netscout - adaptive network monitoring and countering mechanism

Aditya Joshi
Computer Science and Engineering
Indian Institute of Technology Bombay, India
aditya.joshi.2000@gmail.com

G Sivakumar
Computer Science and Engineering
Indian Institute of Technology Bombay, India
siva@cse.iitb.ac.in

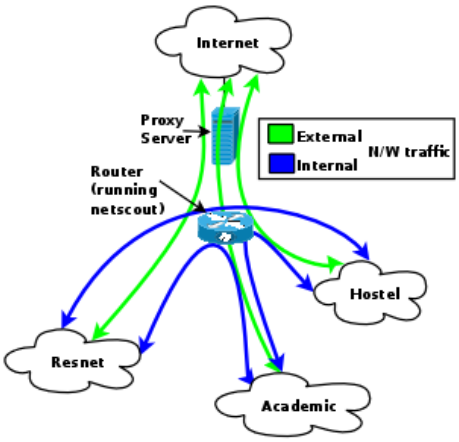

Figure 1: A typical university network topology

Abstract

Networks have policies in place to ensure efficient and fair resource utilization and protection of network installations. An effective monitoring and countering mechanism needs a complete picture of resource protection and utilization ,by monitoring network traffic and diverse activity logs, in order to act on time.

netscout is a python based network filtering, monitoring and countering mechanism driven by policies. Traffic is observed in real time, largely in a passive mode, and past and current user/host activity information is used to make decisions about future traffic monitoring. Such a 'predictive' model allows efficient information gathering and subsequent appropriate responses.

1 Introduction

Consider a typical university network, Figure 1, with interconnected residential (resnet), academic and hostel networks. Connection to the Internet is via one or many router(s) and proxy server(s). The router(s), running netscout[1], act as vantage points to monitor and control network traffic. LDAP logins (Proxy authentication) (if used) can be determined via netscout and be used in addition to HOST IP based information gathering to build user/host network activity profile. In order to maximize the efficiency of netscout to present relevant information, the application should run on one or many of the routers(s) to monitor internal and outgoing traffic, via the proxy server. netscout can be employed in a topology wherein traffic from the main router(s) is also directed to another machine (akin to a t-junction) running netscout. netscout, in position now to monitor network traffic can filter, counter 'hostile' traffic and generate user desired reports for further analysis and action.

2 the netscout advantage

The strength of netscout lies is the ability to employ some 'intelligence' in packet gathering. As LANs tend to generate huge amounts of traffic the network data sample should be small but useful. An approach where a sample of network data is gathered, sans any filters, the probability of finding useful information is proportional to the data size. A large data size though tends to slow down further processing. For example, consider (a) 1000 packets captured and P2P (peer-to-peer) packets looked for in them or (b) looking only for a threshold level of P2P packets. As evident, case (b) is more efficient as the data set captured contains relevant information and also the threshold levels decided by the user allows the user to control the number of P2P hosts he intends to discover.

netscout also tries to obtain a logical picture of the network activity. For example, once netscout has ascertained that a host is involved in P2P activity it then tries to (if activated) obtain the proxy user id being used from that host. This is done by looking at a couple of http packets. The proxy user component now allows netscout to build network profile of a user.

Hosts added to netscout's database via static keyword matching techniques (say P2P), slowly form the database for the future. For instance, hosts identified as involved in P2P activities are first checked for their http traffic. Websites accessed by hosts are stored and

classified. The classification involves looking at keywords in the website's meta section. If website is found to be violating policy then, a further detailed analysis of the host network activity is done in order to ascertain if the host accesses too many policy violating websites. Furthermore, hosts accessing websites classified as policy violating are monitored for there network activity. In addition to keyword matching techniques few statistical techniques too are employed for traffic monitoring. Parameters such as bandwidth usage and high connections help identify potential network bottlenecks and possible policy violating traffic. In essence, netscout tries to maximize network information in the smallest gathering and processing time.

3 netscout internals

netscout is a python based network filtering, monitoring and countering mechanism. The packet gathering is largely done via existing tools such as ngrep[5],tcpdump[4] and tshark[6]. The gathered data is then analysed and acted upon based on the network policies laid out in the netscout configuration file, which drive the application. Currently, netscout can monitor and detect: (1) unauthorized P2P (DC++), (2) policy violating HTTP traffic, (3) MAC address spoofing, (4) heavy bandwidth usage and (5) 'hostile' traffic against important network resources. The information is persisted in a SQLite database for basing future activity monitoring and reporting user/host network activity. netscout has five modes of operation: **passive**, **protect**, **counter**, **update** and **report** modes. These modes can be activated/deactivated via the following "config" file parameters

```
## Set to true to check for DC++ p2p traffic
dcppMode=false
## Set to true to check for http traffic
httpMode=true
## Set to true to do statistical analysis
statMode=false
## Set to true to protect important resources
(proxy,mail)
protectMode=false
## Set to true to counter offending hosts
counterMode=false
```

4 Passive Mode

The primary data gathering mode of netscout. Here traffic is listened on to specified interface and data obtained based on the configuration file. Based on the configuration the kind of traffic requested (eg http,p2p) to be monitored is heard for. Also, statistical parameters like maximum number of connections to, bandwidth usage are computed based on the thresholds decided in the configuration. Sample session on netscout invoked in passive mode on the eth0 interface.

```
# netscout -i eth0
Screen logging on
nescout version 1.0
Ensure you are root!
Reading Configuration Parameters
Detailed Screen logging on
Using existing db file
netscout passive mode
Checking for tcpdump
tcpdump check ok
Checking for tshark
tshark check ok
Checking for ngrep
ngrep check ok
Checking if root permissions available
ROOT permissions available
Creating data directories
getting existing host info
Found existing 0 hosts
getting existing user info
Found existing 0 users
```

P2P (DC++)

Peer-to-peer shares[3] tend to generate lot of network traffic and create potential network bottlenecks. In addition to the large network volumes generated, it is the potential copyright infringement that P2P hubs can be used for that causes a need to monitor them. Also, not all P2P shares generate lot of traffic or indulge in copyright infringement, those can be identified by analysing the bandwidth usages , files hosted.

netscout can identify hubs and clients operating in a network without requiring prior knowledge of such hosts and the need to connect to central peer hubs to obtain information. Working in a passive mode and armed with keyword matching and protocol information, netscout can identify peers and also the files shared and traded by peers. Based on the results netscout can form a database for future runs and attempt to identify P2P hubs with more efficiency. The number of HUBS to be looked for can be controlled by setting the following appropriate configuration parameters:

```
## Number of MyINFO packets to look for
(ie number of HUBS to locate)
myinfo=18
## Number of Search Results packets to look
for (to get file list)
sr=26
## Number of HUB-Client Packets to look
for
hubClient=15
```

Sample netscout P2P session:

```
Initialing DC++ P2P scouting
Looking for DC++ HUBS
Listening for MyINFO packets
Obtaining DC++ HUBS IP and PORT
Checking if User ID is already known
Obtaining User id for 10.2.1.1
Scanning HTTP requests for 10.2.1.1
adding host 10.2.1.1 and id pqrs123
Host not been seen in this session
User not been seen in this session
Getting File List for HUB 10.2.1.1
Searching for requested file extensions
Scanning for P2P clients
Scanning for P2P clients for HUB 10.2.1.1
clientList ['10.168.1.2']
Adding client information
Checking if User ID is already known
Obtaining User id for 10.168.1.2
Scanning HTTP requests for 10.168.1.2
adding host 10.168.1.2 and id abcd123
Host not been seen in this session
User not been seen in this session
Adding hub information
```

The file types to be searched for can be set in the ext configuration file parameter:

```
ext=.mpg,.mpeg,.avi,.mov,.rm,.wmv,
.mp3,.wma,.jpeg,.jpg,.png,.gif
```

HTTP

The type and kind of websites allowed access are driven by policies. Hence, monitoring the kind of websites being accessed by a user/host gives insight into the future potential activity of the user/host. Proxies generally hold black or white lists, determined by policies, to control Internet access. netscout, instead of such a static technique, uses keyword matching techniques to classify a website. The classification is done by looking for keywords only the **head** section containing the meta and title information, instead of scanning a complete page. It has been found that few lines of text of the head section are sufficient in most cases to classify the website. The categorization is based on the category-keyword pairs as decided by the policy. A user/host accessing websites that violate policy is assumed to do so in the future too and subsequent runs for netscout keep that in mind when filtering traffic. This automated classification also allows now to identify unlisted websites and keeping the website access lists updated.

MAC spoofing

MAC spoofing often forms the backbone of malicious attacks and detecting it can go a long way in thwarting potential future attacks. netscout, in each run, captures the MAC address of the active hosts. The MAC thus obtained is compared when the host is active again in subsequent runs of netscout. Deviations from past history are reported.

OS Fingerprinting

netscout uses the lightweight passive OS fingerprinting expertise of **p0f** [7]. Once potential policy violating hosts have been identified, netscout tries to garner more information about them. p0f allows an efficient way to learn the OS make of the host allowing better response to threats.

Bandwidth usage

Bandwidth usage monitoring goes in a long way to ensure fair and efficient usage of network resources. netscout explicitly looks at P2P traffic utilizing protocol information, but can also detect traffic bottlenecks. In this case, bandwidth usage of hosts is computed and those high on the list are probed more to obtain information about the type of traffic flowing through them. Another indicator is to look for number of destination connections terminating at a host. Hosts exhibiting high values of incoming connections are also probed for their traffic types and patterns. This way services like LAN multiplayer games, streaming servers may be identified without specifically looking for them.

```
## number of packets to look for connections
terminating on a host in order to raise alarm
connTo = 50
## minimum number of connections termi-
nating on a host in order to raise alarm
connToMin = 20
## ignore activity to and from this host. eg,
a proxy server
ignoreHost = proxy.local.ac
```

5 Protect Mode

A major goal of information filtering is to use it to protect important resources. Mail, Proxy, academic

servers, to name a few, need to be protected from internal threats. netscout monitors the activity to specified important resources and alerts system administrators of deviations from the norm. In addition activity from identified 'rouge' hosts is controlled (bandwidth throttling) via firewall rules (IPTABLES).

```
## Network resources that need to be protected
impRes = proxy.local.ac, smtp.local.ac
## number of packets to look for connections terminating on important hosts
prConnTo = 50
## minimum number of connections terminating on a host in order to raise alarm
prConnToMin = 20
```

6 Counter Mode

Once a threat or bandwidth hoarder is identified steps can be taken to limit the damage. netscout is able to either control bandwidth output from hosts or completely block them from network access for a pre-defined time duration. Also, if the proxy authentication from the host is determined then e-mails can be sent to warn users about their actions.

7 Update Mode

In the update mode netscout looks to improve on previous and add new web categorizations based on the category and keywords pertaining to it, as specified by the configuration file. The website categorization exercise is completely automated and can be carried out, if required, independently during low network traffic hours. netscout can either update unclassified websites or all websites (classified or unclassified) based on the category-keyword pairs as defined. The results of the website categorization may be reviewed, and if required, corrected by authorized personnel.

```
## Enter name value pairs of categories and keywords
Adult = sex,porn
Video = video,stream
SocialNetworking = friends,network,meet
News = news
```

8 Report Mode

After a netscout run has finished the information collected can be retrieved via the report mode. The report mode prints out a user/host wise display of network activity in a specified time period.

```
#netscout -r
netscout report mode
-------------------------------------------------

-------------------------------------------------
HOST:10.2.2.1
Activity: 0
User ID : pqrs123
Time : Sun Oct 26 09:28:37 2008
P2P Act : 0
P2P Type: SERVER
PORT : 411
FILES:
abcd.mp3
23.avi
7.mp3
1x1.gif
-------------------------------------------------

-------------------------------------------------
HOST:10.168.1.2
Activity: 0
User ID : abcd123
Time : Sun Oct 26 09:29:34 2008
P2P Act : 0
P2P Type: CLIENT
SERVER : 10.2.2.47
```

9 Configuration/Policy file

netscout behaviour is controlled by a configuration file where the kind of network activity to be monitored and the kind of countering mechanisms to be employed stated.

10 Future Work

Future work includes:

- Identifying 'tor' endpoints
- More extensive host profiling to identify services running on a system‘

References

[1] *http://netscout.googlecode.com/*, 2008.

[2] *http://www.teamfair.info/wiki/*, 2008.

[3] *https://launchpad.net/linuxdcpp*, 2008.

[4] *http://www.tcpdump.org*, 2008.

[5] *http://ngrep.sourceforge.net*, 2008.

[6] *http://www.wireshark.org*, 2008.

[7] *http://lcamtuf.coredump.cx/p0f.shtml*, 2008.