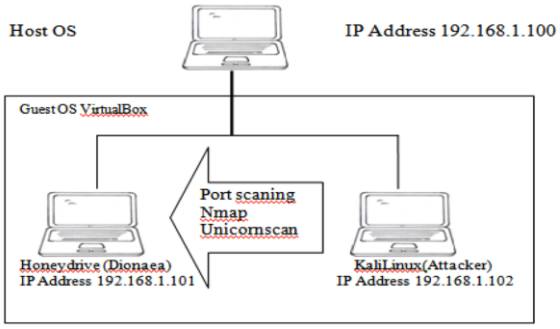
Gambar 1. Kerangka Konsep Penelitian

Konsep penelitian terdiri dari lima tahap. Tahap pertama adalah studi literatur untuk pencarian informasi tentang sumber pustaka, *paper* dari konferensi maupun jurnal, dan buku-buku baik cetak maupun elektronik yang berkaitan dengan topik penelitian. Tahap kedua adalah identifikasi dan perumusan masalah, Tahap ketiga adalah perancangan sistem untuk membuat rancangan sistem *honeypot* pada sistem *virtual* beserta konfigurasi perangkat lunak yang dibutuhkan. Tahap keempat adalah implementasi sistem, dimana melakukan konfigurasi aplikasi dan perangkat yang sudah dirancang kemudian mensimulasikan *port-port* yang dilakukan untuk melakukan penyerangan. Tahap

terakhir adalah pengujian sistem honeypot, dimana dilakukan pengujian serangan dengan teknik *port scanning* dan eksploitasi layanan yang ada pada sistem *honeypot*. Kerangka konsep penelitian digambarkan dalam Gambar 1.

3.2 Topologi Jaringan

Walaupun implementasi sistem ini menggunakan *virtual*, namun tetap harus dibuat topologi jaringan sistem, agar sistem yang berjalan dan pengujian sistem dapat efektif (Purnomo, 2010).



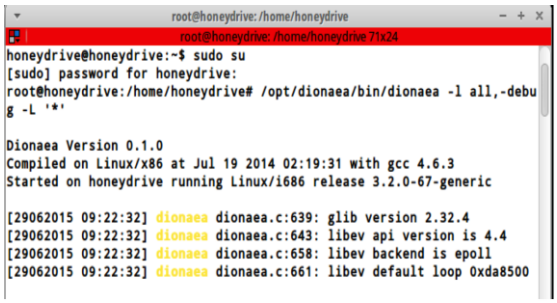
Gambar 2. Topologi Jaringan *Honeypot Dionaea*

Topologi jaringan sederhana sebagaimana Gambar 2 diatas adalah topologi sistem *virtual* antara *guest* dengan *host*. PC *Server* dengan *dionaea* terdapat pada IP 192.168.1.101 yang berfungsi untuk mengalihkan trafik dari penyerang. PC *Client* (penyerang) dengan sistem operasi *kali linux* terdapat pada IP 192.168.1.102, berguna untuk mensimulasikan penyerangan terhadap *host* yang dibuat oleh *dionaea* menggunakan teknik eksploitasi layanan dan teknik *port scanning*. Simulasi penyerangan tersebut nantinya akan menghasilkan *log* yang dapat digunakan untuk melakukan analisa sistem.

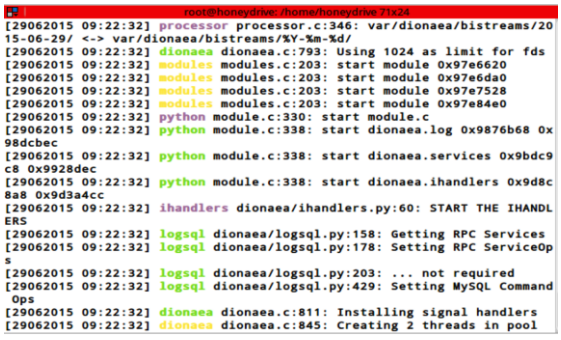
4. HASIL DAN PEMBAHASAN

4.1 Konfigurasi Dionaea

Dionaea membutuhkan konfigurasi agar dapat berjalan sesuai dengan rancangan sistem. Konfigurasi *Dionaea* menggunakan mesin *virtual* dengan distro *Honeydrive*. Gambar 3 dan 4 merupakan hasil konfigurasi *dionaea*:



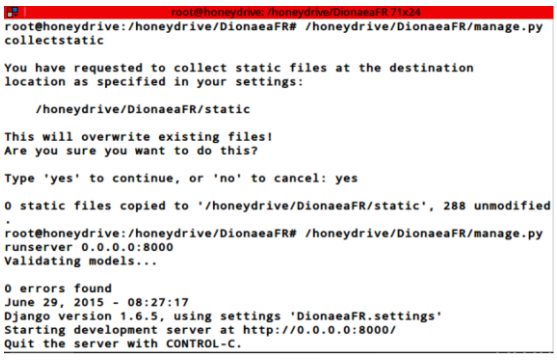
Gambar 3. Konfigurasi Dionaea



Gambar 4. Dionaea Berhasil Dijalankan

4.2 Konfigurasi DionaeaFR

Untuk melakukan konfigurasi *DionaeaFR*, kumpulkan berkas statis yang dibutuhkan oleh *DionaeaFR*, kemudian jalankan perintah: */opt/dionaeaFR/manage.py collectstatic*. Setelah pengumpulan data statis selesai, lalu *DionaeaFR* dapat dijalankan dengan perintah: */opt/dionaeaFR/manage.py runserver* 0.0.0.0:8000



Gambar 5. DionaeaFR Berhasil Dijalankan

4.2 Pengujian Serangan

Pengujian serangan akan disimulasikan sesuai dengan topologi jaringan yang sudah dibuat. PC *client* akan melakukan serangan dengan teknik *port scanning* dan *exploit*. Kedua teknik serangan dijelaskan sebagai berikut ini.

Pengujian pertama, teknik *port scanning* bertujuan untuk mengetahui *port* mana saja yang terbuka pada sistem (Cahyanto, 2014). Perangkat lunak yang digunakan untuk mengetahui *port* mana saja yang terbuka adalah *nmap* dan *unicornscan*. *Nmap* (*Network Mapper*) adalah aplikasi atau *tool* yang berfungsi untuk melakukan *port scanning*. Aplikasi ini digunakan untuk mengaudit jaringan yang ada, sehingga dapat melihat *host* yang aktif di jaringan, *port* yang terbuka. Hasil *port scanning* sistem ditunjukkan dalam Gambar 6.

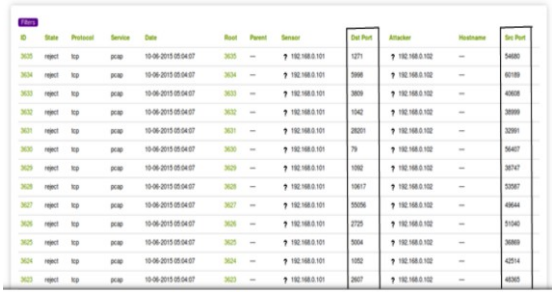
```
root@kali:~# nmap 192.168.1.101

Starting Nmap 6.47 ( http://nmap.org ) at 2015-06-29 04:42 EDT
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled.
 Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 192.168.1.101
Host is up (0.00046s latency).
Not shown: 989 closed ports
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
42/tcp   open  nameserver
80/tcp   open  http
135/tcp  open  msrpc
443/tcp  open  https
445/tcp  open  microsoft-ds
1433/tcp open  ms-sql-s
5060/tcp open  sip
5061/tcp open  sip-tls
8000/tcp open  http-alt
MAC Address: 08:00:27:38:D1:EC (Cadmus Computer Systems)

Nmap done: 1 IP address (1 host up) scanned in 0.73 seconds
```

Gambar 6. Hasil Scanning Menggunakan Nmap

Pada saat melakukan serangan *port scanning* dengan *Nmap*, *Dionaea* mencatat semua aktivitas yang dilakukan oleh *Nmap*. Setiap serangan ke *port* tertentu akan diberikan *attackid* sehingga dapat diketahui detail tiap serangan dan jumlahnya. Gambar 7 menampilkan hasil catatan *log* yang berhasil tersimpan *honeypot dionaea*. Gambar tersebut merupakan serangkaian serangan yang dilakukan oleh *Nmap* dengan cara melakukan *port scanning* TCP dengan sumber *port* yang sama.



Gambar 7. Hasil log yang dicatat oleh Dionaea

Sedangkan *Unicornscan* adalah aplikasi yang secara fungsional sama seperti *nmap*, hanya saja aplikasi ini berjalan dalam bentuk *command line*. Penggunaan *unicornscan* melengkapi hasil yang tidak berhasil diperoleh *nmap*. Gambar 8 menampilkan perintah yang digunakan *unicornscan* untuk mencari *port* yang terbuka. *Unicornscan* menemukan *port UDP* dari mesin target yang terbuka yaitu *port* dengan nomor 55208 dengan alamat IP 192.168.0.101.

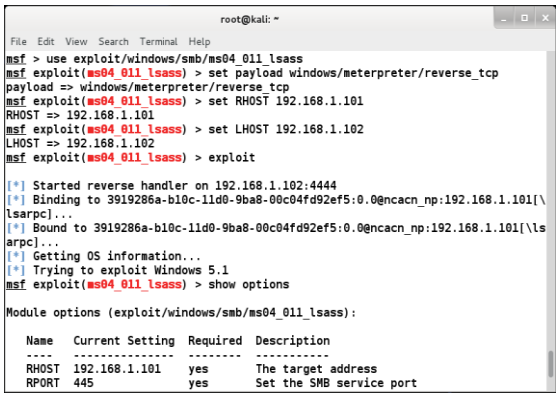
```
root@kali:~# unicornscan -i eth0 -E 192.168.1.101 -m U
ICMP closed            hosts2-ns[   81]        from 192.168.1.101  ttl 64
ICMP closed                talk[  517]        from 192.168.1.101  ttl 64
ICMP closed       av-emb-config[ 2050]        from 192.168.1.101  ttl 64
ICMP closed             unknown[32767]        from 192.168.1.101  ttl 64
ICMP closed          filenet-tms[32768]        from 192.168.1.101  ttl 64
ICMP closed          filenet-rpc[32769]        from 192.168.1.101  ttl 64
UDP open               unknown[60872]        from 192.168.1.101  ttl 64
root@kali:~#
```

Gambar 8. Hasil Scanning Menggunakan Unicornscan

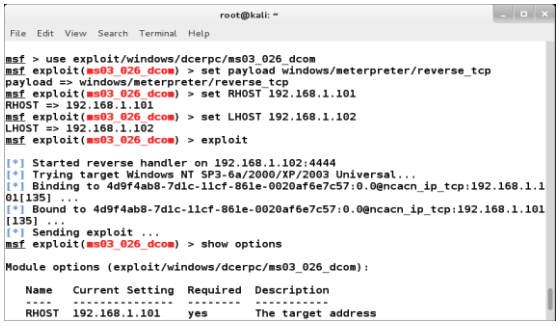
Pengujian kedua adalah eksploitasi layanan menggunakan teknik *exploit* yang terdapat pada *Metasploit Framework*. *Metasploit Framework* merupakan *tools* untuk melakukan eksploitasi terhadap sistem operasi *windows* berdasarkan kelemahan perangkat lunak. Eksploitasi layanan terdiri dari MS04_011_LSASS, MS03_026_DCOM, MySQL_Payload. *Exploit MS04_011_LSASS* merupakan eksploitasi layanan SMB pada *port* 445. Layanan SMB merupakan layanan yang dapat digunakan untuk melayani fitur *file sharing* atau *printer sharing* pada sistem

operasi *windows*. Gambar 9 menampilkan hasil eksploitasi layanan SMB menggunakan *Metasploit Framework.*

Pengujian kedua adalah eksploitasi layanan menggunakan teknik *exploit* yang terdapat pada *Metasploit Framework*. *Metasploit Framework* merupakan *tools* untuk melakukan eksploitasi terhadap sistem operasi *windows* berdasarkan kelemahan perangkat lunak. Eksploitasi layanan terdiri dari MS04_011_LSASS, MS03_026_DCOM, MySQL_Payload. *Exploit MS04_011_LSASS* merupakan eksploitasi layanan SMB pada *port* 445. Layanan SMB merupakan layanan yang dapat digunakan untuk melayani fitur *file sharing* atau *printer sharing* pada sistem operasi *windows*. Gambar 9 menampilkan hasil eksploitasi layanan SMB menggunakan *Metasploit Framework.*



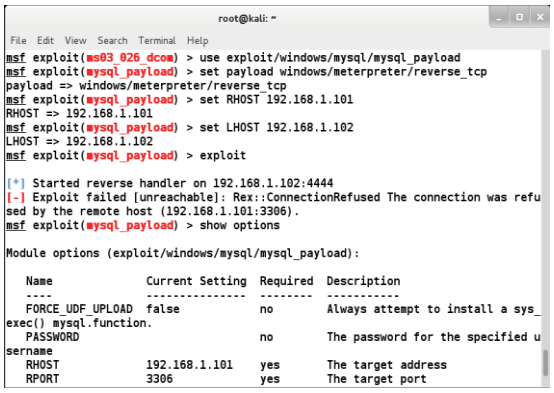
Gambar 9. Eksploitasi MS04_011_LSASS



Gambar 10. Eksploitasi MS03_026_DCOM

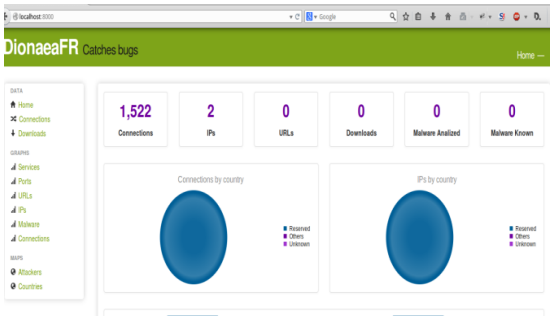
Exploit *MS03_026_DCOM* merupakan eksploitasi layanan MSRPC (*Microsoft Remote Procedure Calls*) pada *port* 135. Gambar 10 menampilkan hasil eksploitasi layanan MSRPC.

Exploit MySQL_Payload merupakan eksploitasi pada layanan basis data MySQL menggunakan *port* 3306. Gambar 11 menampilkan hasil eksploitasi dengan *MySQL_Payload*.



Gambar 11. Eksploitasi MySQL_Payload

Pengujian ketiga serangan tersebut dilakukan ke *server* yang sudah dibuat sesuai dengan desain topologi jaringan. *Server* yang diserang adalah *honeypot dionaea*. *Dionaea* dapat mencatat aktivitas serangan, seperti pada Gambar 12.



Gambar 12. Statistik Serangan Dionaea

Pada gambar diatas, *Dionaea* berhasil menangkap dan mengenali serangan yaitu pada IP target serangan (192.168.1.101) dan IP penyerang (192.168.1.102) serta jumlah data yang tersimpan sebanyak 1.522. *Honeypot Dionaea* telah berhasil membuat layanan

palsu sebagai target serangan dan mencatat serangan/aktivitas yang dianggap membahayakan sistem.

5. KESIMPULAN DAN SARAN

Dari hasil pengujian dapat disimpulkan bahwa Dionaea dapat digunakan sebagai *server* palsu atau *server* tiruan sehingga dapat melindungi *server* asli ketika *server* tiruan tersebut mengalami serangan. Pengujian *server* tiruan tersebut berbasis *Dionaea* menggunakan *Metasploit Framework*, dan melibatkan tiga teknik *exploit* yaitu *MS04_011_LSASS, MS03_026_DCOM,* dan *MySQL_Payload.* Berdasarkan simulasi serangan yang sudah dikerjakan, dapat diketahui bahwa penggunaan *honeypot* dapat menunjang keamanan jaringan, namun *honeypot* tidak dapat melindungi sistem operasi khususnya *windows*. Ditemukan banyak kelemahan pada sisi aplikasi, sehingga sisi kelemahan pada aplikasi tersebut dapat dimanfaatkan oleh penyerang untuk menguasai sistem seperti yang sudah ditunjukkan pada pengujian serangan.

Penelitian selanjutnya disarankan memperhatikan implementasi *honeypot* harus seimbang antara keamanan pada aspek jaringan dengan keamanan pada aspek sistem operasi, karena teknologi selalu berkembang maka tingkat keamanan sistem operasi selalu berkembang dan sistem operasi selalu diperbaharui. Kelemahan penggunaan *honeypot* hanya berfungsi untuk membuat sistem tiruan adalah jika konfigurasi sistem tiruan ke sistem asli dapat diketahui maka sistem asli juga dapat diketahui, sehingga disarankan untuk melakukan konfigurasi tingkat *advance*. Disamping itu *honeypot* akan lebih baik lagi apabila dikombinasikan dengan *firewall* dan IDS (*Intrusion Detection System*) sehingga ketika penyerang ingin melakukan serangan maka diharapkan sudah ditangani oleh *firewall* dan IDS.

DAFTAR PUSTAKA

Arief, Muhammad.2012. *Implementasi Honeypot Dengan Menggunkan Dionaea Dijaringan Hotspot FIZZ*. Politeknik Telkom: Bandung

Bruteforce Lab Team.*Honeydrive*.Diakses Tanggal 02 April 2015 http://bruteforce.gr/honeydrive

Cahyanto, T.A., 2015. BAUM-WELCH Algorithm Implementation For Knowing Data Characteristics Related Attacks On Web Server Log. *Proceeding IC-ITECHS 2014*, *1*(01).

Cahyanto, T.A. and Prayudi, Y., 2014, June. Investigasi Forensika Pada Log Web Server untuk Menemukan Bukti Digital Terkait dengan Serangan Menggunakan Metode Hidden Markov Models. In *Seminar Nasional Aplikasi Teknologi Informasi (SNATI)*.

Dionaea Project Team.*Dionaea*. Diakses tanggal 17 Maret 2015 http://dionaea.carnivore.it/

Ion.*Visualizing Dionaea's results with DionaeaFR*. Diakses tanggal 15 Maret 2015 http://bruteforce.gr/visualizing-dionaeas-results-with-dionaeafr.html

Nugroho, Ardianto Setyo.2013.*Analisis Dan Implementasi Honeypot Menggunakan Honeyd Sebagai Alat Bantu Pengumpulan Informasi Aktivitas Serangan Pada Jaringan*.Institut Sains & Teknologi AKPRIND: Yogyakarta.

Purbo, Onno W.2008.*Keamanan Jaringan Internet*. Jakarta: PT Elex Media Komputindo.

Purnomo,2010.*Membangun Virtual PC Dengan VirtualBox*. Penerbit Andi: Yogyakarta.

Sentanoe, Stewart.*Instalasi Dionaea*. Diakses tanggal 02 Maret 2015

http://honeynet.idsirtii.or.id/honeynet/?p=129

Umayah, Nurhasanah.2012.*Perancangan dan Implementasi Honeypot pada Virtual Private Server sebagai Penunjang Keamanan Jaringan*. Politeknik Telkom:Bandung.