Superimposing Permutational Covert Channels onto Reliable Stream Protocols

Jamie Levy
Dept. of Math. & Comp. Science
John Jay College, CUNY
New York, NY 10019, U.S.A.
jamie.levy@jjay.cuny.edu

Jaroslaw Paduch
jpaduch@jjay.cuny.edu

Bilal Khan
bkhan@jjay.cuny.edu

Abstract

In this paper, we present a new implicit encoding technique that makes use of lower-layer packet reordering to superimpose covert messages onto a reliable data stream. In particular, since the TCP layer provides a reliable in-order data stream over the unreliable network layer's IP datagram service, we can encode covert messages by artificially permuting IP packets before they leave the source and reading the permutation at the destination prior to delivering the payload to TCP. Applying such permutations will not adversely affect TCP's ability to reconstitute the transport layer data stream, since TCP is designed to be robust against out of order network layer packet delivery. We describe the design and operation of PERMEATE*, an open-source covert channel toolkit which implements such a permutational covert channel over TCP, and we provide a quantitative assessment of it's efficacy and efficiency as a covert channel.*

1. Introduction

A covert channel is a mechanism for steganographically superimposing illegitimate data onto a legitimate network data stream. Although the illegitimate message is sent unencrypted, it remains unnoticed since it is being carried "inside" a legitimate flow in such a manner that it does not essentially alter the semantics of the legitimate data when it arrives at the intended recipient. To date, most covert channels devised can be placed in one of two broad categories: they either (i) *explicitly* encode secret information in unused portions of packet headers, or (ii) *implicitly* encode secret information in inter-packet timing intervals.

- Explicit encodings (storage channels) are more commonly seen in literature and are easier to implement. The basic strategy is to embed covert information into fields that are either not used or can be readily changed with little damage to packet processing and semantics. To encode the covert message, appropriate bits are changed in the packet headers; to decode, these bits are read off the header fields at the destination. Several implementations of storage channels built using TCP, IP and UDP headers are readily available on the web [1, 2, 3, 4, 5, 6, 7]. There are also implementations that use ICMP [4, 8, 9, 10, 11], HTTP [12, 13, 14, 15], DNS [16, 17] and MSN [18] protocols.

- Implicit encodings (timing channels) convey a message based on the time in between successive packet transmissions. According to the "Orange Book" [19], a timing channel is possible whenever "one process is allowed to signal information to a second process by modulating its own use of system resources" . For example, one design might follow Morse code: three packets sent across the wire in a short amount of time might be conveyed as an 'S' (dot dot dot), while sending out three more packets spaced out with a larger amount of time in between them might convey the message 'O' (dash dash dash), etc. In order for timing channels to be successful, both parties must be operating in a steady state with respect to network and CPU load, since otherwise abrupt changes in the traffic environment will be yield errors in the covert channel. More problematically, channels error rates degrade when instrumented in the wide area, since traffic shaping/multiplexing at intermediate (network layer) routers has the effect of reducing the variance of inter-packet time distributions. Thus, in order for the bimodal distribution of interpacket timings to remain separable at the destination, one requires extreme differences in the interpacket timings used to encode covert bits. This endows the shape of the legitimate traffic with a very low entropy, making it appear bursty

[14] A. Dyatlov, "Firepass." [Online]. Available: http://www.gray-world.net/pr_firepass.shtml

[15] D. Uid, "hcovert." [Online]. Available: http://sourceforge.net/projects/hcovert/

[16] T. M. Gil, "NSTX." [Online]. Available: http://thomer.com/howtos/nstx.html

[17] T. Pietraszek, "DNScat." [Online]. Available: http://tadek.pietraszek.org/projects/DNScat/

[18] W. Zheng, "MsnShell." [Online]. Available: http://gray-world.net/pr_msnshell.shtml

[19] "U.S. Department of Defense Trusted computer system evaluation," 1985. [Online]. Available: http://csrc.ncsl.nist.gov/publications/secpubs/rainbow/std001.txt

[20] H. Meer and M. Slaviero, "It's all about the timing..." [Online]. Available: http://www.sensepost.com/research/squeeza/dc-15-meer_and_slaviero-WP.pdf

[21] T. V. Vleck, "Timing Channels." [Online]. Available: http://www.multicians.org/timing-chn.html

[22] H. Meer and M. Slaviero, "Squeeza." [Online]. Available: http://www.sensepost.com/research/squeeza/

[23] daemon9, "Project Loki," in *Phrack Volume 49*. [Online]. Available: http://www.phrack.org/issues.html?issue=49\&id=6\&mode=txt

[24] ——, "LOKI 2(the implementation)," in *Phrack Volume 51*. [Online]. Available: http://www.phrack.org/issues.html?issue=51\&id=6#article

[25] E. Skoudis and L. Zeltser, "Malware: Fighting malicious code." O'Reilly, 2004.

[26] FuSyS, "007shell." [Online]. Available: http://packetstormsecurity.org/groups/s0ftpj/007shell.tgz

[27] FX, "Cd00r." [Online]. Available: http://www.phenoelit-us.org/stuff/cd00r.c

[28] R. Bejtlich, "Chained covert channels," in *The Tao of Network Security Monitoring*. Addison-Wesley, 2005.

[29] C. M. Nyberg, "Sadoor." [Online]. Available: http://packetstormsecurity.org/UNIX/penetration/rootkits/index7.html

[30] T. Redaelli, "Helldoor." [Online]. Available: http://utenti.gufi.org/~drizzt/codes/helldoor/

[31] Bioforge, "Hacking the linux kernel network stack," in *Phrack Magazine, vol 61*. [Online]. Available: http://www.phrack.org/archives/61/p610x0d_Hacking_the_Linux_Kernel_Network_Stack.txt

[32] J. Rutkowska, "The implementation of passive covert channels in the linux kernel." [Online]. Available: http://www.invisiblethings.org/papers/passivecovertchannelslinux.pdf

[33] I. S. Moskowitz and M. H. Kang, "Covert channels - here to stay?" in *Proceedings of the 9th Annual Conference on Computer Assurance*. National Institute of Standards and Technology, 1994, pp. 235–244. [Online]. Available: http://citeseer.ist.psu.edu/moskowitz94covert.html

[34] F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn, "Information hiding — A survey," *Proceedings of the IEEE*, vol. 87, no. 7, pp. 1062–1078, 1999. [Online]. Available: citeseer.ist.psu.edu/petitcolas99information.html

[35] I. Moskowitz, R. Newman, D. Crepeau, and A. Miller, "Covert channels and anonymizing networks," in *Proceedings of the Workshop on Privacy in the Electronic Society (WPES)*, 2003. [Online]. Available: citeseer.ist.psu.edu/moskowitz03covert.html

[36] J. F. Kurose and K. W. Ross, *Computer Networking: A Top-Down Approach*. Addison-Wesley, 2006.

[37] S. J. Murdoch and S. Lewis, "Embedding Covert Channels into TCP/IP," in *Proceedings of the Information Hiding Workshop*, 2005.

[38] W. Myrvold and F. Ruskey, "Ranking and unranking permutations in linear time," *Information Processing Letters*, vol. 79, no. 6, pp. 281–284, 2001. [Online]. Available: citeseer.ist.psu.edu/myrvold00ranking.html

[39] P. Biondi, "Scapy." [Online]. Available: http://www.secdev.org/projects/scapy/

[40] A. Orebaugh, G. Ramirez, J. Burke, and L. Pesce, *Wireshark & Ethereal Network Protocol Analyzer Toolkit (Jay Beale's Open Source Security)*. Syngress Publishing, 2006.

[41] J. Toledo, "EtherApe." [Online]. Available: http://etherape.sourceforge.net/

[42] V. Jacobson, "Congestion avoidance and control," *Computer Communication Review*, vol. 18, pp. 314–329, Aug. 1988.

[43] V. Paxson and M. Allman, "RFC2988: Computing TCP's retransmission timer," United States, 2000.