

Detection of Denial of Service attacks using AGURI

Ryo Kaizaki
Keio Univ.
kaizaki@sfc.wide.ad.jp

Kenjiro Cho
SonyCSL
kjc@csl.sony.co.jp

Osamu Nakamura
Keio Univ.
osamu@wide.ad.jp

Abstract

Denial of Service attacks is divided into two types, one is logic attack and the another one is flooding attack. Logic attack exploits security holl of the software such as operating system and web server bugs, then causes system crash or degrade in the performance. Logic attack can be defended by upgrading software and/or filtering particular packet sequences.

Comparing each packets of the flooding attack and the other normal communication traffics, the only difference is the number of the packets. Flooding attack creates enormous amount of packets. Therefore, to protect systems from flooding attacks, the same method for logic attacks can not be used. During the network operations, flooding attack is usually detected by using traffic monitoring tools such as MRTG. However those tools will not detect the attack automatically.

In this paper, the method for automatic detection of the flooding attacks is described. For the monitoring tools, AGURI, that we have developed, is used. Using the traffic pattern aggregation method, AGURI can monitor the traffics in a long term and detect flooding attacks.

1 Introduction

Internet is the packet switching network, sharing the every resources such as the bandwidth of the links and router's processing unit. Resource management should be done by every end node. For example, congestion controls can be done only by end nodes. Denial of Service attacks, especially flooding attack, is ill behavior on the end node. However current Internet does not have any mechanisms to control this ill behavior. During the network operations, it is very important to detect the flooding attacks as soon as possible. After detecting the flooding attacks, operators can take several actions such as filtering the packets from ill behaving hosts and discovering the attacker.

Denial of Service attacks is divided into two types[1], one is logic attack and the another one is flooding attack. Logic attack exploits security holl of the software such as operating system and web

server bugs, then causes system crash or degrade in the performance. Logic attack can be defended by upgrading software and/or filtering particular packet sequences.

Comparing each packets of the flooding attack and the other normal communication traffics, the only difference is the number of the packets. Flooding attack creates enormous amount of packets. Therefore, to protect systems from flooding attacks, the same method for logic attacks can not be used. During the network operations, flooding attack is usually detected by using traffic monitoring tools such as MRTG[2]. However those tools will not detect the attack automatically.

In this paper, the method for automatically detecting the flooding attacks is described. AGURI[3], that we have developed, is used as a monitoring tool. Using the traffic pattern aggregation method, AGURI can monitor the traffics in a long term and detects flooding attacks.

2 Traffic monitoring for flooding attacks

There are several types of flooding attacks.

1. the large number of the bytes
2. the large number of the packets
3. packets with ill behavior protocols such as sync attack

The traffic with the large number of the bytes for the single destination degrades the performance of the end system and the routers that switching this traffic. And recent routers incur more damages by recieving the large number of packets rather than bytes.

That traffic can be monitored by using SNMP[4]. MRTG is good graphic interface for the detecting the unusual traffic. But it is not sufficient for detecting the flooding attacks. There is limitation of gathering the information using SNMP. The number of the bytes and the packets for the each interface on the routers can be collected. However the number of the byte and the packets to the single hosts can not be collected. If the bandwidth of the link was occupied in general condition, particular

attacks could not be detect by using SNMP/MRTG monitoring, because total bandwidth of the link is not changed.

For detecting the flooding attacks, we should know the normal conditions of the networks. It needs for large number of the traffic data. SNMP is simple mechanisms for collecting the data from the routers and switches. It is needed for aggregation mechanisms for storing the data. NeTraMet and FlowScan which are flow based monitoring tools can monitor specific type of the traffic, such as number of bytes and packets in a long term on HTTP, FTP, IPv6 etc. However these tools require the fixed rule sets. So those tools can not detect unexpected traffic pattern.

3 AGURI

AGURI is an aggregation based traffic profiler targeted for long term measuring. AGURI adapts itself to spatial traffic distribution by aggregating small volume flows into its root. AGURI does not need a pre-defined rule set and is capable of detecting an unexpected increase of unknown packet patterns or flooding attacks.

Figure 1 shows the concept of aggregation: small entries are aggregated into its root. It is the basic algorithm of AGURI's aggregation that monitoring every packets and , at the end, aggregating entries whose counter value is less than an aggregation threshold.

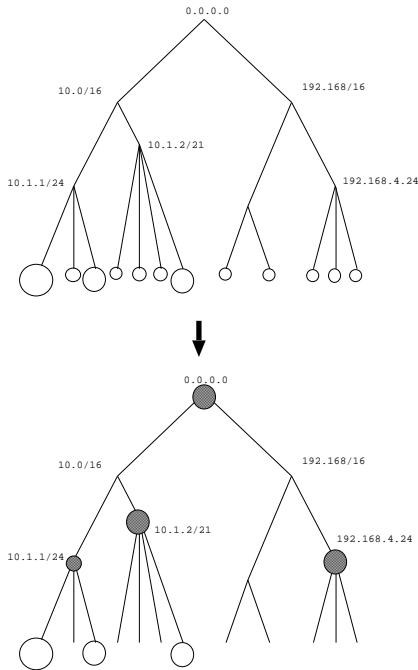


Figure 1: aggregation concept

In figure 1, each circle shows entries and its counter value is indicated by its size. Each filled dot shows sets of aggregated entries whose counter

value is less than an aggregation threshold. For example, the filled dot “10.1.2/21” shows set of aggregated entries whose counter value is less than an aggregation threshold and whose IP address is included in address block “10.1.2/21”.

Figure 2 shows an example of aguri's summary output. A summary consists of header part and body part.

The header part describes version, start-time of profiling, end-time of profiling and average-rate of all traffic. The header part starts with %.

The body part contains 4 profile types:

1. source ip address
2. destination ip address
3. source protocol
4. destination protocol

```

%%!AGURI-1.0
%%StartTime: Thu Mar 01 00:00:00 2001 (2001/03/01 00:00:00)
%%EndTime: Sun Apr 01 00:00:00 2001 (2001/04/01 00:00:00)
%AvgRate: 14.91Mbps

[src address] 4992392109177 (100.00%)
0.0.0.0/0 87902964189 (1.76%/100.00%)
0.0.0.0/1 206637364377 (4.14%/14.78%)
0.0.0.0/2 205796877844 (4.12%/7.12%)
60.0.0.0/6 97928228974 (1.96%/3.00%)
    62.52.0.0/16 51875058871 (1.04%/1.04%)
    64.0.0.0/8 100831910967 (2.02%/3.51%)
    64.0.0.0/9 74610984109 (1.49%/1.49%)
128.0.0.0/2 142349668983 (2.85%/13.33%)
128.0.0.0/3 197067746696 (3.95%/10.48%)
128.0.0.0/5 202911635757 (4.06%/5.45%)
133.0.0.0/8 69142535628 (1.38%/1.38%)
    150.65.136.91 54123094932 (1.08%)
192.0.0.0/4 212653628837 (4.26%/38.41%)
192.0.0.0/6 8885538654 (1.78%/1.78%)
202.0.0.0/7 235853368912 (4.72%/14.70%)
202.0.0.0/9 117196493427 (2.35%/6.77%)
    202.12.27.33 160473669718 (3.21%)
    202.30.143.128/25 60239291958 (1.21%/1.21%)
    203.178.143.127 94031811680 (1.88%)
204.0.0.0/6 228960094456 (4.59%/17.68%)
204.0.0.0/8 125458765333 (2.51%/7.58%)
    204.123.7.2 87103414877 (1.74%)
    204.152.184.75 165733431144 (3.32%)
206.0.0.0/7 164036959478 (3.29%/5.51%)
206.128.0.0/9 53526598302 (1.07%/1.07%)
207.0.0.0/8 57628266965 (1.15%/1.15%)
208.0.0.0/4 282590640975 (5.66%/31.72%)
208.0.0.0/6 116047154301 (2.32%/22.20%)
209.0.0.0/8 140888988219 (2.82%/11.78%)
    209.1.225.217 238192306019 (4.77%)
    209.1.225.218 209160635530 (4.19%)
210.0.0.0/7 154008321340 (3.08%/3.08%)
216.0.0.0/9 192899750315 (3.86%/3.86%)
%LRU hits: 86.82% (1021/1176)

```

Figure 2: Example of AGURI summary output

In the address profile, each row shows an address entry and is indented by the prefix length. The first column shows the address and the prefix length of the entries. The second column shows the cumulative byte counts. The third column shows the percentages of the entry and its subtrees.

Using AGURI's script, we can archive summaries with minimum disk space. This enables long term measurements.

Thus, AGURI achieves long term traffic monitoring and detecting characteristic flows without a pre

defined rule set.

4 Design

For detection of flooding attacks, this paper defines original parameter “Deviation(D)” between characteristic of packet-pattern in long-term and current characteristic.

If the parameter “D” is high, we can guess current packet-pattern is unusual.

Based on the idea, following 2 schemes are needed.

- long-term traffic archiving and current traffic monitoring
- method of calculating “Deviation(D)”

4.1 long-term traffic monitoring

We use AGURI to archive characteristic of traffic in a long term.

AGURI uses a traffic profiling technique in which records are maintained in a prefix based tree a compact summary which is produced by erentries.

Figure3 shows tree structure of archiving summaries. In figure3, AGURI generate hourly summary “A” by aggregating minutes summaries “1”-“12”. We can see various summaries of time scale granularity .

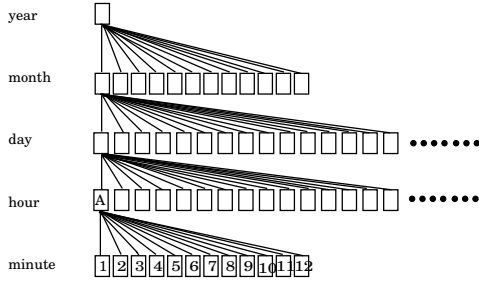


Figure 3: archiving structure of AGURI

4.2 Calculation of “Deviation”

This paper defines original parameter “Deviation(D)” between characteristic of packet pattern in long term and current characteristic.

Figure 4 shows a basic example to calculation of Deviation.

In Figure 4, T1 is traffic summary tree in a long term and T2 is current traffic summary tree. “Distance” is different node depth in tree structure. a, b and c are expressions of node in the tree structure.

In this basic model: all nodes are located as same position and have different value, “Deviation(D)” can be calculated by the following calculs.

$$D = (T1[a] - T2[a])^2$$

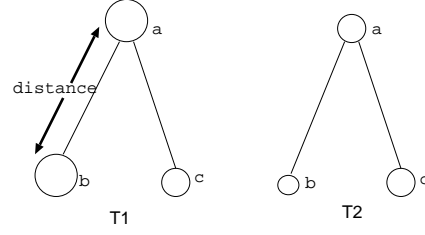


Figure 4: Basic model

$$+(T1[b] - T2[b])^2 + (T1[c] - T2[c])^2$$

In this calculs, T1[a] is an expression of an average throughput rate of node “a” in T1. Defining node a,b,c,...,i,... ,this calculs lead us to abstract following calculs.

$$D = \Sigma(T1[i] - T2[i])^2$$

However, on real traffic pattern using AGURI, a few nodes are located as same position. We have to consider sameness of nodes in which aggregated different depth.

Figure 5 shows a example which nodes aggregated in tree structure with different depth.

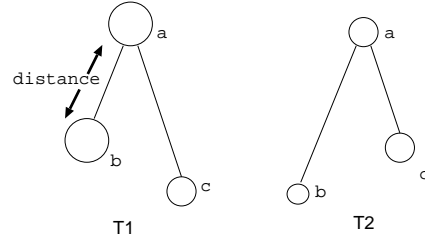


Figure 5: Tree-structure of real traffic

If we had forced to adjust basic model to this tree structure, there had had no relations between T1[b] and T2[b], T1[c] and T2[c]. However, it is naturally expected that T1[b] contains T2[b] and T1[c] is a element of T2[c]. Thus, we have to consider relations between T1[b] and T2[b], T1[c] and T2[c].

Figure 6 shows a virtual breakdown idea to compare different aggregated nodes in depth.

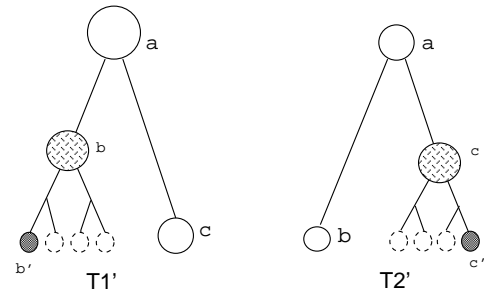


Figure 6: Virtual breakdown of aggregated node

Based on the virtual breakdown method, it happen to recursive breakdown, using this method both T1 to T2 and T2 to T1 at the same time.

Therefore, we go through 2 phases. In first phase, breaking down T2 based on T1 tree structure.[D1] Second phase is the other.[D2]

“Deviation(D)” can be calculated by following calculs.

$$D = \frac{\Sigma(T1[i] - T2'[i])^2 + \Sigma(T1'[i] - T2[i])^2}{2}$$

With the use of virtual breakdown aggregated node algorithm, we can calculate Deviation(D) from different aggregated nodes in tree structure.

5 Evaluation

5.1 Infomation of sample data

We have done a sample evaluation using 1 month long traffic data from the WIDE Internet[5] backbone. This traffic data is taken from a trans pacific link, which contains flooding attacks.

Figure 7 shows monthly traffic pattern. This monthly sample data contains some flooding attacks. Especialy, in this evaluation we had focused at 14th October.

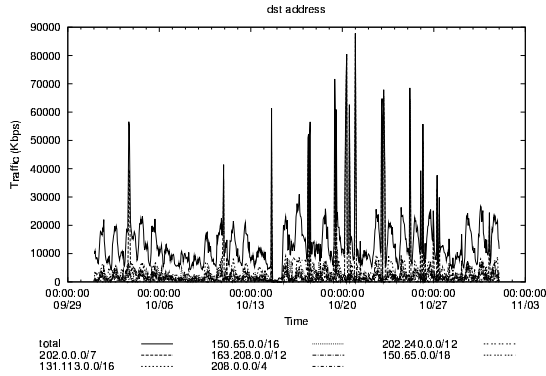


Figure 7: Characteristic of sample traffic in a month

Figure 8 shows 24 hours traffic pattern at 14th October. This figure shows that there was a flooding attack against router1 and router2.

Figure 9 shows 2 hours traffic pattern at 13:00 -15:00 14th October. In Figure 9, “router1” and “router2” are expression of each routers interface. These figures show detailed information which is cased by flooding attacks. We can see that flooding attacks starts at 13:16 and ends at 14:36 toward router1 and router2.

5.2 Time granulation

We have calculated the “Deviation” of the current 2 minutes with 4 types of parameters.

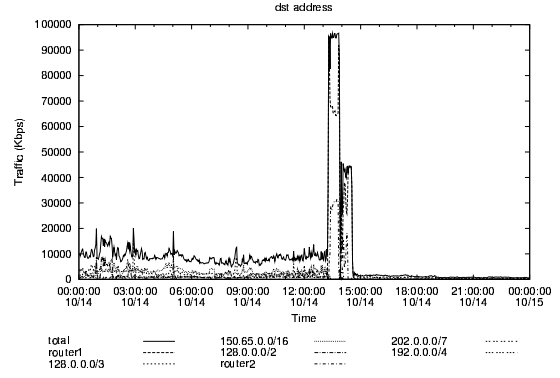


Figure 8: Characteristic of sample traffic at 14th Oct.

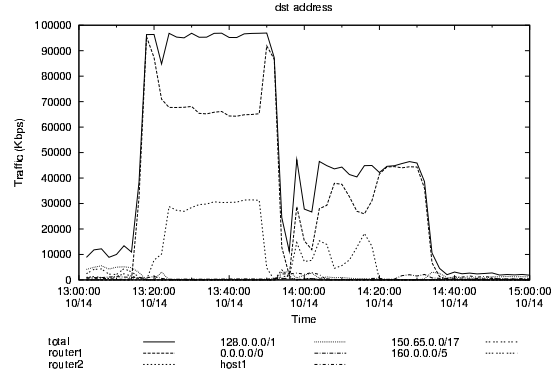


Figure 9: Characteristic of sample traffic at 13:00-15:00 in 14th Oct.

- 1) “Deviation” compared with current 30 days
- 2) “Deviation” compared with current 24 hours
- 3) “Deviation” compared with current 1 hours
- 4) “Deviation” compared with current 5 minutes

Evaluating flooding attack detection with “Deviation”, we had prepared 4 types of parameters. In following 4 figures, Deviations are calculated based on between current 2 minutes traffic date and each length of term.

- 1) Time series plot of Deviation between current 2 minutes and current 30 days (Figure 10)

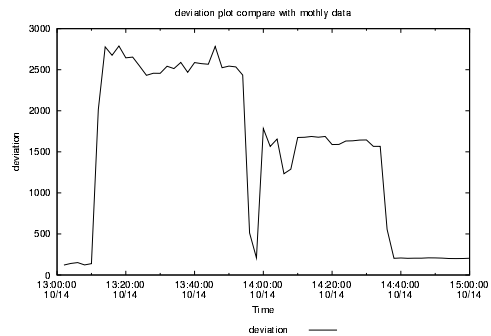


Figure 10: Compared with 30 days data

Figure10 shows that “Deviation” is strongly related to traffic patterns that consist of packets toward router1 and router2.

- 2) Time series plot of Deviation between current 2 minutes and current 24 hours (Figure 11)

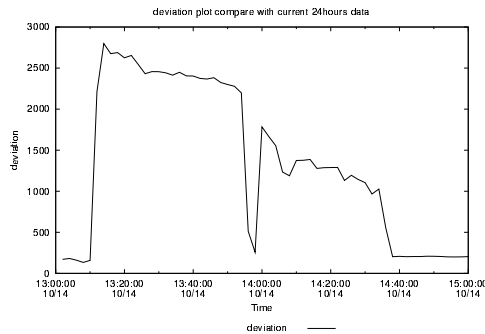


Figure 11: Compared with 24hours data

Figure11 shows that “Deviation” is related to traffic patterns that consist of packets toward router1 and router2. However, the relations are gradually going down by continuing flooding attacks.

- 3) Time series plot of Deviation between current 2 minutes and current 1 hour (Figure 12)

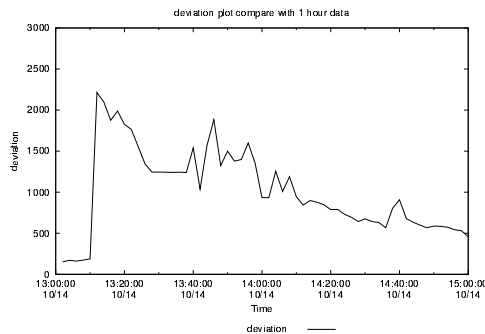


Figure 12: Compared with 1 hour data

Figure12 shows that long term flooding attacks cause “Deviation” to be going down, because flooding attacks traffic would be contained 1 hour traffic.

- 4) Time series plot of Deviation between current 2 minutes and current 5 minutes (Figure 13)

Figure13 shows that long term flooding attacks cause “Deviation” to be going down and that the levels of “Deviation” are very low in long term flooding attacks. Moreover, when flooding attacks end, “Deviation” increases rapidly, because traffic data for current 5 minutes was full of flooding-attack packets.

In figure 10, 11, 12 and 13, when characteristic of current 2 minutes data appears similar to characteristic of long term data, the levels of Deviation is low. At the same time, while characteristic of current 2

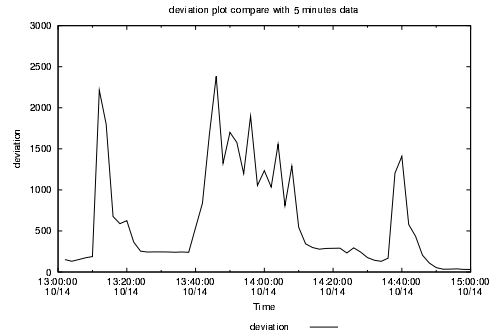


Figure 13: Compared with 5minutes data

minutes data does not have similar to characteristic of long term data, the levels of Deviation is high.

As above, the method of “Deviation” can obviously detect start point of flooding attacks in any time scale. However, in case of flooding attacks which continue for a long time, levels of Deviation depend on length of time to which compares traffic data.

6 Conclusion

The automatic detecting method of the flooding attacks without fixed rules is proposed. The basic concept is how to detect unusual traffic patterns with deviation between usual patterns and recent. The traffic monitoring tool “AGURI” that has been developed by our project, is very useful to realize the characteristics of the traffic pattern, because AGURI can aggregate the traffic data without discarding the traffic characteristics. Algorithm of virtual breakdown aggregated node in the tree is very powerful to calculate deviation of the traffics.

In this paper, we provided the simple evaluation of our method, using real traffic data which taken from WIDE Internet backbone. The flooding attacks can be detected with big deviation value. More detailed evaluation such as probability on wrong detection should be done in future research.

References

- [1] R.Needham, “Denial of Service: An Example”, Communications of the ACM volume 37, November 1994
- [2] Oethker, Tobaias, “MRTG - The Multi Router Traffic Grapher”, USENIX System Administration Conference, December 1998
- [3] Kenjiro Cho, Ryo Kaizaki, Akira Kato, “AGURI: An Aggregation-Based Traffic Profiler”, QoS2001, September 2001
- [4] J.Case, M.Fedor, M.Schoffstall, K.Davin, “A Simple Network Management Protocol(SNMP)”, RFC1157, May 1990
- [5] “WIDE Project,”, <http://www.wide.ad.jp>