REPOSITOR, Vol. 2, No. 4, April 2020, Pp. 475-484
ISSN : 2714-7975
E-ISSN : 2716-1382

Implementasi Multiple Honeypot dengan Raspberry Pi dan Visualisasi Log Honeypot mengunakan ELK Stack

Ilfan Arif Romadhan*[1], Syaifudin[2], Denar Regata Akbi[3]
[1,2,3]Teknik Informatika/Universitas Muhammadiyah Malang
ilfanarif@outlook.com*[1], saifuddin@umm.ac.id[2], dnarregata@umm.ac.id[3]

Abstrak

Perlindungan terhadap keamanan jaringan merupakan hal yang sangat penting untuk dilakukan. Mengingat kemudahan dalam mengakses jaringan memungkinkan adanya gangguan dari pihak yang ingin menyerang, merusak, bahkan mengambil data penting. Honeypot memang tidak menyelesaikan masalah pada keamanan jaringan, namun honeypot membuat penelitian tentang serangan menjadi lebih sederhana dengan konsep yang mudah untuk dimengerti dan dimplementasikan. Penelitian ini menerapkan beberapa honeypot menggunakan Raspberry pi dan ELK *stack untuk monitoring hasil yang didapatkan oleh honeypot. Tujuan dari penelitian ini untuk merancang sistem yang mampu mendeteksi serangan pada jaringan menggunakan honeypot. Raspberry pi digunakan sebagai sensor honeypot untuk pemantauan ancaman keamanan terbukti hemat biaya dan efektif menggantikan komputer desktop.* ELK *stack memudahkan pemusatan data dari berbagai sumber dan membuat analisis log yang awalnya rumit untuk dianalisis menjadi lebih menarik.*

Kata Kunci: *Keamanan jaringan, Multiple Honeypot, Raspberry Pi, ELK Stack*

Abstract

Protection of network security is very important to do. Given the ease in accessing the network allows for interference from parties who want to attack, destroy, and even retrieve important data. Honeypot does not solve the problem on network security, but the honeypot makes research about attacks become simpler with concepts that are easy to understand and implement. This research applies some honeypot using Raspberry pi and ELK stack for monitoring result obtained by honeypot. The purpose of this research is to design a system capable of detecting attacks on a network using a honeypot. Raspberry pi is used as a honeypot sensor for monitoring proven cost-effective and cost-effective security threats to replace desktop computers. The ELK stack facilitates the convergence of data from multiple sources and makes log analysis initially complex for analysis to be more interesting.

Keywords: *Network Security, Multiple Honeypot, Raspberry Pi, ELK Stack*

1. Pendahuluan

Seiring meningkatnya ilmu pengetahuan dan perkembangan teknologi informasi, akses data dalam jaringan menjadi sangat mudah. Salah satu kemudahan dalam teknologi pada saat ini adalah internet, orang sangat bergantung pada internet untuk menyebarkan informasi yang berharga [1]. Di sisi lain, karena ketergantungan yang tinggi pada internet, beberapa orang memanfaatkan kelemahan internet untuk memberikan gangguan kepada pengguna lainya.

Gangguan terjadi karena adanya pihak yang ingin menyerang, merusak, bahkan mengambil data-data penting. Gangguan tersebut umumnya diketahui dari gejala aneh yang terjadi. Kurangnya informasi tentang penyerang seperti siapa yang menyerang, mengapa mereka menyerang, bagaimana mereka menyerang, dan kapan serangan dilakukan, menjadi masalah yang patut untuk dicermati.

Untuk menangani hal tersebut, dibutuhkan alat bantu untuk mendeteksi serangan yang masuk ke dalam jaringan. Salah satu alat bantu dalam keamanan jaringan yang bisa digunakan adalah *honeypot*. *Honeypot* merupakan sistem yang sengaja digunakan dengan harapan untuk diserang dan dieksploitasi [2]. *Honeypot* mempunyai nilai tambah dalam penelitian untuk mempelajari ancaman dan resiko keamanan jaringan. *Administrator* dapat menganalisa aktivitas penyerang menggunakan *honeypot*. Secara umum, *honeypot* dibagi menjadi tiga tingkat, yaitu

low interaction, medium interaction dan *high interaction*. Semakin tinggi tingkat interaksi pada *honeypot*, maka semakin besar data yang ditangkap dan semakin besar juga resiko yang diterima [3].

Biasanya *honeypot* diimplementasikan menggunakan satu komputer *desktop* atau lebih, namun dalam penelitian ini *honeypot* akan dipasang pada *raspberry pi*. Keuntungan menggunakan *raspberry pi* antara lain mempunyai harga yang relatif lebih murah dan mengkonsumsi daya yang lebih sedikit daripada menggunakan komputer *desktop* [4]. *Raspberry pi* dapat digunakan sebagai sensor *honeypot* untuk pemantauan keamanan jaringan menggantikan komputer *desktop* pada umumnya [5]. Selain itu, *raspberry pi* mudah disesuaikan, dan bisa diletakkan dimana saja karena ukurannya yang cukup kecil.

Seiring dengan sulitnya menganalisis *log* yang dihasilkan oleh *honeypot*, maka dibutuhkan alat visualisasi untuk mempermudah dalam menganalisis *log honeypot*. Dalam penelitian ini, hasil serangan *honeypot* divisualisasikan menggunakan ELK *stack*, dimana ELK *stack* ini adalah kombinasi dari *elasticsearch, logstash* dan *kibana* [6].

1.1 *Sumber log*

Dalam penelitian ini *log* didapatkan dari beberapa *honeypot* yang diimplementasikan pada *raspberry pi. Honeypot* tersebut terdiri dari *dionaea, suricata,* dan *cowrie.*

1. ***Dionaea****, honeypot* yang mengemulasi layanan untuk menangkap malware dengan target protokol seperti *Server Message Block* (SMB) [7]. *Honeypot dionaea* menjebak eksploitasi *malware* melalui kerentanan yang sengaja dibuka untuk diserang, tujuan utamanya adalah mendapatkan salinan *malware* [8].
2. ***Suricata***, honeypot yang mampu mendeteksi *Intrusion Detection System* (IDS), *Intrusion Prevention System* (IPS), *Network Security Monitoring* (NSM) dan *Packet Capture* (PCAP). Di buat *Open Information Security Foundation* (OISF) pada tahun 2009 [9].
3. ***Cowrie,*** *medium interaction honeypot* yang dikembangkan oleh Michel Oosterhof pada tahun 2015. *Cowrie* bertujuan untuk mencatat serangan *brute force* dan interaksi yang dilakukan oleh penyerang [10]
4. ***Raspberry pi***, komputer *single-board* yang dikembangkan oleh *raspberry pi Foundation* di Inggris. Komputer *single-board* ini mempunyai biaya lebih murah, sering digunakan dalam bidang akademis, penelitian, dan sistem *embedded* [11].

1.2 *Manajemen log*

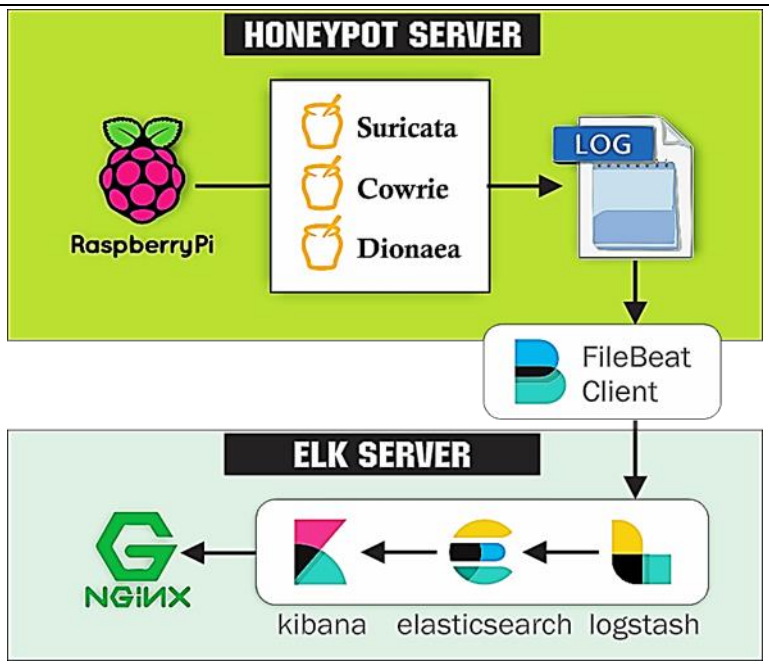
Dalam penelitian ini manajemen *log* yang dihasilkan oleh *honeypot* menggunakan ELK *stack.* ELK *stack merupakan* platform manajemen dan analisis *log* yang komplet. Analisis *log* membantu dalam mendeteksi pelanggaran keamanan, penyalahgunaan aplikasi, serangan berbahaya, dan sebagainya [6]. ELK *stack* mempunyai beberapa unsur, antara lain:

1. ***Elasticsearch***, mesin pencari yang berkemampuan dalam pencarian dan analisis data secara *realtime*. *Elasticsearch* mempunyai beberapa fitur seperti pencarian multibahasa, *geolocation, autocomplete*, *JSON* dan *RESTful API* yang memudahkan *elasticsearch* dalam mengelola data.
2. ***Logstash*** membantu dalam membangun jaringan *pipeline* yang dapat memusatkan pengolahan data. Menggunakan berbagai *plugin input* dan *output* untuk memudahkan dalam *parsing* dan memproses format yang berbeda dalam skala besar.
3. ***Kibana,*** memvisualisasikan data yang tersimpan pada *cluster elasticsearch*. *Kibana* menyediakan antarmuka berbasis *browser* yang memudahkankan dalam membuat *dashboard* dengan cepat. *Kibana* menyajikan data dalam bentuk *histogram, geomaps,* diagram lingkaran, grafik, tabel, dan lain-lain.
4. ***Filebeat*****,** *plug in logstash* yang bertugas sebagai agen pada *server* sumber untuk mengirim data ke ELK *stack*. *Filebeat* menggantikan *plug in logstash* yang lama yaitu *logstash forwarder* atau *lumberjack* [12].

2. Metode Penelitian

2.1 *Skema Sistem*

Skema ini menjelaskan rancangan secara garis besar tentang bagaimana cara kerja dari sistem yang dibuat. Pada Gambar 1 dapat dilihat skema yang telah dirancang oleh penulis.



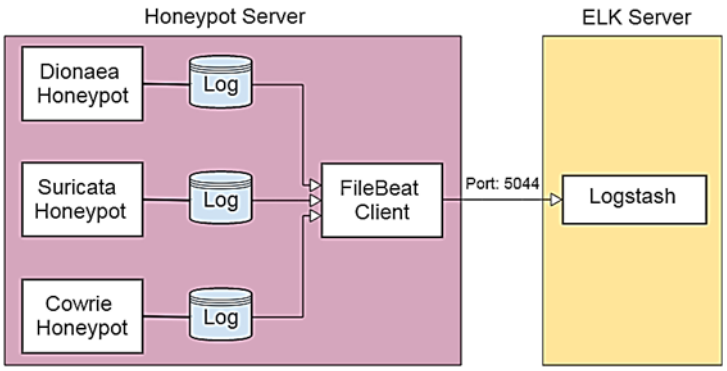
Gambar 1. Skema Sistem

Dalam skema di atas menggambarkan perangkat keras, perangkat lunak, dan alur dari sistem secara umum. Terdapat tiga *honeypot* yang dipasang pada raspberry pi, *honeypot* akan menghasilkan log yang berisi catatan serangan. ELK *server* akan mengolah *log* dari *honeypot* dan menampilkanya dalam bentuk visualisasi yang menarik.

Alur kerja dari sistem yaitu ketika terjadi serangan maka *honeypot* akan menangkap serangan dan mencatat serangan tersebut. Data serangan yang didapatkan akan disimpan dalam *log* masing-masing *honeypot.* Data *log* dari *honeypot* dikirim ke ELK *server* menggunakan *plug in filebeat*. *Log* akan diolah ELK *stack* untuk divisualisasikan dan akan ditampilkan menggunakan *dashboard* pada *web browser.*

2.2 *Arsitektur Honeypot Server*

Pada arsitektur ini *raspberry pi* sebagai *server* yang digunakan untuk menjalakan *honeypot.* Ditunjukkan Gambar 2 dari arsitektur *honeypot server.*



Gambar 2. Arsitektur Honeypot Server

Pada *honeypot server* terdapat tiga *honeypot* yang berbeda. *Honeypot* terserebut antara lain *suricata* yang bekerja dengan cara *monitoring* jaringan, mencatat paket pada lalu lintas, IDS, dan IPS. *cowrie* bekerja dengan cara mencatat serangan *brute force* dan interaksi yang dilakukan oleh penyerang, dan *dionaea* bekerja dengan cara menjebak eksploitasi *malware* melalui kerentanan yang sengaja dibuka untuk diserang. Masing-masing *honeypot* menghasilkan log

untuk dianalisis. *Log* tersebut akan dikirim dari *honeypot server* menuju ELK *server* menggunakan *filebeat* client dengan *port* 5044.

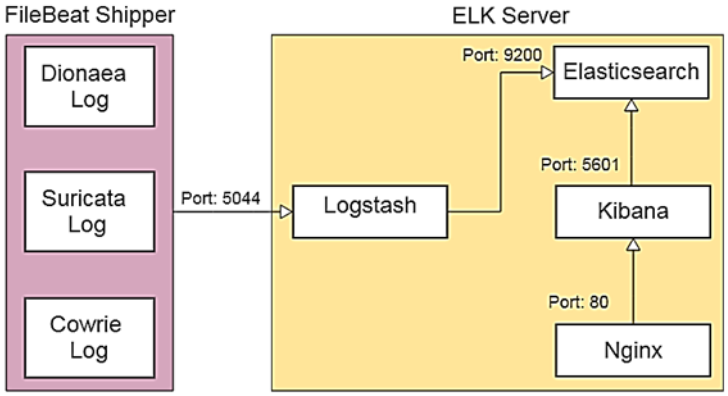
Ditunjukkan pada Tabel 1 beberapa port yang dibuka oleh masing-masing *honeypot* untuk mendeteksi kejadian dalam jaringan.

Tabel 1. Port yang Dibuka Honeypot

Jenis Honeypot	Port	Tingkat Interaksi
Dionaea	42, 53, 123, 80, 443, 5060/ 5060/ 5061, 21, 69, 445, 1433, 3306	Rendah
Suricata	80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 311, 383, 591, 593, 631, 901, 1220, 1414, 1741, 1830, 2301, 2381, 2809, 3037, 3057, 3128, 3702, 4343, 4848, 5250, 6080, 6988, 7000, 7001, 7144, 7145, 7510, 7777, 7779, 8000, 8008, 8014, 8028, 8080, 8085, 8088, 8090, 8118, 8123, 8180, 8181, 8222, 8243, 8280, 8300, 8500, 8800, 8888, 8899, 9000, 9060, 9080, 9090, 9091, 9443, 9999, 10000, 11371, 34443, 34444, 41080, 50002, 55555	Rendah
Cowrie	22, 23	Menengah

2.3 *Arsitektur ELK Server*

Pada arsitektur ini ELK *server* akan digunakan untuk mengolah log yang dihasilkan *honeypot* menjadi visualisasi yang menarik. Ditunjukkan pada Gambar 3 arsitektur ELK *server*. *Logstash* akan menerima *log* yang dikirim dari plug in *filebeat* menggunakan *port* 5044. *Logstash* mengumpulkan data tersebut kemudian dikirim menuju *elasticsearch* menggunakan *port* 9200. Pada *elasticsearch* data akan diolah dan dianalisis untuk ditampilkan dalam visualisasi menggunakan *kibana*. *Kibana* berjalan pada *port* 5601 dan berinteraksi dengan data yang tersimpan dalam indeks *elasticsearch*. Menggunakan *kibana* akan mempermudah dalam memvisualisasikan data dalam berbagai grafik, tabel, dan lain-lain. Pada penelitian ini akses *kibana* menggunakan *localhost,* jadi untuk memungkinkan akses dari eksternal menggunakan *proxy nginx.* Selain itu, *nginx* akan menggunakan *file* htpasswd. users untuk mengkonfirmasi pengguna yang diperbolehkan mengakses *kibana.*



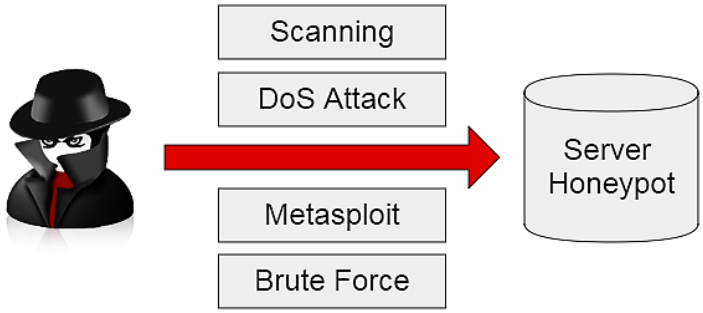
Gambar 3. Arsitektur ELK Server

2.4 *Rancangan Pengujian*

Pengujian pada penelitian ini menggunakan metode *black box.* Pengujian ini dilakukan untuk membuktikan sejauh mana kesesuaian sistem yang dibangun bekerja dengan fungsionalitas dan tujuan perancangan. Adapun beberapa rancangan pengujian didasarkan terhadap beberapa hal, antara lain apakah sistem yang dirancang mampu menangkap dan memberikan peringatan serangan, hasil yang ditangkap oleh *honeypot*, dan visualisasi *log* dari *honeypot* yang ditampilkan oleh ELK *stack.*

2.4.1 *Skenario Serangan Honeypot*

Simulasi serangan dilakukan dengan dua tahap dengan cara menyerang *server honeypot.* Serangan yang dilakukan dapat dilihat pada , pengujian serangan ini dilakukan dengan cara menyerang *server honeypot* secara langsung. Adapun pada serangan tahap pertama, penulis yang akan bertindak sebagai penyerang dalam pengujian ini, ditunjukkan pada Gambar 4.



Gambar 4. Skenario Serangan Honeypot

Dalam simulasi serangan terdapat empat cara yang dilakukan antara lain *scanning*, serangan DoS, *metasploit*, dan serangan *brute force*. Ditunjukkan pada Tabel 2 penjelasan dari serangan yang dilakukan.

Tabel 2. Perintah Simulasi Serangan Honeypot

Opsi Serangan	Deskripsi
Scanning menggunakan Nmap	IP target serangan: 192.168.1.6 IP penyerang: 192.168.1.5 Perintah : nmap –Su –St -A –O 192.168.1.2
Serangan *Brute force* menggunakan Hydra	IP target serangan: 192.168.1.6 IP penyerang: 192.168.1.5 Perintah 1: Hydra –L nama.txt –P passwd.txt 192.168.1.6 ssh Perintah 2: Hydra –L nama.txt –P passwd.txt 192.168.1.6 ftp
Serangan DoS menggunakan *Low Orbit Ion Cannon* (LOIC)	IP target serangan: 192.168.1.6 IP penyerang: 192.168.1.5 Perintah yang dilakukan yaitu *flooding* dengan IP tujuan 192.168.1.6, port 445, metode TCP, dan jumlah *flood* 1.000
Serangan MS17-10 menggunakan *metasploit*	IP target serangan: 192.168.1.20 IP penyerang: 192.168.1.19 Perintah : msf exploit(eternalblue_doublepulsar) > use auxiliary/scanner/smb/smb_ms17_010

Serangan tahap kedua dilakukan hari pada hari senin tanggal 9 oktober 2017 di kelas keamanan jaringan dari jam pertama s.d. jam ketiga. Skenario yang dilakukan yaitu mahasiswa yang mengikuti kuliah tersebut melakukan *scanning* untuk mencari *server* yang membuka banyak *port,* kemudian melakukan serangan *brute force* pada *server* tersebut.

2.4.2 *Pengujian Performa*

Pengujian performa dilakukan untuk mengetahui performa sistem ketika dilakukan penyerangan pada *honeypot* dan performa sistem ketika pengiriman *log* dari *server honeypot* menuju *server* ELK. Parameter yang diuji adalah penggunaan sumber daya *memori* dan CPU. Untuk mendapatkan informasi performa tersebut yang sedang diuji, penulis menggunakan perangkat lunak *glances*. Perangkat lunak tersebut bekerja dengan memberikan informasi tentang seberapa besar penggunaan CPU dan *memori* pada saat menerima serangan.

3. Hasil Penelitian dan Pembahasan

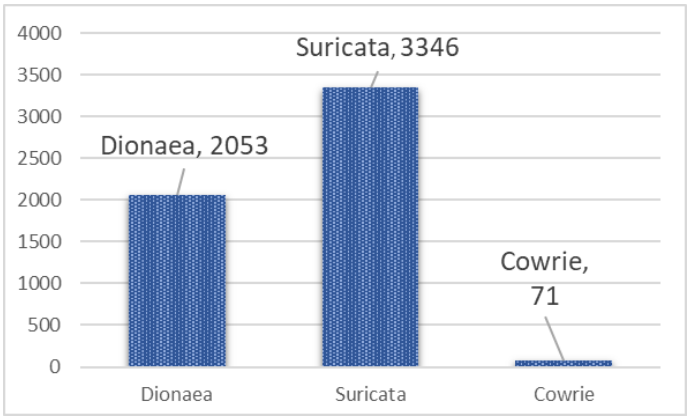
3.1 *Pengujian Serangan Honeypot Tahap Pertama*

Dalam pengujian *honeypot*, dilakukan serangan terhadap masing-masing *honeypot* sesuai dengan rancangan yang telah dibuat. Selain itu, juga ditunjukkan hasil visualisasi dari *log* yang dihasilkan oleh *honeypot*.

3.1.1 *Scanning Nmap*

Pada tahap ini percobaan *scanning* yang penulis lakukan menggunakan *Nmap dengan perintah* nmap –sU –sT -A –O 192.168.1.6*.* Pada hasil scanning tersebut terdapat beberapa *port* yang berstatus *open*, port tersebut bisa digunakan penjahat sebagai celah untuk diserang. Hasil percobaan tersebut ditangkap oleh *dionaea, suricata dan cowrie.*

Berdasarkan percobaan dan hasil yang ditampilkan pada Gambar 5 ini, kejadian yang ditangkap oleh *suricata* sebanyak 3346, *cowrie* sebanyak 71, dan dionaea sebanyak 2053 kejadian. *Suricata* menangkap kejadian paling banyak pada percobaan ini.

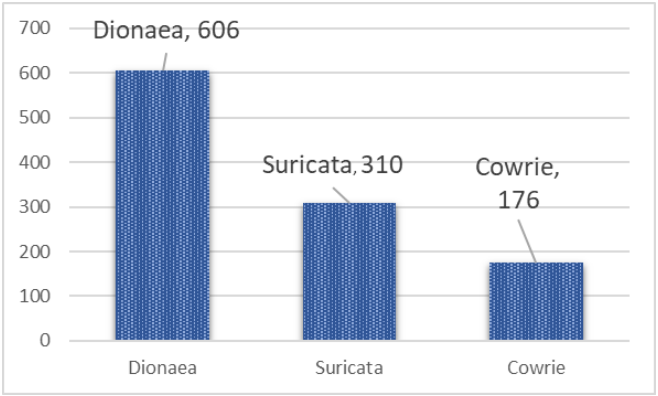


Gambar 5. Jumlah Kejadian Ditangkap Honeypot Terhadap Scanning Nmap

3.1.2 *Serangan Brute Force*

Setelah dilakukan scanning *port* mengunakan *Nmap*, terdapat port 21 dan 22 dengan status *open*. Biasanya port 21 digunakan FTP *server* untuk tukar menukar data dan SSH menggunakan *port* 22 yang digunakan untuk menghubungkan antara komputer satu ke komputer lainya di internet. Pada dapat ditunjukkan serangan *brute force* SSH yang penulis lakukan menggunakan *hydra.*

Kemudian percobaan serangan *brute force* pada port FTP yang penulis lakukan menggunakan *hydra***.** Percobaan serangan *brute force* SSH dapat ditangkap oleh *suricata dan cowrie,* namun *dionaea* tidak. Sebaliknya pada percobaan serangan *brute force* FTP, *suricata* dan *dionaea* dapat menangkap serangan*,* namun *cowrie* tidak. Seperti ditunjukkan pada Gambar 6 berdasarkan percobaan serangan *brute force* pada SSH dan FTP yang dilakukan, *suricata* menangkap kejadian sebanyak 310, *cowrie* sebanyak 176, dan *dionaea* sebanyak 606 kejadian. *Dionaea* menangkap kejadian paling banyak kejadian pada serangan ini.

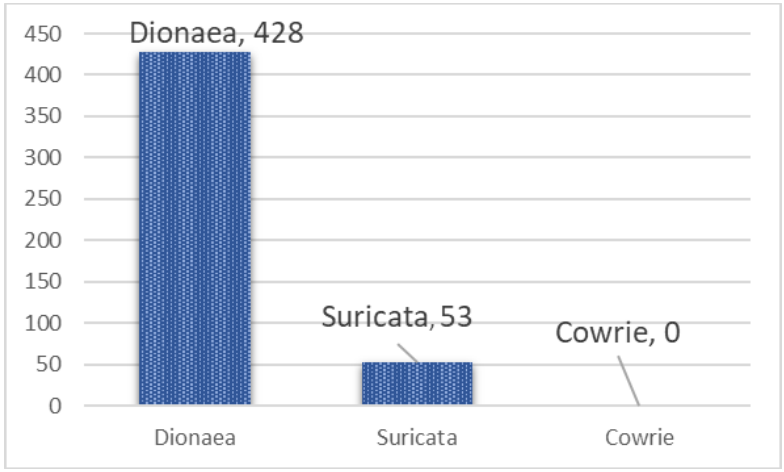


Gambar 6. Jumlah Kejadian Ditangkap Honeypot Terhadap Serangan Brute Force

3.1.3 *Serangan DoS*

Serangan berikutnya yaitu serangan DoS, ditunjukkan pada Gambar 7 serangan DoS menggunakan LOIC. Serangan ini dilakukan dengan IP tujuan 192.168.1.6, port 445, metode TCP, dan jumlah flood 1.000.

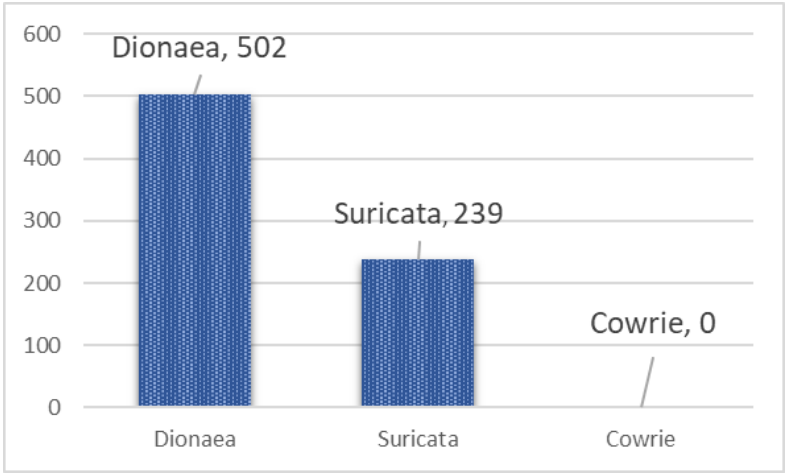
Berdasarkan percobaan tersebut kejadian yang ditangkap oleh *suricata* sebanyak 53, *dionaea* sebanyak 428 kejadian dan *cowrie* tidak berhasil menangkap kejadian. *Dionaea* menangkap kejadian paling banyak pada serangan ini.



Gambar 7. Jumlah Kejadian Ditangkap Honeypot Terhadap Serangan DoS

3.1.4 *Serangan Metasploit*

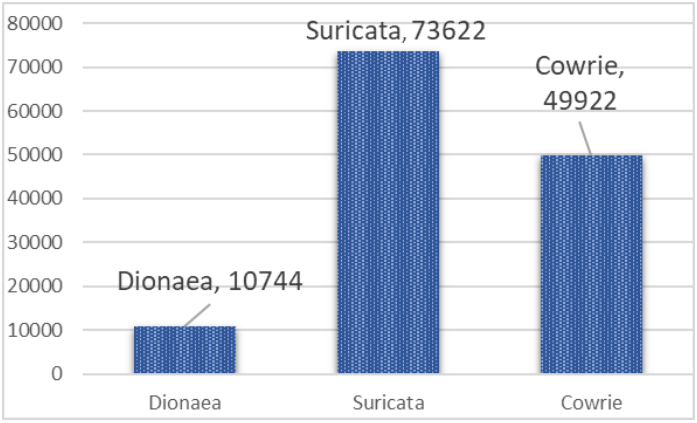
Pada percobaan serangan ini dilakukan dengan cara eksploitasi yang digunakan oleh *malware* jenis *wannacry ransomware* untuk mengeksploitasi kerentanan. Serangan *metasploit* ini menggunakan modul SMB MS17-10. Berdasarkan percobaan tersebut kejadian yang ditangkap oleh *suricata* sebanyak 239, *dionaea* sebanyak 502 kejadian dan *cowrie* tidak berhasil menangkap kejadian. Ditunjukkan pada Gambar 8 *dionaea* menangkap kejadian paling banyak pada serangan ini.



Gambar 8. Jumlah Kejadian Ditangkap Honeypot Terhadap Serangan Metasploit

3.2 *Pengujian Serangan Honeypot Tahap Kedua*

Pada tahap ini, pengujian dilakukan dengan serangan *Nmap* dan *brute force* dilakukan oleh mahasiswa yang mengikuti mata kuliah keamanan jaringan. Berdasarkan percobaan tersebut kejadian yang ditangkap oleh *suricata* sebanyak 73.622, *dionaea* sebanyak 10.744 kejadian dan *cowrie* sebanyak 49.922 kejadian. Ditunjukkan pada Gambar 9, *suricata* menangkap kejadian paling banyak pada serangan ini.



Gambar 9. Jumlah Kejadian Ditangkap Honeypot Terhadap Percobaan Serangan Pada Kelas Keamanan Jaringan

Dari hasil monitoring performa, dapat diperoleh data seperti ditunjukkan pada Tabel 3. Selama proses percobaan *scanning, dionaea* mengkonsumsi penggunaan CPU paling besar yaitu 32,4% dan pemakaian memori paling tinggi yaitu *suricata* dengan jumlah 16,9%.

Tabel 3. Hasil Monitoring Sistem ketika Scanning Nmap

Nama Service	CPU Terpakai	Memori Terpakai
Suricata	7,5 %	16,9 %
Dionaea	32,4 %	3,0 %
Cowrie	23,6 %	4,1 %
Filebeat	15,6 %	2,6 %

Pada saat dilakukan percobaan serangan *brute force* SSH, proses *cowrie* mengkonsumsi penggunaan CPU paling besar, yaitu 44,0% dan pemakaian memori paling tinggi yaitu *suricata* dengan jumlah 16,9%. Sedangkan ketika serangan *brute force* FTP, proses *dionaea* mengkonsumsi penggunaan CPU paling besar yaitu 20,2% dan pemakaian memori paling tinggi yaitu *suricata* dengan jumlah 16,9%, seperti ditunjukkan pada 4 dan Tabel 5.

Tabel 4. Hasil Monitoring Sistem ketika Serangan Brute Force SSH

Nama Service	CPU Terpakai	Memori Terpakai
Suricata	5,1 %	16,9 %
Dionaea	0,3 %	2,7 %
Cowrie	44,0 %	4,1 %
Filebeat	11,8 %	2,0 %

Tabel 5. Hasil Monitoring Sistem ketika Serangan Brute Force FTP

Nama Service	CPU Terpakai	Memori Terpakai
Suricata	8,1 %	16,9 %
Dionaea	20,2 %	2,7 %
Cowrie	0 %	0 %
Filebeat	4,0 %	2,0 %

Kemudian ketika percobaan serangan DoS menggunakan LOIC. Dari hasil *monitoring* pada Tabel 6, proses *dionaea* mengkonsumsi penggunaan CPU paling besar yaitu 73,6% dan pemakaian memori paling tinggi yaitu pada *dionaea* dengan jumlah 44,5%.

Tabel 6. Hasil Monitoring Sistem ketika Serangan LOIC

Nama Service	CPU Terpakai	Memori Terpakai
Suricata	42,6 %	17,5 %
Dionaea	73,6 %	44,5 %
Cowrie	0,0 %	3,7 %
Filebeat	0,0 %	1,6 %

Dari hasil monitoring proses pada Tabel 7, saat percobaan serangan *metasploit, dionaea* mengkonsumsi penggunaan CPU paling besar, yaitu 14,7% dan pemakaian memori paling tinggi yaitu pada *suricata* dengan jumlah 16,9%.

Tabel 7. Hasil Monitoring Sistem ketika Serangan Metasploit

Nama Service	CPU Terpakai	Memori Terpakai
Suricata	4,5 %	16,9 %
Dionaea	14,7 %	2,7 %
Cowrie	0,0 %	0,0 %
Filebeat	0,8 %	2,2 %

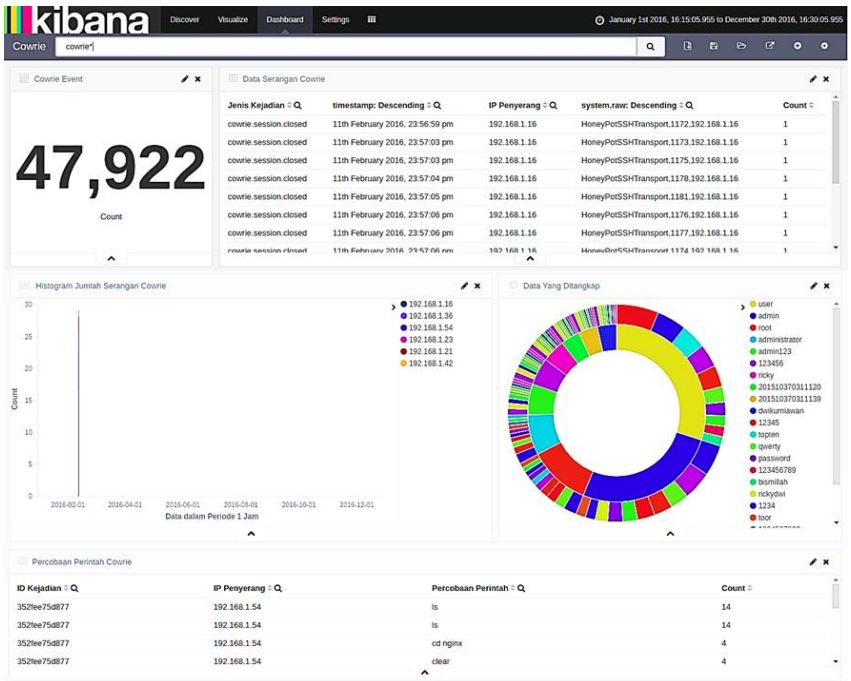
Ditunjukkan pada Tabel 8 perbandingan kemampuan yang dimiliki oleh *honeypot* yang digunakan untuk mendeteksi kejadian dan mengungkap informasi dari berbagai jenis serangan yang dilakukan. Dengan tabel tersebut dapat diketahui perbedaan dari beberapa honeypot yang digunakan.

Tabel 8. Perbandingan Kinerja Honeypot

Nama *Honeypot*	Jumlah Peringatan Keseluruhan	Mendeteksi Perangkat Lunak Serangan	Mendeteksi Percobaan *Log*in	Mengungkap Username/ Password Penyerang
Suricata	77.570	Ya	Ya	Tidak
Dionaea	14.333	Tidak	Tidak	Ya
Cowrie	50.169	Tidak	Ya	Ya

Setelah melakukan beberapa percobaan serangan, penulis menunjukkan hasil visualisasi dari log yang diperoleh masing-masing honeypot. Ditunjukkan pada Gambar 10, sebagai contoh tampilan dari serangan yang dihasilkan oleh *cowrie honeypot* pada saat serangan tahap kedua.

Cowrie honeypot dapat mendeteksi semua usaha *login* ke *server* dan memberikan informasi yang sangat berharga. Informasi tersebut terdiri dari *username* dan *password* yang dilakukan oleh penyerang untuk mencoba memasuki *server,* sesuai dengan fungsi dari *cowrie honeypot.* Pada hasil pengujian ini membuktikan bahwa ELK *stack* dapat mengumpulkan *log* dari sumber yang berbeda *host* dan membuat analisis dari *log* yang awalnya rumit menjadi lebih menarik.



Gambar 10. Visualisasi Log Cowrie Honeypot Saat Serangan Tahap Kedua

4. Kesimpulan

Salah satu tujuan *honeypot* adalah memberikan *server* palsu, dimana penyerang akan membuang-buang waktu untuk melakukan serangan. Hal ini dapat digunakan oleh *administrator* sistem melakukan tindakan untuk mengamankan *server* yang sebenarnya dari pengguna yang tidak sah. *Honeypot* mengumpulkan data hanya jika seseorang berinteraksi dengannya. *Server honeypot akan menangkap serangan dalam jaringan sesuai dengan port yang dimiliki oleh masing-masing honeypot. Jika serangan berkomunikasi dengan port* yang dibuka, *maka sistem akan menangkap serangan tersebut. Honeypot* memang mempunyai informasi yang sangat berharga, namun tidak langsung berguna tanpa analisis secara teliti oleh yang berpengalaman dibidang keamanan jaringan.

Logstash mempunyai plugin filter untuk memproses data berformat *json*, jadi tidak memerlukan pekerjaan tambahan untuk mengirim *log honeypot* yang berformat *json* dari *raspberry pi* menuju *logstash.* Menggunakan ELK *stack* akan memberi keuntungan untuk mencari *log* lebih mendalam berdasarkan *timestamp*. Berbagai jenis bentuk untuk visualisasi *log* seperti diagram, tabel, dan lain-lain sercara *real time.*

5. Saran

Penelitian ini berfokus pada bagaimana cara implementasi beberapa *honeypot* pada *raspberry pi* dan memvisualisasikan hasil *log honeypot* pada ELK *stack*. Masih banyak ditemukan kekurangan pada penelitian sehingga memungkinkan digunakan sebagai tumpuan pada pengembangan penelitian selanjutnya**.**

Penulis memberikan saran antara lain untuk penelitian kedepanya dapat menerapkan berbagai sensor *honeypot* yang baru pada *raspberry pi* kemudian menempatkan *server honeypot* berdasarkan penempatan yang sebenarnya, sehingga sistem ini sungguh-sungguh dimanfaatkan bagi pembaca yang hendak meneruskan penelitian. Hasil yang didapat oleh *honeypot* sangat banyak, sehingga dapat dilakukan beberapa penelitian untuk mempelajari kemampuan dan hasil yang diperoleh oleh *honeypot* dalam mengungkap informasi penyerang secara mendalam. Kemudian dapat dilakukan tindakan kelanjutan seperti *blocking access* terhadap *akses ilegal.*

Referensi

[1] D. K. Bhattacharyya dan J. K. Kalita, *DDoS Attacks: Evolution, Detection, Prevention, Reaction, and Tolerance*. CRC Press, 2016.

[2] L. Spitzner, *Honeypots: tracking hackers*. Boston: Addison-Wesley, 2003.

[3] R. C. Joshi dan A. Sardana, *Honeypots: a new paradigm to information security*. Enfield, N.H. : Boca Raton, FL: Science Publishers ; Distributed by CRC Press, 2011.

[4] S. Mahajan, A. M. Adagale, dan C. Sahare, “Intrusion Detection System Using Raspberry PI Honeypot in Network Security,” *Int. J. Eng. Sci.*, vol. 6, no. 3, hal. 2792, 2016.

[5] C. Lim, M. Marcello, A. Japar, J. Tommy, dan I. E. Kho, “Development of Distributed Honeypot Using Raspberry Pi,” 2014.

[6] S. Chhajed, *Learning ELK stack: build mesmerizing visualizations, and analytics from your logs and data using Elasticsearch, Logstash, and Kibana*. 2015.

[7] C. Moore dan A. Al-Nemrat, “An Analysis of Honeypot Programs and the Attack Data Collected,” in *International Conference on Global Security, Safety, and Sustainability*, 2015, hal. 228–238.

[8] “Dionaea Documentation.” [Daring]. Tersedia pada: http://dionaea.readthedocs.io. [Diakses: 25-Jul-2017].

[9] J. S. White, T. Fitzsimmons, dan J. N. Matthews, “Quantitative analysis of intrusion detection systems: Snort and Suricata,” *SPIE Def. Secur. Sens.*, hal. 875704–875704, 2013.

[10] M. Oosterhof, “Cowrie Honeypot,” *Security Intelligence*. [Daring]. Tersedia pada: http://www.micheloosterhof.com/cowrie/. [Diakses: 23-Jun-2017].

[11] A. Pajankar, Arush Kakkar, Matthew Poole, TotalBoox, dan TBX, *Raspberry Pi: Making Amazing Projects Right from Scratch!* Packt Publishing, 2016.

[12] J. Turnbull, *The Logstash Book*. James Turnbull, 2013.