Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) DOI: 10.25126/jtiik.202073339
Vol. 7, No. 3, Juni 2020, hlm. 591-598 p-ISSN: 2355-7699
Akreditasi KEMENRISTEKDIKTI, No. 30/E/KPT/2018 e-ISSN: 2528-6579

IDENTIFIKASI MALICIOUS HOST DALAM LOCAL AREA NETWORK MENGGUNAKAN TEKNIK GRAPH CLUSTERING DAN FILTERING

Khafidzun Fadli*[1], Achmad Basuki[2], Eko Setiawan[3]

[1, 2, 3] Fakultas Ilmu Komputer, Universitas Brawijaya
Email: [1]khafidzunfadli@student.ub.ac.id, [2]abazh@ub.ac.id, [3]ekosetiawan@ub.ac.id
*Penulis Korespondensi

(Naskah masuk: 06 Maret 2020, diterima untuk diterbitkan: 27 April 2020)

Abstrak

Keamanan pada *Local Area Network* (LAN) sekarang ini adalah masalah serius yang harus diperhatikan. Penyebab LAN menjadi tidak aman dikarenakan teknologi *firewall* tidak mampu melindungi *host* (komputer) dalam LAN dari penyebaran *malware*. Penyebaran *malware* yang terdapat dalam LAN dilakukan oleh *host* di dalam LAN yang disebut sebagai *malicious host*. Tindakan untuk mengurangi penyebaran *malware* dalam LAN dapat dilakukan dengan mengidentifikasi *malicious host*. Penelitian ini mengusulkan metode identifikasi *malicious host* berdasarkan aktivitas *Address Resolution Protocol* (ARP) *request* dengan menggunakan teknik *graph clustering-filtering*. Teknik *graph clustering-filtering* merupakan langkah-langkah pengelompokan serta penyaringan *node* dan *edge* berdasarkan parameter dari *graph* seperti *weight edge*, *out-degree node* dan *weight out-degree node* yang bertujuan untuk mengidentifikasi *malicious host*. Berdasarkan parameter dari *graph* seperti *out-degree node* dan *weight out-degree node,* penghitungan persentase aktivitas *host* dapat dilakukan untuk menunjukkan seberapa besar tingkat aktivitas *host* dalam melakukan *broadcast* ARP *request,* sehingga hasil penghitungan persentase aktivitas *host* dapat menentukan *host* yang diidentifikasi sebagai *malicious host.* Hasil penerapan teknik *graph clustering-filtering* terhadap 511 *node* dan 4144 *edge* didapatkan melalui pengamatan dan pengambilan data selama 3 jam dalam LAN kampus dapat divisualisasikan menjadi hanya 22 *node* dan 328 *edge*. Hasil penghitungan berdasarkan persentase jumlah aktivitas *host* menunjukkan 22 *node* menjadi 6 *node* yang diidentifikasi sebagai *malicious host*. Dengan demikian, visualisasi *graph* menggunakan teknik *graph clustering-filtering* dan persentase aktivitas *host* dapat mengidentifikasi jumlah *host* yang diidentifikasi sebagai *malicious host*.

Kata kunci: *LAN, malicious host, teknik graph clustering-filtering*

IDENTIFICATION OF MALICIOUS HOST IN LOCAL AREA NETWORK USING GRAPH CLUSTERING AND FILTERING TECHNIQUE

Abstract

Local Area Network (LAN) security is a serious problem to consider. The cause of LAN becomes insecure because firewall technology is not able to protect the host (computer) in LAN from spreading malware. The spread of malware contained within a LAN is carried out by hosts in the LAN which are referred to as malicious hosts. Actions to reduce the spread of malware in the LAN can be done by identifying malicious hosts. This paper proposes a method of identifying malicious hosts based on Address Resolution Protocol (ARP) request activities using graph clustering-filtering techniques. Graph clustering-filtering techniques are steps of grouping and filtering nodes and edges based on graph parameters such as weight edges, out-degree nodes and weight out-degree nodes that aim to identify malicious hosts. Based on parameters from the graph such as out-degree node and weight out-degree node, the calculation of the percentage of host activity can be done to show how much the level of host activity in broadcasting an ARP request, so that the result of calculating the percentage of host activity can determine a host that is categorized as a malicious host. The results of graph visualization using graph clustering-filtering technique can display fewer nodes and edges, from 511 nodes and 4144 edges to 22 nodes and 328 edges observed and collected in a LAN within 3 hour in the campus LAN. The results of the calculation of the percentage of host activity show hosts from 22 nodes become only 6 nodes which are suspected as malicious hosts. Overall, graph visualization with graph clustering-filtering techniques and the percentage of host activity can find a number of hosts identified as malicious hosts.

Keywords: *LAN, malicious host, graph clustering-filtering technique*

1. PENDAHULUAN

Keamanan dari *Local Area Network* (LAN) sangat serius untuk diperhatikan sekarang ini. Berdasarkan laporan intelijen keamanan microsoft pada tahun 2018, tingkat *malware encounter* di seluruh dunia dalam periode Januari sampai Desember 2018 mencapai rata-rata 5,10% (Microsoft, 2018). Tingkat *malware encounter* merupakan persentase komputer yang melaporkan adanya *malware*. Lima negara dengan tingkat *malware encounter* tertinggi yaitu Ethiopia (26,33%), Pakistan (18,94%), Palestina (17,50%), Bangladesh (16,95%) dan Indonesia (16,59%). Tingkat *malware encounter* yang semakin tinggi disebabkan karena teknologi *firewall* dalam LAN tidak mampu untuk melindungi komputer yang terdapat dalam LAN. *Host* (komputer) yang tidak terlindungi dapat terinfeksi *malware* melalui *email phising*, pemasangan *pirated software*, anti virus gratis dan kegiatan *browsing* ke *website* yang membahayakan. Penyebab *malware encounter* lainnya adalah sistem operasi kadaluarsa yang terdapat pada komputer, sehingga *host* (komputer) yang terdapat dalam LAN dapat terinfeksi *malware*. Dengan demikian, LAN masih memiliki celah keamanan yang berfokus pada *host* dan keamanannya serius untuk diperhatikan.

Dengan berfokus pada *host*, LAN mempunyai banyak *host* yang terhubung dan beberapa dari *host* tersebut mungkin dapat menyebarkan *malware*. *Host* yang menyebarkan *malware* atau yang mengganggu kinerja jaringan dan membahayakan *host* lain dalam LAN merupakan *malicious host*. Identifikasi *malicious host* dalam LAN perlu untuk dilakukan, agar *malicious host* dapat ditemukan dan dapat dilakukan tindakan lebih lanjut terhadap *malicious host* tersebut. Dengan demikian, penyebaran malware dalam LAN semakin berkurang dan LAN menjadi lebih aman.

Beberapa peneliti sudah mengusulkan metode identifikasi *malicious host* dalam LAN, seperti identifikasi *malicious host* dengan mengamati data trafik *Domain Name Server* (DNS) (Marko dan Vilhan, 2012). Metode tersebut sesuai untuk mengidentifikasi *malicious host* pada aktivitas *browsing*. Selanjutnya ada usulan metode identifikasi *malicious host* dalam LAN dengan mengamati data trafik *firewall* (Bond, 2009), data *output* PRADS (*Passive Real-time Detection System*) (Desta, 2014) dan data *output* Honeypot (Valli, 2009). Ketiga metode tersebut sesuai untuk mengidentifikasi *malicious host* pada aktivitas membahayakan. Namun, keempat metode tersebut tidak dapat digunakan untuk mengidentifikasi aktivitas awal *malicious host* sebelum menyebarkan *malware*. Usulan metode lain untuk mengidentifikasi *malicious host* dalam LAN adalah dengan mengamati paket ARP *request* (Ochiai, 2019). Metode tersebut sesuai untuk mengidentifikasi *malicious host* pada aktivitas awal dalam menyebarkan *malware*, karena *malicious host* dalam LAN melakukan *broadcast Address Resolution Protocol* (ARP) *request* pada awal aktivitas (Whyte, Kranakis dan Oorschot, 2005; Hubballi, dkk., 2011; Matsufuji, dkk., 2019).

ARP *request* bertujuan untuk menemukan *host* lain yang aktif dan mencari celah serta menyebarkan *malware* pada *host* tersebut. Peningkatan jumlah ARP *request* ke *host* lain dalam LAN yang dilakukan oleh *host* tertentu dalam waktu tertentu, menjadikan *host* tersebut sebagai kategori *malicious host*. Dengan demikian, identifikasi *malicious host* dalam LAN dapat dilakukan dengan mengamati paket ARP *request* yaitu dengan menangkap paket ARP *request* dalam LAN menggunakan aplikasi *tcpdump* (Tcpdump, n.d.).

Namun, hasil tangkapan ARP *request* masih sulit untuk mengidentifikasi *malicious host* dalam LAN. Oleh karena itu, transformasi ARP *request* dan visualisasi *graph* diperlukan untuk mempermudah identifikasi *malicious host* dalam LAN (Ochiai, 2019). Transformasi ARP *request* bertujuan untuk mengubah paket ARP *request* menjadi bentuk data visualisasi *graph* dengan format *Comma Separated Values* (CSV) dan visualisasi *graph* bertujuan untuk mengubah data visualisasi *graph* menjadi bentuk *graph* G=(V,E) (Chakraborty, dkk., 2018). V adalah himpunan dari *node (host)* dan E adalah himpunan dari *edge (link),* sehingga dengan visualisasi *graph* dapat mempermudah identifikasi *malicious host*.

Jika dalam visualisasi *graph* terdapat himpunan *node* (V) dan *edge* (E) yang sangat banyak, maka hasil visualisasi *graph* sulit untuk dibaca dan identifikasi *malicious host* sulit untuk dilakukan. Oleh karena itu, penelitian ini mengusulkan metode identifikasi *malicious host* berdasarkan aktivitas ARP *request* dengan menggunakan teknik *graph clustering-filtering*. Tahapan proses dalam metode tersebut terdiri dari 4 proses. Proses pertama adalah melakukan penangkapan paket ARP *request* dalam LAN. Proses kedua adalah melakukan transformasi paket ARP *request* menjadi bentuk data visualisasi *graph*. Proses ketiga adalah melakukan visualisasi *graph* dengan aplikasi *Gephi* menggunakan teknik *graph clustering-filtering*. Proses terakhir adalah melakukan penghitungan persentase aktivitas *host*. Hasil visualisasi *graph* menggunakan teknik *graph clustering-filtering* dan hasil penghitungan persentase aktivitas *host* dapat mengidentifikasi *malicious host*.

2. AKTIVITAS MALWARE DALAM LAN

Malware menyebar ke dalam LAN dimulai dengan adanya aktivitas *broadcast* ARP *request* ke seluruh *host* (komputer) dalam LAN. *Broadcast* ARP *request* yang dilakukan oleh *malicious host* bertujuan untuk mengetahui *host* aktif dalam LAN yang kemudian menjadi target untuk penyebaran *malware*

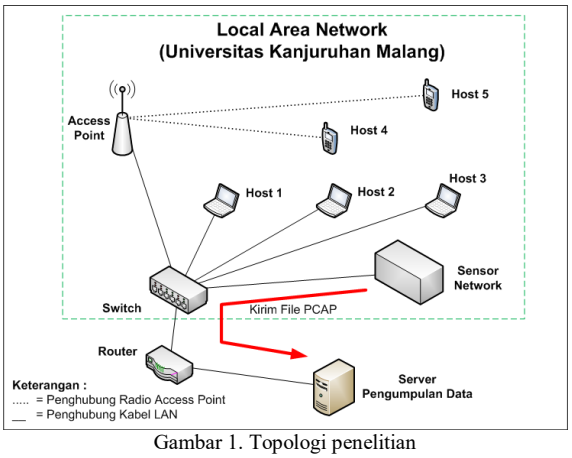
itu sendiri. Paket ARP *request* yang dikirimkan oleh *malicious host* dapat diduga dengan jumlah tidak wajar bila dibandingkan dengan *host* lainnya.

Pada penelitian ini, *malicious host* diidentifikasi berdasarkan jumlah aktivitas ARP *request* dari *malicious host* ke *host* lainnya dalam LAN. Namun demikian, membedakan ARP *request* normal dengan yang tidak normal merupakan permasalahan utama dalam penelitian ini. Apabila identifikasi *malicious host* dapat dilakukan, maka penyebaran *malware* dalam LAN dapat dengan cepat dihambat atau bahkan dikurangi.

3. METODE PENELITIAN

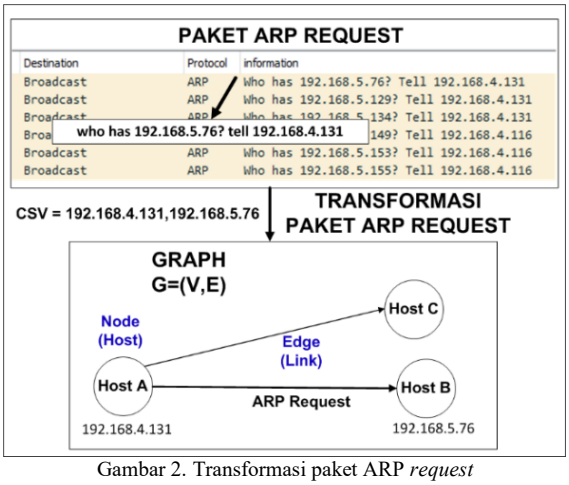
3.1. Pengumpulan Data dan Transformasi

Pengumpulan data untuk identifikasi *malicious host* dalam LAN dilakukan dengan memasang *sensor network* di dalam LAN dan *server* pengumpulan data yang diletakkan di area jaringan *server*. *Sensor network* menggunakan sebuah perangkat *raspberry pi* 3 dengan sistem operasi *raspbian buster lite*. Sedangkan *server* pengumpulan data menggunakan sebuah perangkat komputer *server HP ProLiant ML110* dengan sistem operasi *Ubuntu server* 16.04. *Sensor network* menjalankan aplikasi *tcpdump* untuk menangkap paket ARP *request* selama 3 jam dan setelah itu mengirimnya ke *server* pengumpulan data dengan menggunakan aplikasi *rsync* (Rsync, n.d.). Dalam penelitian ini, *sensor network* di pasang dalam LAN Universitas Kanjuruhan Malang dengan kabel LAN yang dihubungkan ke *switch*. Sedangkan *server* pengumpulan data dipasang di area jaringan *server* Universitas Kanjuruhan Malang yang membentuk topologi seperti pada Gambar 1. Pengumpulan data dilakukan pada saat *host* yang berada dalam LAN Universitas Kanjuruhan Malang banyak yang aktif. Pengumpulan data dilakukan pada hari Senin, 13 Januari 2020 jam 09.00-12.00 dengan tujuan bahwa pada rentang waktu tersebut merupakan jam kerja dan banyak *host* aktif yang terhubung dalam LAN.



Gambar 1. Topologi penelitian

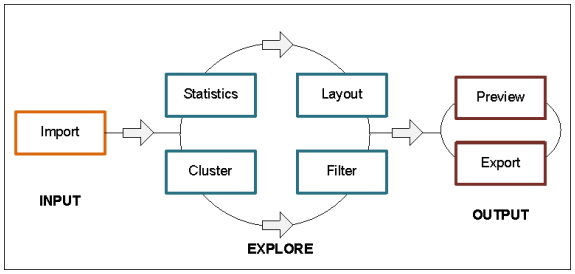
Pengumpulan data dilakukan dengan tangkapan data khusus paket ARP *request*. Paket ARP *request* dalam format PCAP yang terkumpul kemudian ditransformasi ke dalam bentuk hubungan *node* dan *edge* dengan format CSV dalam relasi *graph* G=(V,E), seperti diilustrasikan pada pada Gambar 2. Paket ARP *request* pada dasarnya berisikan permintaan informasi akan kepemilikan sebuah IP *address* "*who has* 192.168.5.76? *tell* 192.168.4.131" seperti tampak pada Gambar 2. IP *address* pengirim ARP *request* disebut sebagai *host* A, sedangkan IP *address* target ARP *request* disebut sebagai *host* B. Dengan demikian, jika direpresentasikan dalam bentuk data visualisasi *graph* dengan format CSV yang dipisahkan dengan koma, maka dapat direpresentasikan sebagai "192.168.4.131, 192.168.5.76" yang menyatakan sebuah relasi 2 buah *node* (*host*) yang dihubungkan oleh *edge* (*link*).



Gambar 2. Transformasi paket ARP *request*

3.2. Visualisasi Graph

Visualisasi *graph* merupakan rekayasa dalam pembuatan gambar berupa sekumpulan *node* yang dihubungkan oleh *edge* yang memiliki tujuan untuk mendapatkan informasi (Sudakov, 2016). Visualisasi *graph* dirumuskan dengan G=(V,E) artinya (V) adalah himpunan dari *node* dan (E) adalah himpunan dari *edge*. Seperti contoh himpunan *edge* (E)={(a,b),(a,c),(b,c)} dan himpunan *node* (V)={a,b,c}, selanjutnya dalam visualisasi *graph* terdapat label *node* yang biasa ditulis dengan angka atau huruf (seperti contoh IP *address*: 192.168.4.20) dan kemudian terdapat label *edge* juga biasa ditulis dengan angka atau huruf (seperti contoh *weight edge*: 10).



Gambar 3. Proses visualisasi *graph* dengan *Gephi*
Sumber : Bastian, Heymann dan Jacomy (2009)

Berdasarkan data visualisasi *graph* yang dihasilkan, proses visualisasi *graph* dapat dilakukan dengan bantuan aplikasi *open source* yaitu *Gephi* (Bastian, Heymann dan Jacomy, 2009; Grandjean, 2015). Proses visualisasi *graph* dengan aplikasi *Gephi* dapat dilihat pada Gambar 3. Proses pertama adalah melakukan *import* data visualisasi *graph* yang berformat CSV, selanjutnya melakukan proses visualisasi *graph* tanpa menggunakan teknik *graph clustering-filtering* dan proses visualisasi *graph* menggunakan teknik *graph clustering-filtering.* Terakhir melakukan *export* hasil visualisasi *graph* dengan format *file* PNG/SVG/PDF.

Ilustrasi hasil visualisasi *graph* tanpa teknik *graph clustering* dapat dilihat pada Gambar 4(a) yang menampilkan jumlah *node* dan *edge* sama dengan jumlah data terkumpul. Jika semakin banyak data terkumpul, maka *node* dan *edge* yang ditampilkan juga semakin banyak. Selain itu, ilustrasi hasil visualisasi *graph* tanpa teknik *graph clustering-filtering* menampilkan ukuran serta warna *node* dan *edge* yang sama, sehingga tidak dapat membedakan *node* satu dengan *node* yang lain. Oleh karena itu, ilustrasi hasil visualisasi *graph* tanpa teknik *graph clustering-filtering* menunjukkan hasil visualisasi *graph* yang sulit untuk dibaca.

3.3. Teknik Graph Clustering-Filtering

Teknik *graph clustering-filtering* merupakan pengelompokan dan penyaringan *node* dan *edge* dengan mempertimbangkan parameter yang ada dalam *graph* (Inoubli, dkk., 2019). Parameter dalam *graph* seperti *weight edge, degree node, in-degree node, out-degree node, weight degree node, weight in-degree node,* dan *weight out-degree node*. Namun, tidak semua parameter itu dipakai untuk proses visualisasi *graph* dalam identifikasi *malicious host*, karena dalam identifikasi *malicious host* yang dijadikan acuan adalah jumlah ARP *request* sebuah hubungan *node,* total jumlah ARP *request* sebuah *node* dan jumlah *host* target sebuah *node*. Oleh karena itu, parameter yang akan digunakan untuk proses visualisasi *graph* dalam identifikasi *malicious host* adalah *weight edge* (jumlah ARP *request* sebuah hubungan *node*), *weight out-degree node* (total jumlah ARP *request* sebuah *node*), dan *out-degree node* (jumlah *host* target sebuah *node*).

Berdasarkan parameter yang digunakan untuk proses visualisasi *graph*, maka penelitian ini mengusulkan teknik *graph clustering-filtering* dengan 3 tahap sebagai berikut:

1. *Clustering host* berdasarkan warna dan ukuran terhadap nilai *weight out-degree node*

$$wd^{-}(HP) = \sum_{i=1}^{n} WE(HP, HT_i) \qquad (1)$$

2. *Clustering link* berdasarkan warna dan ukuran terhadap nilai *weight edge*

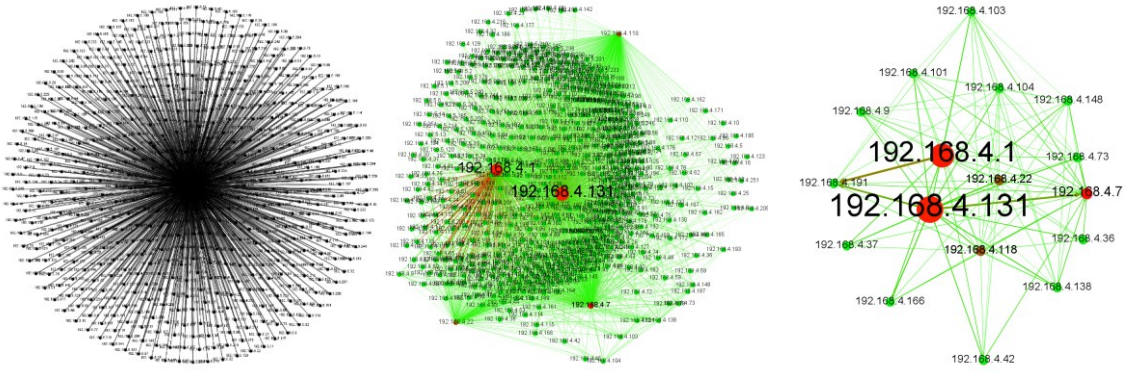
$$WE(HP, HT) = \sum_{i=1}^{n} E_i(HP, HT) \qquad (2)$$

3. *Filtering host* dan *link* terhadap nilai *out-degree node*

$$d^{-}(HP) = \sum_{i=1}^{n} (HT_i) \qquad (3)$$

Pada persamaan (1) merupakan persamaan untuk menghitung *weight out-degree node* (total jumlah ARP *request* sebuah *node*), wd^{-} *adalah weight out-degree node.* HP adalah *host* pengirim ARP *request*, HT adalah *host* target ARP *request, sedangkan* WE adalah *weight edge.* Pada persamaan (2) merupakan persamaan untuk menghitung *weight edge* (jumlah ARP *request* sebuah hubungan *node*), WE adalah *weight edge.* HP adalah *host* pengirim ARP *request*, HT adalah *host* target ARP *request* dan E adalah *edge* (*link*). Pada persamaan (3) merupakan persamaan untuk menghitung *out-degree node* (jumlah *host* target sebuah *node*), d^{-} *adalah out-degree node.* HP adalah *host* pengirim ARP *request*, HT adalah *host* target ARP *request.*

Berdasarkan tahap 1 dan 2, proses visualisasi *graph* yang dilakukan adalah menggunakan teknik *graph clustering*. Tahap 1 adalah melakukan pengelompokan *host* berdasarkan warna dan ukuran *host* terhadap nilai *weight out-degree node* (total



(a) Tanpa teknik *graph clustering-filtering* (b) Teknik *graph clustering* (c) Teknik *graph clustering-filtering*

Gambar 4. Ilustrasi hasil visualisasi *graph*

jumlah ARP *request* sebuah *node*), sehingga jika ukuran *host* yang lebih besar dan warna lebih merah, maka *host* tersebut memiliki nilai *weight out-degree node* atau total jumlah ARP *request* yang banyak. Selanjutnya pada tahap 2 adalah melakukan pengelompokan *link* berdasarkan warna dan ukuran *link* terhadap nilai *weight edge* (jumlah ARP *request* sebuah hubungan *node*), sehingga jika ukuran *link* yang lebih besar dan warna lebih merah, maka *link* tersebut memiliki nilai *weight edge* atau jumlah ARP *request* yang banyak.

Ilustrasi hasil visualisasi *graph* menggunakan teknik *graph clustering* dapat dilihat pada Gambar 4(b) yang menampilkan *node* dan *link* yang sudah dibedakan ukuran dan warna. Namun, ilustrasi hasil visualisasi *graph* menggunakan teknik *graph clustering* menampilkan jumlah *node* dan *edge* sama dengan jumlah data terkumpul, sehingga jika semakin banyak data terkumpul, maka *node* dan *edge* yang ditampilkan juga semakin banyak. Oleh karena itu, ilustrasi hasil visualisasi *graph* menggunakan teknik *graph clustering* menunjukkan hasil visualisasi *graph* yang sulit untuk dibaca.

Dengan melakukan tahap 1 sampai 3, maka proses visualisasi *graph* yang dilakukan adalah dengan menggunakan teknik *graph clustering-filtering*. Selain melakukan pengelompokan pada tahap 1 dan 2, teknik *graph clustering-filtering* melakukan penyaringan *host* dan *link* terhadap nilai *out-degree node* (jumlah *host* target ARP *request* sebuah *node*), sehingga hasil visualisasi *graph* dapat menampilkan jumlah *node* dan *edge* yang lebih sedikit daripada jumlah data terkumpul. Ilustrasi hasil visualisasi *graph* menggunakan teknik *graph clustering-filtering* dapat dilihat pada Gambar 4(c) yang menampilkan *node* dan *link* yang sudah dibedakan ukuran dan warna, serta jumlah *node* dan *edge* yang tampil lebih sedikit daripada jumlah data terkumpul. Dengan demikian, ilustrasi hasil visualisasi *graph* menggunakan teknik *graph clustering-filtering* menunjukkan hasil visualisasi *graph* yang dapat dengan mudah dibaca.

Aturan dalam identifikasi *malicious host* dengan membaca hasil visualisasi *graph* adalah jika melihat *link* yang memiliki ukuran semakin besar dan warna semakin merah, maka *link* tersebut dapat diidentifikasi sebagai *malicious activity*. Jika *malicious activity* tersebut terdapat pada *link* (*host* a, *host* b), maka *malicious activity* tersebut dilakukan oleh *host* a dan kemudian *host* a dapat diidentifikasi sebagai *malicious host*, kecuali *host* tersebut bukan *gateway host*. Sedangkan jika melihat *host* yang memiliki ukuran semakin besar dan warna semakin merah, maka *host* tersebut dapat diidentifikasi sebagai *malicious host*, kecuali *host* tersebut bukan *gateway host*. Dengan demikian, identifikasi *malicious host* dapat dengan mudah untuk dilakukan.

Namun, hasil visualisasi *graph* menggunakan teknik *graph clustering-filtering* masih bersifat subjektif dalam keputusan identifikasi *malicious host*. Oleh karena itu, penelitian ini perlu untuk melakukan penghitungan persentase aktivitas *host* dengan mempertimbangkan parameter dari *graph*.

3.4. Persentase Aktivitas Host

Berdasarkan parameter dari *graph* seperti *weight out-degree node* dan *out-degree node*, *weight out-degree node* merupakan total jumlah ARP *request* dalam sebuah *node*, ketika dalam visualisasi *graph* digunakan untuk *clustering node* dalam membedakan ukuran dan warna *node*. Sedangkan *out-degree node* merupakan jumlah *host* target ARP *request* dalam sebuah *node*, ketika dalam visualisasi *graph* digunakan untuk filtering *node* dan *edge* agar jumlah *node* dan *edge* yang tampil semakin sedikit. Kedua parameter *graph* tersebut merupakan parameter yang dijadikan acuan dalam deteksi *malicious activity* yang dapat menemukan *malicious host* (Matsufuji, dkk., 2019). Oleh karena itu, penghitungan persentase aktivitas *host* dapat dilakukan dengan kedua parameter *graph* tersebut. Persentase aktivitas *host* berfungsi untuk menunjukkan seberapa besar tingkat aktivitas sebuah *host* dalam melakukan *broadcast* ARP *request*.

Dengan demikian, penelitian ini mengusulkan penghitungan persentase aktivitas *host* dengan mempertimbangkan parameter *graph* seperti *weight out-degree node* dan *out-degree node* sebagai berikut:

1. Normalisasi *weight out-degree node*

$$nwd^{-} = \frac{wd^{-}(HP_i) - \min(wd^{-}(HP))}{\max(wd^{-}(HP)) - \min(wd^{-}(HP))} \quad (4)$$

2. Normalisasi *out-degree node*

$$nd^{-}(HP_i) = \frac{d^{-}(HP_i) - \min(d^{-}(HP))}{\max(d^{-}(HP)) - \min(d^{-}(HP))} \quad (5)$$

3. Rata-rata hasil normalisasi *weight out-degree node* dan *out-degree node*

$$rn(HP_i) = \frac{nwd^{-}(HP_i) + nd^{-}(HP_i)}{2} \quad (6)$$

4. Persentase aktivitas *host*

$$p(HP_i) = rn(HP_i)\ x\ 100 \quad (7)$$

Pada persamaan (4) merupakan persamaan untuk menghitung normalisasi *weight out-degree node*, *nwd-* adalah normalisasi *weight out-degree node.* Sedangkan *wd- adalah weight out-degree node,* HP adalah *host* pengirim ARP *request.* Pada persamaan (5) merupakan persamaan untuk menghitung normalisasi *out-degree node*, *nd-* adalah normalisasi *out-degree node.* Sedangkan *d- adalah out-degree node*, HP adalah *host* pengirim ARP *request.* Pada persamaan (6) merupakan persamaan untuk menghitung rata-rata hasil normalisasi *weight out-degree node* dan *out-degree node*, *rn* adalah rata-rata hasil normalisasi dari *weight out-degree node* dan *out-degree node.* Sedangkan *nwd-* adalah normalisasi *weight out-degree node, nd-* adalah normalisasi *out-*

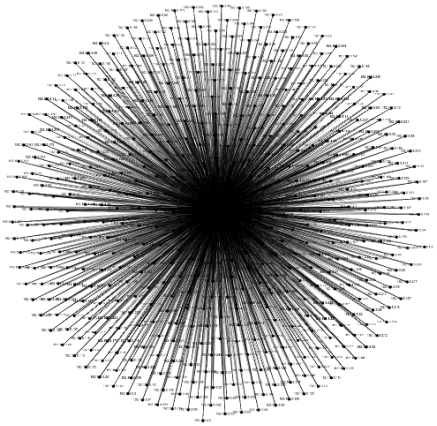
degree node, HP adalah *host* pengirim ARP *request.* Pada persamaan (7) merupakan persamaan untuk menghitung persentase aktivitas *host*, *p* adalah nilai persentase aktivitas *host* (%). Sedangkan *rn* adalah rata-rata nilai hasil normalisasi dari *weight out-degree node* dan *out-degree node,* HP adalah *host* pengirim ARP *request.*

Hasil penghitungan persentase aktivitas *host* menunjukkan bahwa seberapa besar nilai aktivitas *host* dalam melakukan *broadcast* ARP *request* dalam rentang waktu tertentu. Dengan demikian, jika nilai persentase aktivitas *host* semakin besar, maka *host* tersebut dapat diidentifikasi sebagai *malicious host*, kecuali *host* tersebut bukan *gateway host*.

4. HASIL DAN PEMBAHASAN

4.1. Hasil Identifikasi Malicious Host

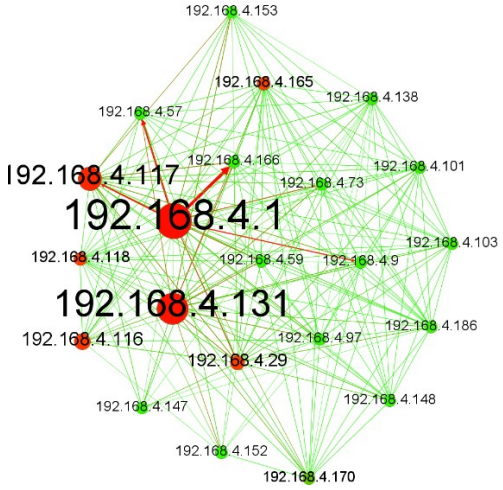
Hasil identifikasi *malicious host* pada 13 Januari 2020 jam 09.00-12.00 dapat dilakukan dengan melihat hasil visualisasi *graph* menggunakan teknik *graph clustering-filtering* dan hasil penghitungan persentase aktivitas *host*. Pertama kali penelitian ini menjelaskan hasil visualisasi *graph* tanpa menggunakan teknik *graph clustering-filtering* yang dapat dilihat pada Gambar 5. Hasil visualisasi *graph* tanpa menggunakan teknik *graph clustering-filtering* berdasarkan data terkumpul yaitu 511 *node* dan 4144 *edge,* setelah diproses visualisasi *graph* dengan aplikasi *Gephi* tanpa menggunakan teknik *graph clustering-filtering* menghasilkan jumlah *node* dan *edge* yang jumlahnya sama dengan data terkumpul. Dengan demikian, *node* dan *edge* yang tampil pada hasil visualisasi *graph* sangat banyak dan sulit untuk dibaca.



Gambar 5. Hasil visualisasi *graph* tanpa menggunakan teknik *graph clustering-filtering*

Sedangkan hasil visualisasi *graph* menggunakan teknik *graph clustering-filtering* menghasilkan visualisasi *graph* yang dapat dilihat pada Gambar 6. Berdasarkan data terkumpul yaitu 511 *node* dan 4144 *edge,* setelah diproses visualisasi *graph* dengan aplikasi *Gephi* menggunakan teknik *graph clustering-filtering* menghasilkan jumlah *node* (22) dan *edge* (328). Dengan demikian, *node* dan *edge* yang tampil pada hasil visualisasi *graph* dapat dengan mudah dibaca.

Dengan mengikuti aturan dalam identifikasi *malicious host*, hasil visualisasi *graph* menggunakan teknik *graph clustering-filtering* yang terlihat pada Gambar 6 menunjukkan bahwa IP *address* 192.168.4.1 memiliki ukuran lebih besar dan warna lebih merah, tetapi IP *address* tersebut bukan *malicious host* melainkan adalah *gateway host*. Sedangkan IP *address* 192.168.4.131, 192.168.4.117, 192.168.4.116, 192.168.4.29, 192.168.4.118, dan 192.168.4.165 adalah IP *address* yang diidentifikasi sebagai *malicious host,* karena memiliki ukuran *node* lebih besar dan warna *node* lebih merah.



Gambar 6. Hasil visualisasi *graph* menggunakan teknik *graph clustering-filtering*

Selain melihat hasil visualisasi *graph* menggunakan teknik *graph clustering-filtering*, keputusan untuk memastikan IP *address* yang diidentifikasi sebagai *malicious host* dapat dilakukan dengan melihat hasil penghitungan persentase aktivitas *host* berdasarkan persamaan (4-7). Hasil penghitungan persentase aktivitas *host* menghasilkan 6 *node* yang memiliki persentase aktivitas *host* terbesar, *node* tersebut dengan IP *address* 192.168.4.131 dengan nilai persentase aktivitas *host* (91,2%), sementara 192.168.4.117 (74,7%), 192.168.4.116 (56,6%), 192.168.4.29 (55,3%), 192.168.4.118 (52,4%), dan 192.168.4.165 (27,3%) yang dapat dilihat pada Tabel 1.

Tabel 1. Hasil Penghitungan Persentase Aktivitas *Host*

IP *Address*	Nilai Persentase Aktivitas *Host*
192.168.4.131	91,2 %
192.168.4.117	74,7 %
192.168.4.116	56,6 %
192.168.4.29	55,3 %
192.168.4.118	52,4 %
192.168.4.165	27,3 %

Dengan demikian, hasil identifikasi *malicious host* pada 13 Januari 2020 jam 09.00-12.00 dengan melihat hasil visualisasi *graph* menggunakan teknik *graph clustering-filtering* dan hasil penghitungan

persentase aktivitas *host* menemukan 6 *node* yang diidentifikasi sebagai *malicious host*. *Node* tersebut yaitu *node* dengan IP *address* 192.168.4.131, 192.168.4.117, 192.168.4.116, 192.168.4.29, 192.168.4.118 dan 192.168.4.165.

4.2. Verifikasi Malicious Host

Berdasarkan hasil identifikasi *malicious host* pada 13 Januari 2020 jam 09.00-12.00, IP *address* yang telah dicurigai sebagai *malicious host* dilakukan verifikasi ke lokasi tempat komputer yang memiliki IP *address* tersebut. Tahap pertama verifikasi adalah proses pemeriksaan pada *router* dalam LAN untuk mencari *hostname* dan MAC *address* berdasarkan IP *address* yang diidentifikasi sebagai *malicious host,* seperti contoh IP *address* 192.168.4.131 dengan *hostname* (*Admin-PC*) dan MAC *address* (*00:00:00:00:00:11*). Tahap verifikasi selanjutnya adalah proses pemeriksaan komputer yang ada di lokasi. Jika saat di lokasi menemukan komputer yang memiliki IP *address* dengan *hostname* dan MAC *address* yang diidentifikasi sebagai *malicious host*, maka proses selanjutnya adalah proses pemeriksaan terhadap komputer tersebut. Pemeriksaan yang dilakukan adalah pemeriksaan sistem operasi komputer dan pemeriksaan program atau service yang berjalan pada komputer.

Hasil pemeriksaan komputer yang diidentifikasi sebagai *malicious host,* semua komputer rata-rata memiliki sistem operasi kadaluarsa dan sedang menjalankan program atau *service* yang membahayakan seperti *tTab virus* dan *browser_assisten.exe*. *tTab virus* merupakan program yang menyelinap ke komputer tanpa persetujuan pengguna dengan memberikan banyak tindakan jahat seperti memodifikasi pengaturan *browser default* dan memunculkan *popup* iklan pada aktivitas web *browser* pengguna. Sedangkan *browser_assistent.exe* merupakan program *adware* yang memunculkan *popup* iklan pada *web browser*. Dengan demikian, semua komputer yang diidentifikasi sebagai *malicious host* terverifikasi sebagai komputer yang membahayakan, sehingga semua komputer tersebut dapat dilakukan tindakan lebih lanjut, agar penyebaran *malware* semakin berkurang dan LAN menjadi lebih aman.

5. KESIMPULAN

Hasil penerapan teknik *graph clustering-filtering* terhadap 511 *node* dan 4144 *edge* yang didapatkan melalui pengamatan dan pengambilan data selama 3 jam dalam LAN kampus dapat divisualisasikan menjadi hanya 22 *node* dan 328 *edge*. Penghitungan persentase aktivitas *host* berdasarkan *weight out-degree node* dan *out-degree node* dapat menunjukkan *malicious host* dari 22 *node* menjadi 6 *node* yang diidentifikasi sebagai *malicious host*. Berdasarkan hasil verifikasi di lapangan, semua *node* yang diidentifikasi sebagai *malicious host* tersebut rata-rata memiliki sistem operasi yang kadaluarsa dan sedang menjalankan program atau *service* yang membahayakan seperti *tTab virus* dan *browser_assisten.exe*. Dengan demikian, teknik *graph clustering-filtering* dan penghitungan persentase aktivitas *host* dapat mengidentifikasi *malicious host*.

DAFTAR PUSTAKA

BASTIAN, M., HEYMANN, S. dan JACOMY, M. (2009) 'Gephi: An Open Source Software for Exploring and Manipulating Networks. BT - International AAAI Conference on Weblogs and Social', *International AAAI Conference on Weblogs and Social Media*, pp. 361–362.

BOND, T. (2009) 'Visualizing Firewall Log Data to Detect Security Incidents', *Global Information Assurance Certification Paper Copyright*.CHAKRABORTY, A. dkk. (2018) 'Application of Graph Theory in Social Media', *International Journal of Computer Sciences and Engineering*, 6(10), pp. 722–729. doi: 10.26438/ijcse/v6i10.722729.

CHAKRABORTY, A. dkk. (2018) 'Application of Graph Theory in Social Media', *International Journal of Computer Sciences and Engineering*, 6(10), pp. 722–729. doi: 10.26438/ijcse/v6i10.722729.

DESTA, D. H. (2014) *Visualization of PRADS Output Data Using Open-source Visualization Tools For Improved Log Analysis*. UNIVERSITY OF OSLO Department of Informatics.

GRANDJEAN, M. (2015) 'GEPHI: Introduction to network analysis and visualization'. [online] Tersedia di: <http://www.martingrandjean.ch/gephi-introduction> [Diakses 27 Maret 2020].

HUBBALLI, N., BISWAS, S., ROOPA, S., RATTI, R. dan NANDI, S. (2011) 'LAN Attack Detection using Discrete Event Systems', *ISA Transactions*. Elsevier Ltd, 50(1), pp. 119–130. doi: 10.1016/j.isatra.2010.08.003.

INOUBLI, W. dkk. (2019) 'A Distributed Algorithm for Large-Scale Graph Clustering', *L'archive Ouverte Pluridisciplinaire HAL*, p. hal-02190913v2. [online] Tersedia di: <https://hal.inria.fr/hal-02190913v2> [Diakses 27 Maret 2020].

MARKO, P. dan VILHAN, P. (2012) 'Efficient Detection of Malicious Nodes based on DNS and Statistical Methods', *IEEE 10th Jubilee International Symposium on Applied Machine Intelligence and Informatics, SAMI 2012 - Proceedings*. IEEE, pp. 227–230. doi: 10.1109/SAMI.2012.6208963.

MATSUFUJI, K., KOBAYASHI, S., ESAKI, H. dan

OCHIAI, H. (2019) 'ARP Request Trend Fitting for Detecting Malicious Activity in LAN', *Advances in Intelligent Systems and Computing*, 935, pp. 89–96. doi: 10.1007/978-3-030-19063-7_8.

MICROSOFT (2018) 'Microsoft Security Intelligence Report', *Microsoft Security Intelligence Report*, 24(Januari-December), pp. 1–19. [online] Tersedia di: <http://download.microsoft.com/download/7/2/B/72B5DE91-04F4-42F4-A587-9D08C55E0734/Microsoft_Security_Intelligence_Report_Volume_16_English.pdf> [Diakses 27 Maret 2020].

OCHIAI, H. (2019) 'LAN-Security Monitoring Project Background : Cyber-Security Research', *Asia Pasific Advanced Network*. [online] Tersedia di: <https://www.lan-security.net/whitepaper.pdf> [Diakses 27 Maret 2020].

Rsync, n.d. [online] Tersedia di: <https://rsync.samba.org> [Diakses 27 Maret 2020].

SUDAKOV, B. (2016) *Graph Theory*. Institute of Technology Zurich. [online] Tersedia di: <https://www2.math.ethz.ch/education/bachelor/lectures/fs2016/math/graph_theory/graph_theory_notes.pdf> [Diakses 27 Maret 2020].

Tcpdump, n.d. [online] Tersedia di: <https://www.tcpdump.org> [Diakses 27 Maret 2020].

VALLI, C. (2009) 'Visualisation of Honeypot Data Using Graphviz and Afterglow', *Journal of Digital Forensics, Security and Law*, (January). doi: 10.15394/jdfsl.2009.1056.

WHYTE, D., KRANAKIS, E. dan OORSCHOT, P. VAN (2005) 'ARP-Based Detection of Scanning Worms within an Enterprise Network', *Annual Computer Security Applications Conference (ACSAC)*.