The Third International Conference on Availability, Reliability and Security

Detection of malcodes by packet classification

Irfan Ahmed, Kyung-suk Lhee
Digital Vaccine and Internet Immune System Lab,
Graduate School of Information and Communication, Ajou University, Korea
{irfan, klhee} @ajou.ac.kr

Abstract

In this paper, we propose an anomaly detection approach that classifies packets into code-type and data-type. Our objective is to detect a packet containing codes flowing into a network port, which normally expects data packets only. The proposed approach can detect potentially malicious packets such as worms, viruses, and shellcodes. We propose a time-efficient algorithm and show the results of our initial experiments.

1. Introduction

Intrusion detection systems (IDS) face with known attacks and unknown attacks. Many IDS are signature-based, which can detect known attacks efficiently such as Snort [1] and Bro [2] but cannot detect unknown attacks effectively. Anomaly detection approaches, on the other hand, can better deal with unknown attacks. Anomaly detection approaches model normal (expected) behavior of the system and anything that largely deviates from the normal behavior is considered as anomalous, which may arise from attempted attacks.

Some attacks such as probing a network or exploiting vulnerabilities of network protocol can be detected by analyzing packet header information or traffic analysis. But the attacks exploiting program vulnerabilities such as malcodes (worms or viruses) may not be handled by just using packet header information. Such attacks may be better detected by analyzing the packet payload.

In this paper, we propose an anomaly detection approach that classifies the packets into code-type and data-type. Our objective is to identify a packet containing codes flowing into a network port, which normally expects data packets only. For example, a web client usually receives html files, documents, or proposed approach can detect potentially malicious multimedia data such as images and sounds (except for a few occasions like ActiveX programs). The code packets (such as worms, viruses, and shellcodes that exploit buffer overflow vulnerabilities) and alert web users.

Our scheme analyzes byte frequency distributions from a large set of program files (frequency vectors), and produces a feature vector that summarizes those information. We then calculate frequency vectors from a large set of code and data packets. Using the frequency vectors (of code / data packets) and the feature vector (of program files), we calculate a set of Mahalanobis distance (MD) values. The Mahalanobis distance values derived from code packets are lower than those derived from data packets, so we can determine a threshold that distinguishes code packets and data packets.

The rest of paper is organized as follows. Section 2 presents related work. Section 3 describes the proposed approach, the challenges faced by this approach, and the augmented approach that deals with the problem. Section 4 shows experiment results. Section 5 presents conclusions and our future work to enhance the proposed approach.

2. Related work

Statistical approach is used by many anomaly detection systems such as NIDES [3], SPADE [4], ALAD [5], PHAD [6, 7, 8], and NATE [9], but they rely on the packet header information rather than packet payload.

MADMID [10], EMERALD [11,12], STAT [13] extract and reconstruct information such as bytes transferred and session duration. Some approaches use payload information but in a very limited way. For example, NETAD [14] uses first 48 bytes of packet for feature selection. However, since 40 bytes are required for TCP and IP headers, it uses at most 8 bytes of payload. There are approaches that use payload information [15, 16, 17, 18], but do not use byte frequency distribution.

Our approach employs n-gram and byte frequency distribution, similar to Wang and Stolfo [19, 20]. Kruegel et al. [21] also used byte frequency distribution, but rather than using full byte frequency distribution, they used six ranges (0, 1-3, 4-6, 7-11, 12-15, and 16-255) of byte distribution. Wang and

*This research is supported by the Ubiquitous Computing and Network (UCN) Project, the Ministry of Information and Communication (MIC) 21st Century Frontier R&D Program in Korea.

0-7695-3102-4/08 $25.00 © 2008 IEEE
DOI 10.1109/ARES.2008.100

Stolfo [20] derive multiple models of the normal behavior according to each network port and packet size. Their approach also needs to adapt to the changes in the data stream on each port. In contrast, our approach is not port or packet size-specific, and is much less sensitive to the changes in the data stream on the network.

3. The proposed approach

3.1. Basic scheme

To identify codes in a packet, we initially tried decoding the packet to find valid machine instructions. However, since every bye pattern maps to a valid x86 opcode, it is quite difficult to distinguish a valid instruction this way [22]. Therefore, we opt for a statistical and data-mining approach.

During the learning phase, our scheme analyzes n-byte frequency distributions from a large set of program files (frequency vectors). An n-gram [23] is the sequence of n adjacent bytes in packets or files. A sliding window with width n is passed over to whole payload or file and occurrence of each n-gram is counted (Figure 1 shows examples of 1-gram frequency vectors). From frequency vectors we calculate a feature vector (mean and standard deviation of each n-byte pattern frequency). We use program files in generating feature vectors in order to capture the generic program pattern.

We then calculate frequency vectors from a large set of code and data packets. Using the frequency vectors (of code / data packets) and the feature vector (of program files), we calculate a set of Mahalanobis distance values. The Mahalanobis distance values derived from code packets are lower than those derived from data packets, so we can determine a threshold that distinguishes code packets and data packets.

During the detection phase, each incoming packet is scanned to compute a frequency vector. Using the frequency vector and the feature vector we compute the Mahalanobis distance. If the MD value is lower than the threshold value, then it is considered a code packet.

In this paper, we initially used "artificial packets" by dividing program and data files into 1500-byte stream (since usual MTU is 1500). Section 4 shows experiment results using real packets. We also used the simplified Mahalanobis distance by Wang and Stolfo [20], which is

$$d(x,\overline{y}) = \sum_{i=0}^{n-1}(|\, x_i - \overline{y_i}\,| / \overline{\sigma_i})$$

,

Where $\overline{y}$ is a mean value, $\overline{\sigma}$ is a standard deviation (from the feature vector), and x is a n-byte pattern frequency of the incoming packet.

3.2. 3-gram and threshold value

An instruction consists of multiple bytes (even an opcode alone may consists of multiple bytes), so byte sequence information would be significant in identifying codes in a packet. Therefore, higher order n-gram would be beneficial.

In this paper, we tried increasing the order until we can clearly distinguish code and data packets. Figure 2 shows that using 3-gram we can distinguish code and data with sufficient accuracy (10,000 packets are used for each type). Table 1 show the optimum threshold value for each type as well as the optimum threshold for all types.

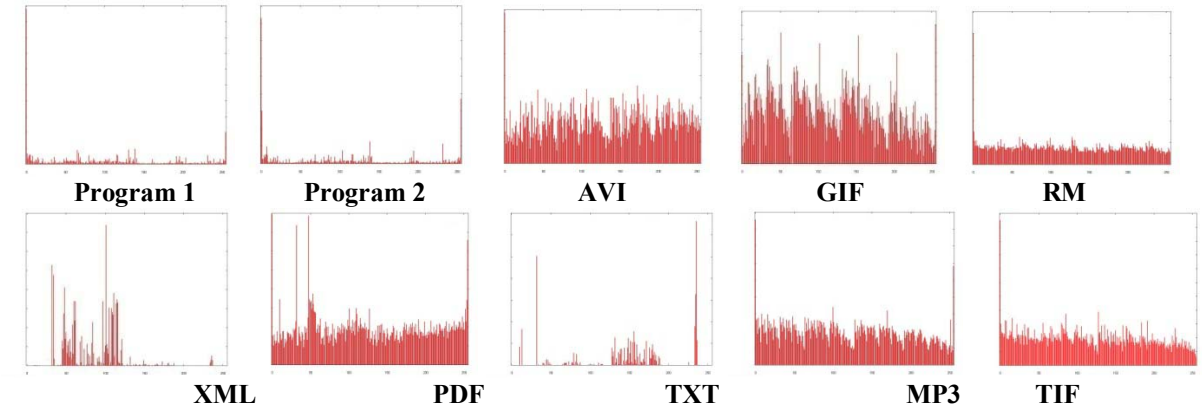


Figure 1. Frequency vectors of program and data files.

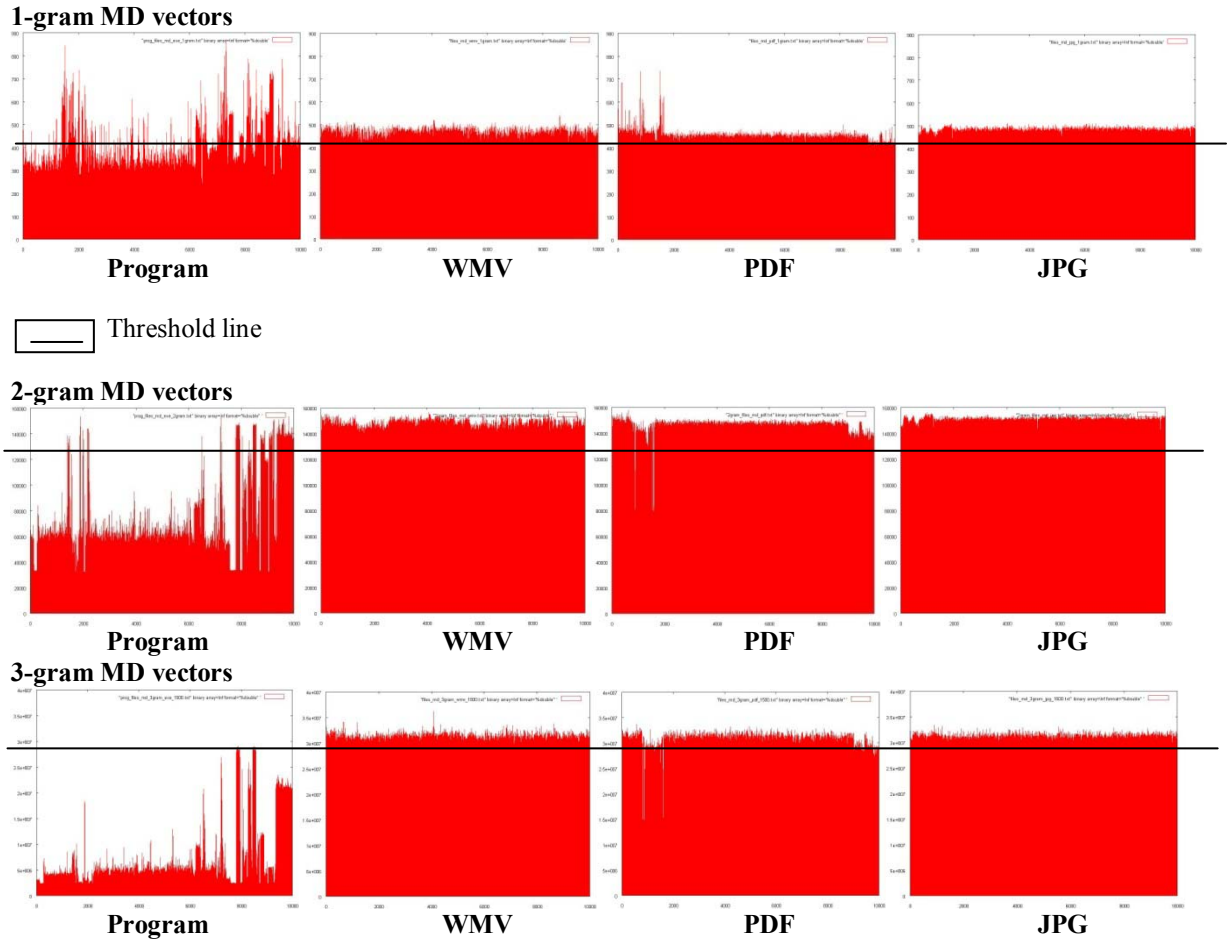


Figure 2: MD vectors of code and multimedia types of packets, using 1, 2, and 3-gram.

	Optimum threshold MD value	Right Hits				False Negatives		False Positives	
		# of program packets	Prog (%)	# of data packets	Data (%)	# of program packets	Prog (%)	# of data packets	Data (%)
JPG	28111620.0	9950	99.5	9889	98.89	49	0.49	111	1.11
MP3	26239000.0	9747	97.47	9960	99.6	252	2.52	40	0.4
PDF	21689270.0	9657	96.57	9603	96.03	342	3.42	397	3.97
WMV	27684800.0	9885	98.85	9934	99.34	114	1.14	66	0.66
All Files	21689270.0	38632	96.58	39518	98.80	1368	3.42	482	1.20

Table 1. Packet classification statistics from code and data packets.

3.3. Problem in the basic scheme

While 3-gram was sufficient to distinguish code packets from multimedia packets, the basic scheme cannot distinguish code packets from text-type packets even if we increase n. As shown in Figure 3, MD values derived from text-type packets are close or lower than those derived from code packets.

We conjecture that this problem is due to the difference in the number of unique n-byte patterns. Table 2 shows that while program and multimedia files have large number of unique 3-byte patterns, text-type files have substantially less number of unique 3-byte patterns.

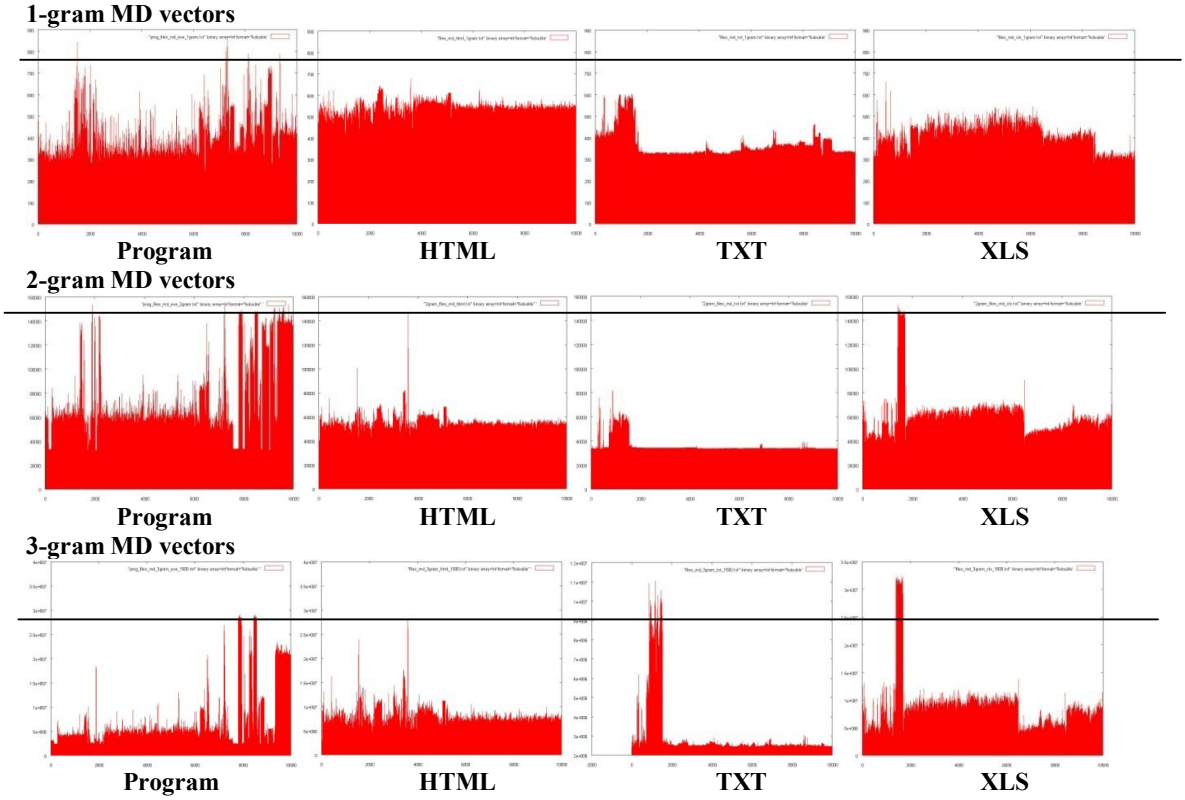


Figure 3. MD vectors of code and text-types of packets, using 1, 2, and 3-gram.

Files Classification	File Type	Number of 3-gram permutations in files (A)	B (%) = (A / total number of 3-gram permutations× 100)
Text-type Files	XML	13926	0.08
	HTML	33845	0.20
	TXT	54427	0.32
	XLS	692463	4.13
	DOC	4654411	27.74
Multimedia Files	TIF	6701610	39.94
	PPS	7338333	43.74
	PDF	9874294	58.86
	JPG	16191234	96.51
	MP3	16425993	97.91
	WMV	16661987	99.31
	RM	16774686	99.98
	AVI	16777216	100.00
Program File	Program	16654060	99.26

Table 2. The number of unique 3-byte patterns in various types of files.

3.4. Dual threshold testing

The problem of the basic scheme is that it can only distinguish multimedia packets vs. code or text-type packets. To distinguish code vs. text-type packets, our solution in this paper is to use another threshold. Specifically, we derive a feature vector from text-type files, and derive a set of Mahalanobis

distance values using this feature vector and the frequency vectors (of code / data packets). From this Mahalanobis distance values we derive the second threshold that distinguish codes and text-type data.

The detection algorithm is now as follows. We first calculate an MD value of the incoming packet using the feature vector of program files. If the MD value is greater than the first threshold, then it is considered a multimedia packet. Otherwise, we calculate another MD value using the feature vector of text-type files. If the MD value is greater than the second threshold, then it is considered a code packet. Figure 4 shows a flowchart of our scheme.

Figure 5 show that using 3-gram we can distinguish code vs. text-type packet with adequate accuracy. However, as Table 3 shows, the false negative rate is much higher than that of the basic scheme. In Section 5, we discuss possible solutions to improve the accuracy as our future work.

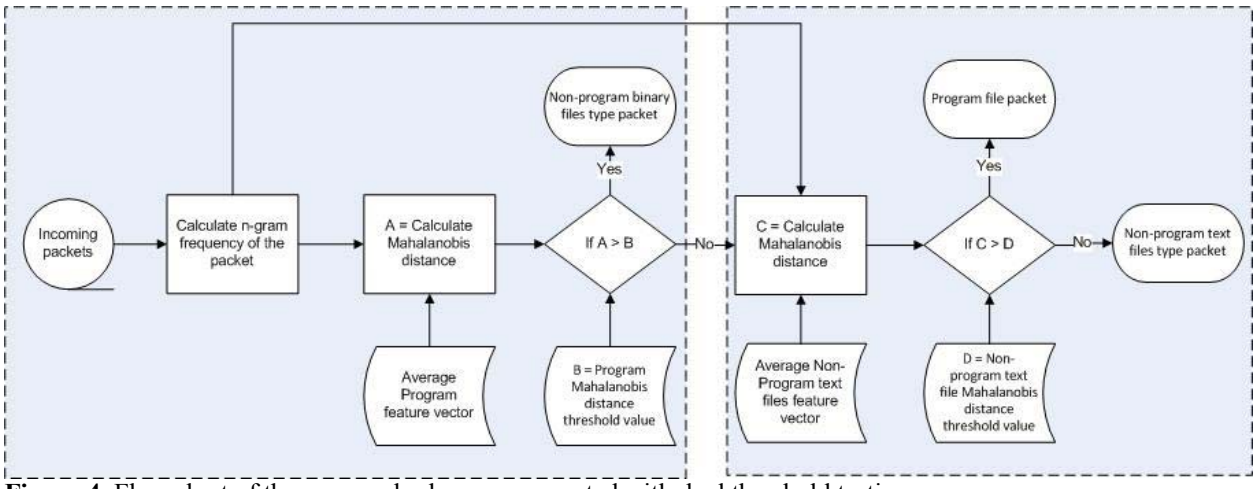


Figure 4. Flow chart of the proposed scheme, augmented with dual threshold testing.

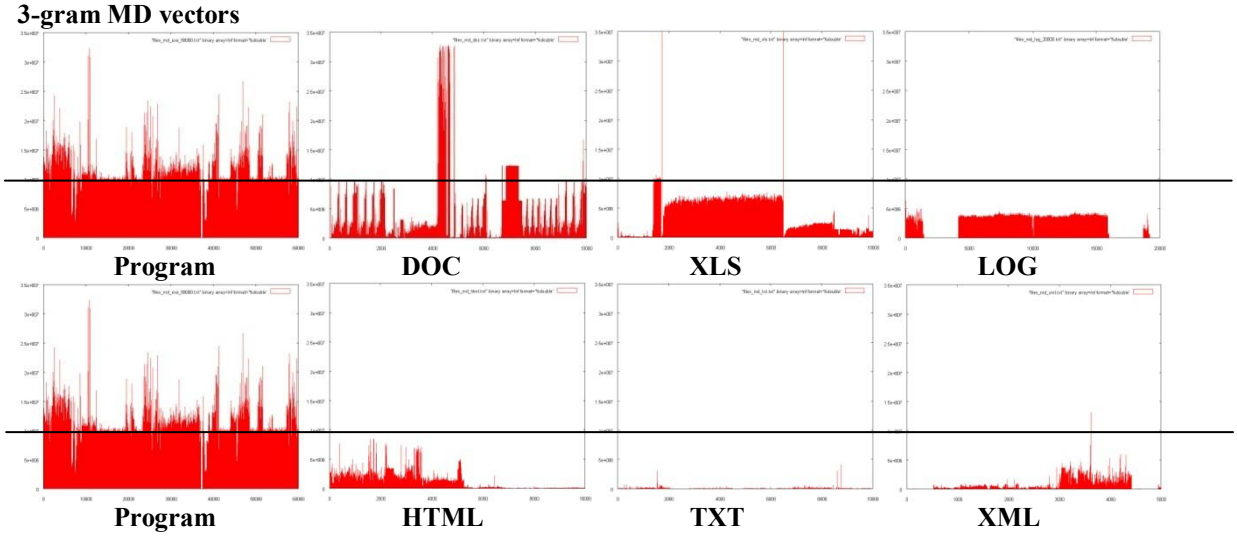


Figure 5. MD vectors of code and text-types of packets, using the feature vector of text-type files.

	Threshold MD Value	Right Hits				False Negatives		False Positives	
		Program	Prog (%)	Non Prog	Non Prog(%)	Program	Prog (%)	Non Prog	Non Prog(%)
TXT	394462.6	98123	98.123	99780	99.78	1877	1.877	220	0.22
XML	3655786.00	91195	91.195	99280	99.28	8805	8.805	720	0.72
HTML	5504284.00	91344	91.344	97990	97.99	8656	8.656	2010	2.01
LOG	4198904.00	90487	90.487	99855	99.855	9513	9.513	145	0.145
XLS	6924106.00	85608	85.608	96290	96.29	14392	14.392	3710	3.71
DOC	6708201.00	86456	86.456	86770	86.77	13544	13.544	13230	13.23
All Files	6521949.00	51838	86.39714	58140	96.9	8162	13.6029	1860	3.1

Table 3. Packet classification statistics from code and data packets using the feature vector of text-type files.

3.5. Time overhead of higher-order n-gram

As we increase the order of n-gram (up to a certain point), the result will be more accurate. However, the time overhead may also increase if we are not careful. We use n-dimensional array to store n-byte patterns. Although memory space will increase exponentially as we increase n, it may not be critical if we use 64-bit architecture. However, a naïve algorithm that iterates through the array will increase the run-time exponentially as well, which is critical because the proposed scheme operates on every packet and therefore needs to be fast.

By iterating through the packet rather than iterating through the array, we can keep the run time independent from the size of n. That is, the run-time depends on the packet size, not the total number of n-gram permutations. A naïve algorithm and a fast algorithm are shown below.

3.5.1 Naïve Algorithm

Pre-computed Values for Mahalanobis Distance calculation:

$\overline{y}$ ← Mean values taken from normal program files

$\overline{\sigma}$ ← Standard deviation values taken from normal program files

During Mahalanobis distance calculation:

a. Calculate the byte count of 3-gram of unknown packet
   PACKET_SIZE ← *length*(packet)
   for j ← 0 ***to*** PACKET_SIZE ***downto*** 2
   do count(3-gram byte pattern on j^{th} position)

b. Calculate the byte sequence frequency and Mahalanobis distance

for i ← 0 ***to*** 255
  for j ← 0 ***to*** 255
    for k ← 0 ***to*** 255
} *main cause of delay*

$$MD \leftarrow MD + (|x_{i,j,k} - \overline{y_{i,j,k}}| / \overline{\sigma_{i,j,k}})$$

MD → Mahalanobis Distance value

x is a n-byte pattern frequency of the incoming packet.

3.5.2 Fast Algorithm

Pre-computed Values for Mahalanobis Distance calculation:

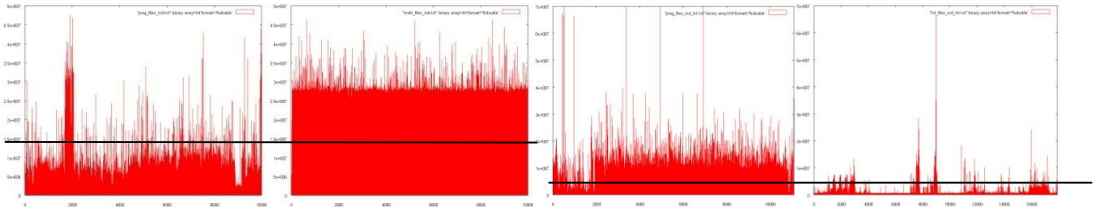
$\overline{y}$ ← Mean values taken from normal program files

$\overline{\sigma}$ ← Standard deviation values taken from normal program files

T ← Total sum of the ratio of mean ($\overline{y}$) and standard deviation ($\overline{\sigma}$) taken from normal files

During Mahalanobis distance calculation:

a. Calculate the byte count of 3-gram of unknown packet
   PACKET_SIZE ← *length*(packet)
   for j ← 0 ***to*** PACKET_SIZE ***downto*** 2
   do count(3-gram byte pattern on j^{th} position)



(a) Program **(a) Multimedia packets** **(b) Program** **(b) Text-type packets**

Figure 6. Part (a) and (b) show the dual threshold testing on real packets of program, multimedia, and text-type packets.

Part (a)		Part (b)	
False Positives (%)	**False Negatives (%)**	**False Positives (%)**	**False Negatives (%)**
2.37	5.97	5.056	18.24

Table 4. False positive and negative rates.

b. Calculate the byte sequence frequency and Mahalanobis distance

Initialize MD ← 0 E←0
for j ← 0 ***to*** PACKET_SIZE ***downto*** 2 } *less iteration*

$$MD \leftarrow MD + (|x_j - \overline{y_j}| / \overline{\sigma_j})$$

$$E \leftarrow E + (\overline{y_j} / \overline{\sigma_j})$$

End for
MD ← MD + T – E

4. Experiments with real packets

Experiments were done on real packets by extracting payloads from the captured packets. We used an FTP server to transfer program and data files. During the file transfer, packets were captured using tcpdump [24] . Tcptrace [25] is used to process packets captured in tcpdump files. The multimedia types we used are are PDF, AVI, RM, WMV, JPG, MP3, TIFF, GIF, and the text-types we used are HTML, XLS, DOC, TXT, XML.

Figure 6 shows the results, and Table 4 shows the false positive and negative rates. We used the same feature vectors threshold values used in Table 1 and 4 to obtain the results. The results are somewhat different from our initial observation using artificial packets. The false positive / negatives are generally higher than before, which is partly because that the threshold value is not calibrated for the real packets.

5. Conclusions and future work

We proposed an anomaly detection approach that uses n-gram and Mahalanobis distance to identify packets containing potentially malicious codes. Our scheme seems effective in distinguishing code packets from multimedia packets, but is less effective in distinguishing code packets from text-type packets. We analyzed why text-type contents are difficult to distinguish, and suggested the difference in the number of unique n-byte patterns as a possible cause. We showed the dual threshold testing approach as our initial attempt to increase the accuracy.

For our scheme to be practical, we need to reduce the false positive / negative level further. We plan to pursue two directions as our future work. First is to use a multivariate model that includes the Mahalanobis distance metric and the number of unique n-byte patterns. Second is to modify our basic scheme such that only a significant subset of unique n-byte patterns is considered in calculating MD values. The rationale is that insignificant n-byte patterns might act as noises in computing MD values.

6. References

[1] Jack Koziol, "Intrusion Detection with Snort", SAMS, 2nd Edition of Book, May 2003.

[2] Vern Paxson, "Bro: A System for Detecting Network Intruders in Real-Time", In proceedings of 7th USENIX Security Symposium, San Antonio, Texas, 1998.

[3] Harold S. Javitz, Alfonso Valdes, "The NIDES statistical component: Description and Justification" Technical report, SRI International, Computer Science Laboratory, 1993.

[4] J. Hoagland, "SPADE", Silicon Defense, http://www.silicondefense.com/software/spice, 2000.

[5] Matthew V. Mahoney, Philip K. Chan, "Learning Nonstationary Models of Normal Network Traffic for Detecting Novel Attacks", In proceedings of SIGKDD, 2002, pp. 376-385.

[6] Matthew V. Mahoney, Philip K. Chan, "PHAD: Packet Header Anomaly Detection for Identifying Hostile Network Traffic", Technical report, Florida Institute of Technology CS-2001-4, April 2001

[7] J. Bhawnani, "Design and Implementation of an Anomaly-Based Intrusion Detection System using Statistical Analysis of Network Traffic", Master's Thesis, State Univeristy of New York at Binghamton, May 2003.

[8] Victor A. Skormin, Douglas H. Summerville, James S. Moronski, James L. Sidoran "Application of Genetic Optimization and Statistical Analysis for Detecting Attacks in a Computer Network", In proceedings of the real-time Intrusion Detection NATO Symposuim, Lisbon, Portugal, May 2002, pp.27-29.

[9] Carol Taylor, Jim Alves-Foss, "NATE – Network Analysis of Anomalous Traffic Events, A Low-Cost approach", New Security Paradigms Workshop, 2001, pp. 89-96

[10] Matthew V. Mahoney, Philip K. Chan, "An Analysis of the 1999 DARPA/Lincoln Laboratory Evaluation Data for Network Anomaly Detection", RAID 2003, 2003, pp. 220-237.

[11] Phillip A. Porras, Peter G. Neumann, "EMERALD: Event Monitoring Enabled Responses to Anomalous Live Disturbances", National Information Systems Security Conference, 1997.

[12] Peter G. Neumann, Phillip A. Porras, "Experience with EMERALD to Date," In proceedings of the 1st USENIX Workshop on Intrusion Detection and Network Monitoring, Santa Clara, CA, April 11-12,1999, pp. 73-80

[13] Matthew V. Mahoney, Philip K. Chan, "Learning Models of Network Traffic for Detecting Novel Attacks", Florida Tech, Technical report 2002-08.

[14] Matthew V. Mahoney, "Network Traffic Anomaly Detection Based on Packet Bytes," In Proceedings of 18th ACM Symposium of Applied Computing, 2003, pp. 346-350.

[15] Thomas Toth, Christopher Kruegel, "Accurate Buffer Overflow Detection via Abstract Payload Execution," In 5th Symposium on Recent Advances in Intrusion Detection (RAID), LNCS, Springer Verlag, Switzerland, October 2002, pp. 274-91.

[16] Janak J. Parekh, Ke Wang, Salvatore J. Stolfo, "Privacy-Preserving Payload-Based Correlation for Accurate Malicious Traffic Detection", In proceedings of Sigcomm, 2006, pp. 99-106.

[17] Douglas H. Summerville, Nnamdi Nwanze, Victor A. Skormin, "Anomalous Packet Identification for Network Intrusion Detection", In workshop of Information Assurance United States Military Academy, West Point, NY, June 2004, pp. 60-67

[18] Nnamdi Nwanze, Douglas H. Summerville, Victor A. Skormin, "Real-Time Identification of Anomalous Packet Payloads for Network Intrusion Detection", In workshop of Information Assurance United States Military Academy, West Point, NY, June 2005, pp. 448-449

[19] Ke Wang, Janak J. Parekh, Salvatore J. Stolfo, "Anagram: A Content Anomaly Detector Resistant to Mimicry Attack", RAID 2006, LNCS , pp. 226 - 248.

[20] Ke Wang, Salvatore J. Stolfo, "Anomalous Payload-based Network Intrusion Detection", RAID 2004, pp. 203-222

[21] Christopher Kruegel, Thomas Toth, Engin Kirda, "Service Specific Anomaly Detection for Network Intrusion Detection", In Symposium on Applied Computing (SAC), Spain, March 2002, pp. 201-208

[22] Christopher Kruegel, Engin Kirda, Darren Mutz, William Robertson, Giovanni Vigna, "Polymorphic Worm Detection Using Structural Information of Executables", In RAID 2005, 207-226

[23] Marc Damashek, "Gauging similarity with n-grams: language independent categorization of text", Science 267(5119), 1995, pp. 843-848.

[24] http://www.tcpdump.org

[25]http://www.tcptrace.org