Detecting Coordinated Distributed Multiple Attacks

S. Mukkamala, K. Yendrapalli, R. B. Basnet, A. H. Sung
Department of Computer Science
Institute for Complex Additive Systems Analysis
New Mexico Tech
(srinivas|krishna|rbasnet|sung@cs.nmt.edu)

Abstract

This paper describes results concerning the robustness and generalization capabilities of kernel methods in detecting coordinated distributed multiple attacks (CDMA) using network audit trails. We also evaluate the performance of denial of service detection models built using the key features in detecting a new attack scheme; CDMA. The data is generated by carrying out the attack (CDMA) in a closed environment at New Mexico Tech Information Assurance Laboratory.

We use traditional support vector machines (SVM), biased support vector machine (BSVM) and leave-one-out model selection for support vector machines (looms) for model selection. We also evaluate the impact of kernel type and parameter values on the accuracy of a support vector machine (SVM) performing CDMA classification.

We show that classification accuracy varies with the kernel type and the parameter values; thus, with appropriately chosen parameter values, CDMA can be detected by SVMs and BSVMs with higher accuracy and lower rates of false alarms.

1. Introduction

The predominant intent of DoS attacks is to prevent or disrupt the use of victim's network or resources. This is done by exploiting the target computer or network vulnerabilities like buffer overflow, protocol stacks etc. The attack is magnified by using multiple systems leading to a Distributed Denial of Service attack. The reason for DoS attacks to be so effective is that the security of networks and in turn the hosts it comprises of is highly interdependent. In many cases the attacker uses many intermediate systems as attack machines, rather than the attackers own machine. Thus the security of the victim network depends on the security of other networks as well. Defending against DoS attacks is far from a consummate science. Several techniques like rate limiting, packet filtering and tweaking software parameters help in limiting DoS attacks, but only in situations where the DoS attacks consume fewer resources than are available.

Distributed Denial of Service (DDoS) attacks have been around for many years and will probably still be present for many to come. Over the years the black hat community (the "hackers") has increased the sophistication and potency of their DDoS attacks [1]. We have seen an increase in the number of nodes and a decrease in the time it takes to install a DDoS agent once a computer has been compromised [2]. Pre-scanned targets are now becoming more common.

CDMA is a type of distributed DDOS attack, where the attacker uses facilitators or compromised systems in a coordinated way. By distributing the attack and varying the type of attack the source attacker exhibits a decreased intensity of activity; therefore, the attack becomes harder to detect [1]. Meanwhile, the concentrated effect on the victim is sufficient to overload network peripherals and systems, resulting in denial of service [1].

As CDMA is a variation of attack, data was not available from standard locations. Attack data was generated in a controlled environment at New Mexico Tech Information Assurance Laboratory ensuring all real time considerations. The CDMA attack scenario used is shown in Figure 1.

Efforts on how to define and characterize denial of service (DoS) attacks through a collection of different perspectives such as bandwidth, process information, system information, user information, and IP address is being proposed by several researchers [1-6].

This paper attempts to explore to generalize and detect more damaging attacks, ones that are highly synchronized (something we haven't seen in abundance in the wild) and that use a wide attack portfolio. Most DDoS attacks target only a handful of vulnerabilities but we will be attacking many vulnerabilities simultaneously. Once we have shown that a highly blended attack can be successful, meaning it consumed the victim's resources; we will explore how blended the attacks can be. One concern is that the overhead of orchestrating the attacks may become too large or that the attacks are too blended to be effective.

Section two covers the aspects of coordinated distributed multiple attacks. The methodology is given in section three. Brief description of kernel methods used is given in section four. The results of our experiments are

given in section five which is followed by our conclusions with a brief discussion of the work.

2. CDMA Attack Schemes

CDMAs are taking traditional DDoS attacks a step further. Instead of a single attack we target multiple vulnerabilities using a diverse selection of protocols as well as varying the attacks over time. The attack portfolio allows us to strike a larger target area (i.e., more vulnerabilities) than a target with just one vulnerability. This coupled with the constantly changing source makes CDMAs hard to detect and block. Even if most of the attacks are blocked, some may still bypass the target's defenses [7].

In this attack (a schematic diagram is given below as figure 1), a number of compromised systems are used as facilitators in a coordinated manner to launch an attack on a victim's host or network.

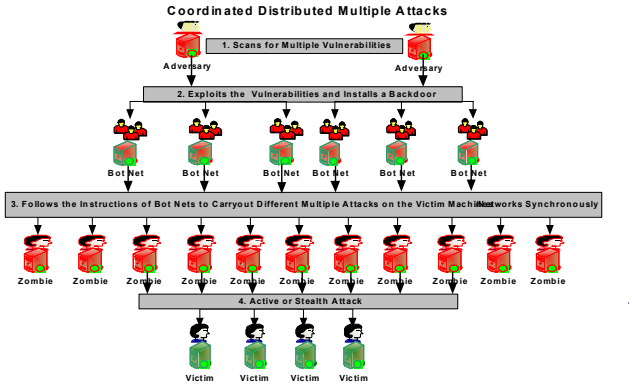

Figure 1: CDMA schematic

A very simple example of a CDMA would be a multi-vector attack. Each attacking node has a different type of attack (whether it is a different vulnerability or a different protocol) that is used throughout the duration of the overall attack.

Multiple attack vectors allow for a large target area on the victim which potentially gives the attack a greater chance of succeeding. The attack vectors are statically assigned and do not change over the duration of the attack.

If the attack vectors are distributed over time, the attack constantly changes but only targets one vulnerability at a time.

Time division attacks are potentially harder to detect because each individual attack type (syn flood, udp flood, etc) is relatively short and could be mistaken for a network anomaly. Such an attack requires a higher level of synchronization than traditional DDoS attacks or the multi-vector attacks.

By combining both the multi-vector attack and the time division attack we get an attack that is highly blended with respect to the attack vector and time. We call this type of attack the "checkerboard" attack because of the grid-like pattern it depicts.

Checkerboard attacks are constantly targeting a wide selection of vulnerabilities and each individual node is continually changing the type of attack it sends. This allows for a very robust attack as well as capitalizing on the stealthy nature of the time division attacks. In this paper we primarily study this type of attack.

In table 1 below are listed the attributes of both traditional DDoS attacks and CDMAs. This illustrates the differences between the two attack schemes.

Table 1: Traditional and CDMA attributes

Traditional DDoS	CDMA
Single attack vector	Attack portfolio
Single protocol	Multi-protocol
Single use of bandwidth	Multiple bandwidth uses muxed together
Single target, many-to-one	Many-to-one or many-to-many targeting

Highly blended attacks such as the checkerboard attack pose a few problems for the instigator. Will such a blended attack be effective? The attack vectors may be so diverse and split into such small time segments that they fail to affect the victim node sufficiently. Synchronization between nodes can also be an area of concern if the attack relies on precise timing. It is very unlikely that the network delay between each attacking node and the victim will be constant or consistent and the attacker should expect even more erratic behavior during an attack as various network buffers become exhausted. These are some of the problems we hope to explore in this paper.

3. Methodology

To evaluate the practicality of CDMAs a testing lab was built in which several experiments were run. Our goal was to determine whether or not highly blended attacks were possible and if so, just how effective were they upon delivery. We accomplished this by subjecting a victim node to several attacks (both traditional DDoS and CDMA) and measured various system parameters such as network performance and memory usage.

An idealistic testing environment consisted of several hundred attacking nodes and a victim node connected to the Internet. From here we could study how attack traffic was affected by the random nature of the Internet. Unfortunately, such a setup was beyond our means. Our solution was to approximate the idealistic layout by using two separate computer laboratories on New Mexico Tech's campus and use the Internet between them as the connecting "cloud". This allowed us to have a controlled and manageable environment but still introduced random

traffic and unknown network configurations. The layout of our testing environment is given in figure 2 below.

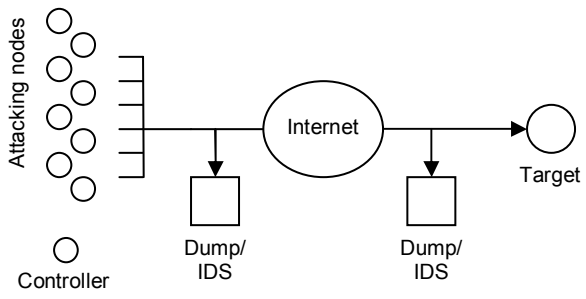

Figure 2: Layout of testing network

As shown in figure 2, Intrusion Detection Systems (IDS) and packet capture programs were installed in between the attacking subnet and the victim subnet. In this paper the IDS is used mainly for validation of the attacks but in future work we hope to use it to explore how stealthy an attack can be. By capturing the packets we can replay them for future tests.

The majority of our experiments attempt to see how well a particular attack affects the victim node, meaning how many resources the attack can consume. To answer this question the victim monitors various performance counters.

Legitimate network packets consume various kinds of shared resources such as bandwidth, memory, processing, and operating system structures. Most of the network peripherals require system and network resources to process the information passing by through the network. An adversary identifies a few activities that are resource intensive and targets the devices with such activity making rendering it nonfunctional. A few possible scenarios of resource exhaustion involve the following: are buffers, file descriptors, address space, disk space, CPU cycle, and bandwidth.

- **IPv4:** The number of datagrams discarded (sent and received) as well as the number of datagrams sent or received are recorded.
- **Main memory:** Page faults, page reads, cache bytes, cache limit, and the number of committed bytes are all recorded.
- **Network interface:** The number of packets sent and received, bytes sent or received per second, the number of packets discarded, and the network interface's output queue length are all recorded.
- **Physical disk:** The current disk queue length, bytes written or read per second, and the number of reads or writes per second are recorded.
- **Processor:** The victim's CPUs are monitored for interrupts and the distribution of processor time (user time, privileged time, etc).

These performance counters present a breadth view of the node's health. Some attacks affect different resources and by monitoring the system as a whole we can see how combined attacks affect critical system resources.

3.1 Attack programs

Our initial experiments used eight attack programs. These programs were found "in the wild", meaning we collected them from several black hat websites. These attacks were chosen because they contained the original source code and were easy to use. All of these programs use a command line interface, which made it easy to write scripts to control them.

The list of attacks used in this paper is given below in table 2.

Table 2: attack programs used

Program	Protocol	Spoofed	Attack type
Ath	ICMP	Yes	Bad data
Bomba	IGMP	No	Oversized packet
Bonk	UDP	Yes	Bad offset
jolt2	ICMP, UDP	No	Fragmentation
Kod	IGMP	No	Fragmentation
Smurf	ICMP, UDP	Yes	Ping flood
Suf	UDP	Yes	UDP flood
Syn	TCP	Yes	Syn flood

To validate the attack scripts several calibration attacks were performed on the victim. Snort was used as the IDS and was placed before the sending nodes and before the victim. If an attack was valid it must (1) be detected by the IDS and (2) consume some resource on the victim. All eight scripts were launched from all of the attacking nodes in a traditional DDoS attack.

3.2 Control code

The control code acted as the glue that held all the attack programs together. Because we did not have a complex routing scheme amongst the attacking nodes a simple client-server relationship was sufficient.

All control information was sent using UDP. Although this is an unreliable protocol and loss may occur (especially during heavily congested conditions such during an attack), we were able to determine that losses were quite rare in our experiments. All control information was sent while the attacking nodes were idling. Since we used a lab environment, regular corruption problems were not a concern. UDP was chosen because it is quick and easy to implement

Node synchronization

One problem faced by the controller was synchronizing all the attacks. This was accomplished by first synchronizing all nodes to the controller and then instructing them to attack at a set time. The controller accomplished this by sending out its current time to all nodes from which the nodes computed a time offset by converting from local time to controller time.

Once the clocks were synchronized the controller informed the nodes of an absolute starting time which was a few seconds in the future. Each node received the same starting time (relative to the controller's clock) and began to count down until the attack. This allowed us to avoid most of the congestion and have a reasonably synchronized attack.

4. Attack Efficacy Results

The CDMA experiment used in these comparisons was a checkerboard attack with the attacks changing every second. This illustrates a heavily interleaved attack with a large number of attack vectors.

Our experiments were focused on exploring the feasibility of CDMAs. We measured the performance of the victim machine during the calibration experiments (using traditional DDoS attacks) and compared this to the performance during a full checkerboard attack. Figure 3 below shows the processor usage on the victim during a highly blended attack compared to the average processor usage for the traditional DDoS attacks. Figure 4 shows the no of bytes received by the victim's machine.

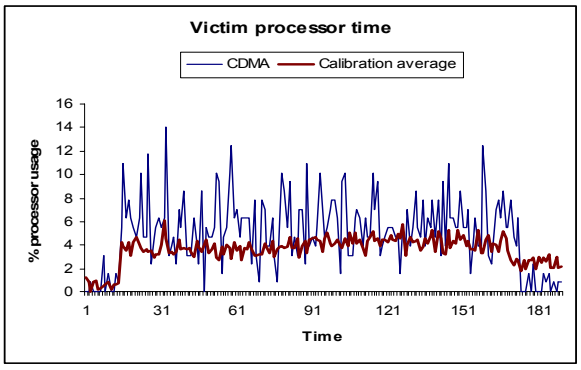

Figure 3: Victim processor time

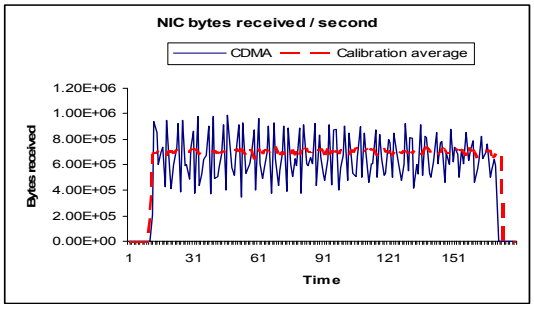

Figure 4: Victim bytes received

Figure 5 shows CDMA interrupts. Figure 6 shows the output queue length. Figure 7 shows the output queue length for 1 second. Figure 8 shows the output queue length for 16 seconds.

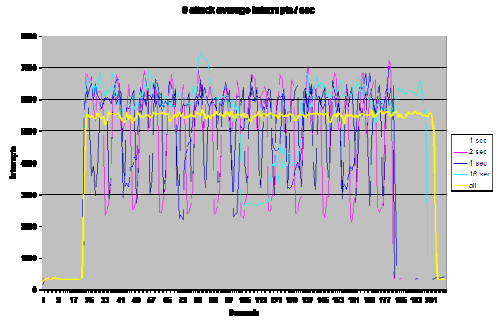

Figure 5: Interrupts CDMA vs. average for all eight

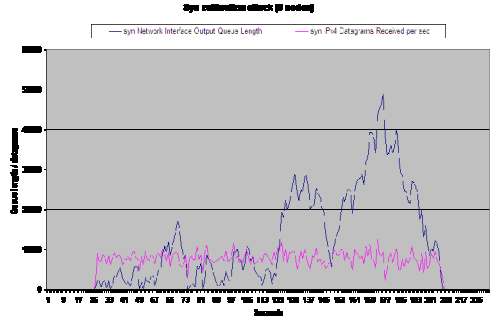

Figure 6: Output queue length calibration syn

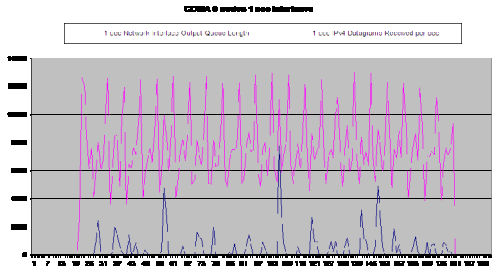

Figure 7: Output queue length CDMA 1 sec

The other performance counters followed the same trend as the processor usage: highly blended attacks are at least if not more effective than traditional DDoS attacks. One critical resource the attacker must take into consideration is bandwidth.

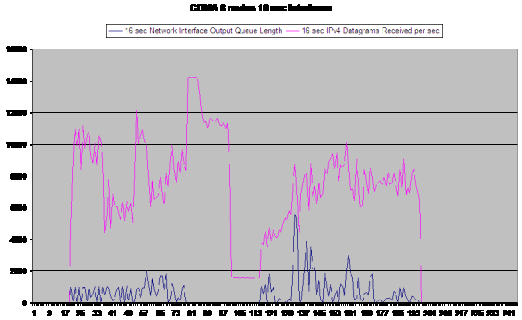

Figure 8: Output queue length CDMA 16 sec

A low bandwidth attack that is just as damaging as a high bandwidth attack is more desirable. Figure 8 below shows that CDMAs use (on average) slightly less bandwidth than their traditional counterparts, despite periodic spikes. Given the complex nature of the new attack this is a very desirable attribute.

The attack portfolio used in this paper utilized only one TCP attack (syn flood) and relied on other protocols to do most of the work. In a real-world network one would expect to see a significant amount of TCP traffic and using a larger proportion of TCP attacks could potentially cause more damage than other protocols. This deficiency in the experiments was particularly noticeable when the victim's output queue length (the number of datagrams waiting to be sent) during a syn flood was compared to a CDMA attack. During the syn flood the queue was severely affected but managed to recover during the CDMA.

As the time between attack changes decreased the de-synchronization between nodes increased. This was due to normal network delays but future efforts will need to take this into consideration. A "rigid" attack is apt to fail and our future research will explore flexible attack schemes that capitalize on the benefits of CDMAs but are still practical in a real-world scenario.

5. Kernel Methods Used for Analysis

In any predictive learning task, such as classification, both a model and a parameter estimation method should be selected in order to achieve a high level of performance of the learning machine. Recent approaches allow a wide class of models of varying complexity to be chosen. Then the task of learning amounts to selecting the sought-after model of optimal complexity and estimating parameters from training data [8,9].

Within the SVMs approach, usually parameters to be chosen are (i) the penalty term C which determines the trade-off between the complexity of the decision function and the number of training examples misclassified; (ii) the mapping function Φ; and (iii) the kernel function such that $K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$. In the case of RBF kernel, the width, which implicitly defines the high dimensional feature space, is the other parameter to be selected [10,11].

We performed a grid search using 5-fold cross validation. First, we achieved the search of parameters C and γ in a coarse scale and then we carried through a fine tuning into the detection faults proper space. CDMA Model selection results obtained through grid search for SVMs are given in Figure9; BSVMs are given in Figure 10 and Looms models are given in Figure 11.

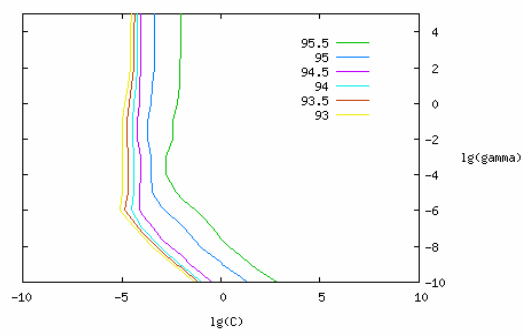

Figure 9: SVM model selection

5.1 Biased Support Vector Machine (B-SVM)

Biased support vector machine (BSVM), a decomposition method for support vector machines (SVM) for large classification problems [8,9]. BSVM uses a decomposition method to solve a bound-constrained SVM formulation. BSVM Uses a simple working set selection which leads to faster convergences for difficult cases and a bounded SVM formulation and a projected gradient optimization solver which allow BSVM to quickly and stably identify support vectors. Leave-one-out model selection for biased support vector machines (BSVM) is used for automatic model selection [11].

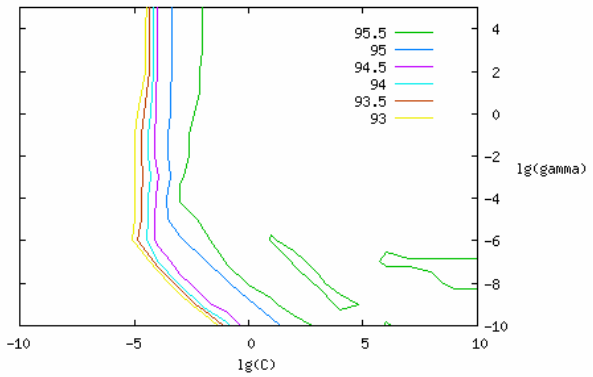

Figure 10: BSVM model selection

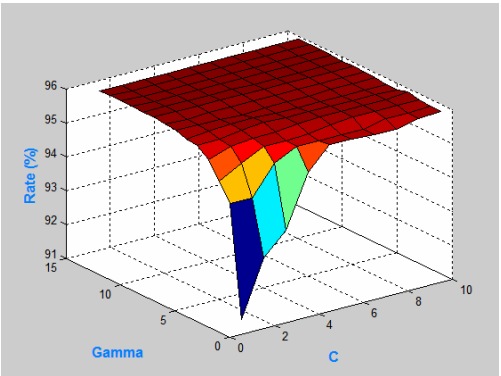

Figure 11: LOOMs model selection

6. Conclusion

One of the goals of this research was to explore the feasibility of launching a highly blended and distributed attack. Our results show that it is very easy to conduct such an attack using simple control code and open source black hat tools.

During the experiments, we observed an increase in the desynchronization among the attacking nodes and can predict that the desynchronization will continue to rise as the attacks become more and more blended. Future attack schemes will need to be resilient to variations among the nodes, perhaps only synchronizing the nodes belonging to the same subnet.

Running only eight attack programs on eight different nodes forced us to spend several days working out the operating system and hardware configurations. Each protocol may be blocked at the node or at the router and a large number of mis-configured nodes can render the attack ineffective.

SVMs easily achieve high detection accuracy (higher than 95%) BSVM performs the best for detecting CDMA. Model selection results using Leave-one-out model selection for support vector machines (looms) based on BSVM are presented in (Figure 11). A grid search for CDMA using SVM and BSVMs (Figures 9 and 10) which seeks the optimal values of the constraint penalty for method solution and the kernel width (C,γ) has been performed. We demonstrate that the ability with which SVMs can classify CDMA attacks is highly dependent upon both the kernel type and the parameter settings.

Acknowledgements

Support for this research received from ICASA (Institute for Complex Additive Systems Analysis, a division of New Mexico Tech), a DOD IASP, and an NSF SFS Capacity Building grants are gratefully acknowledged. We would also like to acknowledge many insightful discussions with Dr. Jean-Louis Lassez that helped clarify our ideas.

References

[1] W. J. Blackert, D. C. Furnanage, and Y. A. Koukoulas, "Analysis of Denial of Service Attacks Using an Address Resolution Protocol Attack", Proc. of the 2002 IEEE Workshop on Information Assurance, US Military Academy, pp. 17-22, 2002.

[2] T. Draelos, et. al, "Distributed Denial of Service Characterization," *Technical Report*, Sandia National Laboratories, 2003.

[3] J. Mirkovic and P. Reiher, A Taxonomy of DDoS Attacks and Defense Mechanisms, ACM SIGCOMM Computer Communications Review, Volume 34, Number 2, April 2004, pp. 39-54.

[4] D. W. Gresty, Q. Shi, and M. Merabti, "Requirements for a general framework for response to distributed denial of service," Proc. Of Seventeenth Annual Computer Security Applications Conference, pp. 422-229, 2001.

[5] K Houle, G. Weaver, "Trends in Denial of Service Attack Technology", CERT Coordination Center, 2001, http://www.cert.org/archive/pdf/DoS_trends.pdf

[6] S. Staniford, V. Paxson, N. Weaver, "How to Own the Internet in Your Spare Time", Proceedings of the 11th USENIX Security Symposium, 2002.

[7] S. Mukkamala, A. H. Sung, "Coordinated Distributed Multi Attacks (CDMA)", Proc. of IEEE International Conference on Advances in Intelligent Systems Theory and Applications, IEEE Computer Society Press, ISBN 2-9599776-8-8, PID 011-04.

[8] O. Chapelle, V. Vapnik, "Model selection for support vector machines", Advances in Neural Information Processing Systems 12, 1999.

[9] V. Cherkassy, "Model complexity control and statistical learning theory", Journal of natural computing 1: (2002) 109–133.

[10] N. Cristianini, J. S. Taylor, "Support Vector Machines and Other Kernel-based Learning Algorithms", Cambridge, UK: Cambridge University Press, 2000.

[11] C. C. Chang, C. J. Lin, "LIBSVM: a library for support vector machines", Department of Computer Science and Information Engineering, National Taiwan University, 2001.