

## Detecting Worms and Abnormal Activities with NetFlow, Part 2

Yiming Gong 2004-09-23

### 1. NetFlow review

In the [first part](#) of this article series, we looked at what NetFlow is and how it can be used in the early detection of worms, spammers, and other abnormal network activity for large enterprise networks and Internet service providers. The article discussed some of the most common methods of flow-based analysis: Top N, Baseline and Pattern Matching techniques.

In this second and final part of the article, we'll look at three additional methods of analyzing the flow, including how to filter our flow results via TCP flags, in order to get a more granular view of network abnormalities. We'll discuss some ICMP issues, and then look at some of the various tools that exist to help implement and analyze our NetFlow solution. Let's get started.

### 2. TCP flags for NetFlow

One difficult task when performing flow-based analysis is that the administrator must evaluate a very large number of flow records. If he is just relying on the Top N, baseline and pattern matching methods, the administrator will merely get a coarse view of network abnormalities. We've seen many times there are moderately intensive worms and other abnormal activities which appear intangible amongst the immense amount of legitimate traffic that is typically found in a large enterprise network. Those malicious hosts will not show up in the Top N lists, nor will we know in advance what key fields and values to 'grep' -- yet these are still malicious hosts that must be addressed.

In order to identify the abnormalities more effectively and accurately, a better way to narrow the analyzable flow records is required. Fortunately, for most types of TCP-based worms and other abnormalities, there is another useful field in flow records: analysis based on the TCP flags.

Worms, by their replicating nature, are programmed to seek as many victims as possible. Typically they send out hundreds or even thousands of probes to large blocks of IP addresses in a very short period of time. If a worm was designed to spread via TCP (as most of them are), during its propagation there will be a lot of corresponding TCP SYN packets sent out as it seeks vulnerable services in other hosts.

#### 2.1 A typical worm's SYN scan process

In the patch a worm takes as it makes its way outside the corporate network, there are three possible results to its SYN scan:

1. The first possibility is that the destination host is alive, and the corresponding vulnerable service that was targeted is running.

As we all know, the three-way handshake that starts a normal TCP connection involves:

- o First a client will send a SYN packet to the destination host.
- o The destination host sends back a SYN/ACK packet.
- o The client acknowledges the destination host's acknowledgment.
- o A connection is established.

Figure 1, below, illustrates this handshake.



**Figure 1: destination host is living and the TCP port is open**

When a SYN packet arrives at the destination port of a host, if the port is open the SYN request sent from the worm will be responded to -- regardless of whether or not the service running on that port is vulnerable. Therefore, the standard TCP three-way handshake will be completed and subsequent packets carrying other TCP flags such as PUSH and ACK will be followed. In the [first part](#) of this article we mentioned that a V5 NetFlow record contains the cumulative OR of TCP flags in the 'entire' connection session. Therefore, using our the NetFlow approach we should expect to see a combination of TCP flags such as ACK/PUSH/SYN/FIN or ACK/SYN/FIN in flow record, flowing in both directions.

2. The second possible result from the worm's SYN scan is that the destination host it attempts to connect to is not living, as shown below in Figure 2.



**Figure 2: connect to a 'dead' destination host**

Because the destination host is not alive, the SYN requests sent out by the worm will receive no response. From our NetFlow perspective, we will get a flow record in which only the SYN bit was set from worm host and sent to destination host.

3. The third possible result is that the destination is alive but connection attempts from the worm are not functional, as shown in Figure 3.



**Figure 3: destination host with closed TCP port**

This simply means the destination host is indeed living but the port which the worm is trying to connect is closed. As we know, to establish a TCP connection a server must listen on that particular port. If a client connects to a server's non-listening port, the server will send back a RST/ACK packet. According to normal TCP implementation guidelines, the host will immediately stop any TCP connection attempts once it receives a RST. From the NetFlow angle, the TCP flag combination in the flow records will show only SYN requests from the worm host to the destination host.

Thus far I have discussed three possibilities when a worm's host SYN scans the network. There is one important fact about the worm to remember: when it tries to propagate, the destination addresses are typically generated at random, and normally there will be a large number of destination hosts that are not living or functional. Therefore in the outgoing traffic we should expect to see a large number of SYN bits set in the flow records associated with the worm-infected host. This characteristic is very useful and may be used as a key point for our flow-based detection of worms and other TCP based abnormalities. In section 2.2, below, we will discuss this detection process.

## 2.2 Three steps to process a flow file for TCP flags

When doing flow-based analysis, a captured flow file can be processed by the following steps:

1. The first step is to search through the flow file, filter out all flow records that have only the SYN bit set, extract the source IP addresses of every flow record, count the occurrence of every unique IP, and then finally sort the records by the number of counts for each one. Following this process, we will end up with a suitable list of potentials. The administrator can set a threshold depending on the network size and traffic volume, whereby hosts whose counters are above the threshold should be considered as potentially malicious, and those under the threshold should be considered as benign.
2. The second step is to search through the flow file again to extract all flow records where the source IP addresses are the ones found in the "potential malicious" list as generated in step 1 above. By taking a second look at the flow file in this way, we will get a detailed connection table for every potential host. The results of this search will be used for our third and final step in this process, and will help us to further identify the behavior of our suspicious hosts.
3. The third step will give us some very meaningful data about the worm-infected hosts on our network. First read the output taken from step 2, and then for each host count the number of appearances of every unique destination port. Sort by the number of occurrences, and we will then

get an IP address and its corresponding active ports table. The following is an example output generated by a little shell script that performs this task, as written by the author.

```
potential host1: 61.236.123.225
-----
 84 times probe on dstport 1025
 76 times probe on dstport 80
 72 times probe on dstport 2745
 64 times probe on dstport 3127
 48 times probe on dstport 6129
```

For a malicious host that is always trying to connect to one or several special ports, we can discover it in our reports by checking the corresponding services registered for the most active host's destination port in the matrix. In other words, using the above example, host 61.236.123.225 was infected with W32.gaobot.sa worm because it is scanning for the MyDoom backdoor on port 3127, the Bagle backdoor on port 2745, and the Dameware port on 6129. These are clearly the characteristics of worm W32.gaobot.sa.

In our goal is to evaluate the propagation of a worm from outside our network into the inside, flow records coming from outside that have the TCP flags RST/ACK set should be looked at. We know that a closed port will send back a RST/ACK to a TCP request, as was shown in Figure 3. If a worm is scanning a large block of living hosts for a certain vulnerability, those hosts with closed ports would send back a RST/ACK. For ingress traffic, if a destination (not source!) host in the flow records receives too many RST/ACK responses, the administrator should check out this destination IP seriously, as it is very likely infected with a worm.

### 3. ICMP issues

One of the purposes of ICMP is to provide feedback about problems in the communication environment of a network. Sometimes a ICMP type/code in the flow records could also be used to help us locate the potential malicious hosts.

The first thing to note is that there is no flow field that is directly named as an ICMP type or ICMP code, as was inferred in part 1 of this article series. Some people have suggested, then, that the type and code of ICMP requests cannot be identified in the captured flow data because that information is not specifically recorded. Does that mean we cannot use ICMP type/code for flow-based analysis? No. In actual fact, we can: ICMP type and code are indeed recorded in the NetFlow data, they are simply stored in the destination port field in the flow record.

For flow-tools, when one needs to obtain the ICMP type and code number, we just need to check the

destination port field (dstPort). If the number appears in hex, we should convert it to decimal, and vice versa.

The following is example output of flow-tools in which the dstPort is in decimal.

srcIP	dstIP	prot	srcPort	dstPort	octets	packets
135.169.9.116	137.54.111.144	1	0	2048	28	1
135.169.9.116	137.62.249.241	1	0	2048	28	1
135.230.255.66	136.129.9.27	1	0	769	112	2
135.32.252.50	136.129.9.27	1	0	769	56	1

We can see that protocol field (prot) is 1, which means ICMP. The destination port is 2048, which is 800 in hex. Here 8 means ICMP type 8, and 00 is the code field for ICMP type 8, which means no code. So we can conclude that 800 is an ICMP echo request. In the same way, 769 is 301 in hex, which is ICMP type 3 and code 01, which means ICMP host unreachable.

There are two interesting types of ICMP packets that can be used for flow-based abnormal detection when analyzing a network's ingress traffic. It is also possible to use pattern matching methods to do ICMP flow analysis, as we will demonstrate.

### 3.1 ICMP destination unreachable

According to ICMP implement guidelines, if the destination network or the destination host is unreachable, the gateway MAY send destination unreachable messages to the source host, as shown below in Figure 4.



Figure 4: destination unreachable

### 3.2 ICMP port unreachable

For UDP requests, hosts with closed ports may send back ICMP port unreachable messages to the source host. If a worm spreads with UDP, it may then trigger many ICMP port unreachable flow records in the packets returned. This is shown below in Figure 5.



Figure 5: destination host with closed UDP port

If a host has an abnormal volume of ICMP port/host/network unreachable set of flow records, that may indicate that the host is acting abnormal.

### 3.3 Pattern matching methods

Another ICMP-based flow analysis method is pattern matching. Some worms and network attacks are carried out using ICMP, as we saw with the W32.Nachi.worm. When a host is infected with the worm, it will send out ICMP echo requests to the outside with a fixed length of 92 bytes. So we simply need to filter out the flow records with ICMP type 8 that have a 92-byte packet length, and hosts infected with this worm will be caught.

## 4. Special zones in the Enterprise

In enterprise networks there are always some servers that are protected with firewalls. Normally these servers should be hardened and have only fixed ports open to outside. Except for those fixed ports, any connection established between the server and outside should be prohibited.

We could use this characteristic to monitor the security of the servers using NetFlow.

### 4.1 ingress traffic

If we find any flow record whereby the destination IP contains a server IP, but the destination port is not in the server's functional port list and additionally the TCP flags in the flow record contains ACK (but not RST/ACK), an alert should be triggered.

The above suggestions perhaps indicates two points. First, it tells us that the firewall in front of the host has something wrong with it, as it has let a connection (which should be prohibited) get established. An exception to this would be that the connection launched by outside incorrectly contains only a ACK packet; regardless, this kind of connection should not have appeared. Secondly, the appearance of this flow record also indicates the server may have an abnormal port open to outside!

### 4.2 egress traffic

When we see any flow record whereby the source IP contains a server IP but the source port is not on the server's list of functioning ports, and additionally the TCP flags in the flow record are not RST/ACK, an alert

should be triggered.

As well, if we spot any data being transferred at the same time as the above, a red alert should be immediately raised! It is quite possible that the server has been broken into. Perhaps a backdoor has been activated, and maybe a new service has been enabled.

## 5. Implementation guidelines

Thus far we have discussed several methods that can be used for flow-based detection of worms and other network abnormal activities, however no detailed implement instructions have been provided. In reality, if you follow the main points of these methods as discussed in this article series, implement will actually be fairly easy.

For instructions on how to enable NetFlow on a specific router, readers can check the corresponding manufacture's website. Some example NetFlow configurations for popular Cisco and Juniper routers can be found at: <http://www.splintered.net/sw/flow-tools/docs/flow-tools-examples.html>

Although there are both commercial and open source solutions for flow file analysis, the author himself prefers the open source solution. Commercial products normally have built-in and fixed functions which are always difficult to extend. Most importantly, commercial flow analysis tools don't have the flexibility as existing open source options.

For open source solutions we have many choices, such as cflowd, SiLK, and flow-tools. All of these work quite well and on many different UNIX platforms.

### **cflowd**

cflowd is the classic traffic flow analysis tool. It can be found at <http://www.caida.org/tools/measurement/cflowd>, however note that it is no longer supported by CAIDA anymore, so consider one of the other tools below.

### **SiLK**

SiLK is a collection of NetFlow tools developed by CERT/AC to facilitate security analysis in large networks. It consists of a packing system and an analysis suite. SiLK provides administrators with great flexibility in its ability to process the flow data, but in my option, SiLK still needs some revisions and enhancements to make it run more smoothly. SiLK can be found at <http://silktools.sourceforge.net>.

### **Flow-tools**

Flow-tools is a powerful and helpful program for NetFlow related work. There are some available add-ons, and overall it provides greater flexibility and controls than many of the other tools. Flow-tools can be found at <http://www.splintered.net/sw/flow-tools/>.

In addition to these, there are some other programs such as FlowScan and CUFlow which can be used for flow-based analysis work. All of these can be considered to be valuable tools.

## 6. Summary

This article series has discussed the flow-based detection of worms and abnormal activities. [Part one](#) talked about the basic concept of NetFlow, and then the first two of the five flow-based analysis methods were put forward. The second part of the article discussed the final three analysis methods. In summary, these five methods of analysis are Top N and Baseline, Pattern Matching, TCP flags, ICMP issues and special zone for large enterprises. With these methods, network administrators can detect network-wide abnormalities much more effectively.

There is no silver bullet for security detection on large network infrastructure, but with NetFlow we may attain further insight into the traffic crossing our entire network -- and make it run better.

### About the author

[Yiming Gong](#) has worked for China Telecom for more than 5 years as a senior system administrator, and now he works as a Technical Manager in China Telecom System Integration Co.Ltd. He also has a [personal homepage](#) focusing on network/system security.

Comments on this article can be sent to the [editor](#).

[Privacy Statement](#)

Copyright 2006, SecurityFocus