CS 353: Programming Assignment 3

Revision History:
3/23 Release

In this assignment you will develop a simple NIDS, a network intrusion detection system. Network Intrusion Detection Systems (NIDS) are placed at strategic points within the network to monitor bidirectional network traffic. Typically, it performs analysis of network traffic and identifies attack either through statistical anomaly detection or by signature matching. Once an attack is identified, the alert can be sent and the traffic can be blocked. This assignment leverages the skills from the first two assignments to develop a statistical anomaly detection NIDS.

The NIDS system has two parts, a monitoring part called the **watchdog** and central decision manager called the **desman**. We discuss these in detail below.

The PARTS

I. Develop a monitoring system, called `watchdog`, to monitor network traffic at multiple locations on the network. The watchdog supports both monitoring real time network traffic at the specified network interface and allows reading pcap files for debugging alert algorithms.

Command Line Options:

> watchdog [-r filename] [-i interface] [-w filename] [-c desmanIP]

where

```
-r, --read          Read the specified file
-i, --interface     Listen on the specified interface
-w, --write         Write the output in the specified log file
-c, --connect       Connect to the specified IP address for the desman
```

When run without any options, it should print the usage instructions and exit.

II. Develop a decision manager, called `desman`, to coalesce traffic reports and alarms from the watchdogs deployed at multiple locations on the network. The manager is always available on a TCP port 11353. It instructs the watchdogs with specific commands and combines the received reports and alarms to provide a holistic view of the network health.

Command Line Options:

> desman [-w filename] [-n number]

where

```
-w, --write         Write the output in the specified log file
-n, --number        The number of watchdogs in the NIDS
```

When run without any options, it should print the usage instructions and exit.

The NIDS Deployment

The desman is first deployed and then one or more watchdogs are deployed on the network. All communication between the watchdogs and the desman occur on TCP port 11353. The deployment phase is completed once all the watchdogs are connected to the desman. The UID can be a sequential numeric counter, that is, 1 for the first watchdog that connects, 2 for the second watchdog, and so on. Once all the watchdogs connect, the desman should command them to monitor using a start command. The output in the log files MUST be as follows on deployment:

Desman Logfile output:
Listening on port 11353…
Incoming watchdog connection from IP <IPaddress>
Assigned <UID> to watchdog at IP <IP>
….
All watchdogs connected…
Issuing start monitoring…

Watchdog Logfile output:
Connecting to desman at <IP>…
Received <UID>
Received start…

Message format:
The message format of the messages from the desman to the watchdog MUST be formatted as follows:

1. UID <numeric>
2. start

Note, that start command allows syncing up the monitoring start times for all the watchdogs on the network. Hence any delays in serially launching the watchdogs will be negated and they will all start to monitor at exactly the same time.

The NIDS Normal Operation

Every watchdog reports the current network traffic status to the desman. The reports are sent periodically, once every second. The report lists the number of packets, bytes, and flows seen during each time period. Each report also contains a serial number for the report, the first report should be numbered 1, followed by the next report numbered as 2, and so on. Once the desman receives reports from all the watchdogs, it adds the observed packet, byte, and flow counts, to get total count of the traffic on the full network for that period of time.
The log file entries for each watchdog and desman MUST be as follows:

Desman Logfile output:
Received report <UID1> <reportnumber> <packets> <bytes> <flows>
Received report <UID2> <reportnumber> <packets> <bytes> <flows>
Total traffic <packets> <bytes> <flows>

Watchdog Logfile output:
report <reportnumber> <packets> <bytes> <flows>

Message format:
The message format of the messages from the watchdog to the desman MUST be formatted as follows:

1. report <reportnumber> <packets> <bytes> <flows>

Note that since all the watchdogs start times will be synchronized by the desman, the time periods will line up to be for the same second within the trace file or network. All the watchdogs will typically be located on the same network, hence communication delays for the start command will be negligible.

The NIDS Alert Operation

Every watchdog also monitors for anomalous network traffic. In this assignment we will define anomalous network traffic as, either packets, bytes, or flow counts that increase by three or more times the amount of traffic observed in the previous time period. For example, lets assume that in time period N, the number of flows is 30. Then if in time period N+1, the number of observed flows is 90 or more, the watchdog should send an alert report to the desman at the end of the time period.

In addition to the information of a periodic report, the alert report also contains the term "alert" and the destination IP address of the host receiving the maximum number of packets, bytes, or flows during that period.

The log file entries for each watchdog and desman MUST be as follows. The "type" in the watchdog log file refers to the type of alert, that is packets, bytes or flows. Please list all "types" that are anomalous.

Desman Logfile output:
Received alert report <UID1> <reportnumber> <packets> <bytes> <flows> <IP>

Watchdog Logfile output:
alert <type>
alert report <reportnumber> <packets> <bytes> <flows> <IP>

Message format:
The message format of the messages from the watchdog to the desman MUST be formatted as follows:

1. alert report <reportnumber> <packets> <bytes> <flows> <IP>

Note that an alert is raised only once when the traffic goes from a low to high value. Also, this anomaly detector will not catch slowly ramping up types of attacks. Typically, there is an increase in more than one type during an attack. For example, an anomalous increase in flows, will typically also cause an anomalous increase in the number of packets. In that case, please list both packets and flows in the watchdog log file as "alert packets flows" . You can choose to list the IP with maximum packet counts, or the IP with the maximum flow counts in the alert report.

Code and Collaboration Policy

You are encouraged to refer to the socket programming tutorials. You can also discuss the assignment and coding strategies with your classmates. However, your solution must be coded and written by

yourself. Please refer to the plagiarism policy in the course syllabus. The submissions will be run through code similarity tests. Any flagged submissions will result in a failing score. Keeping your code private is your responsibility. You are strongly encouraged to pair and test your watchdog and desman implementations with your peers in class.

Submission Instructions

You can develop and test your code using your own machines or DETER. Create a compressed tar file which includes a README, Makefile, and the source code. To submit, create a folder called assign3 with the code, Makefile, and README. Tar the folder assign3. Submit the tar file on blackboard.
The README must contain: USCID, compiling instructions, additional notes on usage if needed.
To evaluate your project we will untar and type
% make

It is a project requirement that your code build without any warnings (when comipiled with -Wall flag). Structure the Makefile such that it creates the desman and watchdog executables (not a.out). For more information please read the make(1) man page.

We will then run your programs using a suite of test inputs. After running the program, we will grade your work based on the output. It is recommended that your implementation be somewhat modular. This means that you should follow good programming practices—keep functions relatively short, use descriptive variable names.

Deadlines

Due on April 22nd by 6pm