7-2012

Employing Entropy in the Detection and Monitoring of Network Covert Channels

Chaim Sanders
Rochester Institute of Technology

Jacob Valletta
Rochester Institute of Technology

Bo Yuan
Rochester Institute of Technology

Daryl Johnson
Rochester Institute of Technology

Peter Lutz
Rochester Institute of Technology

Employing Entropy in the Detection and Monitoring of Network Covert Channels

Chaim Sanders, Jacob Valletta, Bo Yuan, Daryl Johnson, Peter Lutz
Department of Network Security and Systems Administration
Rochester Institute of Technology
Rochester, NY 14623, USA
{ces1509, jrv1197, bxyics, dgjics, phlics}@rit.edu

Abstract*— The detection of covert channels has quickly become a vital need due to their pervasive nature and the increasing popularity of the Internet. In recent years, new and innovative methods have been proposed to aid in the detection of covert channels. Existing detection schemes are often too specific and are ineffective against new covert channels. In this paper, we expound upon previous work done with timing channels and apply it to detecting covert storage channels. Our approach is based on the assumption that the entropy of covert channels will vary from that of previously observed, legitimate, communications. This change in the entropy of a process provides us with a method for identifying storage channels. Using this assumption we created proof of concept code capable of detecting various covert storage channels. The results of our experiments demonstrate that we can successfully detect existing and unpublished covert storage channels accurately.

***Keywords*— covert channel; security; detection; entropy**

1. Introduction

Since Lampson [1] originally introduced the idea of covert channels on trusted systems, network based approaches have become more prevalent [2]. These network based approaches make it extremely challenging to secure traditional networks. Covert channels also prove problematic for individuals, companies, and countries in securing their information or data. In general there have been a number of different approaches, each with varying levels of success for detecting and dealing with network covert channels [2]. Most of these proposed schemes are designed to detect a specific covert channel and rely solely on signatures.

Recently, there has been research into using statistical methods in order to better detect covert channels, with mixed results [31]. Typically the major hindrance of these experiments is due to the large number of possibilities at different levels of network communications where a covert channel might occur. While there is recent research into a general detection scheme for network based timing channels [3], the same type of general solution has not been implemented for storage channels.

In this paper we propose an entropy based detection mechanism for storage based network covert channels. Entropy, as defined by Shannon, is a measure of the uncertainty associated with a random variable [4]. While we observe that a network has many different types of data, we postulate that much of it remains constant with the exception of application payloads and checksum fields. From this we concluded that the creation of a storage channel should behave as a statistical outlier when compared to the normal operation of a network.

In order to detect this we use a large baseline estimate of the network traffic and analyze new entries into the network based on the calculated entropy levels that have been seen. The effect of a given packet will then modify the previous entropy estimation allowing, in time, the system to learn what is acceptable for the network and adapt to new technologies, as well as the detection of storage channels.

2. Background and Related Work

Covert channels have historically been broken up into two separate categories: storage channels and timing channels [7]. More recently a non-ambiguous definition has been devised that defines a storage covert channel as a communication channel in which “the output alphabet consists of different responses all taking the same time to be transmitted” and a timing covert channel as a channel where “the output alphabet is made up of different time values corresponding to the same response” [10, 12]. Additionally, the idea of behavioral based covert channels has been introduced in [11]. These categories are further broken down by the system they use (network or single system), their stealthiness, and bandwidth. The main characteristic of a covert channel is the aim to hide the fact that a transmission is taking place [5]. Early research has shown it is almost impossible to completely eliminate covert channels. Accordingly, the US government has stated that a covert channel with a bandwidth under 1 bit per second is acceptable [7]. Thus the main goal of research in this field is to develop defenses against covert channels such as removing them, disrupting them, managing their threat and alerting to their presence. Since there is always such a large amount of data, the techniques of alerting, managing and disrupting are more realistic and prevalent in research [2].

There has been extensive research devoted to detecting timing channels. Research started with [6] and the identification of covert channel capacity for auditability. Additional auditability techniques were introduced in [15] which allowed for audited applications that depended on relative timing of operations. The first techniques to prevent timing channels were presented in [9, 14] in the forms of pumping and artificial noise injection, these techniques introduce artificial timing delays to communications in order to eliminate timing channels. In [8] the idea of pumping is evolved into that of jammers which can modulate noise and timing in order to reduce the overall capacity of a given channel. Recent research has established a basis for statistically determining a timing channel from amongst legitimate data and alerting on it [3, 5]. Throughout this whole time frame, individuals who developed covert channels also presented signature based defenses for detecting their own specific covert channels [10, 22, 25].

While many effective techniques have been developed to defend against timing based channels less published work is available to detect covert storage channels. We see early work done on calculating bandwidth and capacity of the channels by [17]. Additionally, [11, 16] discuss the possibility of detecting storage channels at the development stage of production. Recently, [17, 19] used network normalization to remove avenues that could potentially carry a covert channel. As with timing channels, individuals authors also presented signature based methods to detect their own storage channel [20, 21, 23, 24]. In our paper we expound upon previous detection techniques available for timing channels [3], applying them to storage channels to provide a viable solution to detect network based storage channels.

3. ENTROPY CALCULATION

3.1 Entropy

Our hypothesis for detection of storage based covert channels focuses on the mathematical formula known as information entropy. As previously stated, entropy is a measure of the uncertainty associated with a random variable. Putting this simply, entropy tells us how much randomness exists in a set of values for a given variable. In this case, the random variable is a field in a network protocol header. By this logic, we are able to gather the values for fields and, after collecting a minimum amount, calculate the randomness of the field value. At some point this randomness value should reach a plateau, if the field itself isn't based on a random function. At this point we assume that this is the normal entropy for that field on the network, and that a significant change in entropy would be a sign of the existence of a covert storage channel. As a quick example, the RFC for ICMP denotes that the code field for an ICMP Echo Request should always be zero. If we assume that all devices on the network are obeying the RFC we should be able to say with certainly that the randomness of this field, over time, should equal zero. The presence of a covert storage channel utilizing the code field would thus create a spike in entropy, triggering an alert.

4. TEST CHANNELS

We selected a number of covert storage channels in order to test how effective our algorithm was at detecting variations on a network. We developed or implemented the following covert channels which had varying levels of bandwidth and stealthiness along with network impact. Each test channel was chosen because it uses a different network protocol, giving us a wider array of covert channels to test.

4.1 ICMPv4 & ICMPv6 Based Covert Channels

In order to first identify a basic covert channel we developed a very simple technique to send data via the ICMP code field. According to the RFC this field should always be zero [29]. This technique was based loosely on the ideas presented in [26] and enabled us to have a moderately stealthy channel from which to develop our test methods around. This concept was then expanded, to include both the identification and sequence numbers in the ICMP Echo header.

To add another protocol to our tests, we created a ICMPv6 covert channel based loosely on [27, 20]. This allowed us to expand our abilities to detect different protocols beyond those based on IPv4. This covert channel uses techniques similar to that of the ICMPv4 variant discussed earlier..

4.2 UDP Based Covert Channel

After demonstrating success with an ICMP covert channel, we examined a storage based covert channel presented in [28]. This covert channel modulates the length fields found in network and transport layer protocols to send packets with even or odd lengths, which in turn translate to a one or zero. This gives the potential to transmit a bit of data per packet. Using standardized protocols such as UDP and IP this model proposes a minimum bandwidth ratio of 1:11680, but can easily be increased depending on the maximum transmission unit for a network.

4.3 TCP Based Covert Channel

Our last test channel was a very simple covert channel found in the TCP protocol. This channel is one of the classic examples of a storage based covert channel and is presented in [14]. This covert channel uses an unused field in the TCP header to send up to 4 bits of data per packet. It should be noted that this is a dated and deprecated covert channel.

5. EXPERIMENTATION

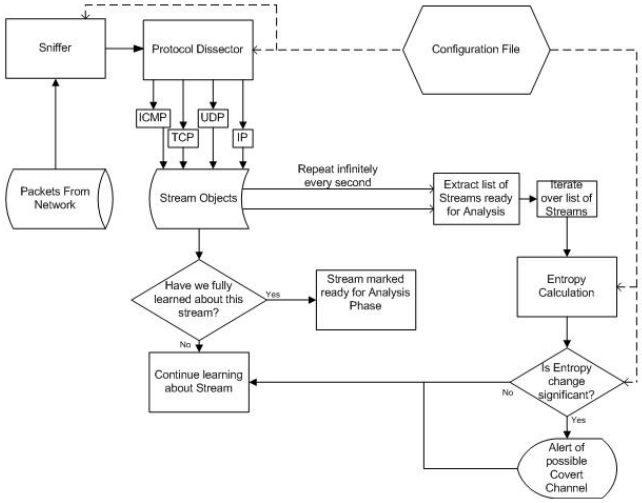
To reinforce our hypothesis that storage channels can be detected using entropy, we developed a proof of concept (PoC) program capable of capturing, dissecting and

calculating entropy associated with network traffic. This program uses the network protocol dissectors built into Scapy, a packet manipulation module for Python. After creating our proof of concept code and the test channels described above, we conducted a few experiments to test the overall effectiveness of the code. Our experiment consisted of using random packet captures from an active private network; each packet capture was 50,000 packets in size and was captured at different points during the day. In each packet capture one of the covert channels that we implemented was executed. We parsed these packets into our software and evaluated our success based on the outcome.

5.1 Proof of Concept Design

The overall flow of the PoC is summarized in the figure below [*Figure 1*]. The program has two phases: a learning phase, and a analysis phase. Even though the program is designed to be a proof of concept, it is still designed to be robust and expandable. This would allow us to move from just a proof of concept to a usable, implementable piece of code.

Figure1PoCFlow



5.1.1 Learning Phase

In the learning phase, the program establishes a network entropy profile by capturing and storing network traffic. The PoC allows a customizable window size of traffic to be specified. This is designed to provide flexibility for various network implementations. The window size is a requirement of the program that specifies how much traffic will be taken in before the actual entropy comparisons are calculated. This is done to establish a base entropy calculation for the network. By definition the larger the window size the more accurate comparisons will be, leading to less false positives. The window size also serves as the buffer for incoming packets during the analysis phase, therefore, a larger window size, although allowing for more comparisons, also requires exponentially more computation. Another feature we have added during this phase is the ability to manually specify which protocols should be analyzed, thereby decreasing the amount of computation required. Once the specified threshold of network traffic has been reached, the program transitions to the analysis phase.

5.1.2 Analysis Phase

The analysis phase is responsible for monitoring and alerting deviations in entropy based on the profile established in the learning phase, as well as to continuously learn and maintain the entropy profile. This process creates a network dependent behavioral based approach to detecting storage based covert channels. We provided a configurable sensitivity level which allows the code to be customizable to a given network. The various setting of the sensitivity parameter control, to some degree, the amount of false positives produced by the code. The sensitivity parameter works hand in hand with the window size parameter specified in the previous section to create a scalable sensitivity that allows the window size to range from a value as small as 64 packets to as large as 500,000 packets and remain accurate.

5.2 Experimental Setup

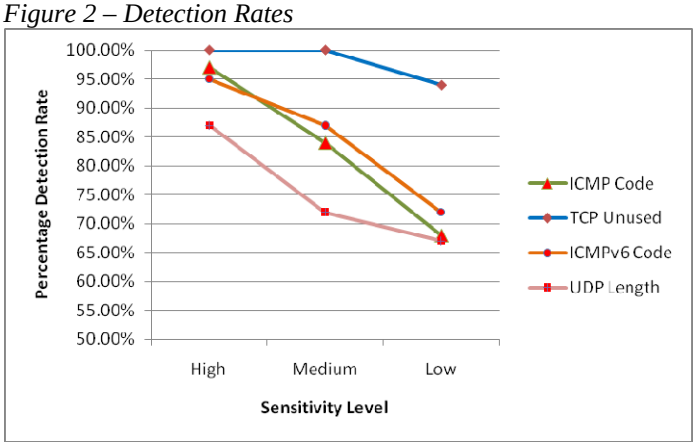
We had a total of four experiments that we conducted in which we attempted to detect our test covert channels. Although each tests was performed independently the same application and principles were used for each. For all four of our experiments we used the same default settings. For our experiments we choose the relatively low training window size of 256, that is, the program took its base assessment of the network from the first 256 packets and started evaluating entropy after that point. While with a higher window size we would see initially lower false positive results the total detection rate should arrive at the same level of effectiveness do to its ability to learn the network. The reason why we selected a low window size is so that we didn't discard vital packet information where our covert channels might be occurring in the experiment data. In real world usage training could be done on many thousands of packets prior to full scale usage. The experiments were all designed to transmit the same message for each channel to maintain consistency for detection purposes. We choose the words "this is a test hello world". Additionally, for each experiment we used 3 different levels of sensitivity, what these levels indicate is the amount of variation in entropy on a given field that was allowed without reporting the event as suspicious. The three levels: low, medium and high had 10.04%, 8.007%, and 3.45% variation allowed respectively.

5.3 Experimental Results

Our experiments indicated that our original notion that covert channels will behave as statistical entropy outliers was correct. With high sensitivity we achieved, on average, a 94.75% detection rate with a false positive rate less than 1%. While that is an accomplishment, having several hundred false positives could present an issue without proper screening. Low sensitivity achieved nearly as impressive detection rates with an average of 75.25% of covert packets

detected with a far lower false positive rate of 124.75 or 0.25%. Ultimately, we believe that the experimental results show that given proper training and configuration our method would result in lower false positive rates while maintaining the expected high detection rates. It is also important to note that false positives that were detected are activity that is not normally seen on the network. That being the case, although it is not a covert channel, the packet has a high probability of being other unwanted traffic. Additionally, although we can see detection rates drop off at low sensitivity in a monitored environment, only one detection of a covert channel should be enough to alert the maintainer of the network to its presence and allow for it to be disabled accordingly.

Figure 2 – Detection Rates



5.3.1 ICMP Code Field Results

When testing for the ICMP Code channel we originally used high sensitivity, although this approach generated many more false positives, 408 precisely, it alerted us to 97% of the packets that contained the covert channel. When we moved to medium sensitivity where an 8.007% change in entropy per second would alert us, we received similarly high results, achieving an 84% detection rate with 263 false positives. Moving to low sensitivity we only achieved a 68% detection rate but we also only had 126 false detections over our entire capture of 50,000 packets.

5.3.2 TCP Unused Field Channel Results

The TCP unused field channel can only transmit 4 bits per packet, as a result very many packet have to be sent in order to get a full message across. This turned out to be a double edged sword in terms of detection for our algorithm. In the short term such a large amount of random traffic was immediately detected on all sensitivity settings. However, we found that if the message was very large the extreme amount of traffic would eventually become expected by the algorithm causing detection rate to drop. Using our test message this channel generated 26 packets, on both high and medium we had a 100% detection rate for this particular channel. When on low sensitivity we dropped to 94%. While the detection rate was high we continued to maintain a comparative steady level of false positives getting 367 false positives on high sensitivity, 197 false positives on medium sensitivity, and 103 false positives on low sensitivity.

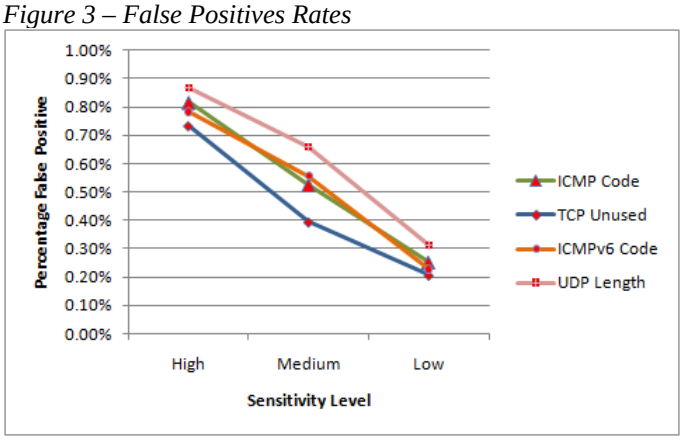
5.3.3 ICMPv6 Code Channel Results

Being very similar to the original ICMP channel we expected similar results in terms of detection rates and false positives. We were indeed not disappointed, the variation that we had is most likely a result of the different traffic between the two captures. Our results had 392, 278, and 114 false positives on high, medium, and low sensitivity levels respectively. We achieved slightly better detection rates compared to the regular ICMP channel as well - detecting 72% on low sensitivity, 87% on medium sensitivity, and 95% on high sensitivity. A reason for these slightly better rates could be due to the fact that although the network captures contain IPv6 traffic it is relatively low due to the amount of IPv4 packet. This discrepancy, although true in real life, also makes it harder for our covert channel to blend in thereby making it easier to detect.

5.3.4 UDP Length Channel Results

Our final channel dealt with UDP length and had very interesting properties where it didn't try to modify an actual field but rather modified the packet itself in order to add length. While we expected to be able to detect this channel we expected the results to be lower detection rates then the other channels. We had very similar false positives to the other channels having 433, 329, 156 false positives with regards to high, medium and low sensitivity respectively. The detection rates, while existent, were lower here being 87, 72, and 67 percent

Figure 3 – False Positives Rates



6. LIMITATIONS

In this section we discuss possible limitations of our entropy based system to detect covert channels. A large requirement imposed by this system is the need for intense packet analysis. This scanning can be done on existing packet captures but is most effective on a live network. This type of

packet analysis is most analogous to deep packet inspection and requires an in-line component that will slow down the network.

Additionally, at this time, our approach is only possible on protocols for which the specification is known and using unencrypted transmissions. The reason for this is due to the entropy calculation on a per field basis. As a result of the requirements there are several countermeasures to this type of detection, the most obvious being encryption. Other countermeasures include: using unknown protocols and using non-network based covert channels. As with most entropy based systems adjusting the system correctly to minimize false positives, while still reporting legitimate covert channels is pivotal.

As a final limitation, our tool is still not one-hundred percent error free. Because of its fully adaptive capabilities, our proof of concept is prone to false-positive results, especially in high entropy fields. This means that the tool, while effective, might require additional human examination of network traffic.

7. CONCLUSIONS AND FUTURE WORK

We expounded on previous work for using entropy to detect timing channels and applied it successfully to the detection of storage based covert channels. We designed and created a system which is implemented for some of the more prevalent packet types and allows easy expansion in the future for other types. We utilized a standard entropy calculation in order to identify a number of different covert storage channels.

In the future we would like to explore the connections between other suspicious activity and the use of covert channels. Additionally, we would like to see if the current detection system could be adapted to not only detect covert channels, but also to block and filter them. This type of system could take advantage of normalization on only suspect packets thereby making the previous work on normalization more effective, turning our passive warden system into an active warden system. Furthermore, we want to investigate ways to further reduce the false positive rate while keeping the detection rate stable.

REFERENCES

[1] B. Lampson. “A Note on the Confinement Problem.” Commun. ACM, vol. 16, no. 10, Oct. 1973, pp. 613–615.

[2] S. Zander. “A Survey Of Covert Channels and Countermeasures In Computer Network Protocols.” IEEE Communications Surveys, vol.9, no. 3, 3rd Quarter 2007.

[3] S. Gianvecchio and H. Wang. “Detecting Covert Timing Channels: An Entropy-Based Approach.” Proceedings of the 14th ACM conference on Computer and communications security (CCS '07). ACM, New York, NY, USA, pp. 307-316.

[4] C. Shannon. “A mathematical theory of communication.” Bell System Technical Journal Vol. 27, July and Oct. 1948).

[5] V. Berk et al. “Detection of Covert Channel Encoding in Network Packet Delays.” Technical Report TR536, Revision 1, revised November 2005.

[6] J. K. Millen. “Finite-state Noiseless Covert Channels.” Proc. Computer Security Foundations Workshop 11, Franconia, NH, June 1989, pp. 81-86.

[7] U.S. Department of Defense. Trusted computer system evaluation "The Orange Book". DoD 5200.28-STD Washington: GPO:1985, 1985

[8] J. Giles and B. Hajek. “An information-theoretic and game-theoretic study of timing channels.” IEEE Transaction on Information Theory, volume 48, pages 2455-2477, September 2003.

[9] M. Kang, I. Moskowitz, and D. Lee. A network version of the pump. In Proceedings of the IEEE Symposium in Security and Privacy, pages 144-154, May 1995.

[10] CR. Tsai. "A Formal Method for the Identification of Covert Storage Channels in Source Code." Proceedings of the 1987 IEEE Symposium on Security and Privacy, pp. 74-87.

[11] D. Johnson et al. “Behavior-Based Covert Channel in Cyberspace.” Intelligent Decision Making Systems, pp. 311-318.

[12] I. S. Moskowitz and M. H. Kang. “Covert channels - Here to stay?” Proceedings of the 9th Annual Conference on Computer Assurance (COMPASS’94). National Institute of Standards and Technology, pp. 235–244.

[13] H. Wei-Ming, “Reducing Timing Channels with Fuzzy Time,” Proc. IEEE Computer Society Symp. Research in Security and Privacy, May 1991, pp. 8–20.

[14] T. Handel and M. Sandford. “Hiding data in the OSI network model.” First International Workshop on Information Hiding(Cambridge, U.K.), May-June 1996.

[15] P. M. Melliar-Smith and L. E. Moser. “Protection against covert storage and timing channels.” Proc. Computer Security Foundations Workshop IV, June 1991, pp. 209–214.

[16] L. E. Moser, “Data dependency graphs for Ada programs,” IEEE Transactions on Software Engineering, vol. 16, no. 5, pp. 498-509.

[17] CR. Tsai and V. D. Gligor. “A Bandwidth Computation Model for Covert Storage Channels and its Applications.” Proc. IEEE Conf. on Security and Privacy, 1988.

[18] M. Handley and V. Paxson. “Network Intrusion Detection: Evasion, Traffic Normalization, and End-to-End Protocol Semantics.” Proceedings of the 10th conference on USENIX Security Symposium - Volume 10, 2001.

[19] G. R. Malan et al. “Transport and Application Protocol Scrubbing.” Proc. IEEE Conf. Computer Communications (INFOCOM), Mar. 2000, pp. 1381–1390.

[20] N. B. Lucena et al. “Covert Channels in IPv6.” Proc. Privacy Enhancing Technologies (PET), May 2005, pp. 147–166.

[21] A. Singh et al. “Malicious ICMP Tunneling: Defense against the Vulnerability.” Proc. 8th Australasian Conf. Information Security and Privacy (ACISP), July 2003, pp. 226–235.

[22] N. Schear et al. “Glavlit: Preventing Exfiltration at Wire Speed.” Proc. 5th Wksp. Hot Topics in Networks (HotNets), Nov. 2006.

[23] C. H. Rowland. “Covert Channels in the TCP/IP Protocol Suite.” First Monday, Peer Reviewed Journal on the Internet, July 1997.

[24] daemon9. “LOKI2: The Implementation.” Phrack Magazine, vol. 7, no. 51, Sept. 1997.

[25] S. Cabuk et al. “IP Covert Timing Channels: Design and Detection.” Proc. 11th ACM Conf. Computer and Communications Security (CCS), Oct. 25–29 2004, pp. 178–187.

[26] T. Sohn et al. “Covert Channel Detection in the ICMP Payload Using Support Vector Machine.” Lecture Notes in Computer Science, Volume 2776, 2003, pp. 461-464.

[27] R. P. Murphy. “V00d00n3t – Ipv6 / ICMPv6 Covert Channel.” Slides from DEFCON, 2006.

[28] B. Yuan et al. “A Covert Channel in Packet Switching Data Network.” The Second Upstate New York Workshop on Communications and Networking, Rochester, NY, November 2005.

[29] J. Postel. “Internet Control Message Protocol.” Network Working Group, RFC 792, Sept. 1981, pp. 14.

[30] E. Tumoian and M. Anikeev. "Detecting NUSHU Covert Channels Using Neural Networks." Technical report, Taganrog State University of Radio Engineering, 2005.