An Efficient Network Attack Visualization Using Security Quad and Cube

Beom-Hwan Chang and Chi Yoon Jeong

Security quad and cube (SQC) is a network attack analyzer that is capable of aggregating many different events into a single significant incident and visualizing these events in order to identify suspicious or illegitimate behavior. A network administrator recognizes network anomalies by analyzing the traffic data and alert messages generated in the security devices; however, it takes a lot of time to inspect and analyze them because the security devices generate an overwhelming amount of logs and security events. In this paper, we propose SQC, an efficient method for analyzing network security through visualization. The proposed method monitors anomalies occurring in an entire network and displays detailed information of the attacks. In addition, by providing a detailed analysis of network attacks, this method can more precisely detect and distinguish them from normal events.

Keywords: Network security visualization, network monitoring, security situational awareness.

I. Introduction

Security visualization has recently emerged as a means to address an open research challenge: rapidly detecting and classifying the enormous traffic anomalies in today's large networks [1], [2]. Such techniques, which are aimed at the situational awareness of the human user by visually representing security information, are enabling automated network security monitoring and analyses to be performed in an accurate and timely manner; however, most security visualization tools have suffered in terms of their ability to detect small insignificant attacks under enormous amounts of activities or to differentiate between suspicious and legitimate behaviors due to a limited display space. Also, it takes a lot of time before the behavior of an unknown attack is detected and classified, and/or correlated as a known attack, for a proper response.

Visualization of network security is a way of representing the security situation of a network using security events as diagrams of abstract graphs in 2D or 3D space. These security events including traffic data and alert messages are generated in various network and/or security devices, such as routers, switches, intrusion detection system (IDS), intrusion prevention system (IPS), virtual private network (VPN), firewalls, and others [3], [4]. Security visualization techniques are classified by dividing them into signature-based, anomaly-based, or both signature-based and anomaly-based visualization methods, similar to IDS. One way to classify the existing security visualization techniques is based on the nature of their data source. Some of the tools developed are NVisionIP [5], VisFlowConnect-IP [6], SnortView [7], and PortVis [8].

The primary goal of our research is to find and prevent enormous traffic attacks and the unknown and trifling attacks

Manuscript received Sept. 29, 2010; revised Feb. 1, 2011; accepted Feb. 24, 2011.

This work was supported by the Technology Innovation Program funded by the Ministry of Knowledge Economy (MKE, Korea), (No. 10035237).

Beom-Hwan Chang (phone: +82 42 860 0623, email: bchang@etri.re.kr) and Chi Yoon Jeong (email: iamready@etri.re.kr) are with the Software Research Laboratory, ETRI, Daejon, Rep. of Korea.

http://dx.doi.org/10.4218/etrij.11.0110.0570

buried in huge amounts of traffic. Another goal is to accurately detect suspicious and legitimate behavior, and instinctively provide the properties and patterns of anomalies, attacks from the same attacker, and unknown attacks.

The remainder of this paper is organized as follows. In section II, we present related work on security visualization. In section III, we propose an efficient visualization using security quad and cube (SQC) to detect network attacks. In section IV, we describe the experimental results used to evaluate our studies. Finally, we conclude this paper in section V with some brief remarks.

II. Related Work

A significant amount of work has been introduced in the area of network security visualization. In this section, we focus our study on work that is closely related to SQC design objectives: network security visualization and attack detection.

Many visualization tools have been introduced to enhance network security. IDS RainStorm [9] visualizes alarms generated by an IDS and provides a main view presenting an overall representation of the entire destination IP range and a zoom view that provides more information on a user selected range of IP addresses. Zooming and drilling down for further details can be performed at the user's discretion. SnortView [7] uses a 2D time diagram and three main frames: source address, alert, and source-destination matrix all focus on dealing with false alerts. All data is sorted by time. The IP addresses in the source address frame are listed and sorted vertically. The vertical axis in the alert frame represents the list of source IP addresses, and the horizontal axis represents the time. Each colored glyph, or icon, is an IDS alert. Each color displays different priorities, red being the highest and blue the lowest. Each shape (square, circle, star, and so on) of the glyph represents a type of attack. There is a detailed window at the bottom to provide further information.

Le Malécot and others [10] introduced an original visualization design that combines 2D and 3D representations of network traffic and activity. Both representations are based on the same interactive grid representation of the network space, and are linked together as they provide complementary functionalities. The 3D representation provides an overview of the communication between several network zones, while the 2D representation provides a detailed view of selected parts of the 3D representation. Ball and others [11] presented VISUAL, an IP address oriented technique that simply maps values of each byte of an IP address onto horizontal and vertical axes of the display spaces. VISUAL provides insight on networks with many internal and external hosts, shows the relative activity of the hosts, displays them in a constant relative position, and reveals the ports and protocols used.

IDGraphs [12] uses flow records and presents one of the following over a particular time frame: the number of successful connections or the number of unique IP and destination port pairs used. To avoid overlapping pixels, luminance is used to show the data density. In IDGraphs, the central visualization is a flow-level trace plotted over time on the horizontal axis and the aggregated number of unsuccessful connections on the vertical axis. PortVis represents a destination port as a 2-byte number mapped to a matrix with x and y axes and produces images of network traffic mainly by choosing axes that correspond to important features of the data, such as time and port number, creating a grid based on these axes and then filling each cell of the grid with a color that represents the network activity there.

The spinning cube [13] plots traffic in a 3D cube of source IP versus destination IP versus destination port axes. The amount of network activity is visualized interactively using color, displaying certain attacks, particularly port scans, clearly. NVisionIP [5] uses a graphical representation of an entire class-B network to allow users to quickly visualize the current state of the network. The galaxy view gives a visual picture of the current state of an entire class-B network. Both the small multiple view (SMV) and machine view are presented with more detailed information on abnormal traffic patterns for the set of hosts selected. VisFlowConnect-IP [6] uses a compact data structure and efficient algorithms for generating statistics and visual representations, thus enabling it to run uninterrupted for long periods and on large data sets. The value of VisFlowConnect-IP, specifically for security situational awareness, is that any security event, with only a few minor exceptions, will be reflected as a traffic flow.

However, these tools only focus on visualizing network traffic to assist users in understanding network events and detecting single attacks in an ideal situation. We must also detect smaller and less significant attacks under enormous amounts of activity and identify suspicious or legitimate behavior under ambiguous situations when such attacks are mixed within normal activities.

III. Security Quad and Cube

1. Aggregation and Distinct Dispersion

Routers, switches, firewalls, IDS, IPS, VPN, anti-virus software, and servers produce heterogeneous event data as traffic information or attack alerts. The data from multiple devices is categorized and tagged with a descriptor that includes information about the nature of the event. Heterogeneous events can therefore be evaluated and

normalized in a common format. We chose five fields typically used to define an event: protocol number (*prt*), source IP address (*src*), source port number (*spt*), destination port number (*dpt*), and destination IP address (*dst*) known from [14] and [15]. These elements are commonly included in security events and best express the meaning of an event.

Aggregation is a very useful method for characterizing and monitoring network anomalies and attacks [16]. Almost all denial-of-service (DoS) attacks, worms, port scans, host scans, and flash crowds have either a unique or diverse source IP address, destination IP address, source port number, and/or destination port number [14], [15]. This means that these activities show up concentrated in one point or in many scattered points if they are aggregated by two or three fields in an event. In order for this aggregation to be achieved, we first group the events by protocol number and then aggregate each event using two fields of *src*, *spt*, *dpt*, and *dst*. For example, the aggregaton using *src* and *spt* is denoted by *Agg*(*dpt*, *dst*), *Agg*(*src*, *spt*, *x**, *y**), or *Agg*(1100). Next, each distinct dispersion value of unselected remainders, D_x and D_y, is numerically calculated by

$$\begin{aligned} S = \{&Agg(src, spt, x^*, y^*),\ Agg(src, x^*, dpt, y^*),\\ &Agg(src, x^*, y^*, dst),\ Agg(x^*, spt, dpt, y^*),\\ &Agg(x^*, y^*, dpt, dst),\ Agg(x^*, spt, y^*, dst)\}, \end{aligned} \quad (1)$$

$$D_x = \frac{d(x^*)}{n(S)}, D_y = \frac{d(y^*)}{n(S)}, \quad (\text{where, } 0 < D_x, D_y \le 1),$$

where S is the set of events that are derived to form *Agg*(*x*, *y*), which is defined by the set of aggregated events except both *x* and *y*. Also, $d(x)$ is a number of unique *x*, $n(S)$ is a number of elements in set *S*, and *x** is all possible values of *x*.

2. Security Quad and Cube

A security quad is a square comprised of all distinct dispersions of the coordinate values of the *x*-axis and *y*-axis. This square aims to group two attributes from four main attributes of the traffic, such as *src*, *spt*, *dpt*, and *dst*, and analyzes the relation between the remaining two attributes. Thereafter, it detects the group of events showing an uncommon relation and displays it instinctively on the security quad. Figure 1 shows the security quad is created from *Agg*(*,*, *dpt*, *dst*), each color displays different TCP or UDP port numbers.

There is a disadvantage in representing events using the security quad in that, while similar movements of events are displayed well, it is impossible to reflect their weighted value. For example, if two distinct dispersions, D_x and D_y, of S_1 are equal those of S_2, then event groups, S_1 and S_2, are displayed in the same point, even if $n(S_1)$ does not equal $n(S_2)$. In order to

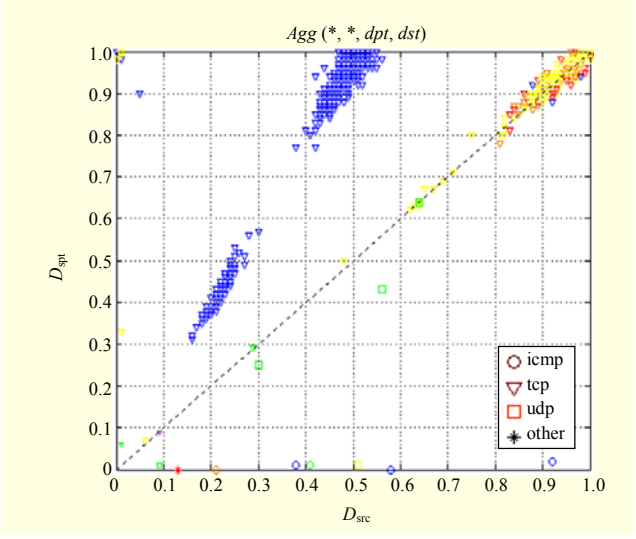


Fig. 1. Security quad using *Agg*(*x**, *y**, *dpt*, *dst*).

solve this disadvantage, the security cube assigns the weighted value of an event group to the *z*-axis and displays the severity using the weighted value of the event group. The weighted value E^2 is double the compensation entropy or number of events and is calculated from (2), where $p(i)$ is the probability mass function of outcome *i*. The value of the compensation entropy is less sensitive to the amount of network traffic compared to entropy value *H*, which is a measurement of data randomness.

$$\begin{aligned} H &= -\sum_{i=1}^{n} p(i) \cdot \log_2 p(i), \\ E &= H \times \sqrt{\frac{dn}{n}}, \\ E^2 &= \sqrt{E_x \times E_y}. \end{aligned} \quad (2)$$

Figure 2 shows a security cube calculated by *Agg*(*, *, *dpt*, *dst*) and (2). In a 3D security cube, *x*-axis and *y*-axis coordinates are calculated using a unique dispersion degree of two features with a 2D security quad.

The E^2 values from the *z*-axis denote the amount of network attack. That is, if the E^2 values are equal, it denotes the same number of attempted attacks or the same amount of attempted network attacks. It also means that there is a large probability of an attack from a similar attacker. The greater the value of E^2 is, the higher the risk shown. In order to effectively and clearly display such risk levels, dots of different sizes, shapes, and colors can be displayed according to the calculated entropy values.

The events representing an allegorical pattern of an identical attacker or similar anomaly show up as adjacent points on SQC because their behavior is similar. Behavior-based detection

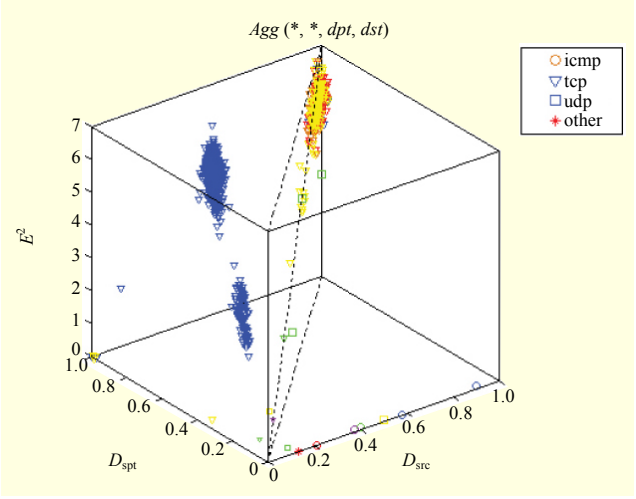


Fig. 2. Security cube by *Agg*(*x**, *y**, *dpt*, *dst*).

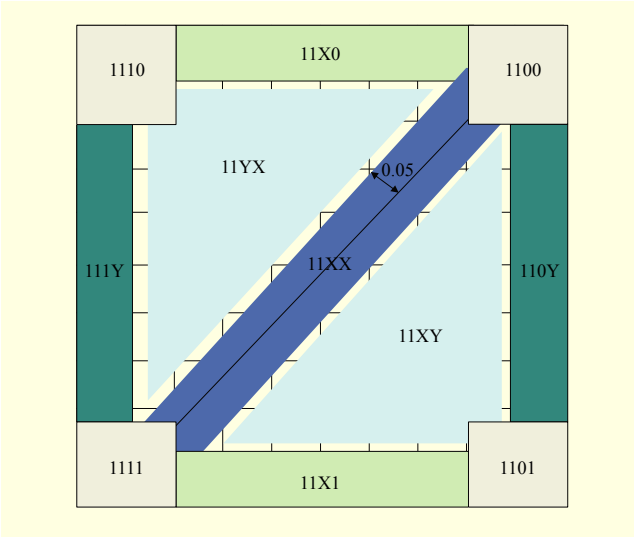


Fig. 3. LBC for *Agg*(1100).

algorithms using distinct dispersion and entropy go well beyond typical volume or signature-based attacks such as DoS and DDoS attacks, viruses, and worms [7].

3. Attack Clustering

Attack clustering is necessary to group similar attack patterns in SQC and to find an attacker quickly. There are many clustering algorithms, such as *K*-means clustering and self-organization map (SOM), but they are not adequate for clustering huge amounts of traffic data. Because they use iterative techniques to find optimal solutions, they require a lot of processing time. Therefore, we propose two efficient clustering algorithms: location-based clustering (LBC) and grid-map clustering (GMC) for attack clustering.

Algorithm 1. LBC

```
1: begin
2: input E
3: for k ← 0 to EVT_NUM do
4:     x ← E[k]·x
5:     y ← E[k]·y
6:     if x < 0.1 then
7:         if y < 0.2 then
8:             class[k] ← "1111"
9:         else if y < 0.8 then
10:            class[k] ← "111Y"
11:        else
12:            class[k] ← "1110"
13:        end if
14:    else if x < 0.2 then
15:        if y < 0.2 then
16:            class[k] ← "1111"
17:        else if y ≥ 0.8 then
18:            class[k] ← "1110"
19:        else if [|ax+by+c|/√(a²+b²)]a=1,b=-1,c=0 ≤ 0.05
20:            class[k] ← "11XX"
21:        else
22:            class[k] ← "11XY"
24:        end if
25:    else if x < 0.8 then
26:        if y < 0.1 then
27:            class[k] ← "11X1"
28:        else if y ≥ 0.9 then
29:            class[k] ← "11X0"
30:        else if [|ax+by+c|/√(a²+b²)]a=1,b=-1,c=0 ≤ 0.05
31:            class[k] ← "11XX"
28:        else if x/y < 1 then
29:            class[k] ← "11XY"
32:        else
33:            class[k] ← "11YX"
34:        end if
35:    else if x < 0.9 then
36:        if y < 0.2 then
37:            class[k] ← "1101"
38:        else if y ≥ 0.8 then
39:            class[k] ← "1100"
40:        else if [|ax+by+c|/√(a²+b²)]a=1,b=-1,c=0 ≤ 0.05
41:            class[k] ← "11XX"
42:        else
43:            class[k] ← "11YX"
44:        end if
45:    else
46:        if y < 0.2 then
47:            class[k] ← "1101"
48:        else if y < 0.8 then
49:            class[k] ← "11Y1"
50:        else
51:            class[k] ← "1100"
52:        end if
53:    end if
54: end for
54: output class
55: end
```

A. Location-Based Clustering

The LBC method quantifies the object space into a finite number of cells forming a grid structure and then performs clustering operations based on the location of each grid. Figure 3 shows a security quad with divided areas by *Agg*(1100).

In Fig. 3, a 2D security quad is sectored into 11 areas based on network behavior analysis, generally abnormal traffic behavior shows up concentrated in one point or in many scattered points, and if traffic data is present in a predefined area, the traffic data is classified as traffic having the feature of that predetermined area.

Algorithm 1 is a method of classifying abnormal traffic based on the areas of a security cube, which are sectored as in Fig. 3. In order to classify abnormal traffic, algorithm 1 determines where abnormal traffic is based on the value of the *x*-axis. Then, the area of the abnormal traffic is finally determined using the value of the *y*-axis. In Fig. 3 and Algorithm 1, *x* denotes a unique dispersion degree value of a destination port, and *y* is a unique dispersion degree value of a destination address. If the value of *x* is smaller than 0.1, and the value of y is smaller than 0.2, the LBC algorithm determines that the traffic belongs to the area, 1111. If the value of *x* is smaller than 0.1, and the value of *y* is equal to or larger than 0.2, it determines that the traffic belongs to the area, 111Y. If the value of *x* is smaller than 0.1, and the value of *y* is larger than 0.8, it determines that the traffic belongs to the area, 1110. According to such a scheme, traffic belonging to each area is defined and classified based on the characteristics of that area. This method has the advantage of quick classification, but has the limitation of seriously influencing an abnormal traffic classification result in a predefined area.

B. Grid-Map Clustering

The GMC method partitions each dimension of SQC into the same number of equal length intervals and calculates the grid features from the existing grid data. The grid features are the top three ports and their ratio within the grid. Although a network attack program can easily change the source or destination port, network traffic generated from the same attack program has similar distinct dispersions. Also, network traffic using the same port has the same network attack patterns. Therefore, distinct dispersion and the port number are important information in classifying network attacks.

Figure 4 shows grid-map clustering using a similarity of grids in a 2D security quad divided by $N \times N$ lattices. This method calculates the similarity between grids at $L(x,y)$ and with its 8 neigborhoods using the features of each grid. The comparison uses (3) and algorithm 2.

In Fig. 4, grouping is performed by comparing a predetermined lattice $L(x, y)$ with adjacent lattices in the arrow direction, which goes from $L(0, 0)$ to $L(n, n)$, or in the reverse direction, which goes from $L(n, n)$ to $L(0, 0)$. Information such as the port list for each protocol, frequency per port, rate per port for all data sets, locations, and areas of a group in a 2D security quad are extracted from the group that is determined as the same group through the grouping operation. The extracted information is used to determine abnormal traffic conditions.

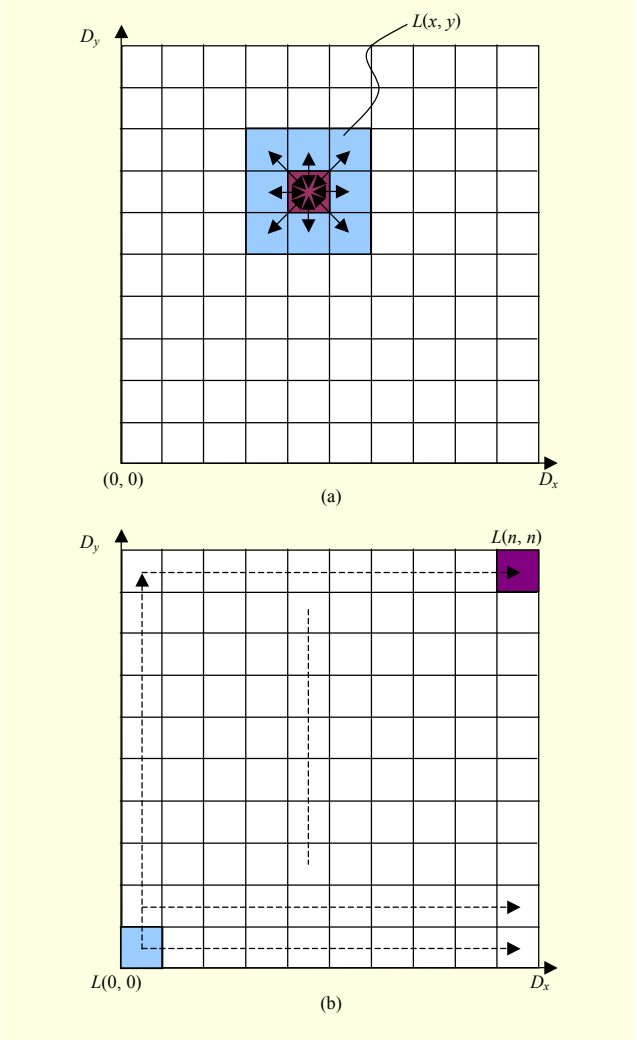


Fig. 4. GMC.

$$S = \sum_{k=0}^{2} \sum_{i=0}^{N} \sum_{j=0}^{N} w_{ijk} \cdot r_{ik} \cdot r_{jk} \,, \tag{3}$$

$$\text{where} \quad w_{ijk} = \begin{cases} 1 \,, \text{ if } p_{ik} = p_{jk} \,, \\ 0 \,, \text{ if } p_{ik} \neq p_{jk} \,. \end{cases}$$

A similarity, $s(x, y)$, between a predetermined lattice L and adjacent lattices, for example, can be detected by multiplying the function values with the parameters, such as weight w_{ijk} (where k is the protocol), and p_{ik} and p_{jk} are the port numbers and r_{ik} and r_{jk} are the ratios of the top N ports. Network attacks

using the same ports or that have a similar distinct dispersion show a high similarity. If the similarity between two grids exceeds a threshold, the two grids are in the same cluster. Since the proposed method does not use iterative techniques and calculates a similarity using the property of the grid, we can speed up the clustering method over existing methods. Attack clustering is calculated as follows:

Algorithm 2. GMC.

```
1: begin
2: input E
3: label ← 0
4: for k ← 0 to EVT_NUM do
5:    x ← E[k]·x
6:    y ← E[k]·y
7:    Cell[x][y] ← number and count of DPT
8: end for
9: for i ← 0 to GRID_NUM do
10:    for j ← 0 to GRID_NUM do
11:       if n(Cell[i][j]) >= min_element then
12:          class_map[i][j] ← label
13:          label ← label + 1
14:       else
15:          class_map[i][j] ← 0
16:       end if
17:       P[i][j] ← top port number of top N ports
18:       R[i][j] ← the ratio of top N ports
19:    end for
20: end for
21: for i ← 0 to GRID_NUM do
22:    for j ← 0 to GRID_NUM do
23:       if class_map[i][j] > 0 then
24:          for k ← 1 to 8 do
25:             calculate the similarity for N-th
                neigborhood using P[i][j] and R[i][j]
26:             if similarity > min_similarity then
27:                class_map[i][j] ← MIN(class_map[i][j],
                                         neigborhood)
28:             end if
29:          end for
30:       end if
31:    end for
32: end for
33: output class_map
34: end.
```

4. Spectrum Analysis of IP address

It is still perfectly fine to define whether suspicious behavior at each point is an attack using only SQC because each point can be either an attack or the normal behavior of the server. Therefore, we perform two detailed analyses for each point. One is a spectrum analysis of *src* and *dst* using a parallel coordinate chart, and the other is the uniformity and continuity test of *spt* and *dpt*.

Figure 5 shows an example of a parallel coordinate graph

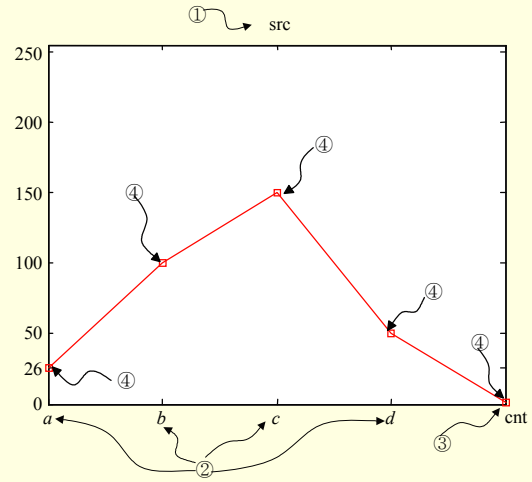


Fig. 5. Parallel coordinate graph of IP addresses.

based on the parallel coordinate division of the source IP address. ① denotes a title indicating the attribute of an IP address (for example, a source IP address or a destination IP address). ② references an IP address represented by an Internet address scheme. The IP address generally has a length of 32 bits, and includes four attribute fields, *a*, *b*, *c*, and *d*, each of which is composed of 8 bits. The IP address is divided into four 8-bit subnetwork values. The divided subnetwork values, where one subnetwork value is composed of one attribute field, are represented on each parallel axis on the *x*-axis in the form of identifiers, that is, *a*, *b*, *c*, and *d*. Finally, ③ denotes the number of events, which increases whenever an event composed of *a*, *b*, *c*, and *d* is generated. The event count is represented as the last parallel axis on the *x*-axis.

The numerical values, 0, 26, 50, 100, 150, 200, and 250, shown on the *y*-axis are used to improve the identification of the range of IP address ②. The value of *a* is 26, which is the first attribute field of IP address ② and is represented on the *y*-axis to improve the identification performance. The values of *b*, *c*, and *d* are 100, 150, and 50, respectively, which are the other attribute fields of IP address ② and are represented in the forms of points ④ at the points where the parallel axes intersect the *y*-axis. The points denoted by ④ may be represented in triangle or rectangle shapes. Of course, event count ③ is also represented in the shape of a point. To improve the identification performance, the parallel coordinate division display links points ④ and event count ③ on a parallel coordinate chart in order to draw a line graph. Finally, to identify suspicious behavior, we assign a unique color according to the A-class IP address of the events and confirm the line spectrum displayed on the parallel coordinate chart. If it is normal behavior, the line spectrum is made up of very different colors, and the group of lines consists of many A-class

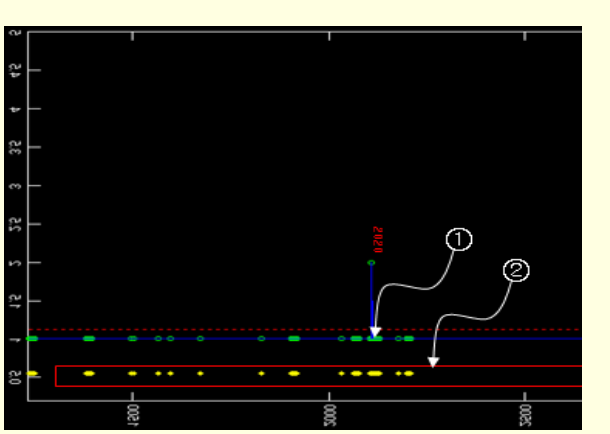


Fig. 6. Uniformity and continuity tests.

IP addresses. Moreover, the lines over 5% of the entire events do not appear. If it is an attack, the spectrum is very simple and a large group of lines still exists.

Figure 6 shows the uniformity (①) and continuity (②) tests of *spt* and *dpt* used to detect port scan attacks. In Fig. 5, the *x*-axis represents the used port range, and the *y*-axis denotes the occurring counts of each port. In general, the characteristics of port scan attacks have a uniform occurrence and a continuing range of destination port distribution.

IV. Experiments and Results

To evaluate the ability to recognize attacks of SQC, we designed both the actual traffic test data generated in a real network and the experimental test data using attack tools.

Figure 7 shows six quads as a summary of each anomaly during a five-minute period using the Netflow data exported from an IP-router with 155 Mbps interfaces on APII (via KREONET-StarTap) between Korea and the USA. The left-top quad calculated by *Agg*(*, *, *dpt*, *dst*) represents a cluster of an anomaly causing several traffic flows from the various destination port numbers and destination IP addresses, which lasts for as long as the specific source IP address and source port number. The right-top quad calculated by *Agg*(*, *spt*, *, *dst*) represents a cluster showing uncommon relations between the destination IP address and source port number that comprise several traffic flows through an aggregation of the specific source IP address and the specific destination port number.

SQC allows a real-time analysis of incoming security events via correlation of the characteristics, which is calculated using the cluster using the LBC and GMC methods as well as the occupancy rates, in terms of their unique variances and entropies based on the IP addresses and port numbers assigned to the source and destination. We can precisely analyze a wide variety of attacks occurring across a network based on the similarity of clustering. Each anomaly detected is displayed as a dot, and similar events are grouped together into a cluster. Dots have varying shapes according to their protocol. Ports are mapped using different colors for greater visual distinction. In our implementation, we provide summary information in Fig. 7, such as the cluster pattern and port information, displayed next to each cluster. Essential details on the relationship between the source IP address, source port, destination port, and destination IP address are also displayed. This information is extremely useful for quickly responding to an attack, as it enables rapid determination of the attack type and ports used for the attack. We expect that these models and methods will help developers and researchers implement their security management systems for network security.

Figure 8 shows the locations of network attacks on the six security quads calculated by an LBC algorithm. A host scan attack appears on the security quads using *Agg*(1100), *Agg*(1010), and *Agg*(0110). A port scan attack is observed in the quads made by *Agg*(1100), *Agg*(1001), and *Agg*(0101). A DoS attack is displayed on the quads created by *Agg*(1010), *Agg*(1001), and *Agg*(0011). A DDoS attack is observed on the quads using *Agg*(0110), *Agg*(0101), and *Agg*(0011). As shown in Fig. 8, the host scan, port scan, and DDoS attacks are displayed in various locations of the quads calculated by *Agg*(1010), *Agg*(1001), and *Agg*(0011) because there are many different variations of a 4-tuple (*src*, *spt*, *dpt*, *dst*) of an event. For example, if a variation of the destination IP address associated with the host attacks changes 1/2 or 1/3 of the variation of an entire event, the *y*-coordinate value of the attack grows from the bottom of *y*-axis.

Figure 9 shows seven event groups, six attack clusters 1 to 6 and one normal cluster 7 by server activity on a security quad. Also, each event group of clusters is analyzed in detail based on the spectrum techniques. In Figs. 10 and 11, the normal server, worm, host scan attack, and port scan attack can be distinguished through a detailed analysis using the spectrum display of the IP address as well as the port uniformity and continuity test. If it is normal behavior, the line spectrum of IP address is made up of very different colors, however if it is an attack, the spectrum is very simple and has the large group of the same color lines. Also, the traffic is artificially generated and the characteristics of the source and destination ports appear as continuous and uniform.

V. Conclusion

SQC reduces the time required for analyzing security events by easily visualizing their behavior. Security events including network traffic flows correlated with anomalies are different from normal flows. These events appear to have a high

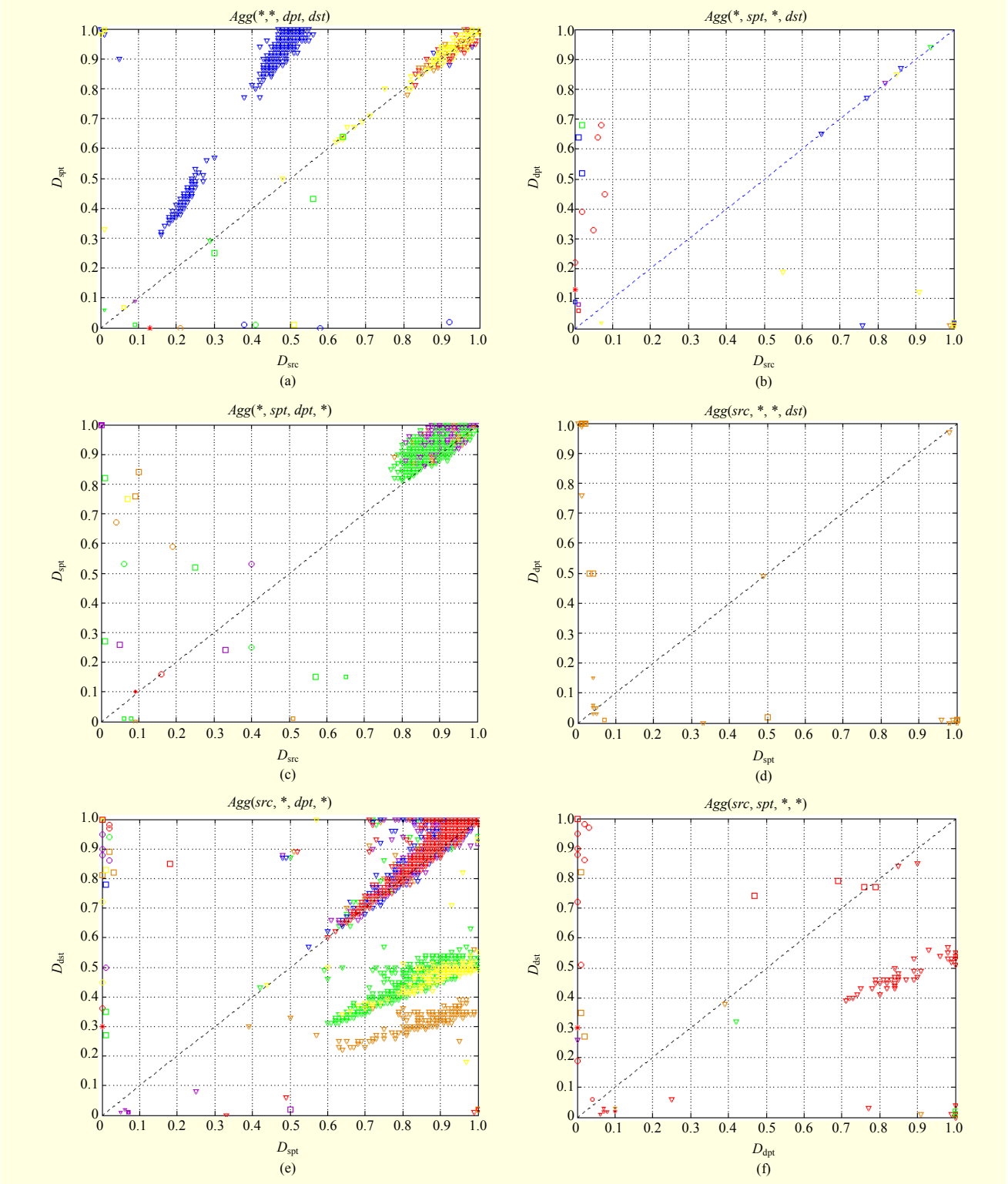


Fig. 7. Security quad (where $n(S) \geq 50$ during 5-min period): shape and color of each dot is determined by a certain protocol and port number.

potential, and they show up as concentrations in one point or as many scattered points on a security quad/cube because their distinct dispersions of *src, spt, dpt*, and *dst* are similar.

Our experimental results showed that SQC can rapidly detect a network attack and classify unknown attacks into a known attack cluster. In addition, SQC is able to detect even the

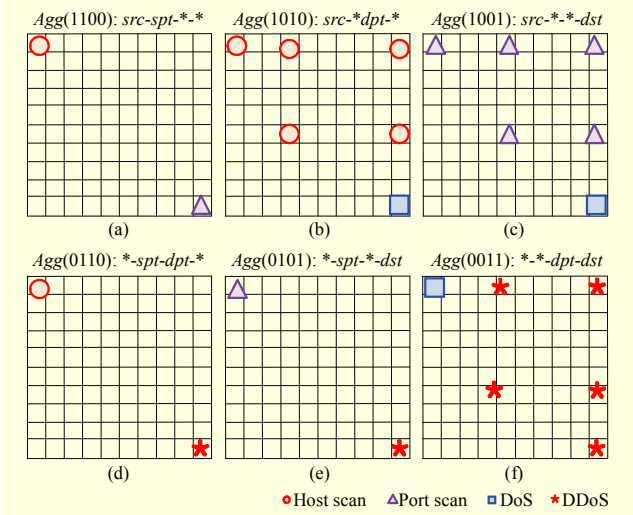


Fig. 8. Attack clusters by LBC algorithm.

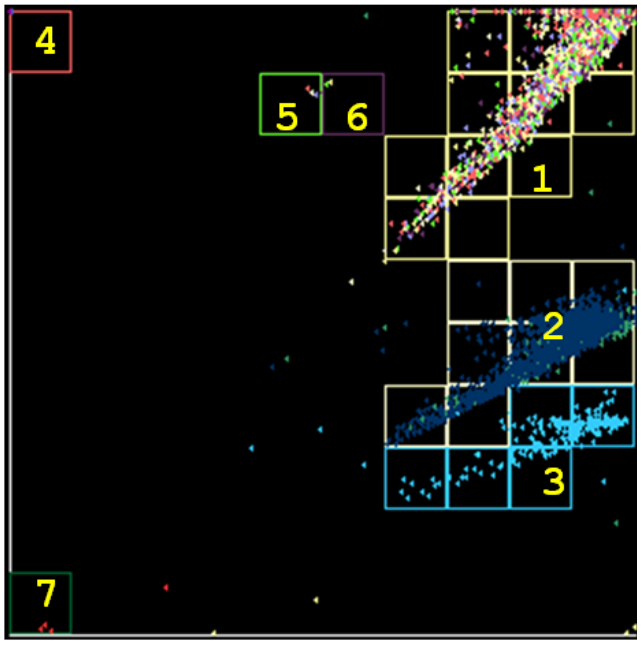


Fig. 9. Attack clusters on quad (1010) by GMC algorithm.

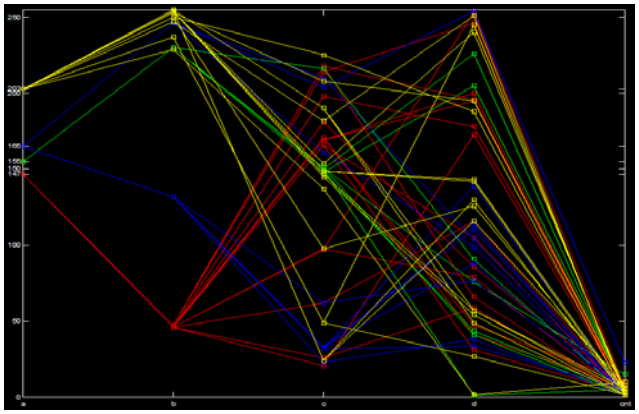
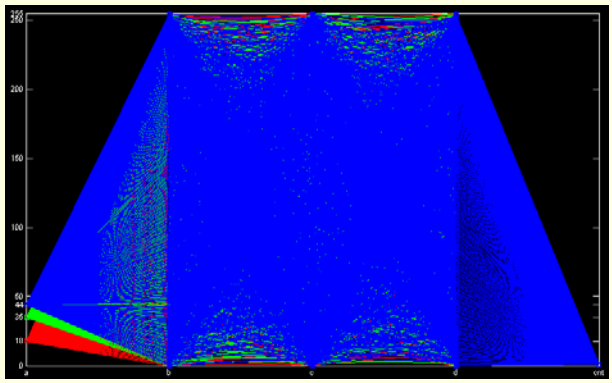
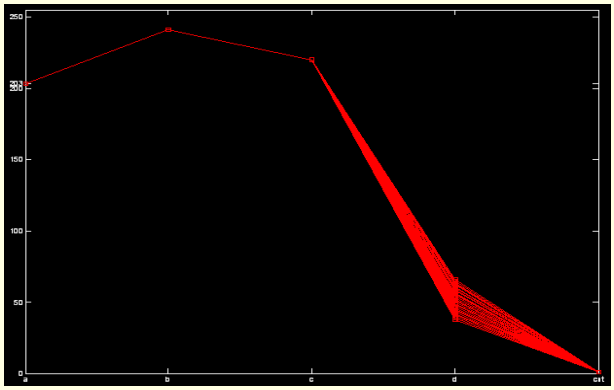


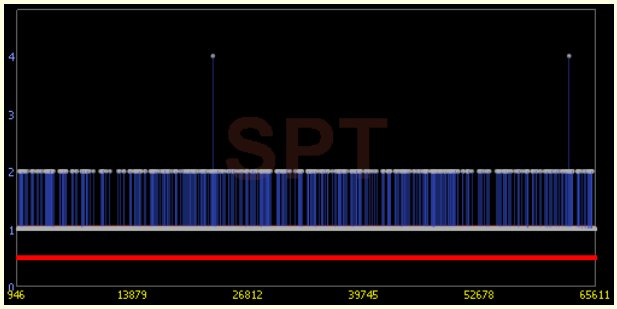
Fig. 10. Detailed spectrum analysis of normal server.



(a) Slammer worm



(b) Host scan attack



(c) Port scan attack

Fig. 11. Detailed spectrum analysis of the attacks: (a) slammer worm, (b) host scan attack, and (c) port scan attack.

smallest anomalies. Also, an effective tool to pinpoint the cause of a network anomaly, SQC, provides detailed information about a detected abnormal activity, including its characteristics and patterns, signaling also whether an attack originates from the same source as another attack or whether or not it is a known type of attack. We expect that SQC will be able to easily detect DoS attacks, worms, port scans, host scans, and traffic anomalies as well as unknown attacks. It will also be very helpful in intuitively recognizing the security of a current network. Without knowing the network security, a network administrator can know the security situation of a network and whether or not it is under attack.

References

[1] E.W. Bethel et al., "Accelerating Network Traffic Analytics Using Query-Driven Visualization," *IEEE Symp. Vis. Anal. Sci. Technol.*, 2006, pp. 115-122.

[2] A. Wagner and B. Plattner, "Entropy Based Worm and Anomaly Detection in Fast IP Networks," *Proc. 14th IEEE Int. WET ICE*, 2005, pp. 172-177.

[3] J. Kim, S. Radhakrishnan, and J. Jang, "Cost Optimization in SIS Model of Worm Infection," *ETRI J.*, vol. 28, no. 5, Oct. 2006, pp. 692-695.

[4] J. Lee et al., "PKG-VUL: Security Vulnerability Evaluation and Patch Framework for Package-Based Systems," *ETRI J.,* vol. 31, no. 5, Oct. 2009, pp. 554-564.

[5] K. Lakkaraju, W. Yurcik, and A.J. Lee, "NVisionIP: Netflow Visualizations of System State for Security Situational Awareness," *Proc. ACM Workshop VizSEC/DMSEC*, 2004, pp. 65-72.

[6] X. Yin, W. Yurcik, and A. Slagell, "The Design of VisFlowConnect-IP: A Link Analysis System for IP Security Situational Awareness," *Proc. 3rd IEEE Int. Workshop Inf. Assurance*, 2005, pp. 141-153.

[7] H. Koike and K. Ohno "SnortView: Visualization System of Snort Logs," *Proc. ACM Workshop VizSEC/DMSEC*, 2004, pp. 143-147.

[8] J. Mcpherson et al., "Portvis: A Tool for Port-Based Detection of Security Events," *Proc. ACM Workshop VizSEC/DMSEC*, 2004, pp. 73-81.

[9] K. Abdullah et al., "IDS RainStorm: Visualizing IDS Alarms," *Proc. IEEE Vis. Comput. Security*, 2005, pp. 1-10.

[10] Le Malécot et al., "Interactively Combining 2D and 3D Visualization for Network Traffic Monitoring," *Proc. 3rd ACM Int. Workshop Vis. Comput. Security*, 2006, pp. 123-127.

[11] R. Ball, G.A. Fink, and C. North, "Home-Centric Visualization of Network Traffic for Security Administration," *Proc. ACM Workshop VizSEC/DMSEC,* 2004, pp. 55-64.

[12] P. Ren et al., "IDGraphs: Intrusion Detection and Analysis Using Histographs," *Proc. IEEE Vis. Comput. Security*, 2005, pp. 39-46.

[13] S. Lau, "The Spinning Cube of Potential Doom," *Comm. ACM*, vol. 47, no. 6, June 2004, pp. 25-26.

[14] Y. Hu, "Adaptive Flow Aggregation–A New Solution for Robust Flow Monitoring under Security Attacks," *Proc. 10th IEEE/IFIP Netw. Operations Manage. Symp.*, 2006, pp. 424-435.

[15] S. Krasser et al., "Real-Time and Forensic Network Data Analysis Using Animated and Coordinated Visualization," *Proc. 6th IEEE Info. Assurance Workshop*, 2005, pp. 42-49.

[16] A. Giani et al., "Attribution and Aggregation of Network Flows for Security Analysis," *Proc. 3rd CERT/CC Annual Workshop Flow Anal.*, 2006.

Beom-Hwan Chang received his BS, MS, and PhD from Sungkyunkwan University, Rep. of Korea, in 1997, 1999, and 2003, respectively. Since 2003, he has been a senior researcher with the Software Research Laboratory, ETRI, Daejeon, Rep. of Korea. His research interests include network security, programmable network, situation awareness, and security visualization.

Chi Yoon Jeong received his BS and MS in electronic and electrical engineering from Pohang University of Science and Technology (POSTECH), Rep. of Korea, in 2002 and 2004, respectively. Since 2004, he has been a researcher in the Software Research Laboratory at ETRI, Daejeon, Rep. of Korea. His research interests include computer vision, network security, and visual pattern recognition.