INL/CON-13-28639
PREPRINT

Real-Time SCADA Cyber Protection Using Compression Techniques

Technologies for Homeland Security

Lyle G. Roybal
Gordon H. Rueff

November 2013

Real-Time SCADA Cyber Protection Using Compression Techniques

Lyle G. Roybal
Idaho National Laboratory
2525 N. Fremont Avenue
Idaho Falls, ID 83415-3755
Lyle.Roybal@inl.gov

Gordon H Rueff
Idaho National Laboratory
2525 N. Fremont Avenue
Idaho Falls, ID 83415-3544
Gordon.Rueff@inl.gov

Abstract— **The Department of Energy Office of Electricity Delivery and Energy Reliability (DOE-OE) has a critical mission to secure the energy infrastructure from cyber-attack. Through the DOE-OE Cyber Security for Energy Delivery Systems (CEDS) program, Idaho National Laboratory (INL) has developed a method to detect malicious traffic on Supervisory, Control, and Data Acquisition (SCADA) networks using a data compression technique. SCADA network traffic is often repetitive with only minor differences between packets. Research performed at INL indicates that SCADA network traffic has traits desirable for using compression analysis to identify abnormal network traffic. An open source implementation of a Lempel-Ziv-Welch (LZW) lossless data compression algorithm was used to compress and analyze surrogate SCADA traffic. Infected SCADA traffic was found to have statistically significant differences in compression when compared against normal SCADA traffic at the packet level. The initial analyses and results are clearly able to identify malicious network traffic from normal traffic at the packet level with a very high confidence level across multiple ports and traffic streams. Statistical differentiation between infected and normal traffic level was possible using a modified data compression technique at the 99% probability level for all data analyzed. The conditions tested were limited in scope and should be expanded to include more realistic simulations of hacking events using techniques and approaches that are better representative of a real-world attack on a SCADA system. Nonetheless, the use of compression techniques to identify malicious traffic on SCADA networks in real time appears to have significant merit for infrastructure protection.**

Keywords-component; SCADA; Cyber Security; Data Compression.

I. INTRODUCTION

Securing the country's energy sector infrastructure from cyber-attack is critical to the well-being of the American people and is a central focus to the Department of Energy (DOE) Office of Electricity Delivery and Energy Reliability (OE) Cybersecurity for Energy Delivery Systems (CEDS) program [1]. The DOE program aims to enhance the reliability and resilience of the nation's energy infrastructure by reducing the risk of energy disruptions due to cyber-attacks [2]. The purpose INL's work in the SCADA (Supervisory Control and Data Acquisition) Protocol Anomaly Detection Utilizing Compression (SPADUC) project was to investigate if and how data compression algorithms could be used to identify cyber-attacks on SCADA networks. SCADA systems operating over network protocols typically contain several static layers of headers. The ratio of the packet header size to the control or response data is often very high. Moreover, the number of header types is often limited because of the repetitive nature of SCADA communications. Control of hardware and system status of actuators and sensors generally occur in regularly timed sequences. Therefore, network traffic on dedicated SCADA systems and at the boundaries of SCADA systems where data transfer are moved to historian and trending functions, tend to be of a monotonous nature. Because large portions of the transmission control protocol (TCP)/internet protocol (IP) message traffic are repetitive, the concept of using compression techniques to differentiate traffic that is "non-normal" was proposed as a way to identify and quarantine malicious traffic at the packet level before its payload is completed. An open source implementation of a Lempel-Ziv-Welch (LZW) lossless data compression algorithm [3,4] was also proposed to analyze surrogate but typical SCADA network traffic and ascertain if malicious traffic could be differentiated from normal SCADA traffic.

LZW compression is a well-known and mature technology used to compress data of any sort (binary or ASCII). It is the method used by commercial product WinZip in the personal computer world. LZW algorithms compress data by using a dictionary technique to store repetitive sequences of bytes as they occur in the data stream. The dictionary is initialized with a default set of entities that usually consists of the 256 ASCII character set as a starting block. When data are compressed, this dictionary is updated every time a new sequence of characters is encountered. So, for example, in a Word document that contains many repeated words, a dictionary entry can be created for the word "tomorrow." In this manner, "tomorrow," which is 8 bytes (or 64 bits), can be represented by as few bits (say 10 bits for example). The compression of this word is 10/64 and is 16% the size of the original entity.

Initial research showed that SCADA network traffic has traits desirable for using compression analysis as a method to identify abnormal network traffic. Primarily, SCADA network traffic is often very repetitive with only minor differences between packets. This results in highly compressible data streams that demonstrate improved compression ratios in subsequent packets. However, two difficulties in using traditional compression techniques were identified. The first is that inter-dependence in the compression ratio metric between packets violates the use of statistical analysis needed to differentiate safe traffic from infected traffic. The second is that the packets or messages in a given communication channel or port compress at significantly different ratios because message types are of different function, form, and length. This led to the development of an analysis technique that groups SCADA traffic by uncompressed packet size. This provided a basis for comparing like message traffic. This use of data compression appears to have some significant merit for infrastructure protection. The preliminary analyses and results presented herein are clearly able to distinguish malicious network traffic from normal network traffic at the packet level with a very high confidence level for the conditions tested.

II. Data Collection and Surrogate Data

A. *Normal Traffict Data Set*

The normal traffic used for this research was collected from a production SCADA network using a mirror port on the networking equipment. The traffic was sorted into channels by service and client as defined by the server IP address, server port, protocol, and client IP address. The five most verbose of these channels were then selected for further analysis.

B. *Malicious Data Set*

In order to simulate malicious traffic, we decided to concentrate on identifying a particular type of exploit. Based on the hypotheses being tested, exploits that result in payload code execution would be the easiest malicious traffic to identify. Other types of exploits, such as spoofed values or denial of service (DoS) based on invalid fields, would likely not be identifiable using compression ratio analysis of the packet because the packets contents would be too similar to the normal packet contents. Therefore, we inserted metasploit payloads into packets of the otherwise normal traffic.

III. Data Analysis

The original concept was to use compression techniques to "train" a dictionary by compressing data at the packet level and appending the results to a dictionary that was allowed to grow in time. The dictionary would be developed or "trained" using clean or normal SCADA traffic. Then, when a malicious packet was encountered, it would contain unusual byte sequences and compress differently from the norm. As it turns out, there were two problems with this initial proposed methodology. It appears that TCP/IP sessions are re-initialized periodically. When this occurs, the nature of the communications headers appear to change, altering the characteristics of the SCADA network stream. This is illustrated in Figure 1, which shows "compression ratio" for each packet as it arrives in time (sequence) for all the normal SCADA traffic on destination port 5026. Compression ratio is defined as the number of encodes (or dictionary entries) generated during packet

Work sponsored by the Office of Electricity Delivery and Energy Reliability under DOE Contract DE-AC07-05ID14517.

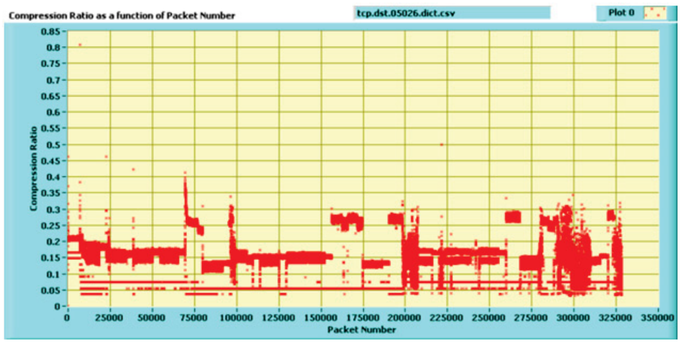

Figure 1. Time sequence of compression ratio for port 5026.

compression divided by the overall packet size. Small compression ratios indicate a packet that was highly compressed by the LZW algorithm.

There are several features that can be extracted from the data presentation in Figure 1. First, the amount of data analyzed is large at over 325,000 packets. Second, although difficult to see in this plot, there is a startup trend in compression ratio where the dictionary is growing with each packet analyzed until the system settles into a compression ratio of around 0.15. Third, the settling in process receives a shock periodically where the compression ratios increase dramatically and later settle in again. These shocks are caused by two factors, we believe. The first factor is the arrival of a batch of packets that are of a new size and message content. The second factor appears to be associated with the establishment of a new session in the TCP client/server relationship. Fourth, it is apparent that the distribution of compression ratios is of a binary nature and not continuous. This is not surprising because packet sizes are an integer value and the number of encodes generated is also an integer value. The binary nature of these distributions becomes more apparent when compressing smaller packets.

After analyzing the data from port 5026, it became apparent that a better way to look at the data was to plot the data as a function of packet size rather than by the sequence in which it arrived. Figure 2 shows the same data as that shown in Figure 1, but plotted as a function of packet byte size. The data group much more naturally when presented this way; however, the session start-up issues identified previously are apparent here by the straight line grouping along many of the packet sizes.

One of our desires for this project was to investigate statistical means to classify normal data traffic against malicious traffic. However, training an LZW dictionary using either good or malicious data violates a fundamental assumption of statistical analysis when attempting to classify data at the packet level using compression ratio as a metric. Statistical analysis requires that each observation is independent unless there is a method to calculate the covariance between observations. In this case, the observation of compression ratio (as defined by the number of encodes divided by the packet size in bytes) is dependent on all previous packets as they have been used to create the dictionary used in calculating encodes.

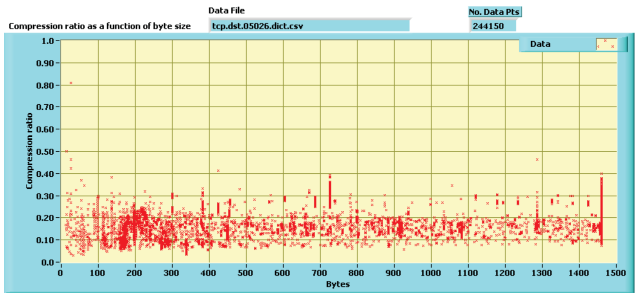

Figure 2. Port 05026 data plotted by packet size.

A. *Static Dictionary Analysis Method*

A static dictionary method was proposed where a fresh or new dictionary was used for each packet in the data stream. The new dictionary was the default ASCII table normally used in LZW compression algorithms. This technique provided a basis for comparing and grouping compression ratios using statistical means because now each compression ratio calculation can be considered an independent calculation. The down side to this approach is that the concept of "training" the compression algorithm was abandoned. However, the hope was that there would still be enough distinction between normal traffic and infected traffic to differentiate them using compression ratio of the packet as a metric.

The data plotted in Figure 3 were generated with each packet using the default start-up dictionary in the LZW compression algorithm. Additionally, since this provided a fairly tight grouping of data, a logarithmic model was fit to the data using a least squares algorithm to model the data. Compression ratio as a function of packet byte size is of the following form:

$$CR = A + B*ln(P). \qquad (1)$$

Where *CR* is compression ratio, *A* and *B* are coefficients from a least squares regression, and *P* is packet size in bytes.

Figure 4 shows a plot of the source TCP port 5026 data. The characteristics of the source and destination data are similar to those shown in Figure 3, as indicated by the fit coefficients and standard deviation calculated for each data set. The fit coefficients and data for the source and destination of port 5026 are close enough statistically that they can be considered one data set/model.

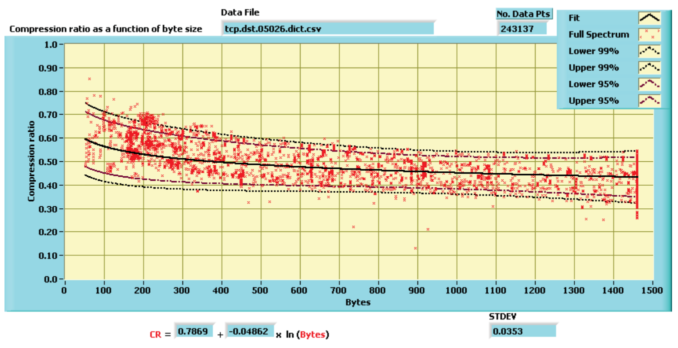

Figure 3. Port 5026 Destination data with default dictionary for every packet.

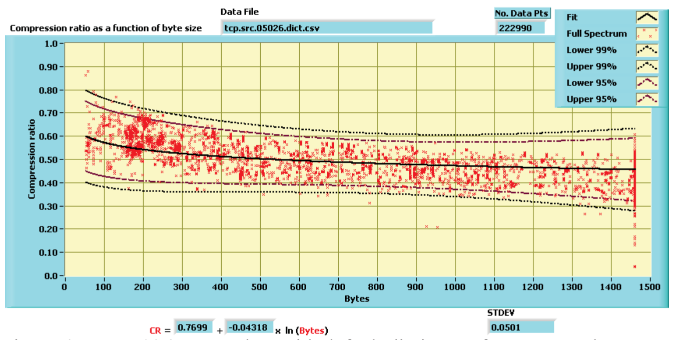

Figure 4. Port 5026 source data with default dictionary for every packet.

B. *Static Dictionary Infected Data Analysis*

Analysis of normal SCADA network traffic appears to group well enough that it will generate a statistical model that has reasonably small error bounds. The next step in the process of defining a way to discriminate malicious traffic from normal network traffic was to look at how infected data compresses and compare it to normal SCADA traffic. Figures 5 and 6 show compromised data sets for the destination and source TCP ports 5026 using a fresh dictionary for the compression of each packet in the network traffic stream. Observations and comparisons to the non-infected data streams are as follows:

- Infected data sets are close enough statistically to be considered one model as evidenced by the fit coefficients calculated for each of the four data sets shown. This is consistent with the normal traffic data sets shown in Figures 3 and 4.
- The compression ratios for infected data sets compared to normal traffic data sets are significantly different for packet sizes less than about 1000 bytes. For larger packets sizes, the difference between the two data sets starts to overlap statistically. This will tend to create large numbers of false negatives and false positive when trying to distinguish malicious traffic from normal traffic which is highly undesirable.
- Start-up or new sessions start issues that are more prevalent at larger packet sizes as seen by the straight lines in the plots of both the normal and compromised data sets.
- Although this is not shown in the data plots presented here, the variability in compression ratio is much larger for small packet sizes of less than about 70 bytes.
- The infected data are not statistically independent from the normal network traffic data in this study because of the way the compromised data was generated (i.e., by inserting malicious information into existing packets). This should be considered a preliminary analysis and drawing absolute conclusions based on these data sets needs to be investigated more thoroughly and compared with independent data sets.

- Confidence levels grow more uncertain when the prediction moves away from the byte size centroid of the data set as seen in all the figures below. This will cause prediction problems when trying to distinguish between normal and infected data at the extreme ends of the predictive models.

C. *Split Packet Analysis*

Compression ratios for packet sizes of less than 300 or 400 bytes appear to be sufficiently different between normal network traffic and infected traffic to categorize data as good or likely suspect based on statistical inferences. However, the difference between the normal network traffic data and infected network data at larger packet sizes are not sufficient to statistically classify the packet as belonging to either class with the high confidence levels needed when dealing with these extremely large data sets. We believe the primary reason for this is that the size of the injected simulated malicious software used for this study was 314, 247, and 341 bytes respectively for the exploits used. So, for initially small packets of network traffic, the infected portion of the packet is large. While for large network packets (e.g., greater than 1000 bytes) the infected portion of it is less than 30% of the total packet size. Therefore, it becomes harder to distinguish malicious network traffic imbedded in larger network packets based on compression techniques. This is simply because they are a smaller portion of the total packet size and do not affect overall compression of the packet significantly enough to distinguish from normal traffic.

A scheme was devised to subdivide all network packets that were larger than 450 bytes and analyzed these sub-packets individually while attributing the analysis results to the entire packet. The algorithm used to break apart larger network data packets was to divide the total packet size by 300 and round the results to the nearest integer. The resulting number was used to break the packet in the "sub-packets." Network packets that were smaller than 450 bytes were left intact by using this method. At 450 bytes, and larger multiples of 300 bytes, the network packet was subdivided into multiple sub-packets. Each sub-packet was compressed separately and the original packet was assigned a compression ratio from the sub-packet that provided the largest compression ratio.

This greatly improved the ability to distinguish malicious data embedded in the larger network data packets while minimally affecting the computed compression ratios for normal or non-infected network traffic as shown in Figure 6. Figure 7 shows a plot of compression ratio data for normal network traffic on the destination side of port 5026 using the split-packet analysis method described above. Compare this plot to Figure 3 and it is evident that the compression ratio stays relatively constant after 450 bytes which is where the split packet algorithm starts to modify the packet analysis. Figure 8 shows a plot of the compression ratio data for infected network data on the destination side of port 5026 and should be compared to Figure 4. These data clearly indicate the advantage of using the split packet analysis for larger network packets containing infected data. The compression ratio is nearly constant with a mean of about 0.82. This can be compared to Figure 5 where the mean of the compression ratio drops significantly as packet size increases. This allows a much easier distinction between normal and infected traffic at the larger packet sizes.

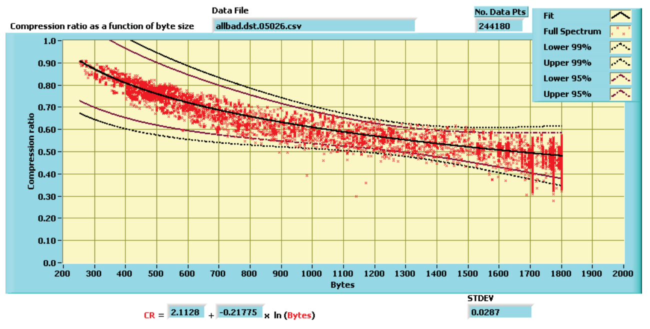

Figure 5. Metasploit infected data at TCP port 5026 destination (default dictionary for each packet).

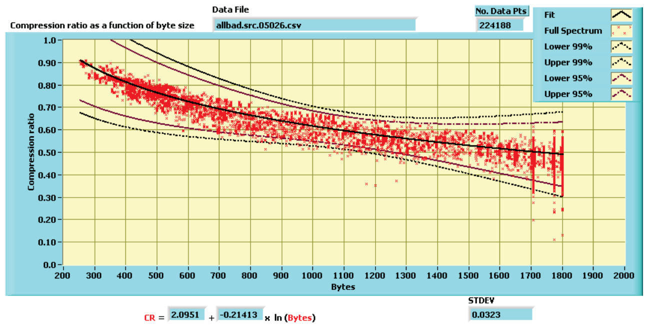

Figure 6. Metasploit infected data at TCP port 5026 source (default dictionary for each packet).

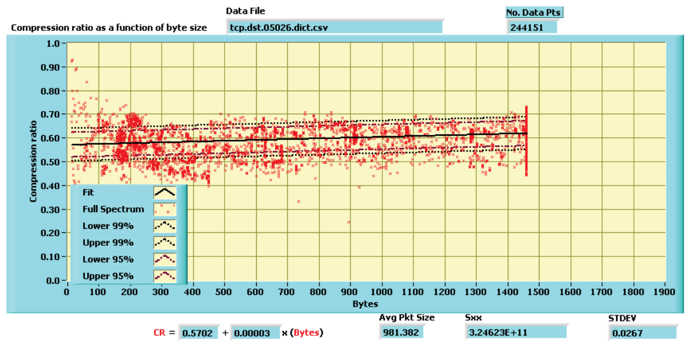

Figure 7. Normal traffic data plotted as a function of byte size using the split packet analysis for TCP port 5026 destination data.

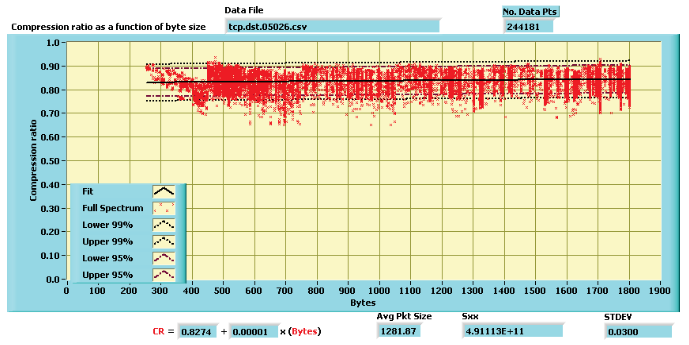

Figure 8. Data plotted as a function of byte size for metasploit infected network traffic using the split packet analysis on TCP port 5026 destination data.

D. Statistical Analysis on a per Byte Size Basis

A variation of the split packet analysis is shown in Figures 9 and 10, and displays the standard deviation for each packet size encountered during the network traffic analysis in which there were more than 20 occurrences of that packet size. For normal distribution assumptions and analysis generally more than 30 observations are required to derive valid statistics on an event. Twenty occurrences were chosen for this analysis. These figures provide the mean and three-sigma standard deviation about the mean for individual packet sizes. Three-sigma standard deviations include the 99th percentile of all occurrences within the data set. The purpose of this presentation was to show a potential way of training an algorithm by creating statistical bounds for normal and malicious traffic tied to packet size. Each distinct packet size would have an acceptable upper and lower boundary that is calculated in the training set. If a packet arrives that falls outside of the established 3-sigma boundaries it would be flagged as suspect. The main difference between this analysis and the previous split-packet analysis is that statistics are created at specific packet sizes rather than generating one model that spans the spectrum of packet sizes.

IV. Conclusions and Recommendations

The use of compression techniques to identify malicious traffic on SCADA networks in real time appears to have some significant merit for infrastructure protection. The preliminary analyses and results presented herein are clearly able to identify malicious network traffic at the packet level at a very high confidence level for the conditions tested. However, the conditions tested are rather limited in scope and should be expanded into more realistic simulations of hacking events using techniques and approaches that are representative of a real-world attack on a SCADA system. Some specific recommendations are as follows:

1. Develop and implement a real-time software prototype capable of sitting on a real SCADA network and collect data based on known, normal SCADA traffic. This would allow a better characterization of the network traffic from both a volume and statistical variability aspect. This activity should be done over several weeks of operations at a minimum.

2. Improve the attack simulation technique employed such that multiple attack types can be simulated and that the attack proceeds along more realistic time events rather than infecting every packet on the simulated network with mal-code as was done in the proof of principle study.

3. Implement a simulated attack on at SCADA system with real end goals of system compromise by using known techniques used by hackers to achieve these goals. The goals of this effort would be characterize real malicious traffic and to determine if this traffic can be discriminated against normal traffic for real-time protection against an attack.

4. Repeat items one and two above for multiple types of SCADA systems and attack implementations as time and money allow.

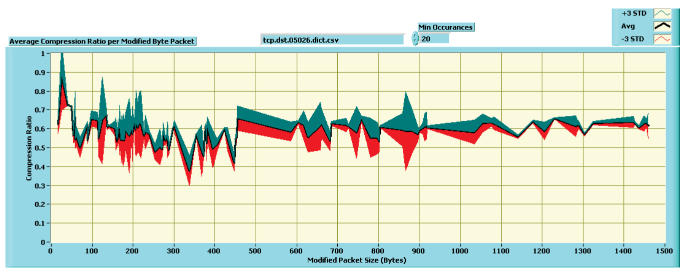

Figure 9. Normal traffic data plotted as a function of byte size using the split packet analysis for TCP port 5026 destination data with statistical analysis displayed for each byte with more than 20 data points.

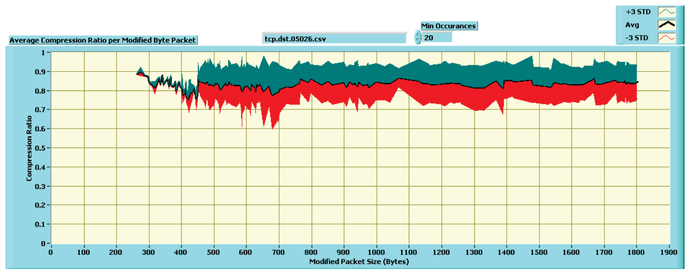

Figure 10. Data plotted as a function of byte size for metasploit infected network traffic using the split packet analysis on TCP port 5026 destination data with statistical analysis displayed for each byte with more than 20 data points.

Disclaimer

Acknowledgment

This work was done for the Department of Energy under contract number DE-AC07-05ID14517 with funding from the U.S. Department of Energy – Office of Electricity Delivery and Energy Reliability under the DOE Idaho Operations Office.

References

[1] Energy Delivery Systems Cybersecurity, Department of Energy's Office of Electricity Delivery and Reliability (OE) homepage, http://energy.gov/oe/technology-development/energy-delivery-systems-cybersecurity.

[2] National SCADA Test Bed – Fact Sheet, U.S. Department of Energy, Office of Electricity Delivery and Energy Reliability.

[3] J. Ziv and A. Lempel, "A Universal Algorithm for Sequential Data Compression," IEEE Transactions on Information Theory, Vol. 23, Issue 3 [pp. 337-343], May 1977.

[4] T. A. Welch, A Technique for High Performance Data Compression, Computer, vol. 17, no. 6, pp. 8-19, June 1984.