12-22-2004

Conversation Exchange Dynamics: A New Signal Primitive for Computer Network Intrusion Detection

John C. McEachen
Naval Postgraduate School, Monterey, California

John M. Zachary
University of South Carolina - Columbia

Junling Wang
University of South Carolina - Columbia

Kah Wai Cheng
Naval Postgraduate School, Monterey, California

CONVERSATION EXCHANGE DYNAMICS: A NEW SIGNAL PRIMITIVE FOR COMPUTER NETWORK INTRUSION DETECTION

John C. McEachen, John M. Zachary**,Junling Wang** and Kah Wai Cheng**

*Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, California

**Department of Computer Science
University of South Carolina
Columbia, South Carolina

ABSTRACT

As distributed network intrusion detection systems expand to integrate hundreds and possibly thousands of sensors, managing and presenting the associated sensor data becomes an increasingly complex task. Methods of intelligent data reduction are needed to make sense of the wide dimensional variations. We present a new signal primitive we call conversation exchange dynamics (CED) that accentuates anomalies in traffic flow. This signal provides an aggregated primitive that may be used by intrusion detection systems to base detection strategies upon. Indications of the signal in a variety of simulated and actual anomalous network traffic from distributed sensor collections are presented. Specifically, attacks from the MIT Lawrence Livermore IDS data set are considered. We conclude that CED presents a useful signal primitive for assistance in conducting IDS.

Index Terms—Intrusion detection, network diagnostics, statistical mechanics.

I. INTRODUCTION

Understanding network behavior for the purposes of diagnosis and intrusion detection is currently a major effort in the quest to build secure, robust and dependable computing systems. Specifically, intrusion detection systems (IDS) are detection security mechanisms that monitor a computer system or network, attempt to detect malicious activity, and raise an alarm to system or security administrators. IDSs can be classified as either anomaly-based detection or signature-based detection [1]. The former approach detects *anomalous* behavior, which may be a superset of *undesirable* behavior, and generally suffers from high false alarm rates. The latter signature-based approach may reduce false alarm rates but generally depends on a well-defined security policy to base detection on. Furthermore, signature-based intrusion detection systems are unable to detect events for which a signature is not defined in their signature database.

The recent trend in intrusion detection is to deploy a large number of sensors throughout an organization's network with the hope of gaining visibility into all areas of the network ([2],[3]). The unfortunate consequence of this approach is that the network administrator is overwhelmed with a plethora of somewhat information. Consequently, a new trend in IDS has developed in the area of data reduction [4].

We present a novel approach to modeling distributed system with a high number of interacting entities. This problem is notoriously complex, and our model seeks to provide some level of data reduction so as to distinguish what is an anomaly from what is typical network activity. Consequently, this technique has the potential to find applicability to a wide variety of systems (beyond computer network systems).

Efforts related to our approach can be found in [5], [6] and [7]. These approaches all consider the problem from the abstraction of determining statistical properties of the network. In this paper, however, we intend to focus on the analysis of the underlying state descriptors with the hope of extending these to global properties in the future.

II. DESCRIBING NETWORK CONVERSATION FLOW

As was stated before, the goal of our approach is to reduce standard network data into a useful, reproducible, and meaningful form, ultimately to allow accurate detection of network anomalies. The notion of state will be more carefully defined below, but quickly summarizes into activity levels of various conversation groups within the network. One end product is a real time three-dimensional signal description of the configuration of the network.

The network is constructed as a state space of information sources and sinks. As information quanta move throughout a network, the state space is updated accordingly. States are represented as a vector $\vec{v}$ of sources and sinks. The analogy used is that of *buckets* and *balls*. Information moves between nodes represented as buckets as indivisible balls. As the network moves information around, this is represented as balls being passed between buckets. For example, a series of n packets transmitted from a node N_a to another node N_b would be modeled as n balls moved from bucket X and placed in bucket Y. The association of node N_a with either bucket X or Y depends on the nature of the conversation. The bucket can be defined using any combination of conversation characteristics including the affiliation of who is talking (individual hosts or networks), the language they are speaking (TCP, UDP, or ICMP), or the job they are performing (client or server).

In its simplest form, each node in a network is associated with one or more buckets and the total number of packets exchanged between nodes is modeled as moving balls from bucket to bucket. The collection of all buckets together with the allowable distribution range of balls forms a *bucket state space*.

The bucket state space includes an initial distribution of balls among the buckets (corresponding to a computer network with an expected distribution of information). This initial distribution forms an *initial condition* for the bucket state space. Likewise, the allowable bucket state

Published in 3rd Workshop on the Internet, Telecommunications and Signal Processing, Adelaide, Australia, December 20-22, 2004.
Edited by Beata J Wysocki & Tadeusz A Wysocki. ISBN: 0-9756934-0-9.

spaces are assumed to be represented by a set of *boundary conditions*.

A state transition causes a shift in the distribution of information between the buckets. In other words, this model translates network behavior into bucket state transitions by selecting a ball from the bucket matching the source characteristics of the packet and moving that ball into the bucket matching the destination characteristics of the packet, thereby redistributing the information and transitioning the state.

Figures 1 and 2 show the importance of the state walk, and how the model provides more information than a simpler model. Both signals represent a state space walk made up of two conversation groups, bucket A and bucket B, and they start with 5 balls each. As stated before, a network exchange can cause a state transition in three ways: by moving a ball from bucket A to bucket B, by moving a ball from bucket B to bucket A, or by either moving a ball from bucket A to bucket A or bucket B to bucket B, essentially resulting in no change for that period.

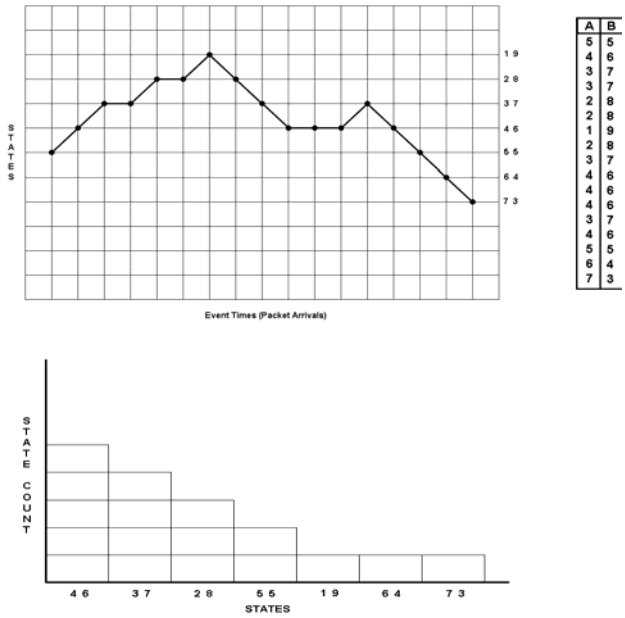

Fig 1. A sample state walk for a two bucket model. (Top left) A plot of the state walk over time. (Top right) A plot of the bucket sizes over the course of the state walk. (Bottom) A ranked histogram of the bucket states over the entire time period of this state walk.

The paths shown in the figure 1 and 2 share a few notable similarities. The paths begin and end at the same state and undergo the same number and types of state changes, including five changes up, seven down, and four no changes. However, the transitions that make up those changes produce remarkably different paths, shown on the left hand side of the figures, and those paths produce remarkably different state counts, shown on the right hand side of the figures. Thus it is shown that this model provides a number of unique discriminators including number and type of state changes, starting and ending point and the general path in-between, and the state counts.

By examining the manifold, or canyon, developed by collecting various state histograms over time, anomalies can be easily spotted as perturbations in the normal flow of the canyon. The cause of such dramatic changes in practice ranges from a single transition to many thousands of packets.

An anomaly can cause one of two effects in the canyon signal. Either new states are visited, or previously visited states are seen more often. The first effect will cause a spike oriented along the z-axis and the latter along the y-axis. An anomaly that is orthogonal to the normal traffic flow will tend to cause a spike oriented along the z-axis, due to the new states visited. An anomaly that is parallel to the normal traffic flow will tend to cause a spike oriented along the y-axis, due to the revisiting of previously visited states. The magnitude of the potential spike is what determines the ability of the operator to detect the anomaly. The orientation of the anomaly with respect to the normal traffic flow will determine the magnitude of the perturbation. A single packet anomaly that is orthogonal to the normal traffic flow will cause a large perturbation in the signal, where a packet that is parallel to the normal traffic flow will cause a relatively small perturbation. The less orthogonal the anomaly is to the normal traffic flow, the larger the number of anomalous packets required to cause a noticeable perturbation in the signal response.

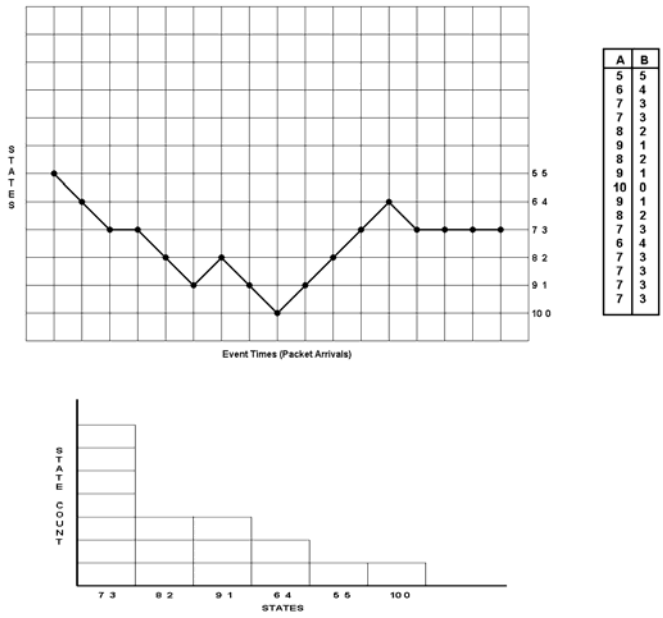

Fig 2. An alternative state walk for a same two bucket model as shown in fig 1. (Top left) A plot of the state walk over time. (Top right) A plot of the bucket sizes over the course of the state walk. (Bottom) A ranked histogram of the bucket states over the entire time period of this state walk. Note how the histogram varies for this state walk even though both examples end in the same state.

For example, figure 3(a) represents all of the possible bucket states that are contained in the bucket state space for a system consisting of three buckets (a, b, and c) each containing four balls. Each of the blue nodes represents a

different bucket state. The number of balls in a given bucket is given by the lines parallel to the side opposite the vertex of interest. Each of the vertices corresponds to the case where all of the balls are in the associated bucket. The purple node represents the initial ball distribution, or initial bucket state of $\{4,4,4\}$. In figure 3(b), the number of balls in bucket 'c' is constant at four. The thick green line represents the nine possible bucket states based on a conversation between the remaining two buckets.

An example of the results of a single packet anomaly, that is orthogonal to the normal traffic flow, can be seen in figure 4. In this case the packet caused a ball to move from bucket 'c' into the conversation between buckets 'a' and 'b'. The result is a new line of possible bucket states. This new line contains ten possible bucket states. Given that the data from any given sample window is averaged over a historic window of time there are now nineteen possible bucket states, which is more than double the original number of nine. This results in a run out (in the z-axis direction) of the canyon signal. This type of anomaly is very easy to detect even though it was caused by a single packet.

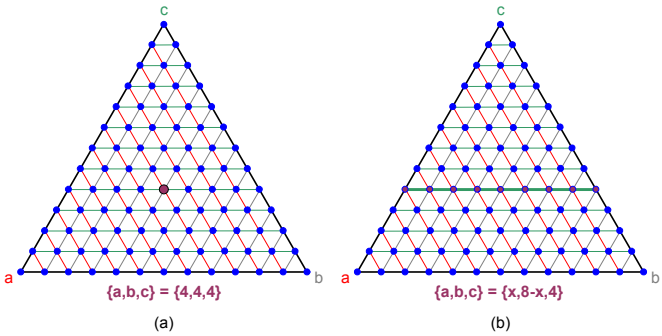

Fig. 3. Graphics that depict the total bucket state space for a system containing three buckets each with four balls. Each node corresponds to a different bucket state. – (a) The purple node corresponds to the bucket state of $\{4,4,4\}$. (b) The green line represents the range of possible bucket states for a conversation between buckets 'a' and 'b'.

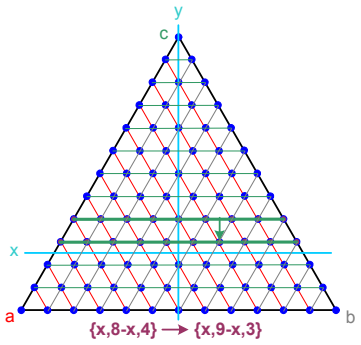

Fig. 4. A graphic depicting the results of an anomalous packet that is orthogonal to the normal traffic flow. The anomalous packet causes a ball to move from bucket 'c' into the conversation between buckets 'a' and 'b'. The results is the state walk moves from the line at $c = 4$ to the line at $c = 3$.

The more orthogonal anomalous traffic is to the normal traffic flow, the greater effect the anomaly will have on the signal response. Since it is not possible to know all the expected anomalous traffic in advance, the key is to create a collection of bucket spaces that provide a tight classification of critical traffic. For example, traffic should be parsed by functional group, like web servers, as opposed to grouping servers and clients together.

This notion of orthoganlity is illustrated in figure 5. In this example the signal response for two diametric attacks sorted through the bucket space are shown. This bucket space in particular is designed to show conversations of SMTP traffic. In the first, an ICMP denial of service attack shows insignificant response in the signal response. On the contrary, a mailbomb attack using the SMTP protocol clearly identifies the anomalous SMTP traffic.

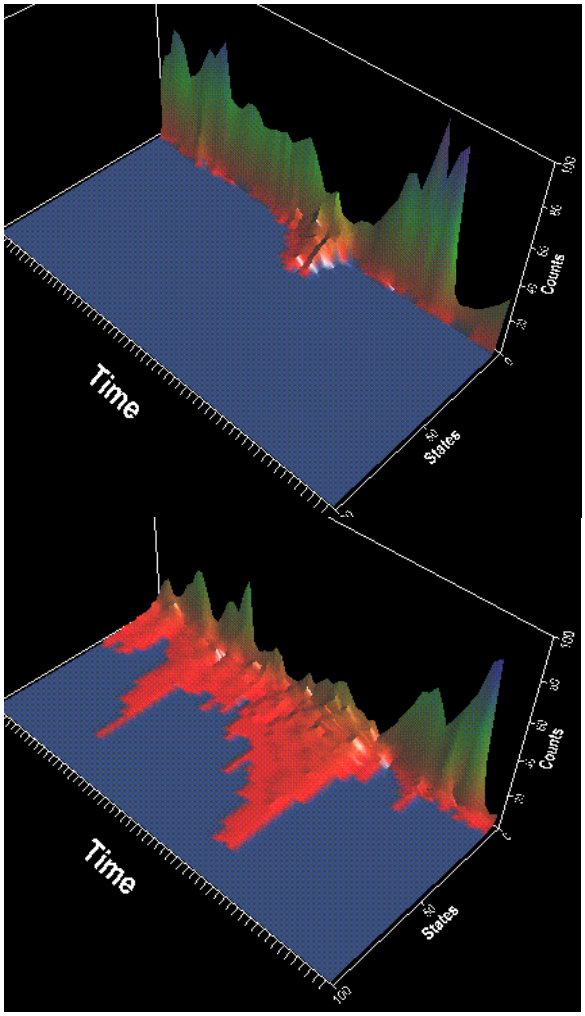

Fig. 5. (Top) Signal response of an orthogonal attack (ICMP flood) against a bucket space monitoring e-mail exchanges (SMTP). (Bottom) Signal response of a mailbomb attack against the same mail server.

There is a limit to the number of buckets a configuration can have, ideally, multiple instances of the system should be run concurrently to allow for smaller bucket spaces. This is also beneficial in reducing the complexity of interpreting the signal space which increases with the number of buckets. The next section presents some of the analysis on actual network traffic.

III. EXPERIMENTATION AND ANALYSIS

This section is divided into two parts: controlled experiments conducted laboratory test equipment and results from real traffic on operational networks.

A. Laboratory experiments

The purpose of these experiments was to show how the bucket state histogram varies as the bucket space description deviates from the actual network configuration. Two categories of experiments are discussed. The first shows the effect of an increasing number of rogue web servers on the bucket space histogram. The second shows the effect of a single out-of-profile packet on the bucket space histogram.

The simulation network consisted of a trusted subnet of 10 web servers connected to an untrusted internet of 1000 clients. In order to simulate typical network traffic, Spirent TeraMetrics traffic generators running TeraCaw software were used. This system is capable of simulating millions of client/server sessions using various application level protocols, beyond the initial three-way handshake. The system was configured such that any where between 400 and 800 users (using the 100 client machines) would randomly access the 10 web servers. Each web access consisted of establishing a TCP connection with the server, an HTTP GET message and then a 64-byte HTTP Response message. The connection was then terminated with a RESET.

The bucket space definition included a list of "authorized" web servers, identified an address space associated with trusted users, and considered ports below 1024 to be service ports. Consequently, for figures 5 through 9 the bucket space was partitioned as follows:

1. A trusted IP address, a web server, and a service port
2. A trusted IP address, a web server, and not a service port
3. A trusted IP address, not a web server, and a service port
4. A trusted IP address, not a web server, and not a service port
5. Not a trusted IP address and a service port
6. Not a trusted IP address and not a service port

The number of possible bucket states, N, can be determined as:

$$N = \binom{n+k-1}{n-1} \tag{1}$$

where n is the number of buckets and k is the number of balls in the system. Thus for the experiments above, with six buckets and initially four balls each (k = 24), there are theoretically N = 118,755 possible bucket states.

In the control run the bucket space description matches the actual network topography. Figure 5 illustrates the average bucket sizes and bucket space histograms over a period of two minutes (120 seconds). The typical load on the network was approximately 1000 packets per second. The bucket histogram is shown at two scales in this example.

Since the bucket space definition is aligned with the actual traffic patterns, the number of non-zero bucket states is small. Hence the histogram tail is very short.

Perhaps more importantly, this display also illustrates the smoothing effect defined by the central limit theorem on traffic that has been shown to be highly self-similar in nature on a per-client basis ([8], [9]). In other words, even though all clients arguably have the same heavy-tailed exchange characteristics, the actual distribution of states as shown by the bucket state histograms is highly normal and smooth, particularly when the bucket definitions are aligned with the configuration (i.e. all web servers in the web server list).

Next an anomalous packet was injected into the network for each during the above scenario. The anomalous packet was an UDP packet from one of the web servers to one of the clients. The packet originated from an ephemeral port (1025) with a destination service port on the client (53). Figure 6 shows the effect on the bucket state histogram for this packet a scales comparable to figure 5.

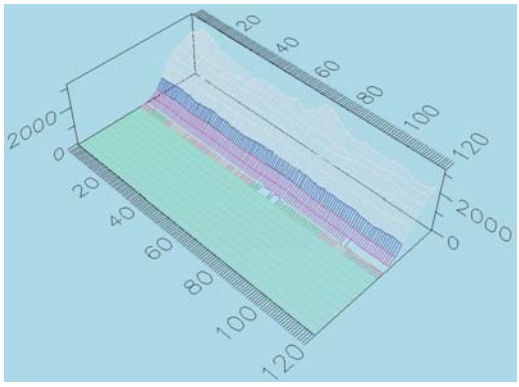

Fig 5.The bucket state histogram over time. Decreasing frequency of bucket states comes out of the graphic. Traffic is confined to external clients visiting internal web servers so the number of bucket states is small.

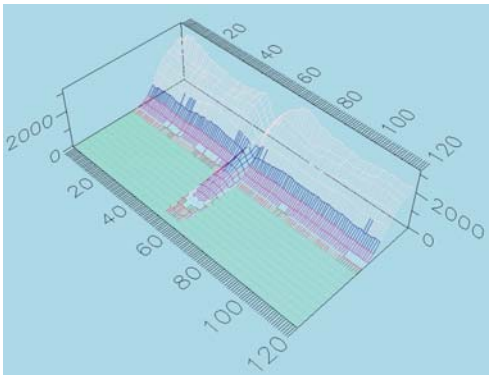

Fig 6. The bucket state histogram with a single UDP injected into the over 200,000 web traffic packets. Compare to figure 5.

The difference caused by the anomaly of the UDP packet should be readily apparent when comparing figures 5 and 6. Keep in mind that during this two minute period, over 200,000 packets were exchanged. The reason for the significant protrusion is because the UDP packet forces a ball to be transferred to a state that would not otherwise be

visited, reducing the counts of the "normal" buckets and altering the frequency of the histograms (hence the notch in the signal response.)

The second experiment illustrates the effect when one web server removed from the configuration file. This web server is now in effect an unauthorized or rogue web server. The top graphic in figure 7 shows the effect of pulling the single server out of the web server group.

One immediately notes the significant variation in the bucket state histogram when compared to figure 5. This is due to the self-similar nature of the requests made to the single rogue web server. This hypothesis becomes more evident in lower graphics of figure 7 where two, four and six web servers are removed from the "web server" list. As we might expect, the increasing number of rogue servers invokes the central limit theorem, producing an increasingly smooth display. We do see and increasingly larger number of active bucket states, however, due to the larger variation in traffic characteristics.

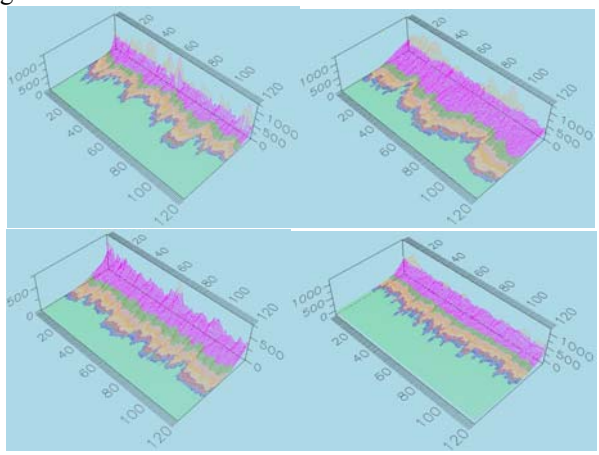

Fig 7. Bucket state histograms after the removal of one (top, left), two (top, right), four (bottom, left) and six (bottom, right) web servers from the web server list. Note the difference between these and the middle of figure 7. As we remove more web servers from the "authorized" list we see increasing smoothness in the display but also and increase in frequency of bucket states.

Next a single UDP packet was injected into all of the scenarios of figure 7. The resulting displays are found in figure 8. Note how the UDP packet gets completely lost in the variable traffic of a single server removed, but is still quire prominent in all the others.

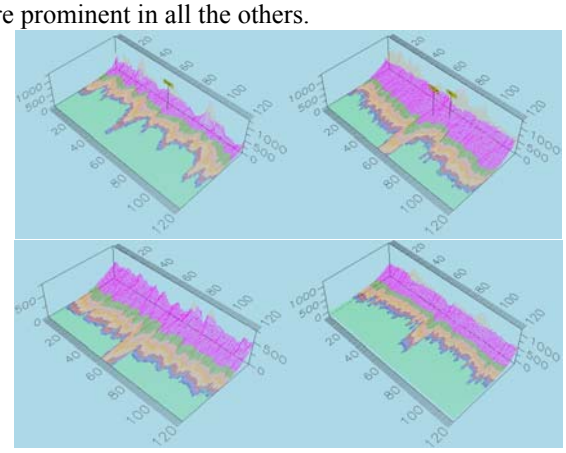

Fig 8. Bucket state histograms for the traffic in figure 7 with a single UDP injected amongst the 200,000 packets. Histograms shown after the removal of one (top, left), two (top, right), four (bottom, left) and six (bottom, right) web servers from the web server list.

B. DARPA Lincoln Lab IDS Test Data

For analysis and configuration of the system we utilized the *Tcpdump* files that were collected by Lincoln Labs using their 1999 Simulation Network. Per references [10], the simulation network was created to conduct evaluations of intrusion detection systems by measuring detections and false alarm rates.

Figure 9 shows the response to attack #41213446, an ICMP flood or "Smurf" attack. This attack is very common and generally easy to detect.

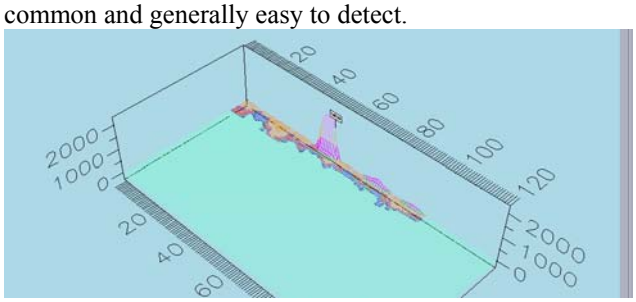

Fig. 9 Response of the system to an ICMP flood, #41213446.

Figure 10 displays the response to a Mailbomb attack, #42155148. This is a denial of service attack directed against the sendmail program. This is accomplished by sending a unique set of strings to the sendmail server.

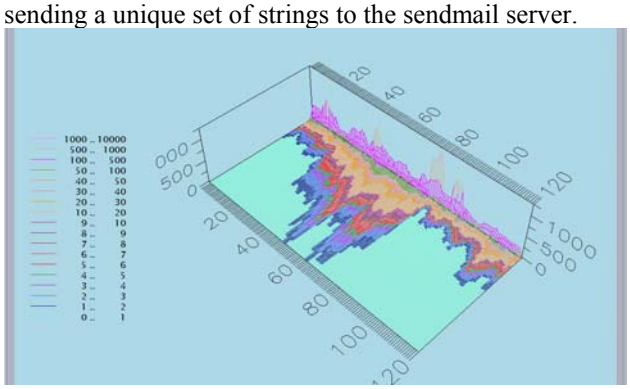

Fig 10. Response of the system to a Mailbomb attack, #42155148.

A similar type of attack to the Mailbomb is the Apache2 attack, #51140100. The response of the system to this attack is shown in Figure 11.

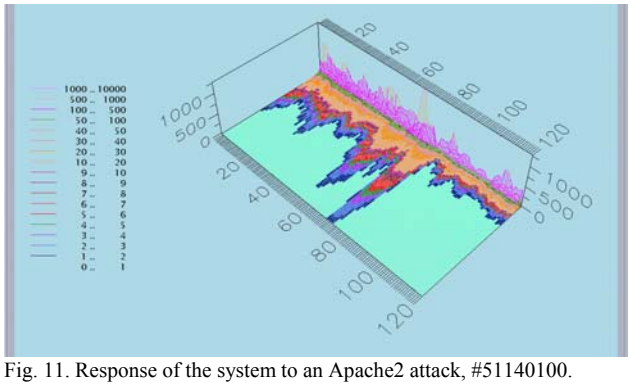

Fig. 11. Response of the system to an Apache2 attack, #51140100.

Finally, the response of the system to a sweep of IP addresses is shown in figure 12. The shape of this response is particularly worth noting.

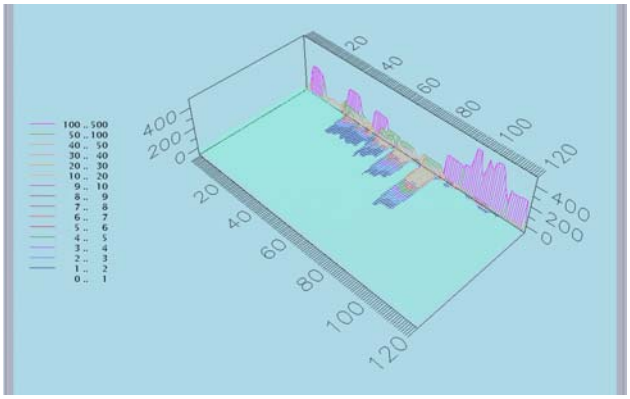

Fig. 12. Response of the system to an IP sweep, #52211313.

C. *Operational network observations*

Figure 13 depicts a flood of ICMP packets originating inside a monitored operational network after normal working hours. This flood increased the state count; thereby producing a large spike on the GUI. This flood consisted of 6,032 ICMP echo requests/replies within a four second time frame. ICMP echo requests and replies are not necessarily anomalous, however the owner of the system was logged off and at home requiring notification of the local CERT for further investigation.

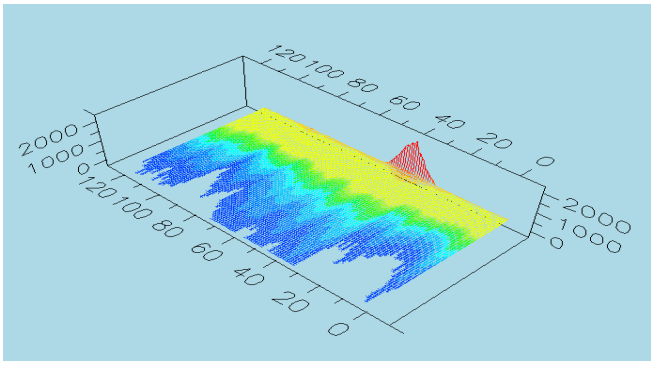

Fig 13. An anomalous event observed on an operational network. Specifically a flood of ICMP packets was released from an internal client over a four second period.

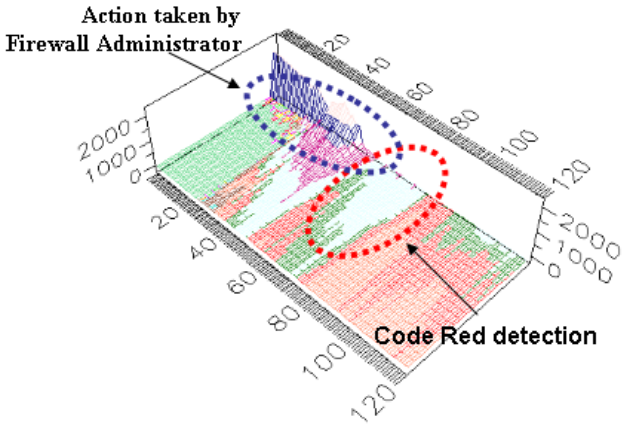

Fig. 14. Response of the system to the Code Red attack indicated by the red circle, followed by the subsequent action of shutting down the firewall by the network administrator.

Figure 14 illustrates the response due to the Code Red worm in August 2002. This example is particularly interesting in that it show how the character of the network changed twice – once when the worm breached the school's network and then when the firewall was shutdown due to detection of the breach.

IV. Conclusion

We have presented a novel approach to characterizing the conversation flow of a computer network using an intelligent form of data reduction based on an Ehrenfest urn model. This approach associates traffic of certain characteristics with a category or bucket and observes the transfer of information from one category to another. Revealing behavior is identified by viewing the histogram of the bucket states, or category arrangements, over time. The approach has been evaluated on both simulated laboratory traffic and operational network traffic.

More importantly, the experiments demonstrated how this approach produces events that can be characterized by Gaussian models. While most anomaly detectors try to apply ML techniques to heavy tailed distributions (e.g. looking at packet length), our approach produces a statistic that can justifiably be analyzed with ML/MSE estimators.

References

[1] S. Axelsson, "Intrusion detection systems: A survey and taxonomy", Chalmers University Technical Report 99-15, March 2000.

[2] P. Porras and P. Neumann, "EMERALD: Event monitoring enabling responses to anomalous live disturbances," Proceedings of the *20th National Information Systems Security Conference*, Baltimore, MD, October, 1997, pp. 353 – 365.

[3] C. Manikopoulos and S. Papavassiliou, "Network Intrusion and Fault Detection: A Statistical Anomaly Approach," *IEEE Network*, October 2002, pp. 76 – 82.

[4] S. Axelsson, "The base-rate fallacy and the difficulty of intrusion detection," *ACM Transactions on Information and System Security (TISSEC)*, 3(3), pp. 186-205, ACM Press.

[5] Burgess, M., "Thermal, nonequilibrium phase space for networked computers," *The American Physical Society*, Volume G2 Number 2, pgs. 1738-1742, August 2000.

[6] Evans, S. C. and Barnett, B., "Network security through conservation of complexity," *Proceedings of IEEE Military Comms. Conf. (MILCOM 2002)*, pgs. 1133 – 1138, Los Angeles, October 2002.

[7] Donald, S. D., McMillen, R. V., Ford, D. A., and McEachen, J. C., "Therminator 2: A real-time system for patternless intrusion detection", *Proc. of the IEEE Military Comms. Conf. (MILCOM 2002)*, pgs. 1498 – 1502, Los Angeles, October 2002.

[8] Crovella, M. and Bestavros, A. "Self-Similarity in world-wide web traffic: Evidence and possible causes," *Proc. ACM Sigmetrics Conf. on Meas. And Mod. Of Comp. Sys.*, May 1996.

[9] Arlitt, M. and Jin, T. "A workload characterization study of the 1998 world cup web site," *IEEE Network*, May/June 2000.

[10] Massachusetts Institute Of Technology, Lincoln Laboratory, DARPA Intrusion Detection Evaluation. [http://www.ll.mit.edu/IST/ideval/index.html]. 14 September 2003.