AFRL-IF-RS-TR-2001-128
Final Technical Report
June 2001

NETWORK ATTACK PROGRAM

TRW ESL

Sponsored by
Defense Advanced Research Projects Agency
DARPA Order No. K297

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-2001-128 has been reviewed and is approved for publication.

APPROVED: W. John Maxey

W. JOHN MAXEY
Project Engineer

FOR THE DIRECTOR:

WARREN H. DEBANY, Technical Advisor
Information Grid Division
Information Directorate

If your address has changed or if you wish to be removed from the Air Force Research Laboratory Rome Research Site mailing list, or if the addressee is no longer employed by your organization, please notify AFRL/IFGB, 525 Brooks Road, Rome, NY 13441-4505. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

NETWORK ATTACK DETECTION

Jack May and James Petersen

Contractor:	TRW ESL
Contract Number:	F30602-99-C-0201
Effective Date of Contract:	30 June 1999
Contract Expiration Date:	31 August 2000
Short Title of Work:	Network Attack Detection
Period of Work Covered:	Jun 99 - Aug 00

Principal Investigator:	Jack May
Phone:	(408) 743-6112
AFRL Project Engineer:	W. John Maxey
Phone:	(315) 330-3617

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

This research was supported by the Defense Advanced Research Projects Agency of the Department of Defense and was monitored by W. John Maxey, AFRL/IFGB, 525 Brooks Road, Rome, NY.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
JUNE 2001

3. REPORT TYPE AND DATES COVERED
Final Jun 99 - Aug 00

4. TITLE AND SUBTITLE
NETWORK ATTACK DETECTION

5. FUNDING NUMBERS
C - F30602-99-C-0201
PE - 62301E
PR - H551
TA - 10
WU - 01

6. AUTHOR(S)
Jack May and James Petersen

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
TRW ESL
1330 Geneva Drive
PO Box 3510
Sunnyvale California 94086-3510

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Defense Advanced Research Projects Agency
3701 North Fairfax Drive
Arlington Virginia 22203-1714

Air Force Research Laboratory/IFGB
525 Brooks Road
Rome New York 13441-4505

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-IF-RS-TR-2001-128

11. SUPPLEMENTARY NOTES
Air Force Research Laboratory Project Engineer: W. John Maxey/IFGB/(315) 330-3617

12a. DISTRIBUTION AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

In our research we developed algorithms to detect attacks on large networks and their network components, such as routers. This approach differs from others that detect attacks on computers. The advantage to network attack detection is that it discovers distributed denial of service (DDoS) types of attacks that cannot be found with conventional techniques.

The algorithms take advantage of changes in emergent properties of large networks to detect attacks. Emergent properties are the statistics of avalanches of lost packets in routers from overloads, and avalanches of communication links approaching their capacity. Statistical data is collected from existing simple network management protocol (SNMP) messages from network components. N-grams are used to detect the changes in the patterns of network management message "conversations" that are caused by the attacks.

A fast large network simulation was developed using self-organizing system (SOS) techniques. This simulation utilized a very simple, but very fast, model that used only the most significant characteristics of the network. The core part of the simulation was less than 100 lines of code that simulated over 1000,000 routers and links per second. In addition to testing the algorithm on real networks, the simulation will be needed for testing attacks that are impractical to implement on operational networks and for planning courses of action.

14. SUBJECT TERMS
Network Attack Detection, Internet, Large Netowkrs, Self-Organizing Systems, Power Law Statistics, SNMP Avalanche Statistics, N-Grams, Network Animation, Denial of Service Attacks, DDoS, DoS, Fast Internet Simulation, Data Flow Simulation

15. NUMBER OF PAGES
28

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Table of Contents

List of Figures

Network Attack Detection Final Report

Introduction

The Federal Government has recognized the United States' reliance on certain national infrastructures. The incapacity or destruction of these infrastructures would have a debilitating impact on the defense or economic security of the United States. One of the categories of threats to these infrastructures is computer–based attacks on the information or communications components that control these critical infrastructures ("cyber threats"). Over the last decade these infrastructures have become increasingly dependent on private sector "cyber" communication systems such as the Internet. Recent news events such as denial-of-service (DoS), DoS tool kits[1], and virus/worm attacks have emphasized the vulnerability of government and civilian organizations to damage caused by these types of attacks on the Internet. Methods for quickly detecting, localizing and responding to attacks to prevent or minimize damages are needed.

Recent work in Self-Organizing Systems (SOS) has shown that complex systems exhibit certain statistical properties[2]. Complex systems exhibit power law statistics and spectrum which plot as straight lines on log-log graphs. In addition, the slope of the power law statistics is a characteristic of a particular complex system. The fractal nature of the Internet traffic is an almost certain indication that large networks are SOS and should exhibit the same kinds of statistical properties.

Project Objective

The objective of the project in this final report is to develop new methods of network attack detection utilizing the inherent statistics of complex systems. It is hypothesized that a large network will exhibit one set of statistics under normal operation and a different one under attack.

A localized attack will change the statistical characteristics in one part of the network. The statistics will no longer be self-similar (the same at all scales) and the statistics will change to a non-fractal, non-power law distribution. An effective wide scale attack will change the slope of the power law curve. Both of these change characteristics are detectable.

The basic approach in this final report develops classification algorithms that use the statistical differences between normal network operation and network operation under attack. The data used in developing the new methods of network attack detection is a combination of collected and simulated data. It is necessary to simulate attack data because it is not currently feasible to run attacks against any real large networks. However, it is possible to collect normal operational data from large networks as long as the collection does not interfere with network operation. Future collection of operational data is planned using the TRW Intranet. The data from network operation will be used to validate the simulation model. This simulator was implemented so that operation and attacks can be easily visualized and changed to aid in understanding and developing methods of countering attacks. This approach has the following advantages:

- It can be used to detect attacks on the network infrastructure, not just attacks on the networked computers.
- It utilizes the inherent large scale characteristics of the network that are impractical for attackers to spoof or control

Task 1. Network Simulation and Visualization

Task Objective

The project objective of the simulator has been to provide simulated attack data for use in testing attack detection algorithms. The simulator design has several objectives.

- It is important to model those key characteristics of the network that will generate the SOS statistics used in attack detection, without adding unnecessary complexity to the simulation.
- The simulation needs to run on a desktop PC and still be able to handle thousands of router nodes.
- The simulation needs to run fast enough to provide quick answers for large networks. It is also important that simulation be easy to modify and maintain so that changes can be quickly implemented.
- It is important to be able to visualize progression and propagation of attacks.

Approach

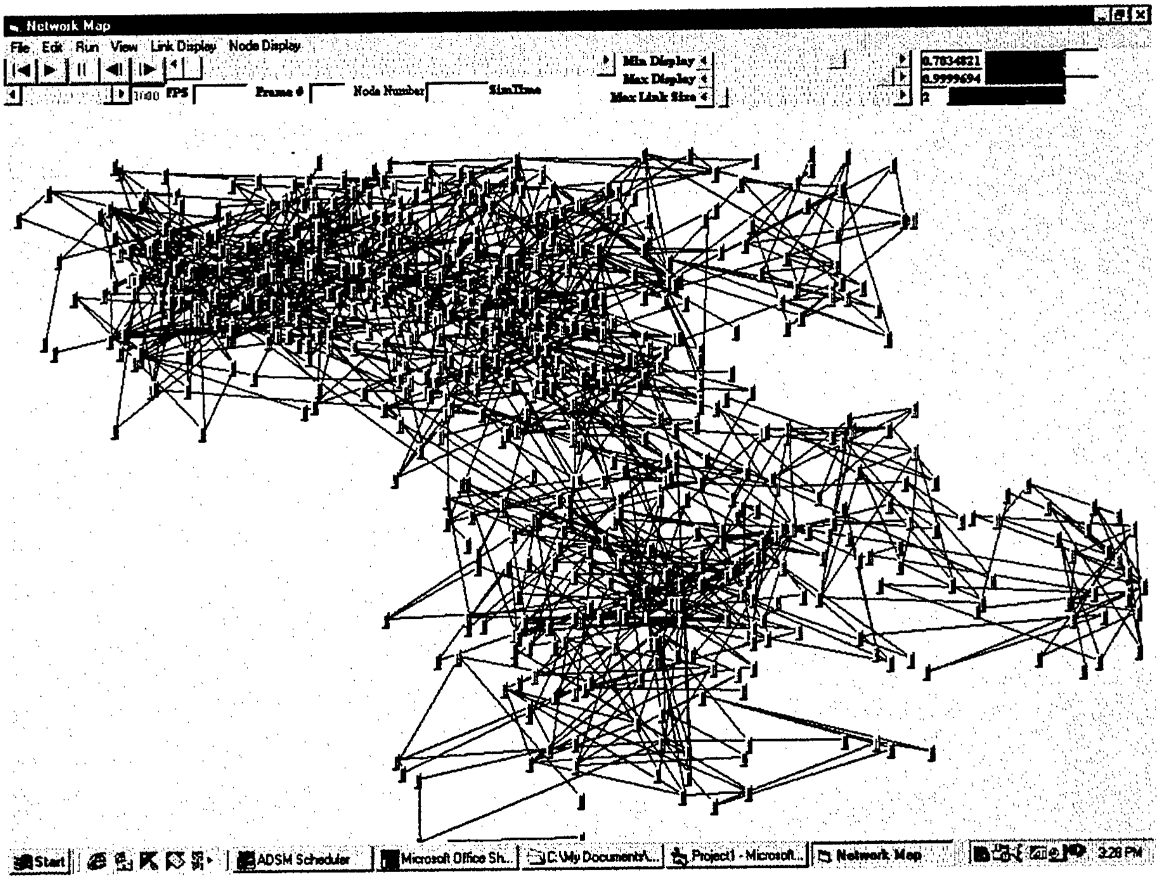
The simulation software was written in Visual Basic 6.0 on a 500 MHz Dell OptiPlex GX1 computer with 128 Mbytes of RAM. The operating system used is Windows 98. Visual Basic was chosen as the implementation language at the top level because it is a productive rapid application development (RAD) environment. We use ActiveX components to integrate code from other languages such as C++. The availability of a large number of commercially available ActiveX components to extend programming functions is also an advantage.

The approach to programming the simulation software has been to develop a flow model and not to try to model the complexity of packets. An iteration of the main simulation loop of the model represents approximately five seconds of network operation. This approach preserves the key characteristics of the network and allows the model to run fast. The main simulation loop for the model is a little over 80 lines of code. It allows 32,000 nodes plus associated links to be processed a second. This is approximately 100K network components per second. The simulation was implemented as an Active-X component so that it could be reused in visualization, attack detection algorithm development, and future coarse of action (COA) tools.

Good models of the topological structure of a network are essential for developing and analyzing networks. Georgia Institute of Technology has developed software to generate graph models that accurately represent the topological properties of real networks[3]. We ported their code from Unix to Windows and compiled it into an ActiveX component that can be called by many Windows development environments. Their Transit-Stub model was used to generate graph files used as the network model input to the simulation software. The simulation software will accept varying sizes of networks.

Network sizes from 40 to 10,000 nodes have been used to test the simulation software. The simulation software was designed to read the file format produced by the Georgia Tech Model and produce a network map of the nodes and links. Figure 1 is an example of a network map produced by the simulation software. This particular network map has 650 nodes and 1802 links.

It is important for the analyst be able to interact and display the simulated data in ways that will allow greater understanding of attacks and avalanches. The simulation software animates the network map allowing simulated data to be visualized. The visualization implementation allows links (lines) and nodes (buttons) to be placed on a form under program control. In addition the links (lines) can be colored, change thickness or disappear under program control. These drawing object parameters have been used to animate the visualization portion of the simulation algorithm.



1000

Min Display 0.7834821

Max Display 0.9999694

Max Link Size 2

Figure 1 Network Map

Algorithm Description

Figure 2 shows the major processing steps in the simulation algorithm.

- (1) The simulation software reads the network map file created by the Transit-Stub model. The simulation software scales and draws the network map in a window. It saves the necessary constants and arrays for the animation process to reference the nodes and links of the network map.
- (2) A randomized initialization file containing the initial state for the network map is created. This procedure sets the initial state for buffers, traffic flows, and routing. Power law statistics are used to initialize network parameters. Those initial parameters are stored to a Network State File. That same file is used to store network parameters at any point in the simulation. The network state file is used to start the animation at the same point each time when running tests under

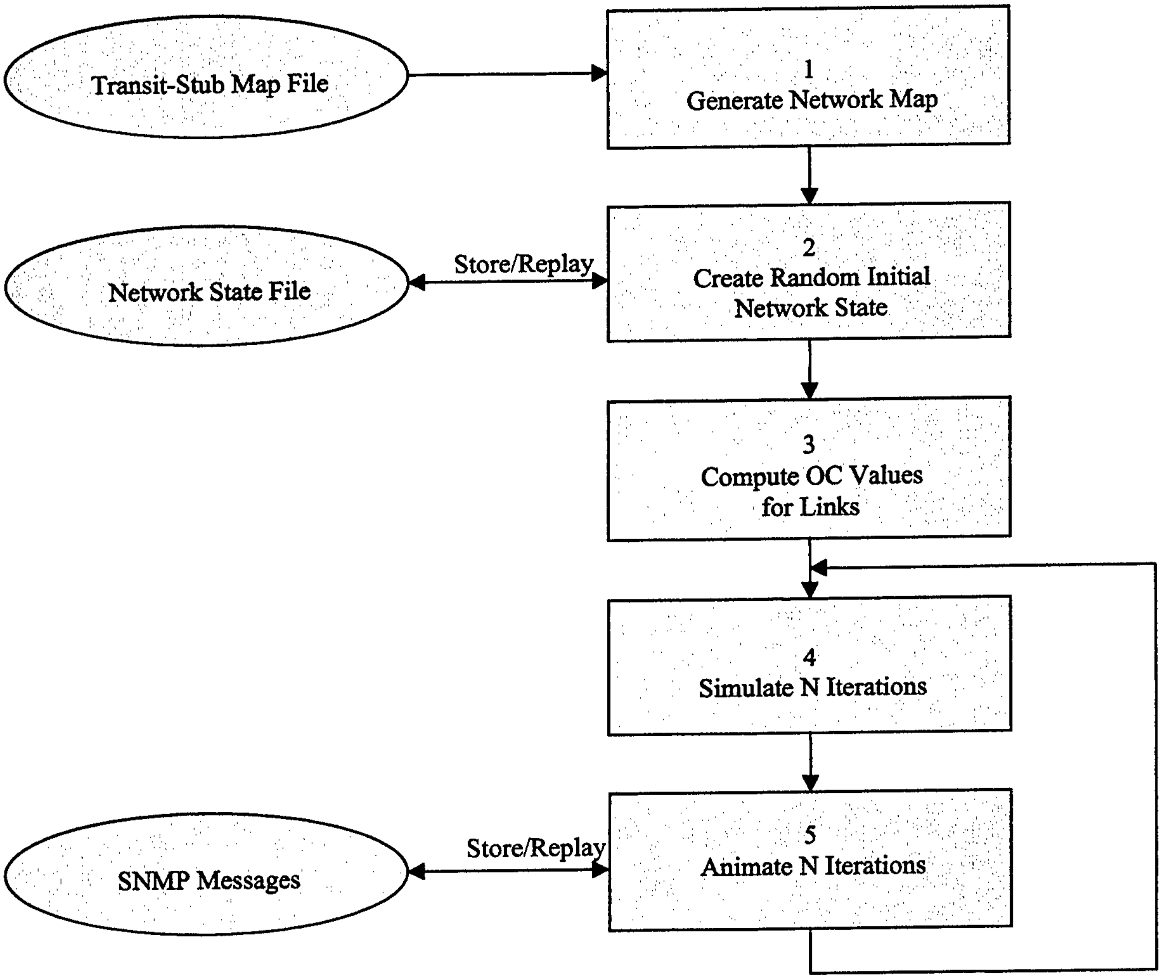


Figure 2 Simulation Algorithm Node/Link Flow Simulation Model

different attack conditions

- (3) Using the initialization file, the link capacities for each node are assigned by running a modified simulation to find average traffic flows for links from each node. The modified simulation turns off congestion control and traffic randomization algorithms to get average steady state flow values. As expected, certain links carry more traffic than others. The nodes are assigned OC numbers (link capacity) relative to the traffic flow they carry. The nodes are then colored on the map according their link capacity OC number.
- (4) The simulation is then run for N iterations. N is a number that will allow all the simulation, animation and output data to fit in the PC memory. The larger the network, the smaller the value of N can be without running out of memory. Keeping the simulation and data in memory allows the simulation and animation to run faster.
- (5) After N iterations have run, the simulated SNMP data is displayed in the animation process. The analyst can slow down, stop, or single step the display, somewhat like a videocassette recorder (VCR). The last N iterations can be displayed over again as many times as desired. In addition, the algorithm allows the analyst to save a section of data (multiple of N iterations) to a file for future replay. The algorithm allows the iterations to continue N at a time (see step 4) until the routine reaches the total number of iterations requested by the analyst.

The animation software allows the analyst to use control sliders to set data display thresholds. For instance, one of the simulation data outputs examined was the percentage loading of links. The control sliders allowed the analyst to display all links carrying a percentage of full capacity between the two thresholds. In addition, another slider controlled how the link widths were scaled (line width) when drawn. If desired, the animation process can be turned off and the simulation then runs faster.

Self Organizing System Simulation Model

Figure 3 illustrates the basic concepts of the simulation model for a single node and associated links. The core simulation loop implements this model. The simulation software scales the flows, data stores and capacities for as many nodes and links as needed for a particular network map. The following concepts are implemented in the model:

- The total traffic flow into a router is assumed to be equal to the total traffic flow out of a router to the link buffers.
- Traffic flows are fractal random processes[4] implemented with a fractal random lookup table i.e. routing is a fractal random process. We generated the fractal random lookup table using fractal traffic generation software[5] we ported to Windows.

- The link buffers have a maximum capacity. If more data flows into a link buffer than it can handle, an overflow occurs and traffic is lost. This overflow results in error messages. The error messages are modeled as flows of lost traffic.

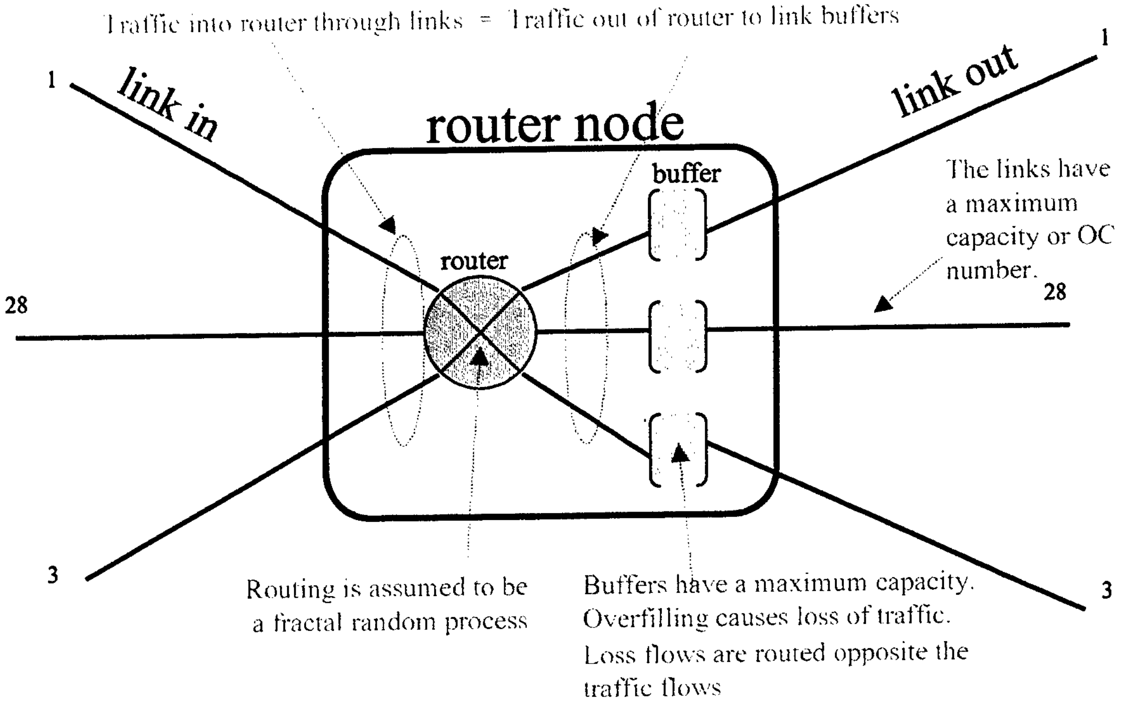

Traffic into router through links = Traffic out of router to link buffers
1
link in
router node
link out
1
buffer
The links have a maximum capacity or OC number.
router
28
28
3
3
Routing is assumed to be a fractal random process
Buffers have a maximum capacity. Overfilling causes loss of traffic. Loss flows are routed opposite the traffic flows

Figure 3 Router Node Model

- Error messages flow opposite the direction of routed traffic.
- Because error messages are a relatively small fraction of the total traffic they are

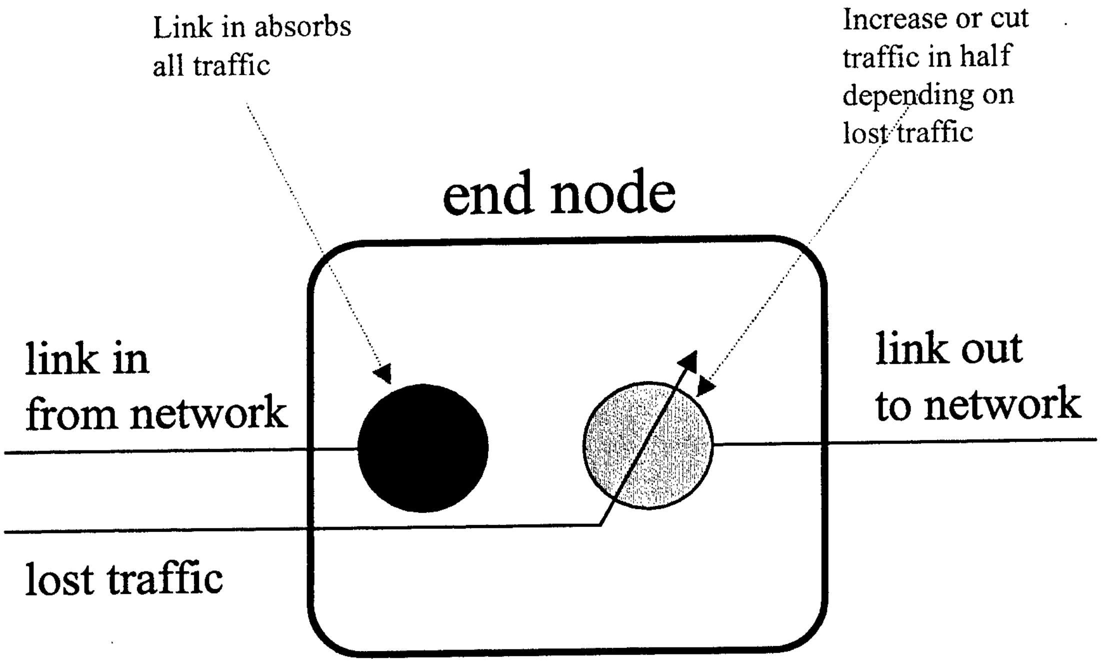

Link in absorbs all traffic
Increase or cut traffic in half depending on lost traffic
end node
link in from network
link out to network
lost traffic

Figure 4 End Node Model

modeled separately from traffic flows and are modeled as not contributing to the total traffic flow.

- All the links from a node have the same capacity. The method for assigning link capacity for each node was discussed previously. The traffic flow out the link can not exceed the link capacity.
- End nodes implement the congestion control mechanism as shown in Figure 4. Only end nodes increase the amount of traffic in the network. End nodes absorb all data routed into then from the network. If the lost traffic flow to an end node is below a threshold then the traffic flow from an end node into the network is increased by a percentage. If the error flows are above the threshold then the end node traffic into the network is cut in half.

We feel that these are the essential characteristics of the Internet that represent how the system operates[6]. The main mechanisms that characterize how the network operates are the routing of data, the buffering of data, link capacity limits, and the congestion control when packets are lost. They create a non-linear system that has self organizing power law characteristics when a large number of elements interact using these basic network characteristics.

There are certainly more details that can be added to the simulation, but experience with other large systems indicates that increased detail will not significantly affect the simulation results[7]. For example details of HTML, FTP, Telnet, ... may have different traffic characteristics, but when all of the protocols are added together in the flow of traffic in a high speed link, their individual characteristic will tend to average out over time and should not have a major impact on the characteristics of network operation.

Characteristics like packet loss and congestion control are a major part of the network which will affect the basic characteristics of the network and will not average out with some other property. Similarly the effects of link capacity and random routing of traffic will not average out with other characteristics. Because these characteristics are fundamental, they are the ones we use in our self-organizing system simulation of the Internet.

A plotting control with associated software was placed on the network map to plot selected log-log statistics of the network. This plot was updated with new data every N iterations. One of the simulated SNMP messages plotted was link loading. Figure 5 shows the number

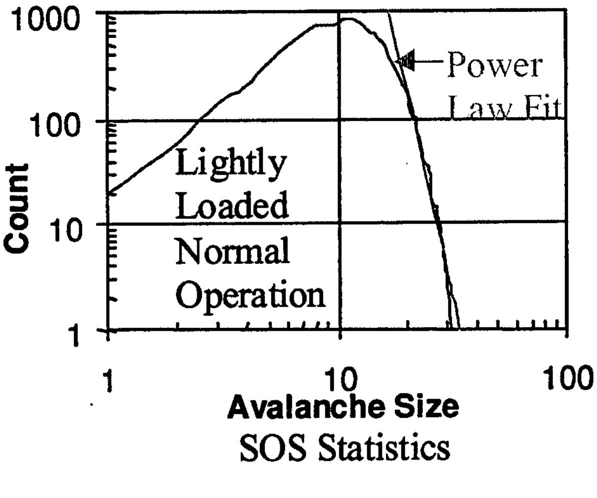


Figure 5 Normal Network Operations

of links near capacity (exceeding 60% loading) during simulated normal operations.

Avalanches in this graph are defined as the total number of links that are loaded beyond a threshold value that we can adjust with sliders in our interface. This is an overly simple avalanche definition which will be improved in the future.

A better avalanche definition would be to count only over threshold links that are closely coupled together. This would mean counting links connected to each other or connected to the same backbone. Regional avalanche information is also needed to locate attacks.

DDoS Attack Results

DDoS and possibly other attacks will reduce the data on adjacent links as Internet congestion control algorithms adjust to the heavy denial of service traffic loads. Our present threshold sliders can measure under load avalanche statistics.

Other important avalanche measures of the network such as packet loss and spectral characteristics of avalanche variations are important to use in the detection algorithms.

The curve in Figure 5, like most power law statistics, deviates from a straight line at the ends of the probability distribution. Most power law curves deviate from a straight line at the ends typically because there is less dependency between the elements of the large system in these parts of the graph. In the network simulation, when there are few link data rates over a threshold value, they tend to be weakly interacting or isolated events. Isolated and weakly interacting events have a different statistical characteristic of avalanches from more strongly coupled events in the other parts of the distribution.

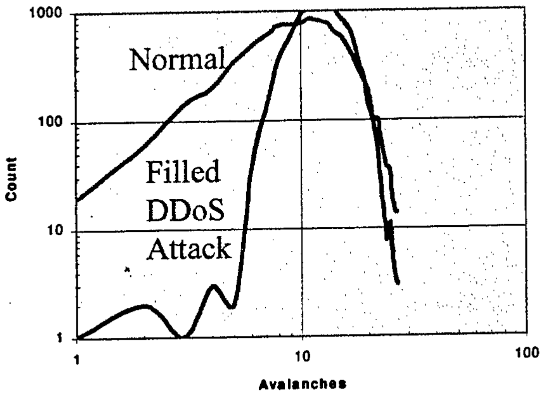


Figure 6 Normal vs Attack Statistics

A " quick and dirty" implementation of a Distributed Denial of Service (DDoS) Attack was accomplished by editing a network state file by hand, routing 99% of the traffic from 5 end nodes to a single end node (i.e. the node under attack). This produced an overload on some of the links the traffic went through. Other links with higher capacity handled the traffic with no problems. In the future, a generalized attack routine needs to be developed to automate this process for several types of attacks. The DDoS attack changed the total statistics of the network as shown in figure 6.

The shapes of the normal network operation and attack operation distribution are significantly different and detection of the difference will be relatively easy.

In second type of DDoS attack (Figure 7), the traffic only overloaded the last link going to the target. The other links were run with the minimum amount of attack traffic to minimize the probability of detection by an automatic algorithm.

Again, the distribution (Figure 7) is significantly different than the normal network operation curve. Detecting the difference is relatively simple

Statistical characteristics of the Internet are continually changing over time. To handle the time varying nature of the network, the attack detection algorithms will look for short term changes in the statistical characteristics caused by the attack being launched. With local information about different parts of the networks, we would use algorithms that would see significant statistical anomalies in different parts of the network.

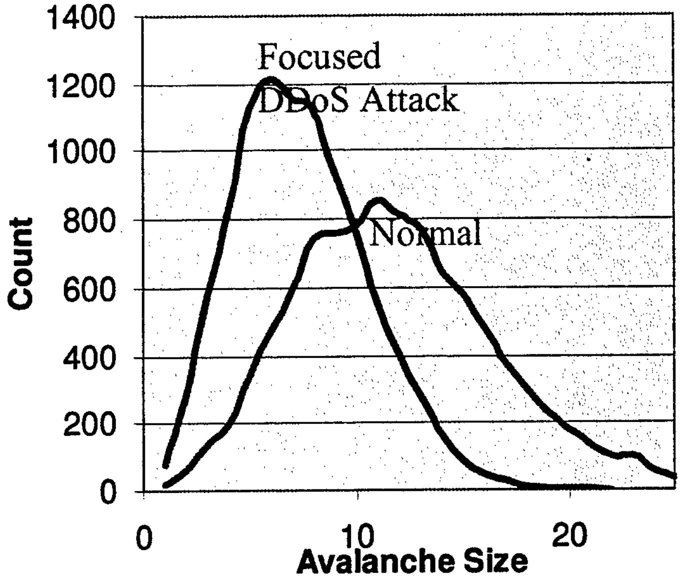


Figure 7 Light traffic DDoS Attack

Simulation Model Verification

To verify our assumptions about modeling the Internet, we installed, debugged and ran the Dartmouth SSF and the S3 simulators. We expected a well defined model of the Internet with a good interface. We found that in both simulators, we would have to make many assumptions and major development would be required. Getting the output data into a form that might be usable for comparison looked very difficult.

Even then it looked like we would not have sufficient capability to compare those packet level Internet simulation models to our traffic flow model.

We have the Mil3 OPNET simulator, which is a well designed commercial package that works at the packet level. We discussed the simulator with our in-house expert user. It may have a better chance of meeting our needs, but there was still a major concern that we would get usable results. Getting those results would use a major part of the program resources to developing the simulation.

This project is working with emergent behavior of large networks which will tend not to appear in the small simulations that we could implement with these other tools. The conclusion is that the only good way to test our simulation techniques is to compare it with a large real network.

The network we plan to use in this testing is the TRW Intranet which is international in extent and used by over 60K people with one or more computers assigned to them. For example, in our Sunnyvale Ca. facility where this project is run, the average employee has more than two computers for their use. We also have about as many people, such as production line workers, that don't have a computer but use computer stations for time worked entry and getting company information.

This large network will show the emergent properties we need and it has even had attack like problem that can be used for testing. For example a DoS type attack has occurred with a faulty set up of some software. We have received oral approval to use the TRW network on this project and are in the process of doing the paper work to get final approval.

Network traffic will vary over time on this network. Since we will have access to all the SNMP messages on the network, we will be able to measure traffic and packet losses in all parts of the network. We can set up our simulation to have the same parameters. As we run our simulator for different sets of parameters, we can see if we are measuring the same statistical characteristics in both the real and the simulated network

The simulated network has small number of parameters and design conditions that can be changed if there are differences in the statistics. For example, differences in packet losses would probably be caused by errors in our model of router cache. Differences in traffic dynamics could require changes in our congestion control algorithm approach.

Task 2. Network Management Message Pattern Detection

Task Objective

The objective of the second part of the program was to develop tools to monitor changes in network management message patterns for attack detection classification. Specifically, we are developing a computationally efficient monitoring system based on incoming SNMP message parameters collected by COTS network management tools.

This is being accomplished by implementing a computationally fast N-gram algorithm for use with TRAP- or GIDO-based parameter inputs in order to monitor potential attacks on a very large network via processing on a single workstation. Our ultimate goal is to fuse the results of avalanche detection, message pattern detection, and attack transient behavior indicators for optimum detection reliability.

Existing Practice

Network traffic is by its nature heterogeneous and stochastic. Current network monitoring systems are not designed to perform adaptive, anomaly-based attack detection against this type of behavior. Instead, existing network monitoring schemes use static thresholds set by a human network manager which in turn generate alarms when exceeded.

The manager manually transfers his understanding of the local network into a set of rules (i.e. threshold levels). When the one or more of the threshold levels are exceeded, an alarm is sent and the manager must correlate the alarm with a corrective action. This existing method is not designed to accommodate the dynamic nature of network behavior.

As a result it cannot reliably detect the precursors of a network attack, let alone distinguish between different attack categories. In addition, as the size and rate of change of the network increases it becomes exponentially more difficult for a human network manager to maintain adequate understanding of the network's behavior.

Approach

Network management messages provide key information on what is happening at each node. That information is typically provided by the equivalent of SNMP messages containing management information base (MIB) variables. These messages provide a different view of the network than avalanche failures.

In particular, they provide an approach for detecting anomalies at the node level, i.e. detecting when one or more nodes are acting significantly different than normal. In our network management message pattern detection approach, we monitor these MIB variables using adaptive statistical techniques based on an N-gram methodology.

Messages are not part of a continuous function so traditional clustering and pattern recognition do not work well. Message pattern recognition is similar to the problem of recognizing patterns in text. A statistical approach for non-continuous distributions called N-grams is often used to do text topic recognition.

These techniques are very general and can be used to recognize the patterns of network management messages. If the pattern changes from normal, this is an indication of a possible attack. Node locations for abnormal message patterns can be used to locate the position of the attack. N-gram algorithms are computationally fast and should allow us ultimately to use one workstation to detect and classify attacks across a very large network.

We began this task by performing an extensive literature search of recently published research in this area. The results of this investigation provided insight into promising message classes and pattern recognition methods that could potentially be incorporated into our design. In addition, this investigation provided confirmation that our N-gram approach to network monitoring has not been attempted before.

Algorithm Description

Figure 8 summarizes the modules and dataflow of our algorithm. We have implemented most of this system in Visual Basic. In the top left box of Figure 8, our algorithm in its current form uses archived IF and IP-level MIB parameters that have been collected off of TRW Sunnyvale's internal Intranet (see Figure 9) via the COTS network management package InterMapper. This data set corresponds to parameters collected at 30 second intervals from three link interfaces and one server.

Ultimately, this data archive activity can be expanded to include parameters from a significantly larger set of links and routers. TRW intranet parameters (incoming and outgoing bytes per second, packets per second, errors per minute, and discards per minute) are used to populate the probability cluster corresponding to "normal" network operation. Ultimately, this "normalcy data" will be expanded to incorporate a much larger set of MIB parameters and will be used to create a family of hyper-plane decision boundaries that can be used to distinguish probability clusters of abnormal nodes. Each node has its own distinct set of decision boundaries.

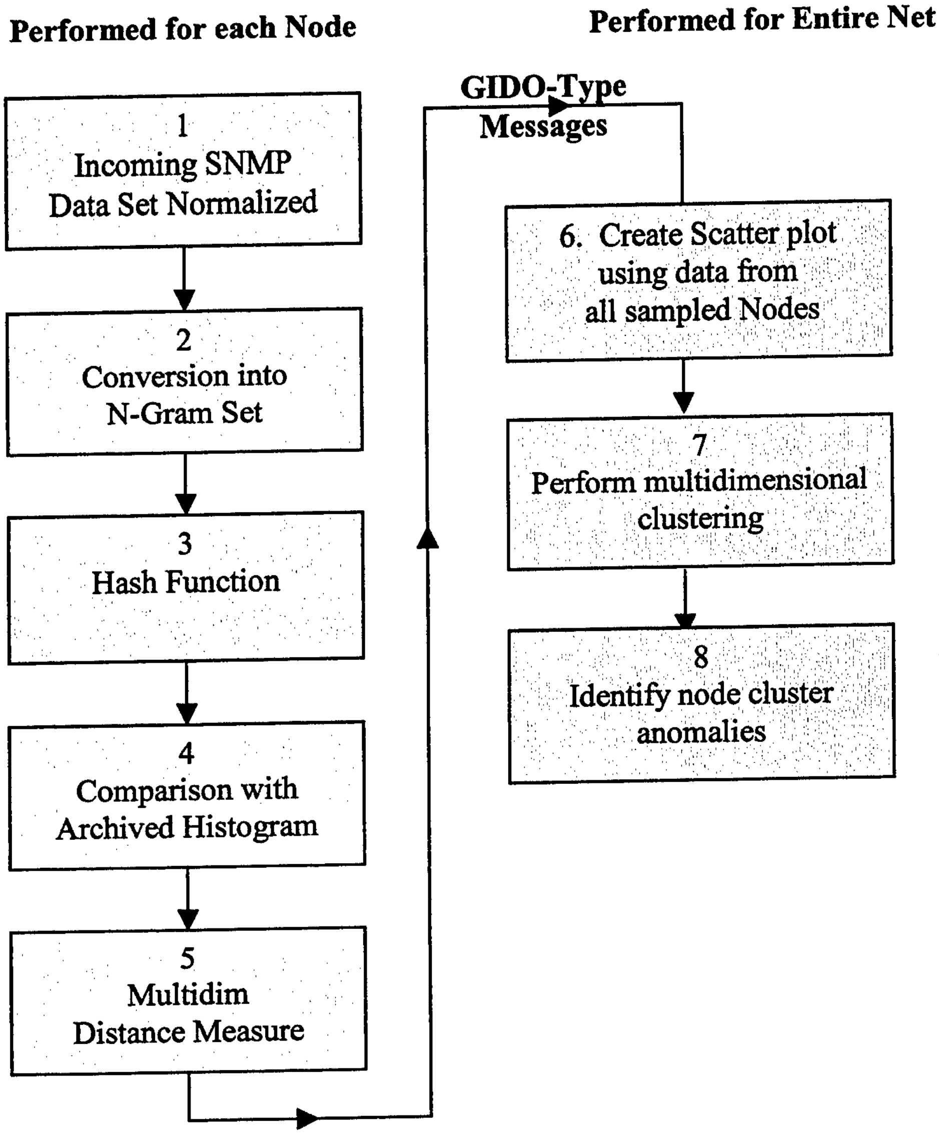


Figure 8 Network Management Message Pattern Detection Algorithm

Once the incoming data set has been normalized, the next module (i.e. Box 2 in Figure 8) generates a purely statistical characterization of the MIB parameters based on N-grams. The N-gram techniques have previously been used by USG in high-speed text topic recognition, text compression, and language recognition.

In our application, the N-gram recognition algorithm is used to detect when the pattern of network messages has changed from normal to an anomalous state. N-gram algorithms have advantages in being able to quickly handle large numbers of messages with probability distribution accumulations that suppress noise. N-gram algorithms are designed to work with non-smooth probability distributions, like network management messages.

As each new SNMP parameter is received, it is normalized and quantized into one of 5 levels. It is then used to generate a new N-gram vector. Each N-gram vector contains eight elements:

1. The current (i.e. new) sample received at time t
2. The preceding sample value received at time t-1
3. The preceding sample value received at time t-2
4. The preceding sample value received at time t-3

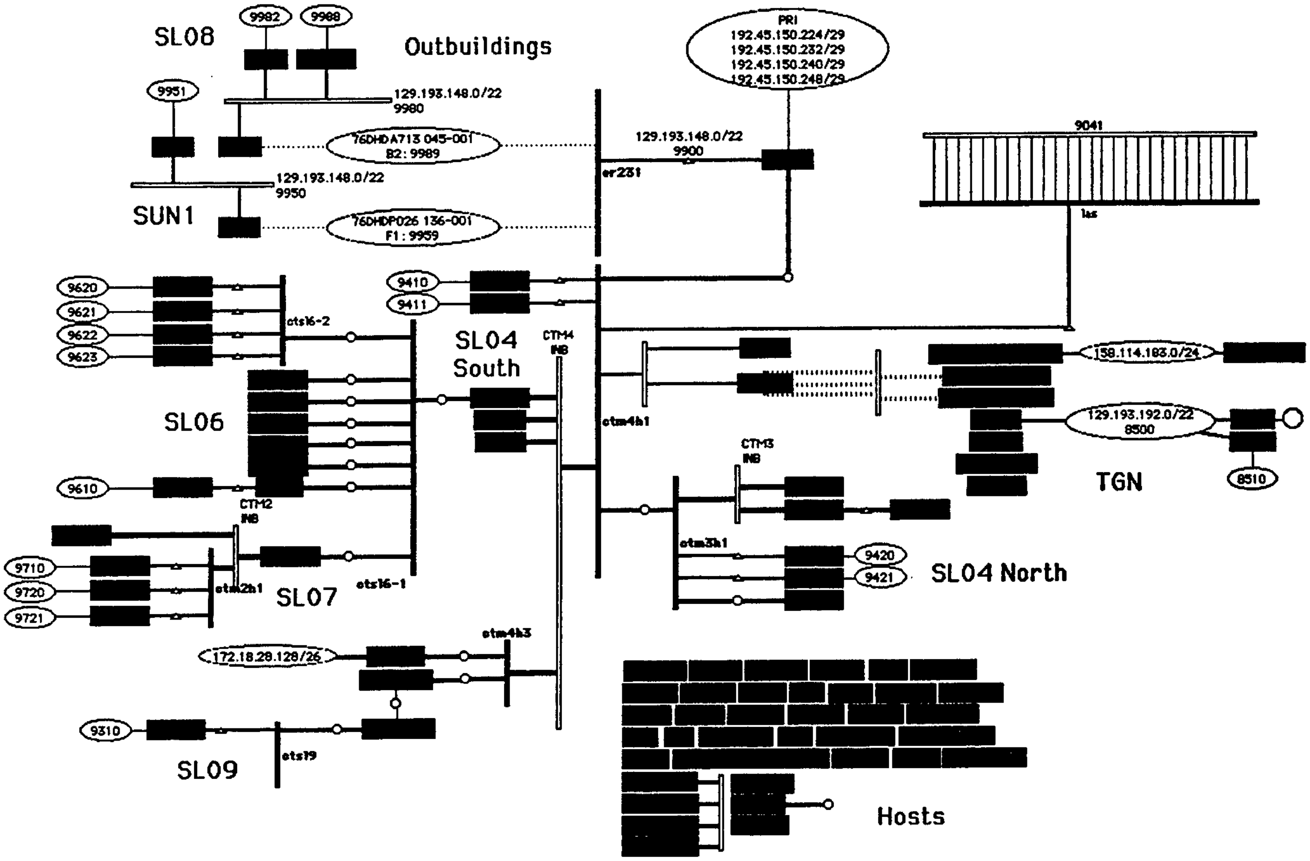


Figure 9 TRW Sunnyvale Intranet

5. The preceding sample value received at time t-4
6. The average of the past 10 sample values
7. The average of the past 100 sample values
8. The average of the past 1000 sample values

In summary, the algorithm in its current form generates eight N-gram vectors (one for each of the eight SNMP parameter categories). Each of these N-gram vectors contains eight elements that reflect recent history of that particular SNMP parameter.

Each N-gram vector corresponds to a numerical value between 1 and 390,624. This range is large and using it directly would require impractical amounts of memory to accumulate the histogram. To avoid this problem, conventional hashing techniques are used to create a smaller address that can be used to address a practical size memory.

As a result, each N-gram vector is indexed to improve the computational efficiency of the algorithm accomplished by running each N-gram vector through a hash function based on modular arithmetic (referred to in Box 3 of Figure 8). The numerical values of each N-gram vector are thus stored and retrieved using these indices as pointers to memory using a hash key as its index.

The hash function reduces the required size of the search table by two orders of magnitude – i.e. from 390,624 to 4031. Our algorithm ignores collisions (i.e. different N-grams being mapped to the same index value) since these appear to occur fairly infrequently.

Each index value generated by the hash function was then compared with a histogram of archived values (Box 4 in Figure 8). We use a distance measure to rank the relative "normalcy" of each N-gram index value, and a geometric distance measure is generated based all eight dimensions (Box 5 in Figure 8).

Decisions

Figure 10 shows a sample of our system's GUI display. The upper eight boxes

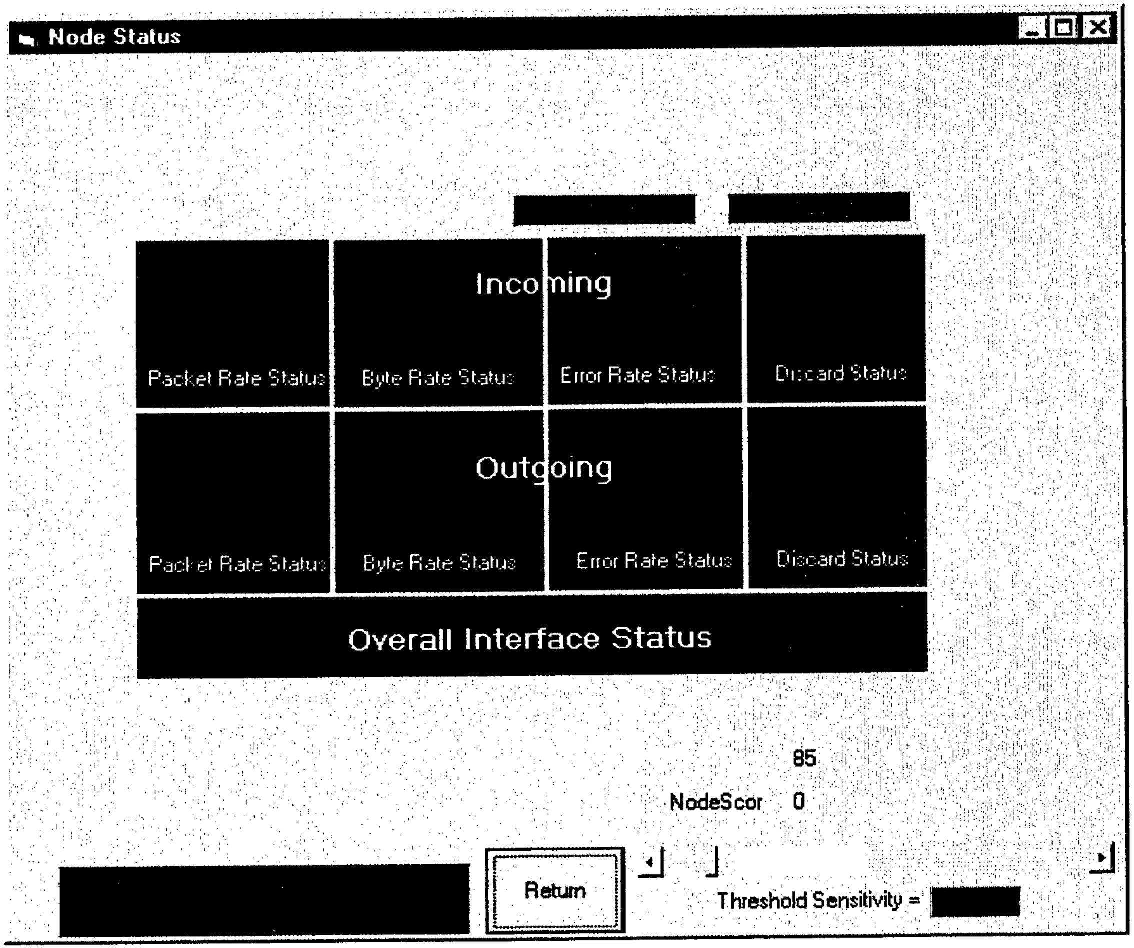


Figure 10 Sample display from node monitor tool GUI

correspond to the relative state of each of the eight MIB parameters. If the distance measure of the current N-gram for a given MIB parameter is small (i.e. the N-gram's hash index value matches a commonly seen hash index value in the archived histogram) then the corresponding box appears bright green.

If the N-gram is less common (i.e. more anomalous) the distance measure increases and the color of the box transitions from green to black to red. The bottom rectangle in the display corresponds to the geometric distance measure produced by the combination of all eight MIB variables.

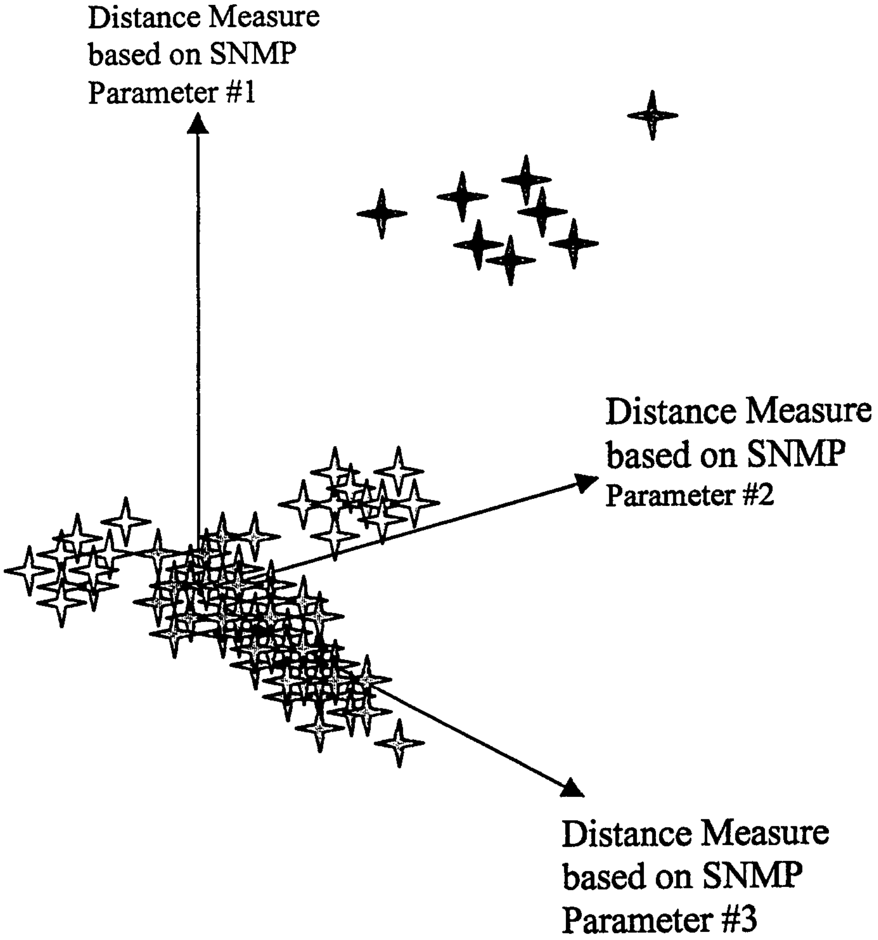


Figure 11 Detection clustering

Because eight separate MIB parameters are being monitored for each node, in effect we have created an eight-dimensional location for each node, where the distance out along each axis (corresponding to each of the eight dimensions) is the probability of a given network management message history. GIDO-type messages from multiple nodes across the network (Box 6 of Figure 8) are then collected to create a network-wide status scatter plot as illustrated in Figure 11. Three axes are shown for illustration purposes with each star representing the current state of a single node, but in addition to the eight of our algorithm there can

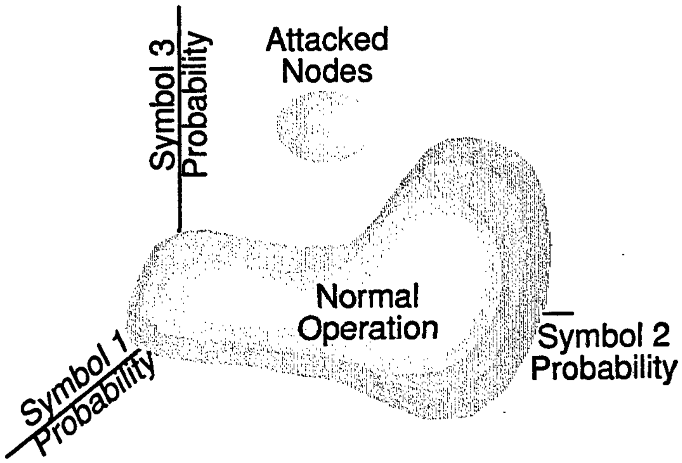


Figure 12 N-gram cluster detection for distinguishing nodes in normal operation from attacked nodes.

ultimately be thousands of dimensions for the histogram space.

Normal operation is going to produce one or more probability clusters centered near the origin as shown in Figure 12. Hyper-plane decision boundaries (Box 7 of Figure 8) can be placed around those clusters by training a decision tree[8].

If an attack is underway, it can be expected to produce an abnormal set of conditions at a node or a set of nodes that no longer respond network management messages (Box 8 of Figure 8). Those abnormalities will create a probability cluster that is different from the normal clusters, as shown in Figure 5. With a significant probability mass outside of the normal range, the detection algorithms will declare that an attack may be underway.

Network Attack Tests

In its current form 8 N-gram vectors per node result in an 8-dimensional cluster set. (Three-dimensions are shown in this illustration.)

As was mentioned earlier, network management messages from TRW's internal Intranet are being used to supply "normalcy" data for our prototype system. Based on our

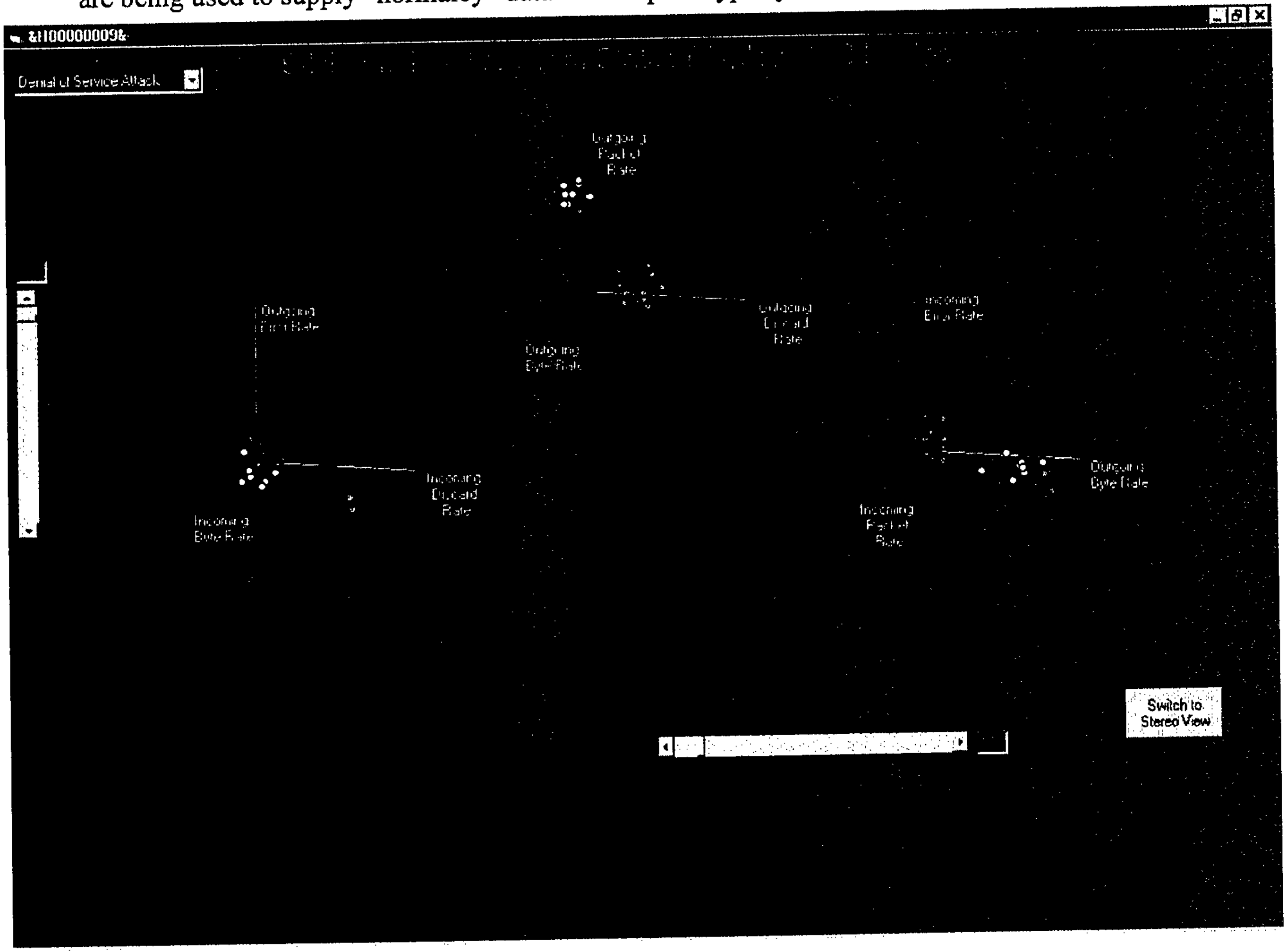


Figure 13 Nine-dimensional N-gram status display of 28 nodes

existing archive, we have extrapolated a set of rough parameter estimates for seven anomalous network conditions:

1. Unusually light traffic load
2. Unusually heavy traffic load
3. Broadcast storm
4. Stalled router
5. Jabbering node
6. Physically degraded link
7. Denial of service attack

Based on the scatter plot concept of Figures 11 and 12, we have developed a prototype visualization tool in Visual Basic. This visualization tool is intended to assist the network manager in simultaneously monitoring the status of all nodes and providing a means for visually observing anomalous clustering patterns. Figure 13 shows a sample of this network-wide scatter plot display in monostatic mode. This tool is also capable of presenting a stereo version of the display for 3-dimensional visualization.

[1] "CERT Advisory CA-2000-1 Denial-of-Service Developments", http://www.cert.org/advisories/CA-2000-01.html

[2] "How nature works", Per Bak, Oxford University Press.

[3] "Modeling Topology of Large Internetworks", http://www.cc.gatech.edu/fac/Ellen.Zegura/graphs.html

[4] Mark E. Crovella and Azer Bestavros, ``Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes," in *IEEE/ACM Transactions on Networking,* 5(6):835--846, December 1997.

[5] V. Paxson, Fast, Approximate Synthesis of Fractional Gaussian Noise for Generating Self-Similar Network Traffic . Computer Communications Review, V. 27 N. 5, October 1997, pp. 5-18.

[6] Bruce S. Davie, Larry L. Peterson, David Clark, "Computer Networks: A Systems Approach, Morgan Kaufmann Publishers, 1999

[7] "One Law To Rule Them All", New Scientist, Nov. 8th, 1997

[8] "The OC1 decision tree software system", http://www.cs.jhu.edu/~salzberg/announce-oc1.html

*MISSION
OF
AFRL/INFORMATION DIRECTORATE (IF)*

The advancement and application of Information Systems Science and Technology to meet Air Force unique requirements for Information Dominance and its transition to aerospace systems to meet Air Force needs.