

An IDS Using NetFlow Data

Daniel Medina

I. INTRODUCTION

IN an effort to monitor network traffic, Columbia University uses various tools. Internal core switches (which we call edges) and external uplinks are polled via SNMP to detect fluctuations in bytes and packets sent and received. NetFlow data is exported from the Cisco (and soon Juniper) routers that serve as uplinks to our commodity Internet provider and Internet2.

This NetFlow data is exported from the major uplinks to a host running flow-gathering software (flow-capture from the flow-tools bundle [1]). It is then scanned using FlowScan, a package created by Dave Plonka [2]. Reports are generated using CUFlow, a FlowScan module written at Columbia by Johan Andersen and maintained by Matthew Selsky [3]. CUFlow generates Top-10 reports of usage by various hosts by bytes, packets, and flows in and out, as well as providing nice graphs of services usage.

A NetFlow record contains information about connections made through the routers. Flows are collected from the routers approximately every five minutes. The aggregated flow files regularly include over 600,000 flow records. The information includes

- Source and Destination IP addresses
- Source and Destination ports
- Type of Service
- Packet any Byte counts
- Protocol type
- Flow start and end timestamps

Cisco details more about NetFlow and tools they have released to use it on their website [4]. Reports generated by CUFlow enable us to do many things, such as detect traffic anomalies (DoS and DDoS attacks), find bandwidth abusers, and determine service traffic patterns. Robert Galloway has an excellent guide to setting up a Cisco router to export NetFlow data, collect with flow-tools, process with FlowScan, and analyze with CUFlow [5].

Using NetFlow and FlowScan we can also, as we will show, detect intrusions and other policy violations launched to and from our network.

D. Medina is a post-undergraduate student in the Columbia University School of Engineering and Applied Science and a Network Analyst in the Network Systems Group in AcIS. He may be contacted via email at medina@columbia.edu

II. WHY NETFLOW?

NetFlow records provide an excellent means analyzing network traffic. Sommer and Feldman point out that NetFlow lies between the coarse SNMP data that may be retrieved and the “complete package” of packet capture [6].

Packet capture is the method used by the network IDS Snort [7], but there are several reasons this was not chosen. Packet capture is a more intrusive though, like NetFlow, passive detection method. A host harvesting packets could potentially contain large amounts of sensitive data that would have to be handled accordingly. There is already a strong push against intrusive network monitoring on university campuses, not the least of which is caused by the fear of an Recording Industry Association of America subpoena [8]. The Electronic Privacy Information Center has criticized many traffic monitoring projects, pointing out the very real concerns of the security of collected information [9].

NetFlow is adequately blind to content, but is still able to reveal what we need to know. It is also lower in volume, and lends itself very well to analysis by intelligent agents. Frank’s chosen features for an AI IDS coincide very nicely with the exported NetFlow records [10]. Plonka and Barford hint at interesting statistical analysis that may also yield fruit [11].

III. SUCCESSES AND FAILURES

There was initial success in identifying simple network policy violations. Several tools were created. All owe their creation to Plonka’s excellent interface to FlowScan. One of the scripts, “Snoopy” displays traffic sent to and from a particular host. This reveals very effectively what is going on. Hosts scanning for ports are immediately identified by their sequence of IPs and regularity of connections. Worm propagation is also easily identified. Due to the (relatively) small size of a stored NetFlow file (about 10 MB), a cache of several days may be saved to analyze past incidents.

Expanding this further, a small script was written to detect hosts scanning for open NetBIOS ports. So many complaints have been received that we were curious what NetBIOS was being used for exactly. Analysis showed that almost no legitimate connections were being made between the outside world and the campus network on the NetBIOS (and associated) ports. Attackers scanned between 16 and 1,000 hosts on our network apiece, every

five minutes. While the high-volume attacks might have been detected, the “slow scans” were insidious. Even more disturbing, compromised hosts on campus were scanning outside hosts at rate of up to 4,000 per five minute interval (that is, per compromised host). These findings have led to the decision to filter NetBIOS ports (TCP and UDP ports 137,138,139 and 445) in the near future.

But how to generalize this success in finding any case of rampant scanning? The solution would be to build an IDS to detect incidents that “look like” scanning. An intelligent agent (as suggested by Frank and others) is the obvious road, since it would maintain the most flexibility and adaptability. As stated above, NetFlow has all the features we would need to create such an agent.

What we lack are labelled samples. We can positively identify some scanning attempts, but many other beasts may go unnoticed. An expert system (or an enterprising young network analyst) is needed to compile data from various researched incidents to use as sample data. Unfortunately, work began at a time of considerable changes. The NetFlow-harvesting host was moved from a 300 MHz Sun host to a dual-processor gigahertz PC running Linux. The speed-up improved the analysis times (CUFlow reports in approximately two minutes per flow file, down from the precarious 4-6 minutes). No longer are we losing valuable information from DoS attacks because the analysis engine lags behind. New routers were deployed which exported data in a new NetFlow format, and data was lost in the transition. Lack of follow-up investigation with the end-users also meant we could not be sure what the exact source of the incidents were.

Eventually, a compromise had to be reached. A full-blown IDS could not be developed at the current time. However, an adaptive system could be designed. Using signatures and thresholds pre-defined in a configuration file, a new FlowScan module called “Nasties” was written. This script is run as a reporting class to FlowScan, flagging hosts that stray from the defined thresholds. If the host traffic matches a signature, it is reported. If the host is not matched, a new signature is added.

This method failed at first, as high-volume traffic on port 80 (www) caused many hosts to be flagged. An anomaly was defined as a host spending all of its time on one particular service. This too failed for normal web use. Thresholds were then introduced not to cover only overall traffic use, but also use of misbehaving hosts. Exceptions were granted for certain known hosts using known services (such as 53/tcp, domain, on our main DNS server).

This narrower set of parameters worked well. NetBIOS scanners (for whom we have no mercy) are flagged nearly always (“slow scanners”, on the order of a handful per

hour may slip by). ICMP-flooders are identified. The module was able to detect scanning on ports that were not pre-defined in signatures (such as microsoft-ds). Another interesting result was the flagging of many hosts using gnutella-based peer-to-peer clients. These hosts connected to over 400 hosts (not including those they contacted inside the campus network) per five minute interval in their queries. A new peer-to-peer client was identified (“Blubster”), and what appears to be a new, previously unseen scan target port was found.

IV. FUTURE WORK

We must investigate end hosts more carefully to determine if the behavior which we are for the most part surmising is actually going on. We should confirm that a host was using peer-to-peer software, rather than assuming that to be the case and overlooking a virus vector (though both cases may be true!). This is difficult in our particular environment, given that the Network Systems group and the Security group report to different heads (the former being under Systems, the latter under User Support Services).

Efforts must be made to streamline analysis of the generated logs. This is easily done by reusing other scripts that parse log files for IPs and mail the responsible parties (via WHOIS lookups) of the host violation with log samples.

Tuning of the signatures and thresholds should be more automated than it stands currently. As of now, thresholds are manually updated and signatures accumulate without being pruned. Destination ports are examined, but source ports are ignored. This may overlook certain attacks that originate from single portss. We may also be able to expand the focus from finding violating sources to include responding “victims”.

With sufficiently analyzed NetFlow samples, we would still hope to create a true intelligent system, using at first simple modular algorithms (such as k-Nearest-Neighbor). Work will continue on the project, regardless. The filtering of NetBIOS ports will deprive it of its favorite target.

V. SUMMARY

A true IDS using AI methods was not constructed, though it is clear that such a direction would be practical and possible. NetFlow records provide a suitably reduced, feature-full dataset. David Plonka’s FlowScan provides an easy-to-use interface to create modules for. Work will continue towards creating an effective, modular IDS for use on exported FlowScan data.

ACKNOWLEDGMENTS

Thanks to David Plonka at the University of Wisconsin - Madison for his excellent FlowScan package.

Thanks to Johan Andersen and Matt Selsky at Columbia University for their CUFlow, which provided a superb sample of what a good FlowScan module looks like.

Thanks to Paul Dokas of the University of Minnesota for his “find_scanners” script.

REFERENCES

- [1] Flow-tools, the flow-gathering software, <http://www.splintered.net/sw/flow-tools/>
- [2] Dave Plonka's FlowScan, <http://net.doit.wisc.edu/~plonka/FlowScan/>
- [3] Johan Andersen and Matt Selsky's CUFlow module for FlowScan, <http://www.columbia.edu/acis/networks/advanced/CUFlow/>
- [4] Cisco's NetFlow website, related tools and whitepapers, <http://www.cisco.com/go/netflow>
- [5] Robert Galloway, *How to build detailed Network Usage Reports using Apache, RRDTool, flow-tools, FlowScan, and CUFlow* <http://www.linuxgeek.org/netflow-howto.php>
- [6] Robin Sommer and Anja Feldman, Saarland University, Germany *Netflow: Information loss or win?*
- [7] Snort, the Open Source Network Intrusion Detection System, <http://www.snort.org>
- [8] Recording Industry Association of America, <http://www.riaa.org>, <http://www.riaa.org/pdf/Universityletter.pdf>
- [9] Electronic Privacy Information Center, <http://www.epic.org>, <http://www.epic.org/privacy/student/p2pletter.html>
- [10] Jeremy Frank, *Artificial Intelligence and Intrusion Detection: Current and Future Directions*
- [11] Paul Barford and David Plonka, *Characteristics of Network Traffic Flow Anomalies*